[МАРКОС ЛОПЕЗ ДЕ ПРАДО]

МАШИННОЕ ОБУЧЕНИЕ: АЛГОРИТМЫ ДЛЯ БИЗНЕСА

$$T^* = \arg\min_T \left\{ \left| \theta_T \right| \geq \mathrm{E}_0 \left[T \right] \left| 2\mathrm{P} \left[b_t = 1 \right] - 1 \right| \right\}$$

$$\theta_T = \max \left\{ \sum_{t|b_t=1}^{T} b_t v_t, - \sum_{t|b_t=-1}^{T} b_t v_t \right\}$$

Advances in Financial Machine Learning

MARCOS LÓPEZ DE PRADO

WILEY

[МАРКОС ЛОПЕЗ ДЕ ПРАДО]

МАШИННОЕ ОБУЧЕНИЕ: АЛГОРИТМЫ ДЛЯ БИЗНЕСА

$$T^* = \arg\min_T \left\{ \left| \theta_T \right| \geq \mathrm{E}_0 \left[T \right] \left| 2\mathrm{P} \left[b_t = 1 \right] - 1 \right| \right\}$$

$$\theta_T = \max \left\{ \sum_{t|b_t=1}^{T} b_t v_t - \sum_{t|b_t=-1}^{T} b_t v_t \right\}$$

ПИТЕР®

Санкт-Петербург • Москва • Екатеринбург • Воронеж
Нижний Новгород • Ростов-на-Дону
Самара • Минск

2019

ББК 32.813
УДК 004.8
П68

Прадо де Маркос Лопез

П68 Машинное обучение: алгоритмы для бизнеса. — СПб.: Питер, 2019. — 432 с.: ил. — (Серия «IT для бизнеса»).

ISBN 978-5-4461-1154-1

Маркос Лопез де Прадо делится тем, что обычно скрывают, — самыми прибыльными алгоритмами машинного обучения, которые он использовал на протяжении двух десятилетий, чтобы управлять большими пулами средств самых требовательных инвесторов.

Машинное обучение меняет практически каждый аспект нашей жизни, алгоритмы МО выполняют задачи, которые до недавнего времени доверяли только проверенным экспертам. В ближайшем будущем машинное обучение будет доминировать в финансах, гадание на кофейной гуще уйдет в прошлое, а инвестиции перестанут быть синонимом азартных игр.

Воспользуйтесь шансом поучаствовать в «машинной революции», для этого достаточно познакомиться с первой книгой, в которой приведен полный и систематический анализ методов машинного обучения применительно к финансам: начиная со структур финансовых данных, маркировки финансового ряда, взвешиванию выборки, дифференцированию временного ряда... и заканчивая целой частью, посвященной правильному бэктестированию инвестиционных стратегий.

16+ (В соответствии с Федеральным законом от 29 декабря 2010 г. № 436-ФЗ.)

ББК 32.813
УДК 004.8

ISBN 978-1119482086 англ.
ISBN 978-5-4461-1154-1

Оглавление

Отзывы на книгу «Машинное обучение: алгоритмы для бизнеса»

«В своей новой книге "Машинное обучение: алгоритмы для бизнеса" известный ученый-финансист Маркос Лопез де Прадо поражает метким ударом карате наивные и часто статистически переподогнанные методы, которые так распространены в современном финансовом мире. Он отмечает, что в современных высокотехнологичных финансах подходы "бизнес в привычном режиме" в значительной степени бессильны, а во многих случаях они фактически демонстрируют тенденцию к потере денег. Но Лопез де Прадо не просто разоблачает математические и статистические грехи финансового мира. Вместо этого он предлагает технически обоснованную дорожную карту для финансовых специалистов по присоединению к волне машинного обучения. Что особенно освежает, так это эмпирический подход автора — его внимание сосредоточено на анализе реальных данных, а не на чисто теоретических методах, которые могут выглядеть красиво на бумаге, но во многих случаях в значительной степени неэффективны на практике. Книга ориентирована на профессионалов в области финансов, которые уже знакомы с методами статистического анализа данных, но она стоит усилий для тех, кто хочет проделать реальную современную работу в этой области».

Д-р Дэвид Х. Бейли, бывший руководитель компании Complex Systems, Национальная лаборатория Лоуренса Беркли. Сооткрыватель алгоритма BBP spigo

«Финансы эволюционировали от сборника эвристик, основанного на исторической финансовой отчетности, до очень сложной научной дисциплины, основанной на компьютерных фермах по анализу массивных потоков данных в режиме реального времени. Недавние весьма впечатляющие достижения в области машинного обучения изобилуют как обещаниями, так и опасностью применительно к современным финансам. В то время как финансы примеряются к нелинейностям и крупным совокупностям данных, на которых преуспевает машинное обучение, они также примеряются к шумным данным и человеческому элементу системы, которые в настоящее время лежат за пределами области действия стандартных методов машинного обучения. Человеку свойственно ошибаться, но если вы действительно хотите все вконец раз***ать, используйте компьютер. Доктор Лопез де Прадо написал первую исчерпывающую книгу, описывающую применение современного

машинного обучения в финансовом моделировании. Книга сочетает в себе новейшие технологические наработки в области машинного обучения с критическими жизненными уроками, извлеченными из многолетнего финансового опыта автора в ведущих академических и промышленных учреждениях. Настоятельно рекомендую эту увлекательную книгу как будущим студентам финансовых факультетов, так и преподавателям и руководителям».

Питер Карр, профессор кафедры финансов и рискового инжиниринга, инженерная школа NYU Tandon

«Маркос — провидец, который неустанно работает над продвижением финансовой сферы. Его стиль письма является всеобъемлющим и мастерски соединяет теорию с приложением. Не часто можно найти книгу, способную преодолеть этот разрыв. Эта книга является обязательной для прочтения как для практиков, так и для технологов, работающих над решениями для инвестиционного сообщества».

Лэндон Даунс, президент и соучредитель IQBit

«Ученые, которые хотят понять современный инвестиционный менеджмент, должны прочитать эту книгу. В ней Маркос Лопез де Прадо объясняет, как портфельные менеджеры используют машинное обучение для выведения, тестирования и использования торговых стратегий. Он делает это с очень необычной комбинацией академической перспективы и обширного опыта в промышленности, что позволяет ему подробно объяснить, что происходит в промышленности и как это работает. Подозреваю, что некоторые читатели найдут фрагменты книги, которые они не понимают или с которыми они не согласны, но каждый, кто заинтересован в понимании применения машинного обучения к финансам, только выиграет от прочтения этой книги».

Профессор Дэвид Исли, Корнелльский университет. Председатель экономического Консультативного совета NASDAQ-OMX

«На протяжении многих десятилетий финансы целиком опирались на чрезмерно упрощенные статистические методы для выявления закономерностей, или шаблонов, в данных. Машинное обучение обещает изменить это, позволяя исследователям использовать современные нелинейные и высокоразмерные методы, аналогичные тем, которые используются в научных областях, таких как анализ ДНК и астрофизика. В то же время применение этих алгоритмов МО для моделирования финансовых задач будет опасно. Финансовые задачи требуют очень четких решений. Книга д-ра Лопеза де Прадо является первой, которая дает четкую характеристику того, что делает стандартные инструменты машинного обучения безуспешными применительно к области финансов, и он первый, кто обеспечил практические решения уникальных задач, с которыми

сталкиваются менеджеры активов. Все, кто хочет понять будущее финансов, должны прочитать эту книгу».

Профессор Франк Фабоцци, бизнес-школа EDHEC. Редактор журнала Portfolio Management («Портфельный менеджмент»)

«Это отрадный отход от накопления знаний, которое наводнило квантитативное финансирование. Лопез де Прадо определяет для всех читателей следующую эру финансов: промышленные научные исследования, приводимые в движение машинами».

Джон Фосетт, основатель и генеральный директор Quantopian

«Маркос собрал в одном месте бесценную коллекцию уроков и методов для практиков, стремящихся применить методы машинного обучения в финансах. Если машинное обучение — это новое и потенциально мощное оружие в арсенале квантитативных финансов, то проницательная книга Маркоса под завязку нагружена полезными советами, которые помогут любопытствующему практику не идти по тупиковому пути или не стрельнуть себе в ногу».

Росс Гарон, глава Cubist Systematic Strategies. Управляющий директор Point72 Asset Management

«Первую волну квантитативных инноваций в финансах возглавила оптимизация Марковица. Машинное обучение — это вторая волна, и она коснется каждого аспекта финансов. Достижения Лопеза де Прадо в области финансового машинного обучения имеют важное значение для читателей, которые хотят быть впереди технологий, а не быть ими замененными».

Профессор Кэмпбелл Харви, Университет Дьюка. Бывший президент Американской финансовой ассоциации

«Как понять сегодняшние финансовые рынки, на которых сложные алгоритмы маршрутизируют ордера, финансовые данные объемисты, а скорость торговли измеряется в наносекундах? В этой важной книге Маркос Лопез де Прадо излагает новую парадигму менеджмента инвестициями, построенную на машинном обучении. Отнюдь не являясь "черно-ящичным" методом, эта книга ясно объясняет инструменты и процесс финансового машинного обучения. Для ученых и практиков эта книга заполняет важный пробел в нашем понимании управления инвестициями в эпоху машин».

Профессор Морин О'Хара, Корнельский университет. Бывший президент Американской финансовой ассоциации

«Маркос Лопез де Прадо выпустил чрезвычайно своевременную и важную книгу по машинному обучению. Страницы этой книги блистают академическими и профессиональными познаниями автора — в самом деле, приходят на ум лишь немногие, если таковые вообще имеются, авторы, более подходящие для объяснения как теоретических, так и практических аспектов этой новой и (для большинства) незнакомой темы. Как новички, так и опытные профессионалы найдут проницательные идеи и поймут, как данный предмет может быть применен новыми и полезными способами. Исходный код на языке Python даст начинающим читателям старт и позволит им быстро оценить данный предмет на практике. Книге суждено стать классикой в этой быстро развивающейся области».

Профессор Риккардо Rebonato, бизнес-школы EDHEC. Бывший руководитель отдела ставок и валютной аналитики PIMCO

«Экскурсия по практическим аспектам машинного обучения в области финансов, наполненная идеями о том, как использовать передовые методы, такие как дробное дифференцирование и квантовые компьютеры, чтобы получить понимание и конкурентное преимущество. Полезный том для практиков финансов и машинного обучения».

Доктор Коллин Уильямс, руководитель отдела исследований, D-Wave Systems

Памяти моего соавтора и друга, профессора Джонатана Боруэина, FRSC, FAAAS, FBAS, FAustMS, FAA, FAMS, FRSNSW (1951–2016)

Очень мало известных нам вещей не могут быть сведены к математическим рассуждениям, а невозможность этого означает, что наше знание о них весьма незначительно и туманно. Но если мы способны математически рассуждать о вещи, то будет столь же глупо отказываться от этого, как искать что-то в темноте на ощупь, имея рядом свечу.

Джон Арбетнот (1667–1735)
Предисловие к «Законам теории вероятностей» (1692)

Об авторе

Д-р Маркос Лопез де Прадо управляет несколькими многомиллиардными фондами для институциональных инвесторов, используя алгоритмы машинного обучения (МО) и суперкомпьютеры. Он основал компанию Guggenheim Partners' Quantitative Investment Strategies (QIS), где разработал высокоэффективные стратегии, которые неизменно обеспечивали превосходные возвраты на вложенный капитал с поправкой на риск. После управления активами в размере до 13 миллиардов долларов Маркос приобрел QIS и успешно развернул этот бизнес в Гуггенхайме в 2018 году.

С 2010 года Маркос является научным сотрудником Национальной лаборатории имени Лоуренса в Беркли (Министерство энергетики США, управление по научным исследованиям). Ему удалось войти в топ-10 самых читаемых авторов в области финансов (согласно рейтингу портала SSRN, открытого электронного репозитория научных статей) благодаря публикациям десятков научных статей, посвященных машинному обучению и суперкомпьютерным вычислениям в ведущих научных журналах. Кроме того, он владеет правами на многочисленные международные патентные заявки на вычислительные идеи (алгоритмический трейдинг).

Маркос получил докторскую степень в области финансовой экономики (2003), вторую степень в области математических финансов (2011) в Мадридском университете Комплутенсе и является лауреатом Национальной премии Испании за академическое мастерство (1999). Он завершил свои постдокторские исследования в Гарвардском университете и Корнельском университете, где преподает курс финансового машинного обучения в инженерной школе. Маркос имеет число Эрдеша 2 и число Эйнштейна 4 согласно рейтингам Американского математического общества.

Дополнительную информацию можно найти на www.QuantResearch.org.

От научного редактора перевода

Это одна из первых, если не самая первая книга по алгоритмическому трейдингу на русском языке, и она безусловно вызовет интерес как у специалистов в области биржевой торговли, так и у специалистов по машинному обучению.

Настоящая книга подчеркивает и раскладывает по полочкам то, как финансовое машинное обучение следует рассматривать в качестве научного процесса. Например, одним из наиболее распространенных ложных допущений, рассматриваемых

в книге, является допущение об одинаковой распределенности и взаимной независимости выборок из финансовых временных рядов.

В первой части обсуждаются различные виды финансовых данных и способы их использования для целей анализа и тренировки. Во второй части обсуждается моделирование, охватывающее важные алгоритмы, их достоинства и недостатки применительно к финансовым данным. Часть третья рассказывает о вреде бэктестирования в современной практике и как правильно его проводить и интерпретировать проверочные статистические показатели. В этой части рассказывается об автоматическом размещении активов, которое решает проблему минимальной дисперсии по сравнению с методами Марковица или паритета рисков. В четвертой части рассматриваются различные передовые приложения, связанные с квантитативным трейдингом, конструированием портфеля, рыночной микроструктурой. Часть пятая не для слабонервных, поскольку она знакомит с различными передовыми методами информатики для ускорения производительности алгоритмов МО на практике, включая квантовые вычисления.

Для того чтобы расширить аудиторию книги за счет разработчиков, работающих с алгоритмами МО, и наоборот, чтобы облегчить трейдерам работу с технологией МО, настоящий перевод снабжен сносками и небольшим глоссарием основных терминов из финансов, машинного обучения, теории вероятностей и статистики, которые облегчат понимание материала книги и помогут «зацепиться» за тему. Почти все термины содержат ссылку на источник.

Книга дополнена переводом статьи, которая так и называется: «Введение в книгу "Машинное обучение: алгоритмы для бизнеса" Маркоса Лопеза де Прадо» от 23 августа 2018 г., опубликованной на портале Quantopian, посвященном квантитативным методам и алгоритмическому трейдингу. В данной статье резюмирована процедура применения алгоритмов МО и даны обобщающие разъяснения предлагаемых автором книги нововведений.

Одним из ключевых понятий биржевой торговли является финансовый возврат (return). В данной книге это понятие проходит красной нитью через все главы, и крайне важно с самого начала понять его правильно. С точки зрения трейдера, если представить ценовую информацию в виде барного временного графика, то упрощенно финансовый возврат в абсолютных величинах — это не что иное, как разница между ценой в момент времени $t - 1$ и ценой в момент времени t. Если трейдер сделал ставку в момент времени $t - 1$, то он получает положительный, нулевой или отрицательный финансовый возврат в момент времени t. С точки зрения портфельного менеджера, это возврат на портфельные инвестиции в течение любого периода оценивания, включающего изменение рыночной стоимости портфеля. Процентное соотношение этого показателя называется возвратностью (rate of return). Цель машинного обучения в финансах в том и состоит, чтобы натренировать автоматически обучающуюся систему предсказывать состояние рынка и цену актива/стоимость портфеля в момент времени $t + 1$ с целью максимизации финансового возврата, или возвратности.

Надеюсь, что данная книга станет настольной у каждого кванта и разработчика автоматически обучающихся систем.

https://ru.tradingview.com/u/capissimo/#published-scripts

От издательства

Экспериментальные решения некоторых упражнений из книги вы можете найти на GitHub по ссылке: https://github.com/BlackArbsCEO/Adv_Fin_ML_Exercises. В этом репозитории вы также найдете ссылки и на другие проекты, основанные на исследованиях д-ра Прадо.

Ваши замечания, предложения, вопросы отправляйте по адресу comp@piter.com (издательство «Питер», компьютерная редакция).

Мы будем рады узнать ваше мнение!

На веб-сайте издательства www.piter.com вы найдете подробную информацию о наших книгах.

ПРЕАМБУЛА

Глава 1. Финансовое машинное обучение как отдельная дисциплина

1 Финансовое машинное обучение как отдельная дисциплина

1.1. Актуальность

Машинное обучение (МО, machine learning) меняет практически каждый аспект нашей жизни.

Сегодня алгоритмы МО выполняют задачи, которые до недавнего времени могли выполнять только эксперты. Что касается финансов, то сейчас самое интересное время внедрять революционную технологию, которая изменит то, как каждый инвестирует из поколения в поколение. Эта книга объясняет научно обоснованные инструменты машинного обучения, которые работали для меня в течение двух десятилетий и помогли мне управлять большими пулами средств для некоторых самых требовательных институциональных инвесторов.

Книги об инвестициях в основном делятся на две категории. С одной стороны, мы находим книги, написанные авторами, которые не практикуют то, чему они учат. Они содержат чрезвычайно элегантную математику, описывающую мир, которого не существует. Просто потому, что теорема верна в логическом смысле, не означает, что она верна в физическом. С другой стороны, мы находим книги, написанные авторами, которые предлагают объяснения, лишенные какой-либо строгой академической теории. Они неумело используют математические средства для описания фактических наблюдений. Их модели переподогнаны и безуспешны при реализации. Научные исследования и публикации отделены от практического применения на финансовых рынках, и многие приложения в мире трейдинга/инвестиций не основываются на надлежащей науке.

Первый мотив для написания этой книги — преодолеть пресловутый разрыв, который разделяет академическую сферу и промышленность. Я был по обе стороны разлома и понимаю, как трудно его пересечь и как легко закрепиться на одной стороне. Добродетель — в равновесии. Эта книга не будет защищать теорию только из-за ее математической красоты и не будет предлагать решение только потому, что оно, похоже, работает. Моя цель — передать те знания, которые приходят только из опыта, формализованные в строгой манере.

Второй мотив обусловлен желанием того, чтобы финансы служили цели. На протяжении многих лет в некоторых из статей, опубликованных в научных журналах и газетах, я выражал свое недовольство текущей ролью финансов в нашем обществе. Инвесторов заманивают рисковать своими состояниями в азартных играх на дичайших уловках шарлатанов, поощряемых СМИ. Когда-нибудь в ближайшем будущем машинное обучение будет доминировать в финансах, наука ограничит гадание, а инвестиции не будут означать азартные игры. И я хотел бы, чтобы читатель сыграл свою роль в этой революции.

Третий мотив заключается в том, что многие инвесторы не в состоянии понять сложность применения машинного обучения к инвестициям. Похоже, это особенно касается дискреционных фирм, которые переходят в «квантоментальное»[1] пространство. Боюсь, что их большие ожидания не оправдаются, не потому, что машинное обучение оказалось безуспешным, а потому, что они использовали машинное обучение неправильно. В ближайшие годы многие фирмы будут инвестировать в готовые к применению алгоритмы МО, напрямую импортированные из академических учреждений или Кремниевой долины, и мой прогноз — они потеряют деньги (на более эффективные решения машинного обучения). Победить мудрость толпы сложнее, чем распознавать лица или водить машину. С помощью этой книги, я надеюсь, вы узнаете, как решать некоторые из проблем, которые делают финансы особенно сложной площадкой для машинного обучения, например переподгонку бэктеста[2]. Финансовое машинное обучение является предметом самим по себе, родственным, но отдельным от стандартного машинного обучения, и эта книга раскроет его для вас.

1.2. Основная причина безуспешности проектов финансового машинного обучения

Процент неудач в квантитативном финансировании высок, в особенности в финансовом машинном обучении. Те немногие, кому это удается, накапливают большое число активов и обеспечивают неизменно исключительную результативность для своих инвесторов. Тем не менее это редкий исход, по причинам, описанным в этой книге. За последние два десятилетия передо мной прошло много людей, фирмы открывались и закрывались. Исходя из личного опыта, в основе всех этих провалов лежит одна критическая ошибка.

---

[1] Неологизм «квантоментальный» образован из двух терминов, «квантитативный подход» и «фундаментальный подход», и означает сочетание новейших квантитативных методов, в том числе на основе машинного обучения, с классическими методами на основе фундаментальных величин. — *Примеч. пер.*

[2] Бэктестирование (*backtesting*), или бэктестинг, — это тестирование на исторических данных с целью получения результатов и анализа риска и возвратности инвестиций, прежде чем рисковать любым фактическим капиталом. — *Примеч. науч. ред.*

1.2.1. Сизифова парадигма

Дискреционные[1] портфельные менеджеры принимают инвестиционные решения, которые не соответствуют конкретной теории или обоснованию (если бы они были, то были бы систематическими портфельными менеджерами). Они потребляют сырые новости и аналитические отчеты, но в основном полагаются на свое суждение или интуицию. Они могут рационализировать эти решения, основываясь на какой-то истории, но для каждого решения всегда есть история. Поскольку никто полностью не понимает логику своих ставок, инвестиционные фирмы просят их работать независимо друг от друга, изолированно, чтобы обеспечить диверсификацию. Если вы когда-либо посещали собрания дискреционных портфельных менеджеров, то наверняка заметили, какими продолжительными и бесцельными они могут быть. Каждый участник кажется одержимым одной конкретной порцией эпизодической информации, и гигантские спорные заявления делаются без фактологических эмпирических доказательств. Это не означает, что дискреционные портфельные менеджеры не могут быть успешными. Напротив, некоторые из них успешны. Дело в том, что они не умеют работать в команде. Соберите вместе 50 дискреционных портфельных менеджеров, и они будут влиять друг на друга, пока в конечном итоге вы не заплатите 50 гонораров за работу одного. Следовательно, вполне разумно, что они работают разрозненно, сводя свое взаимодействие к минимуму.

Всякий раз, когда я сталкивался с применением этой формулы к квантитативным проектам или проектам на основе машинного обучения, это приводило к катастрофе. Менталитет заседаний совета в следующем: сделаем с квантами[2] то, что сработало с дискреционными ПМ-ами. Давайте наймем 50 высоко дипломированных специалистов и потребуем, чтобы каждый из них через полгода предоставил инвестиционную стратегию. Этот подход всегда имеет неприятные последствия, потому что каждый кандидат будет отчаянно искать инвестиционные возможности и в конечном итоге остановится на 1) ложном утверждении, которое выглядит великолепно в переподогнанном бэктесте, либо 2) стандартном факторном инвестировании, то есть переполненной инвесторами стратегии с низким коэффициентом Шарпа, но которая, по крайней мере, имеет академическую поддержку. Оба результата инвестиционный совет разочаруют, и проект будет отменен. Даже если пятеро из этих высоко дипломированных специалистов сделали настоящую находку, полученной прибыли не хватило бы на то, чтобы покрыть расходы на всех 50, поэтому эти пятеро переедут в другое место в поисках надлежащего вознаграждения.

---

[1] Дискреционный означает действующий по своему усмотрению или по своей инициативе. — *Примеч. науч. ред.*

[2] Квант (quant) — биржевой специалист по квантитативным (количественным) методам. — *Примеч. науч. ред.*

1.2.2. Метастратегическая парадигма

Если вас попросят своими силами разработать стратегии МО, знайте, что вы оказываетесь в заведомо проигрышном положении. Для создания одной настоящей инвестиционной стратегии требуется почти столько же усилий, сколько и для создания сотни, а сложности огромны: курирование и обработка данных, инфраструктура высокопроизводительных вычислений (HPC), разработка программного обеспечения, анализ признаков, симуляторы исполнения, бэктестирование и т. д. Даже если в этих областях фирма предоставляет вам общекорпоративные службы, вы будете выглядеть как работник завода BMW, которого попросили собрать всю машину целиком, используя все цеха в округе. Одну неделю вам придется быть сварщиком, следующую неделю — электриком, еще одну неделю — инженером-механиком, потом маляром... Вы попробуете, потерпите неудачу и по кругу вернетесь к сварке. Какой в этом смысл?

Каждая успешная квантитативная фирма, о которой я знаю, применяет метастратегическую парадигму (Lopez de Prado [2014]). Соответственно, эта книга была написана в качестве исследовательского пособия для команд, а не для отдельных лиц. Прочитав все главы, вы научитесь создавать научно-исследовательскую фабрику, а также различные участки сборочной линии. Роль каждого кванта заключается в том, чтобы специализироваться на конкретной задаче, стать лучшим, имея целостное представление обо всем процессе. В этой книге в общих чертах излагается план фабрики, где работа в команде порождает открытия с предсказуемой скоростью, не полагаясь на неожиданную удачу. Именно так Национальная лаборатория им. Лоуренса в Беркли и другие национальные лаборатории США регулярно делают научные открытия, такие как добавление 16 элементов в периодическую таблицу или создание основы для МРТ- и ПЭТ-сканирования[1]. Ни один конкретный человек не несет ответственности за эти открытия, так как они являются результатом усилий команды, где каждый вносит свой вклад. Конечно, создание подобных финансовых лабораторий требует времени и людей, которые знают, что делают, и делали это раньше. Но как вы думаете, что имеет более высокие шансы на успех: эта проверенная временем парадигма организованного сотрудничества или сизифова альтернатива, где каждый отдельный квант катит свой огромный валун в гору?

1.3. Структура книги

Эта книга распутывает паутину взаимосвязанных тем и представляет их в упорядоченном виде. Каждая глава предполагает, что вы прочитали предыдущие. Часть 1 поможет вам структурировать ваши финансовые данные таким образом, чтобы они подходили для обучающихся алгоритмов. В части 2 обсуждаются способы анализа этих данных с помощью обучающихся алгоритмов. Здесь акцент делается

[1] См. Berkeley Lab: http://www.lbl.gov/about.

на проведении исследований и фактическом открытии через научный процесс, в противоположность бесцельному поиску до тех пор, пока не появится какой-то интуитивно понятный (скорее всего, ложный) результат. В части 3 объясняется, как выполнять бэктестирование вашего открытия и оценивать вероятность того, что оно ложно.

Эти три части дают обзор всего процесса, от анализа данных до модельных исследований и оценивания открытий. С учетом этих знаний часть 4 возвращается к данным и объясняет инновационные способы извлечения информативных признаков. Наконец, подавляющая часть этой работы требует большой вычислительной мощности, поэтому часть 5 завершает книгу некоторыми полезными рецептами по высокопроизводительным вычислениям.

1.3.1. Структура в виде производственной цепочки

Добыча золота или серебра была относительно простой задачей в течение XVI и XVII веков. Менее чем за сто лет испанский золотой флот в четыре раза увеличил количество драгоценных металлов, находящихся в обращении по всей Европе. Те времена давно прошли, и сегодня старатели должны разворачивать сложные промышленные методы для извлечения микроскопических частиц слитков из тонн земли. Это не означает, что добыча золота находится на исторических низах. Напротив, в настоящее время шахтеры добывают 2500 тонн микроскопического золота каждый год, по сравнению со среднегодовыми 1,54 тонны, добытыми испанскими конкистадорами на протяжении всего XVI века! Видимое золото — это бесконечно малая часть совокупного объема золота на Земле. Эльдорадо всегда был рядом... если бы Франсиско Писарро[1] мог обменять меч на микроскоп.

Открытие инвестиционных стратегий претерпело аналогичную эволюцию. Если десять лет назад была относительно распространенной ситуация, когда отдельный специалист находил макроскопический альфа (то есть используя простые математические инструменты, такие как эконометрия), в настоящее время шансы на это быстро сходятся к нулю. Специалисты, которые в наши дни ищут макроскопический альфа, вне зависимости от своей квалификации или опыта сталкиваются с колоссальными трудностями. Единственный истинный альфа сегодня микроскопичен, а его отыскание требует капиталоемких промышленных методов. Как и в случае с золотом, микроскопический альфа не означает пониженную суммарную прибыль. Микроскопический альфа сегодня богаче, чем макроскопический альфа когда-либо был в истории. И на нем можно заработать немало денег, но для этого вам потребуется тяжелая артиллерия МО.

Давайте проведем обзор некоторых участков, задействованных в производственной цепочке, в рамках современного менеджера активами.

---

[1] Франсиско Писарро-и-Гонсалес — испанский конкистадор, завоеватель империи инков, основатель города Лима. — *Примеч. ред.*

1.3.1.1. Кураторы данных

Этот участок отвечает за сбор, очистку, индексирование, хранение, корректировку и доставку всех данных в производственную цепочку. Значения могут быть табличными или иерархическими, выровненными или несогласованными, историческими или в режиме реального времени и т. д. Члены команды являются экспертами в области микроструктуры рынка и протоколов передачи данных, таких как FIX[1]. В их обязанностях — разработка обработчиков данных, необходимых для понимания контекста, в котором возникают эти данные. Например, была ли котировка отменена и заменена на другом уровне или отменена без замены? Каждый класс активов имеет свои нюансы. Например, облигации регулярно обмениваются или отзываются; акции подвергаются дроблению, обратному дроблению, дают право голоса и т. д.; фьючерсы и опционы переносятся на другой срок; валюты не торгуются в централизованной книге ордеров. Степень специализации этого участка выходит за рамки данной книги, и в главе 1 будут рассмотрены лишь некоторые аспекты курирования данных.

1.3.1.2. Признаковые аналитики

Этот участок отвечает за трансформирование необработанных данных в информативные сигналы. Эти информативные сигналы обладают некоторой предсказательной силой над финансовыми величинами. Члены команды являются экспертами в области теории информации, извлечения и обработки сигналов, визуализации, маркировки, взвешивания, классификаторов и методов определения важности признаков. Например, признаковые аналитики могут обнаружить, что вероятность активной распродажи особенно высока, когда: 1) предложения по котировочным ценам отменяются-заменяются рыночными ордерами на продажу[2] и 2) ордера на покупку по котировочным ценам отменяются-заменяются лимитными ордерами[3] на покупку на более глубоком уровне торговой книги. Такая находка не является инвестиционной стратегией сама по себе и может использоваться альтернативными способами: исполнение, мониторинг риска неликвидности, обеспечение ликвидности (маркетмейкинг), позиционирование и т. д. Распространенной ошибкой считается то, что признаковые аналитики занимаются разработкой стратегий. Напротив, признаковые аналитики занимаются сбором и каталогизацией библиотек находок, которые могут быть полезны для многочисленных участков. Этому важнейшему участку посвящены главы 2–9 и 17–19.

---

[1] Протокол FIX (financial information exchange, FIX) — протокол обмена сообщениями, разработанный для обмена актуальной информацией по сделкам. Находится в свободном доступе и использовании (http://www.fixprotocol.org). — *Примеч. науч. ред.*

[2] Рыночный ордер (market order) — поручение клиента брокеру немедленно купить или продать биржевой актив по текущей лучшей цене. — *Примеч. науч. ред.*

[3] Лимитный ордер (limit order) — поручение клиента брокеру купить или продать биржевой актив по определенной максимальной или минимальной цене, то есть в заданном диапазоне цен. — *Примеч. науч. ред.*

1.3.1.3. Стратеги

На этом участке информативные признаки трансформируются в реальные инвестиционные алгоритмы. Стратег будет разбирать библиотеки признаков в поисках идей для разработки инвестиционной стратегии. Эти признаки были обнаружены разными аналитиками, изучающими широкий спектр финансовых инструментов и классов активов. Цель стратега — осмыслить все эти наблюдения и сформулировать общую теорию, их объясняющую. Поэтому стратегия — это всего лишь эксперимент, призванный проверить обоснованность этой теории. Членами команды являются аналитики данных с глубокими знаниями финансовых рынков и экономики. Напомним, что теория должна объяснять большую совокупность важных признаков. В частности, теория должна определить экономический механизм, который заставляет агента терять деньги для нас. Что это? Поведенческое смещение? Асимметричная информация? Нормативные ограничения? Признаки могут быть обнаружены «черным ящиком», но стратегия разрабатывается в «белом ящике». Склеивание нескольких каталогизированных признаков не является теорией. После того как стратегия будет завершена, стратеги подготовят исходный код, в котором задействуется полный алгоритм, и отправят этот прототип команде бэктестирования, описанной ниже. Главы 10 и 16 посвящены этому участку, при том понимании, что было бы неразумно в книге раскрывать конкретные инвестиционные стратегии.

1.3.1.4. Бэктестеры

Этот участок оценивает прибыльность инвестиционной стратегии в рамках разных сценариев. Одним из интересных сценариев является то, как стратегия будет работать, если история повторится. Тем не менее историческая траектория является лишь одним из возможных исходов стохастического процесса, и не обязательно наиболее вероятна в будущем. Необходимо провести оценку альтернативных сценариев в соответствии с имеющимися знаниями о слабых и сильных сторонах предлагаемой стратегии. Членами команды являются аналитики данных с глубоким пониманием эмпирических и экспериментальных методов. Грамотный бэктестировщик встраивает в свой анализ метаинформацию о том, как стратегия появилась. В частности, его анализ должен оценивать вероятность переподгонки бэктеста, принимая в расчет число испытаний, потребовавшихся для отшлифовки стратегии. Результаты этой оценки не будут повторно использоваться другими участками по причинам, которые станут очевидными в главе 11. Вместо этого результаты бэктестирования передаются руководству и не подлежат распространению среди других. В главах 11–16 обсуждаются виды анализа, проводимые этим участком.

1.3.1.5. Команда развертывания

Команда развертывания занимается интегрированием кода стратегии в производственную линию. Некоторые компоненты могут использоваться сразу в нескольких стратегиях, особенно если они обладают общими признаками. Членами

команды являются специалисты по алгоритмам и профессиональные математические программисты. Частью их работы является обеспечение того, чтобы развернутое решение было логически идентично полученному прототипу. Группа развертывания также несет ответственность за оптимизацию реализации в такой мере, чтобы свести к минимуму задержку в условиях производства. Поскольку производственные расчеты нередко чувствительны ко времени, данная команда во многом опирается на планировщиков процессов, серверы автоматизации (Jenkins), векторизацию, многопоточность, многопроцессорность, графические процессоры (GPU-NVIDIA), распределенные вычисления (Hadoop), высокопроизводительные вычисления (Slurm) и параллельные вычислительные технологии в целом. Главы 20–22 затрагивают различные аспекты, интересные для этого участка, поскольку они относятся к финансовому машинному обучению.

1.3.1.6. Портфельный мониторинг

Перед развертыванием каждая стратегия проходит *cursus honorum*[1], который подразумевает следующие этапы, или следующий цикл:

1. **Эмбарго**: изначально стратегия запускается на основе данных, полученных после конечной даты бэктеста. Данный этап вводится специалистами по бэктестингу или возникает в результате задержек внедрения. Если поведение стратегии в период эмбарго соответствует результатам бэктеста, стратегия переходит на следующий этап.

2. **Бумажный трейдинг**: в этой точке стратегия выполняется на живых реальных данных. Благодаря этому в результативности будут учитываться задержки, связанные с разбором данных, расчетами, исполнением, и другие промежутки времени между наблюдением и занятием позиции. Бумажный трейдинг будет происходить столько, сколько необходимо для сбора достаточных доказательств того, что стратегия работает так, как ожидалось.

3. **Выпуск**: в этой точке стратегия управляет реальной позицией, будь то в изоляции или в составе ансамбля. Результативность оценивается точно, включая приписываемые риски, финансовые возвраты и издержки.

4. **Переразмещение**: исходя из производственной результативности, размещение финансовых активов в выпускные стратегии часто и автоматически пересматривается в контексте диверсифицированного портфеля. В общем случае размещение в стратегии следует вогнутой функции. Начальное размещение (при выпуске) небольшое. По прошествии времени и по мере того, как стратегия показывает ожидаемую результативность, это размещение увеличивается. По прошествии времени результативность снижается, и размещение постепенно уменьшается.

[1] Cursus honorum (с *лат.* «путь чести») — последовательность военных и политических магистратур, через которые проходила карьера древнеримских политиков сенаторского ранга. — *Примеч. ред.*

5. **Комиссование**: в конечном итоге все стратегии выводятся из эксплуатации. Это происходит, когда они показывают результативность ниже ожидаемой на протяжении достаточно продолжительного периода времени, чтобы заключить, что поддерживающая теория больше не подкрепляется эмпирическими данными.

В общем случае, предпочтительнее выпускать новые вариации стратегии и выполнять их параллельно со старыми версиями. Каждая версия будет проходить вышеуказанный жизненный цикл, и старые стратегии будут получать меньше размещений финансовых активов в порядке диверсификации, принимая во внимание степень достоверности, полученную из их более продолжительной предыстории.

1.3.2. Структура по компонентам стратегии

Многие инвестиционные менеджеры считают, что секрет богатства заключается в реализации чрезвычайно сложного алгоритма МО. Вынужден разочаровать. Если бы это было так же просто, как программирование современного классификатора, то большинство людей в Кремниевой долине были бы миллиардерами. Успешная инвестиционная стратегия является результатом многих факторов. В табл. 1.1 приведены главы, которые помогут вам решить каждую из проблем, связанных с разработкой успешной инвестиционной стратегии.

На протяжении всей книги вы найдете много ссылок на журнальные статьи, которые я опубликовал в разные годы. Вместо того чтобы повторяться, я буду ссылаться на какую-либо из них, где вы найдете подробный анализ рассматриваемого вопроса. Все цитируемые мною публикации можно бесплатно скачать в препринтном формате с моего веб-сайта www.QuantResearch.org.

1.3.2.1. Данные

- Проблема: «мусор на входе — мусор на выходе»[1].
- Решение: работайте с уникальными данными, которые сложно подтасовать. Если вы единственный, кто использует эти данные, то выжимайте из них максимум.
- Где искать:
  - Глава 2: правильно структурировать свои данные.
  - Глава 3: произвести информативные метки.
  - Главы 4 и 5: тщательно смоделировать ряд со значениями, которые не являются одинаково распределенными и взаимно независимыми случайными величинами.
  - Главы 17–19: отыскать предсказательные признаки.

[1] GIGO (*англ.* Garbage In, Garbage Out, «мусор на входе — мусор на выходе») — принцип в информатике, означающий, что при неверных входящих данных будут получены неверные результаты, даже если сам по себе алгоритм правилен. — *Примеч. ред.*

1.3.2.2. Программное обеспечение

- Проблема: специализированная задача требует наличия специальных инструментов.
- Решение: разработать собственные классы. Использование популярных библиотек означает больше конкурентов, которые используют тот же источник.

Таблица 1.1. Общий обзор проблем, решаемых в каждой главе

Часть	Глава	Финансовые данные	Программное обеспечение	Аппаратное обеспечение	Математика	Метастратегия	Переобучение
1	2	X	X				
1	3	X	X				
1	4	X	X				
1	5	X	X		X		
2	6		X				
2	7		X			X	X
2	8		X			X	
2	9		X			X	
3	10		X			X	
3	11		X		X		X
3	12		X		X		X
3	13		X		X		X
3	14		X		X		X
3	15		X		X		X
3	16		X		X	X	X
4	17	X	X		X		
4	18	X	X		X		
4	19	X	X				
5	20		X	X	X		
5	21		X	X	X		
5	22		X	X	X		

- Где искать:
  - Главы 2–22: для всей книги и для каждой отдельно взятой главы мы разрабатывали собственные функции. Ваши проблемы можно решить только таким же образом, просто пользуйтесь примерами из книги.

1.3.2.3. Аппаратное обеспечение

- Проблема: машинное обучение связано с несколькими самыми сложными вычислительными задачами в математике.
- Решение: стать экспертом в области высокопроизводительных вычислений. По возможности сотрудничать с Национальной лабораторией с целью создания суперкомпьютера.
- Где искать:
  - Главы 20 и 22: научиться мыслить в терминах мультипроцессорных архитектур. Всякий раз, как вы программируете библиотеку, структуруйте ее таким образом, чтобы функции вызывались параллельно. В книге вы найдете множество примеров.
  - Глава 21: разрабатывайте алгоритмы для квантовых компьютеров.

1.3.2.4. Математика

- Проблема: процесс математического доказательства может занимать годы, десятилетия и столетия. Ни один инвестор не будет ждать так долго.
- Решение: использовать экспериментальную математику. Решать сложные, трудноразрешимые задачи не путем доказательства, а путем эксперимента. К примеру, авторы публикации Bailey, Borwein and Plouffe [1997] нашли спигот-алгоритм[1] для π (пи) без доказательства, вопреки прежнему представлению, что такая математическая находка была бы невозможной.
- Где искать:
  - Глава 5: познакомиться с экономичными для памяти трансформациями данных.
  - Главы 11–15: существуют экспериментальные методы, которые позволяют оценить ценность вашей стратегии с большей надежностью, чем симуляция на исторических данных.
  - Глава 16: алгоритм, оптимальный внутривыборочно, может показать слабую результативность вневыборочно[2]. Нет никаких математических доказа-

[1] Спигот-алгоритм (spigot algorithm), или алгоритм «втулки для крана». — *Примеч. пер.*

[2] Статистические проверки результативности модельного предсказания обычно проводятся путем разбиения заданной совокупности данных на внутривыборочный (in-sample)

тельств инвестиционного успеха. При проведении своего исследования опираться на экспериментальные методы.

- Главы 17 и 18: применять методы обнаружения структурных сдвигов; квантифицировать величину информации, содержащуюся в финансовом ряде.

1.3.2.5. Метастратегии

- Проблема: любители разрабатывают индивидуальные стратегии, веря, что существует волшебная формула обогащения. В отличие от них, профессионалы разрабатывают методы для массового производства стратегий. Деньги зарабатываются не сборкой одного автомобиля, а построением автомобильного завода.
- Решение: мыслить как предприятие. Ваша цель — управлять исследовательской лабораторией как фабрикой, где истинные открытия рождаются не из вдохновения. В этом заключалась философия Эрнеста Лоуренса (Ernest Lawrence), основателя первой Национальной лаборатории США.
- Где искать:
  - Главы 7–9: выстроить исследовательский процесс, который будет идентифицировать признаки, релевантные по всем классам активов, при этом справляясь с мультиколлинеарностью финансовых признаков.
  - Глава 10: комбинировать многочисленные предсказания в одну ставку.
  - Глава 16: фондировать стратегии, используя робастный метод, который показывает хорошую результативность вневыборочно.

1.3.2.6. Переподгонка, или переобучение

- Проблема: стандартные перекрестно-проверочные методы в финансовом деле не справляются. Большинство открытий в области финансов оказываются

период, используемый для первоначального оценивания параметров и отбора модели, и вневыборочный (out-of-sample) период, используемый для оценивания результативности предсказания. Эмпирические данные, основанные на результатах предсказания вне выборки, как правило, считаются более достоверными, чем данные, основанные на результатах в выборке. В частности, если имеются данные, скажем, за 3 года, необходимые для расчета волатильности, то модель, используемая в течение этого периода, будет «в выборке». Но если использовать исторические данные для предсказания вперед, то оценивание будет выполняться за период времени, для которого нет данных (вне выборки). Таким образом, обычно «вне выборки» — это понятие для «предсказания там, где у нас нет данных». Технически, даже использование модели для оценки сегодняшней волатильности на основе исторической выборки является прогнозом «вне выборки», потому что у нас нет мгновенной волатильности. — *Примеч. науч. ред.*

ложными вследствие множественного тестирования и систематического смещения при отборе[1].

- Решение:
  - Что бы вы ни делали, всегда задавайте себе вопрос — в чем вы можете переобучить модель? Относитесь скептически к своим результатам и постоянно ставьте перед собой новые задачи.
  - Переобучение — это недобросовестное занятие. Оно приводит к многообещающим исходам, которые не могут быть обеспечены. Когда переобучение допускается сознательно, оно становится откровенным научным мошенничеством. Тот факт, что многие ученые его допускают, не может служить его оправданием: они не рискуют ничьим состоянием, даже своим.
  - Это также пустая трата вашего времени, ресурсов и возможностей. Кроме того, отрасль платит только за вневыборочные финансовые возвраты[2]. Вы добьетесь успеха только *после* того, как создадите значительное состояние для своих инвесторов.
- Где искать:
  - Главы 11–15: существует три парадигмы бэктестинга, одной из которых является ретроспективная симуляция. Каждый бэктест подогнан под результат до определенного предела, и вам необходимо отчетливо понимать, до какого.
  - Глава 16: освоить робастные методы размещения финансовых активов, которые не допускают переподгонки внутривыборочных сигналов за счет вневыборочной результативности.

1.3.3. Структура по распространенной ошибке

Несмотря на ряд преимуществ, машинное обучение не является панацеей. Гибкость и сила методов машинного обучения имеют темную сторону. При неправильном

---

[1] Систематическое смещение при отборе (selection bias), или систематическая ошибка отбора, — это ошибка, выражающаяся в появлении у изучаемой выборки признаков, не свойственных генеральной совокупности; возникает в результате применения неподходящего метода отбора. — *Примеч. науч. ред.*

[2] Финансовый возврат (return) на вложенный капитал, также именуемый возвратом на инвестицию или финансовой отдачей, — это, проще говоря, деньги, сделанные или потерянные на инвестиции. Возврат выражается номинально как изменение денежной стоимости инвестиции с течением времени либо в процентах из соотношения прибыли к инвестициям. — *Примеч. науч. ред.*

использовании алгоритмы МО будут путать статистические случайности с закономерностями. Этот факт, в сочетании с низким соотношением сигнал/шум, характеризующим финансы, гарантирует, что небрежные пользователи будут производить ложные открытия с еще возрастающей скоростью. Эта книга демонстрирует несколько самых распространенных ошибок, допускаемых экспертами в области машинного обучения, когда они применяют свои методы к совокупностям финансовых данных. Часть этих ловушек приведена в табл. 1.2 с решениями, которые описаны в указанных главах.

1.4. Целевая аудитория

В этой книге представлены продвинутые методы машинного обучения, специально спроектированные для сложных задач, связанных с совокупностями финансовых данных. Под термином «сложные» (*advance*) я не имею в виду чрезвычайно трудные для понимания или объяснения последние реинкарнации глубоких, рекуррентных или сверточных нейронных сетей. Напротив, настоящая книга отвечает на вопросы, которые старшие исследователи, имеющие опыт применения алгоритмов МО к финансовым задачам, признают критическими. Если вы только приступили к работе с машинным обучением и у вас нет опыта работы со сложными алгоритмами, то эта книга будет (пока) не для вас. Не столкнувшись на практике с проблемами, рассматриваемыми в последующих главах, у вас вполне могут возникнуть сложности в понимании полезности их решения. Прежде чем читать эту книгу, вы, возможно, захотите изучить несколько хороших вводных книг по машинному обучению, опубликованных в последние годы. Я привел список нескольких из них в разделе справочных материалов.

Ключевая аудитория этой книги — инвестиционные профессионалы с уверенной подготовкой в области машинного обучения. Моя цель стоит в том, чтобы позволить вам монетизировать то, что вы узнаете в этой книге, помочь модернизировать финансы и обеспечить реальную стоимость для инвесторов.

Эта книга также рассчитана на аналитиков данных, которые успешно реализовали алгоритмы МО в различных областях за пределами финансов. Если вы работали в Google и применяли глубокие нейронные сети для распознавания лиц, но кажется, что все не так хорошо работает, когда вы запускаете алгоритмы на финансовых данных, то эта книга вам поможет. Иногда вы можете не понимать финансовую обусловленность некоторых структур (например, метамаркировка, тройной барьерный метод, дробное дифференцирование), но имейте в виду: после того как у вас получится управлять инвестиционным портфелем достаточно долго, правила игры станут для вас яснее вместе со смыслом этих глав.

Таблица 1.2. Распространенные ошибки применения МО в финансовом секторе

#	Категория	Ошибка	Решение	Глава
1	Эпистемологическая	Парадигма сизифова труда	Парадигма метастратегии	1
2	Эпистемологическая	Исследование через бэктестирование	Анализ важности признаков	8
3	Обработка данных	Хронологический отбор	Хронометраж объема	2
4	Обработка данных	Целочисленное дифференцирование	Дробное дифференцирование	5
5	Классификация	Маркировка с постоянно-временным горизонтом	Тройной барьерный метод	3
6	Классификация	Выяснение стороны и размера ставки	Метамаркировка	3
7	Классификация	Взвешивание выборок, не являющихся одинаково распределенными и взаимно независимыми	Взвешивание уникальности: последовательное бутстрапирование	4
8	Оценка	Утечки в перекрестной проверке	Прочистка и эмбарго	7, 9
9	Оценка	Прямое бэктестирование	Комбинаторная прочищенная перекрестная проверка	11, 12
10	Оценка	Бэктестовая переподгонка	Бэктестирование на синтетических данных; дефлятированный коэффициент Шарпа	10–16

1.5. Справочная информация

Инвестиционный менеджмент является одной из самых междисциплинарных областей исследований, и данная книга отражает этот факт. Понимание различных разделов требует практических знаний машинного обучения, рыночной микроструктуры, портфельного менеджмента, математических финансов, статистики, эконометрии, линейной алгебры, выпуклой оптимизации, дискретной математики, обработки сигналов, теории информации, объектно-ориентированного программирования, параллельной обработки и суперкомпьютерных вычислений.

Python *де-факто* стал стандартным языком для машинного обучения, и я должен исходить из того, что вы — опытный разработчик. Вы должны быть знакомы

со scikit-learn (sklearn), pandas, numpy, scipy, multiprocessing, matplotlib и несколькими другими библиотеками. Фрагменты программного кода вызывают функции из этих библиотек, используя их привычный префикс, pd для pandas, np для numpy, mpl для matplotlib и т. д. Каждой из этих библиотек посвящен целый ряд книг, и просто невозможно знать достаточно о специфике каждой из них. На протяжении всей этой книги мы будем обсуждать некоторые вопросы, касающиеся реализации, в том числе нерешенные дефекты, о которых нужно помнить.

1.6. Часто задаваемые вопросы

Как алгоритмы МО могут быть полезны в финансовом деле?

Многие финансовые операции требуют принятия решений на основе заранее определенных правил, таких как ценообразование опционов, алгоритмическое исполнение или мониторинг рисков. На сегодняшний день именно здесь имеет место основной объем автоматизации, трансформирующий финансовые рынки в ультрабыстрые, гиперсвязные сети для обмена информацией. При выполнении этих задач машины должны следовать этим правилам с максимальным быстродействием. Ярким примером является высокочастотный трейдинг. Данная тема подробнее рассматривается в публикации Easley, Lopez de Prado and O'Hara [2013].

Алгоритмизацию финансов уже не остановить. В период между 12 июня 1968 года и 31 декабря 1968 года Нью-Йоркская фондовая биржа (NYSE) закрывалась каждую среду, для того чтобы операционный отдел биржи мог привести бумажную работу в соответствие с торгами. Только представьте себе такое! Сегодня мы живем в другом мире, а через 10 лет все станет еще лучше. Потому что следующая волна автоматизации не предусматривает следование правилам, а действует на свое усмотрение. Как эмоциональные существа, находящиеся под влиянием страхов, надежд и личных интересов, люди не особенно хороши в принятии решений, основанных на фактах, в особенности в случаях, когда эти решения связаны с конфликтом интересов. В таких ситуациях инвесторы получают более качественные услуги, когда решения принимает машина, которая опирается на факты, выученные из данных. Это относится не только к разработке инвестиционной стратегии, но и практически ко всем областям финансового консультирования: предоставлению кредита, рейтингованию облигаций, классификации компании, подбору способных сотрудников, предсказанию доходов, прогнозированию инфляции и т. д. Более того, машины будут действовать в рамках законодательства и будут делать это всегда, при условии, что они соответствующим образом запрограммированы. В случае сомнительного решения инвесторы могут обратиться к журналам операций и выяснить, что именно произошло. Гораздо легче постоянно совершенствовать алгоритмический инвестиционный процесс, чем полностью полагаться на людей.

Почему алгоритмы МО инвестируют лучше людей?

Помните, как некогда люди были уверены, что компьютеры никогда не одержат верх над людьми в шахматах? «Своя игра»? Покер? Го? Миллионы лет эволюции (генетический алгоритм) тонко настраивали наш «приматный» мозг для выживания во враждебном трехмерном мире, где законы природы статичны. Теперь, когда дело доходит до распознавания тончайших закономерностей в высокоразмерном мире, где правила игры меняются каждый день, вся эта тонкая настройка оказывается пагубной. Один алгоритм МО может засекать закономерности в стомерном мире так же легко, как и в нашем знакомом трехмерном. И хотя мы все смеемся, когда видим, что алгоритм делает глупую ошибку, следует не забывать, что алгоритмы находятся вокруг нас несравненно меньше, чем наши миллионы лет. Каждый день они становятся лучше, а мы нет. Люди учатся медленно, что ставит нас в заведомо невыгодное положение в быстро меняющемся мире, таком как финансы.

Означает ли это, что технологии скоро заменят живых инвесторов?

Ни в коем случае. Ни один человек не умеет играть в шахматы лучше, чем компьютер. И ни один компьютер не умеет играть в шахматы лучше, чем человек при поддержке компьютера. Дискреционные портфельные менеджеры находятся в заведомо невыгодном положении при заключении ставки против обучающихся алгоритмов, однако существует возможность, что лучшие результаты будут достигнуты путем объединения дискреционных портфельных менеджеров с алгоритмами МО. Это то, что стало называться «квантоментальным» подходом. По всей книге вы найдете методы, которые могут применяться квантоментальными командами, то есть методы, которые позволят вам совмещать человеческие догадки (обусловленные фундаментальными величинами) с математическими прогнозами. В частности, в главе 3 вводится новый метод под названием «метамаркировка», который позволяет вам добавлять обучающийся слой поверх дискреционного.

Чем МО в финансовом секторе отличается от эконометрики?

Эконометрика — это приложение классических статистических методов к экономическим и финансовым рядам. Неотъемлемым инструментом эконометрики является многомерная линейная регрессия, технология XVIII века, которая осваивалась Гауссом еще до 1794 года (Stigler [1981]). Стандартные эконометрические модели не обучаются. И трудно поверить, что нечто такое же сложное, как финансы XXI века, может быть понято чем-то таким же простым, как инвертирование ковариационной матрицы.

Каждая эмпирическая наука должна строить теории, опираясь на наблюдения. Если статистическим инструментарием для моделирования этих наблюдений является линейная регрессия, то исследователю не удастся распознать сложность данных, и эти теории будут крайне упрощенческими и бесполезными. У меня нет сомнений, что эконометрика является главной причиной, почему

экономика и финансы не испытали значительного прогресса за последние 70 лет (Calkin and Lopez de Prado [2014a, 2014b]).

Столетиями средневековые астрономы проводили наблюдения и развивали теории о небесной механике. Эти теории никогда не рассматривали некруговые орбиты, потому что считались нечестивыми и противоречащими божьему замыслу. Предсказательные ошибки были настолько грубыми, что для их объяснения приходилось изобретать еще более сложные теории. И только когда Кеплеру хватило отваги предложить некруговые (эллиптические) орбиты, неожиданно гораздо более простая общая модель смогла предсказать положение планет с поразительной точностью. Что было бы, если бы астрономы никогда не рассматривали некруговые орбиты? Ну а что, если экономисты наконец начнут рассматривать нелинейные функции? Где наш Кеплер? У финансов нет ньютоновских «Начал», потому что отсутствие Кеплера подразумевает отсутствие Ньютона.

Методы финансового машинного обучения не заменяют теорию. Они задают ей направление. Каждый алгоритм МО заучивает закономерности в высокоразмерном пространстве без прямого ориентирования со стороны. После того как мы поймем, какие признаки являются предсказательными для феномена, мы сможем построить теоретическое объяснение, которое можно протестировать на независимой совокупности данных. Студентам факультетов экономики и финансов не мешало бы записываться на курсы машинного обучения вместо курса эконометрики. Эконометрика, возможно (пока еще), достаточно хороша для того, чтобы преуспеть в финансовых академиях, но успех в деловой сфере требует знания машинного обучения.

Что возразить людям, которые отвергают алгоритмы МО как «черные ящики»?

Если вы читаете эту книгу, то, скорее всего, для вас алгоритмы МО являются белыми ящиками. Они представляют собой прозрачные, четко сформулированные, кристально ясные функции распознавания закономерностей. Большинство людей не обладают подобным знанием, и для них машинное обучение выглядит как ящик фокусника: «Откуда взялся этот кролик? Ты морочишь нам голову, колдун!» Люди не верят тому, чего не понимают. Их предрассудки коренятся в невежестве, которое лечится простым сократовым средством: образованием. Кроме того, некоторые из нас обожают напрягать мозги, даже если нейробиологи до сих пор не выяснили точно, как они работают («черный ящик» по своей природе).

Время от времени вы будете встречать луддитов, которых уже не исправить. Нэд Лудд был ткачом из Лестера, Англия, который в 1779 году в приступе бешенства разнес в щепки два ткацких станка. С наступлением промышленной революции разъяренные механизацией толпы саботировали и разрушали всю технику, которую могли найти. Текстильщики испортили столько промышленного оборудования, что английскому парламенту пришлось выпустить два закона, подводящих «порчу оборудования» под преступление, карающееся смертной казнью. Между 1811 и 1816 годами многие районы Англии были охвачены открытым

восстанием, вплоть до того, что британских войск, сражавшихся с луддитами, было больше, чем войск в Пиренейских войнах против Наполеона. Восстание луддитов закончилось жестоким подавлением с помощью военной силы. Будем надеяться, что движение против «черных ящиков» до этого не дойдет.

Почему в книге не рассматриваются специфичные алгоритмы МО?

Эта книга нейтральна по отношению к любому отдельно взятому вами обучающемуся алгоритму. Выберете ли вы сверточные нейронные сети, адаптивное бустирование, случайные леса, метод опорных векторов и т. д., так или иначе вы столкнетесь с типовыми задачами: структурирование данных, маркировка, взвешивание, стационарные трансформации, перекрестная проверка, отбор признаков, установление важности признаков, переподгонка, бэктестирование и т. д. В контексте финансового моделирования ответы на эти вопросы нетривиальны, и требуется разработка платформенно-специфичных подходов. На этом и сосредоточена эта книга.

Какие другие книги вы рекомендуете по данному предмету?

Насколько мне известно, это первая книга, которая предоставляет полное и систематическое рассмотрение методов машинного обучения в конкретном применении к финансам: начиная с главы, посвященной структурам финансовых данных, затем идет глава по маркировке финансового ряда, затем глава по взвешиванию выборки, дифференцированию временного ряда... и заканчивая целой частью, посвященной правильному бэктестированию инвестиционных стратегий. Надо отметить, что имеется небольшое число предшествующих публикаций (в основном в журнальных статьях), в которых описывается применение стандартного машинного обучения к финансовым рядам, но это не то, что предлагается в данной книге. Моя цель была в том, чтобы обратиться к уникальным нюансам, которые делают финансовое машинное обучение особенно сложной задачей. Как и любой новый предмет, оно быстро эволюционирует, и данная книга будет обновляться по мере появления значительных достижений. Если вы хотели бы предложить отдельную тему для рассмотрения в будущих изданиях, пожалуйста, свяжитесь со мной по адресу mldp@quantresearch.org. Я с удовольствием добавлю эти главы с упоминанием имен читателей, которые их предложили.

Не понимаю, о чем идет речь в некоторых разделах или главах. Что делать?

Мой совет: начните с чтения справочных материалов, которые приводятся в конце книги. Когда я писал эту книгу, я должен был исходить из того, что читатель знаком с существующей литературой, иначе эта книга потеряла бы свою нацеленность. Если после прочтения этих справочных материалов разделы по-прежнему останутся малопонятными, то вероятная причина заключается в том, что они связаны с задачей, хорошо понятной инвестиционным профессионалам (даже если об этом нет упоминания в библиографии). Например, в главе 2 будут рассмотрены эффективные методы корректировки фьючерсных цен для переноса контракта на другой срок — задача, известная

большинству практиков, хотя она редко рассматривается в учебниках. Я бы посоветовал вам посетить один из моих регулярных семинаров и задать свой вопрос в конце моего выступления.

Почему эта книга так зациклена на бэктестовой переподгонке?

На это есть две причины. Прежде всего, бэктестовая переподгонка бесспорно является самой важной открытой проблемой в финансовой математике. Это наш аналог противопоставления классов сложности P и NP в информатике. Если бы существовал точный метод предотвращения бэктестовой переподгонки, то мы смогли бы его взять в банк. Бэктест будет почти так же хорош, как немедленное улаживание сделки фактически (кэшем) вместо ее подстегивания на словах. Хеджевые фонды будут с уверенностью выделять фонды портфельным управляющим. Инвесторы будут меньше рисковать и с охотой платить повышенные комиссионные. Регуляторы будут выдавать лицензии управляющим хеджевых фондов на основе достоверных данных о профессиональном опыте и знаниях, не оставляя места для шарлатанов. На мой взгляд, книга по инвестициям, которая не затрагивает этот вопрос, не стоит вашего внимания. Зачем вообще браться за чтение книги, которая посвящена модели ценообразования капитальных активов (capital assets pricing model, CAPM), модели арбитражного ценообразования (arbitrage pricing theory, APT), способам размещения активов, рисковому менеджменту и т. д., когда эмпирические результаты, подтверждающие эти аргументы, отобраны без определения вероятностей их ложного обнаружения?

Вторая причина заключается в том, что машинное обучение является отличным оружием в вашем исследовательском арсенале и опасным, если быть точным. Если бэктестовая переподгонка представляет собой проблему в эконометрическом анализе, то гибкость машинного обучения делает ее постоянной угрозой для вашей работы. Это особенно касается финансов, потому что наши совокупности данных короче, с более низким соотношением сигнал/шум, и у нас нет лабораторий, где мы можем проводить эксперименты с поправкой на все средовые величины (Lopez de Prado [2015]). Книга по машинному обучению, которая не решает эти проблемы, может оказаться скорее вредной, чем полезной для вашей карьеры.

Какие математические обозначения используются в книге?

Когда я начал писать эту книгу, я думал о присвоении одного символа каждой математической величине или функции во всех главах. Это было бы хорошо, если бы эта книга касалась одной темы, такой как стохастическое оптимальное управление. Однако эта книга посвящена широкому кругу математических предметов, каждый со своими собственными обозначениями. Читателям было бы труднее сверяться со справочными материалами, если бы я не следовал стандартам оформления технической литературы, а это означает, что иногда мы должны повторно использовать символы. Во избежание путаницы в каждой главе номенклатура разъясняется по мере использования. Большая часть математики сопровождается фрагментом исходного кода, поэтому в случае сомнений, пожалуйста, всегда ориентируйтесь на исходный код.

Кто написал главу 22?

Бытует расхожее мнение, что машинное обучение — это новая увлекательная технология, изобретенная или усовершенствованная в компаниях IBM, Google, Facebook, Amazon, Netflix, Tesla и т. д. Следует признать, что технологические фирмы стали крупными пользователями машинного обучения, в особенности в последние годы. Эти фирмы спонсировали некоторые из самых известных последних достижений машинного обучения (например, «Своя игра» или го), что, возможно, усилило это восприятие.

Тем не менее читатель может быть удивлен, узнав, что на самом деле национальные лаборатории США являются одними из исследовательских центров с самой продолжительной предысторией и опытом в использовании машинного обучения. Эти центры использовали машинное обучение до того, как оно стало сверхпопулярным, и они успешно применяли его в течение многих десятилетий, производя поразительные научные открытия. Если предсказание того, какие фильмы компания Netflix должна рекомендовать вам посмотреть в следующий раз, является достойным делом, то таким же является понимание скорости расширения Вселенной, или прогнозирование того, какие береговые линии будут наиболее затронуты глобальным потеплением, или предотвращение катастрофического сбоя нашей национальной энергосистемы. Это лишь некоторые из удивительных вопросов, над которыми такие учреждения, как Национальная лаборатория им. Лоуренса в Беркли, работают каждый день, тихо, но неустанно, с помощью машинного обучения.

В главе 22 д-р Хорст Саймон (Horst Simon) и Кешенг Ву (Kesheng Wu) выражают точку зрения заместителя директора и руководителя проекта в крупной национальной лаборатории США, специализирующейся на крупномасштабных научных исследованиях, связанных с большими данными, высокопроизводительными вычислениями и машинным обучением. В отличие от традиционных университетских условий, национальные лаборатории достигают научных прорывов, объединяя междисциплинарные команды, которые следуют хорошо разработанным процедурам, с сильным разделением труда и обязанностей. Такая исследовательская модель производственной цепочки родилась в Национальной лаборатории им. Лоуренса в Беркли почти 90 лет назад и обусловила возникновение метастратегической парадигмы, описанной в разделах 1.2.2 и 1.3.1.

1.7. Благодарности

Д-р Хорст Саймон, который является заместителем директора Национальной лаборатории им. Лоуренса в Беркли, согласился стать соавтором главы 22 вместе с д-ром Кешенгом Ву, который возглавляет несколько проектов в лаборатории Беркли и Национальном научно-исследовательском энергетическом вычислительном центре (National Energy Research Scientific Computing Center, NERSC).

Машинное обучение требует больших объемов вычислительных мощностей, и мои исследования были бы невозможны без их щедрой поддержки и руководства. В этой главе Хорст и Кешэн объясняют, как Национальная лаборатория им. Лоуренса в Беркли удовлетворяет суперкомпьютерные потребности исследователей во всем мире, а также определяющую роль машинного обучения и больших данных в современных научных прорывах.

Профессор Риккардо Ребонато первым прочитал эту рукопись и призвал меня опубликовать ее. Мои многочисленные беседы с профессором Франком Фабоцци на эти темы сыграли важную роль в формировании книги в ее нынешнем виде. Очень немногие люди в академических кругах имеют промышленный опыт Фрэнка и Риккардо, и очень немногие люди в отрасли имеют академическую родословную Риккардо и Фрэнка.

За последние два десятилетия я опубликовал около ста работ по теме данной книги, в том числе журнальные статьи, книги, главы, лекции, исходные коды и т. д. По моим последним подсчетам, эти работы были в соавторстве с более чем 30 ведущими специалистами в этой области, в том числе профессором Дэвидом Х. Бейли (15 статей), профессором Дэвидом Исли (8 статей), профессором Морином О'Харой (8 статей) и профессором Джонатаном М. Борвейном (6 статей). Эта книга в значительной степени также принадлежит им, поскольку она была бы невозможна без их поддержки, понимания и постоянного обмена идеями на протяжении многих лет. Уйдет слишком много времени, чтобы высказать им заслуженную похвалу, поэтому вместо этого я опубликовал следующую ссылку, где вы можете прочитать о наших коллективных усилиях: http://www.quantresearch.org/Co-authors.htm.

Последнее, но не менее важное: я хотел бы поблагодарить некоторых из моих членов исследовательской команды за вычитку книги и оказанную мне помощь в изготовлении некоторых рисунков: Диего Апарисио, д-р Ли Кон, Майкл Льюис, Майкл Лок, д-р Ясенг Жень и д-р Чжибайя Чжан.

Упражнения

1.1. Знакомы ли вам фирмы, которые пытались перейти от дискреционных инвестиций к инвестициям, направляемым алгоритмами МО, либо совместить их в том, что называется «квантоментальные фонды»?

(а) Насколько они в этом преуспели?

(б) Какие культурные трудности сопровождают этот переход?

1.2. Какая самая актуальная открытая проблема в финансовой математике? Если она уже решена, то каким образом?

(а) Регулирующие органы могли бы использовать решение для выдачи лицензий управляющим инвестициями?

(б) Инвесторы могли бы использовать решение для размещения средств?

(в) Компании могли бы использовать решение для поощрения исследователей?

1.3. Согласно журналу *Institutional Investor*, только 17 % активов хедж-фондов управляются методами количественного анализа. Это около 500 000 000 000 долларов, размещенных во всех фондах количественного инвестирования по состоянию на июнь 2017-го, по сравнению с 386 000 000 000 долларов годом ранее. Как вы думаете, что стоит за подобным глобальным перераспределением активов?

1.4. Согласно списку самых успешных компаний, опубликованному в журнале *Institutional Investor*, сколько компаний, занимающихся количественным инвестированием, попали в десятку самых доходных? Как эти данные можно сопоставить с долей активов фондов количественного инвестирования?

1.5. В чем заключается основное различие между эконометрическими методами и МО? Какую пользу экономическому и финансовому сектору может принести обновление статистического набора инструментов?

1.6. Наука имеет поверхностное представление о принципах работы человеческого мозга (мозга любого живого существа). В этом отношении мозг является неким «черным ящиком». Как вы думаете, почему многие критики называют МО «черным ящиком» и защищают дискреционное инвестирование?

1.7. Вы читаете журнальную статью, в которой описывается инвестиционная стратегия. При бэктестировании достигается коэффициент Шарпа в годовом исчислении, превышающий 2, с уровнем достоверности 95 %. Используя эту совокупность данных, вы можете воспроизвести результат в независимом бэктесте. Почему это открытие, по всей видимости, будет ложным?

1.8. Инвестиционные советники страдают от конфликта интересов во время принятия решений от лица своих инвесторов.

(а) Алгоритмы МО могут управлять инвестициями без конфликта интересов. Почему?

(б) Предположим, что алгоритм МО принимает решение, приводящее к убыткам. Алгоритм сделал то, на что он был запрограммирован, и инвестор согласился с условиями программы, что было подтверждено судебной экспертизой логов. В каком смысле эта ситуация лучше для инвестора, по сравнению с убытком, вызванным неправильным суждением дискреционного инвестиционного менеджера? Каково регрессивное требование инвестора в каждом случае?

(в) Целесообразно ли финансовым консультантам сопоставлять свои решения с решениями, принятыми такими нейтральными агентами?

Часть 1
АНАЛИЗ ДАННЫХ

2 Структуры финансовых данных

2.1. Актуальность

В этой главе мы научимся работать с неструктурированными финансовыми данными и из них формировать структурированный массив, пригодный для подачи в алгоритмы МО. В общем случае, вы не желаете потреблять чужую обработанную совокупность данных, так как скорее всего в итоге обнаружите как раз то, что кто-то уже давно знает или выяснит это в ближайшее время. В идеале отправной точкой является совокупность неструктурированных сырых данных, которые вы собираетесь обработать так, чтобы получить информативные признаки.

2.2. Основные типы финансовых данных

Финансовые данные поступают в разных формах и видах. В табл. 2.1 показаны четыре основных типа финансовых данных, упорядоченных слева направо с точки зрения растущего многообразия. Далее мы рассмотрим разную природу и области применения.

Таблица 2.1. Четыре основных типа финансовых данных

Фундаментальные	Рыночные	Аналитические	Альтернативные
Активы	Цена/отдача/предполагаемая волатильность	Рекомендации аналитиков	Спутниковые/CCTV-изображения
Обязательства	Объем	Кредитные рейтинги	Поиск в Google
Продажи	Дивиденды/купоны	Ожидания чистых доходов	Twitter/чаты
Расходы/чистые доходы	Открытые позиции	Новостной сентимент	Метаданные
Макровеличины	Котировки/отмены		
...	Сторона агрессора		

2.2.1. Базовые данные

Фундаментальные данные охватывают информацию, которую можно найти в нормативных документах и бизнес-аналитике. В основном это ежеквартальные бухгалтерские данные. Особый аспект этих данных заключается в том, что они предоставляются с опозданием. Вы должны точно подтвердить дату и время выпуска каждой точки данных, для того чтобы эта информация использовалась в вашем анализе только после того, как стала общедоступной. Распространенной ошибкой новичка является допущение, что эти данные были опубликованы в конце отчетного периода. Это вовсе не так.

Например, фундаментальные данные, публикуемые Bloomberg[1], индексируются по последней дате, включенной в отчет, которая предшествует дате выхода публикации (часто через 1,5 месяца). Другими словами, Bloomberg назначает эти значения дате, когда они не были известны. Вы не поверите, сколько статей публикуется каждый год с использованием несогласованных фундаментальных данных, в особенности в литературе по факторному инвестированию. После того как вы правильно выровняете данные, значительное число сведений в этих документах невозможно воспроизвести.

Второй аспект фундаментальных данных заключается в том, что в них часто данные заполняются задним числом или восстанавливаются. «Заполнение задним числом» означает, что отсутствующим данным назначается значение, даже если эти значения в то время были неизвестны. «Восстановленное значение» — это исправленное значение, которое корректирует неверную начальную публикацию данных. Компания может выпустить несколько исправлений в результатах за последний квартал спустя значительное время после выхода первой публикации, а поставщики данных могут переписывать исходные значения своими исправлениями. Проблема в том, что исправленные значения не были известны на дату первой публикации. Некоторые поставщики данных обходят эту проблему, сохраняя несколько дат публикации данных и значений для каждой величины. Например, для одной ежеквартальной публикации данных о ВВП мы обычно имеем три значения: исходное опубликованное значение и две ежемесячные ревизии. Тем не менее очень часто можно найти исследования, в которых используется окончательное опубликованное значение, где оно назначается по времени первого выхода публикации или даже последнего дня в отчетном периоде. Мы вернемся к этой неточности и ее последствиям, когда будем обсуждать ошибки бэктестирования в главе 11.

Фундаментальные данные чрезвычайно регуляризованы и низкочастотны. С учетом общедоступности на рынке маловероятно, что в них есть хоть какая-то ценность для эксплуатации. Тем не менее они могут быть полезны в сочетании с другими типами данных.

---

[1] Bloomberg L. P. — один из двух ведущих поставщиков финансовой информации для профессиональных участников финансовых рынков. — *Примеч. науч. ред.*

2.2.2. Данные рынка

Рыночные данные включают в себя всю торговую деятельность, которая происходит на бирже (например, на Чикагской товарной бирже (CME)) или торговой площадке (например, MarketAxess). В идеале ваш поставщик данных предоставил вам сырой поток с разнообразной неструктурированной информацией, например сообщениями по протоколу FIX, которые позволяют полностью восстановить торговую книгу или полную коллекцию откликов BWIC (bids wanted in competition, дословно «требуются ставки на конкурентной основе»). Каждый участник рынка оставляет характерный след в торговой истории, и с достаточным терпением вы найдете способ предвосхитить следующий шаг конкурента. Например, алгоритмы средневзвешенной по времени цены (time-weighted average price, TWAP) оставляют очень определенный след, который используется хищническими алгоритмами для опережения их деятельности в конце торгового дня (обычно связанного с хеджированием) (Easley, Lopez de Prado and O'Hara [2011]). Трейдеры у мониторов с графическим интерфейсом часто торгуют округленными лотами, и вы можете использовать этот факт для того, чтобы оценить, какой процент объема исходит от них в данный момент времени, а затем связать его с определенным поведением рынка.

Одним из привлекательных аспектов данных FIX является то, что, в отличие от фундментальных данных, их обработка не тривиальна. Они также обильны, при этом на суточной основе генерируется свыше 10 ТБ данных. Этот факт делает их более интересной совокупностью данных для исследований стратегий.

2.2.3. Аналитические данные

Вы можете представить аналитику как производные данные, основанные на первоисточнике, который может быть фундаментальным, рыночным, альтернативным или даже совокупностью данных другой аналитики. Аналитика характеризуется не содержимым информации, а тем, что она недоступна из первоисточника и что она была обработана для вас определенным образом. Инвестиционные банки и финансово-исследовательские фирмы продают ценные сведения, полученные в результате углубленного анализа бизнес-моделей компаний, их деятельности, конкурентноспособности, перспектив и т. д. Некоторые специализированные фирмы продают статистические данные, полученные из альтернативных данных, например, настроения, извлеченные из новостей и социальных медиа.

Положительным аспектом аналитики является то, что сигнал был извлечен из сырого источника. Отрицательные аспекты заключаются в том, что аналитика может быть дорогостоящей, используемая в ее производстве методология может быть тенденциозной или непрозрачной, и вы не будете единственным ее потребителем.

2.2.4. Альтернативные данные

В публикации Kolanovic and Krishnamachari [2017] проводится различие между альтернативными данными, производимыми физическими лицами (социальные

медиа, новости, результаты веб-поиска и т. д.), бизнес-процессами (сделки, корпоративные данные, правительственные органы и т. д.) и датчиками (спутники, геолокация, погода, CCTV-камеры видеонаблюдения и т. д.). Некоторые популярные спутниковые изображения или видеопотоки включают в себя мониторинг танкеров, интенсивность движения по туннелям или занятость парковочных мест.

Что действительно характеризует альтернативные данные, так это то, что это первичная информация, то есть информация, которая не попала в другие источники. До того как Exxon Mobile сообщила о повышении прибыли, до того как ее рыночная цена взлетела, до того как аналитики написали свои комментарии о своих последних заявках, до всего этого были движения танкеров и интенсивность перемещения бурильных установок и транспортировки по нефтепроводу. Они произошли за несколько месяцев до того, как эта деятельность была отражена в других типах данных. Двумя проблемными аспектами альтернативных данных являются их стоимость и конфиденциальность. Все это шпионское ремесло стоит дорого, и обследуемая компания может возражать, не говоря уже о посторонних свидетелях.

Альтернативные данные дают возможность работать с действительно уникальными, труднообрабатываемыми совокупностями данных. Запомните, что данные, которые трудно хранить, которыми трудно манипулировать и с которыми трудно работать, всегда являются наиболее перспективными. Вы убедитесь, что совокупность данных вполне может быть полезна, если она раздражает вашу команду инфраструктуры данных. Возможно, ваши конкуренты не пытались использовать их по логистическим причинам, сдались на полпути или неправильно их обрабатывали.

2.3. Бары

Для того чтобы применить алгоритмы МО к вашим неструктурированным данным, нам нужно их разобрать, извлечь из них ценную информацию и сохранить эти извлечения в упорядоченном формате. В большинстве алгоритмов предусматривается табличное представление извлеченных данных.

Финансовые специалисты-практики часто называют строки этих таблиц делениями, столбиками или *барами*. Мы можем различать две категории барных методов: 1) стандартные барные методы, которые являются общепринятыми в литературе, и 2) более продвинутые информационно-управляемые методы, используемые искушенными специалистами-практиками, хотя (пока) их невозможно найти в журнальных статьях. В этом разделе мы обсудим вопрос формирования этих баров.

2.3.1. Стандартные бары

Некоторые методы построения баров очень популярны в финансовой отрасли, и большинство API поставщиков данных предлагают некоторые из них. Цель

этих методов заключается в трансформировании поступающего с нерегулярной частотой ряда с данными наблюдений (часто именуемого «неоднородным рядом») в однородный ряд, получаемый на основе регулярного отбора.

2.3.1.1. Временные бары

Временны́е бары получаются путем отбора информации через фиксированные интервалы времени, например раз в минуту. Собираемая информация обычно включает:

- временно́й штамп (timestamp);
- средневзвешенную по объему цену (volume-weighted average price, VWAP);
- цену открытия (open, то есть первую цену);
- цену закрытия (close, то есть последнюю цену);
- максимальную цену (high);
- минимальную цену (low);
- торгуемый объем (volume) и т. д.

Хотя временны́е бары, возможно, наиболее популярны среди практиков и академических исследователей, их следует избегать по двум причинам. Во-первых, рынки не обрабатывают информацию в постоянном временно́м интервале. Час после открытия гораздо активнее, чем час около полудня (или час около полуночи в случае фьючерса). Как биологическим существам, людям имеет смысл организовывать свой день в соответствии с солнечным циклом. Но сегодняшние рынки управляются алгоритмами, которые торгуют с неплотным контролем со стороны человека, и для них процессорные циклы обработки гораздо релевантнее, чем хронологические интервалы (Easley, Lopez de Prado and O'Hara [2011]). Это означает, что временны́е бары избыточно отбирают информацию в периоды низкой активности, а недобирают информацию в периоды высокой активности. Во-вторых, отобранные по времени ряды часто демонстрируют плохие статистические свойства, такие как внутрирядовая корреляция[1], гетероскедастичность[2] и ненормальность финансовых возвратов (Easley, Lopez de Prado and O'Hara [2012]). Модели обобщенной авторегрессионной условной гетероскедастичности (generalized auto regressive conditional heteroscedasticity, GARCH) были разработаны, в частности, для решения проблемы гетероскедастичности, связанной с неправильным отбором.

---

[1] Внутрирядовая корреляция (serial correlation) — это взаимосвязь между наблюдениями одной и той же переменной в течение определенных периодов времени. Если величина внутрирядовой корреляции переменной равна нулю, то это означает, что корреляция отсутствует, и все наблюдения друг от друга не зависят. — *Примеч. науч. ред.*

[2] Гетероскедастичность (heteroskedasticity) — ситуация, когда некоторые диапазоны исхода показывают остатки с более высокой дисперсией (что может говорить о предикторе, который в уравнении отсутствует). — *Примеч. науч. ред.*

Как мы увидим далее, формирование баров в качестве подчиненного процесса торговой деятельности позволяет изначально избежать этой проблемы.

2.3.1.2. Тиковые бары

Идея тиковых баров проста: примеры перечисленных ранее величин (временной штамп, средневзвешенная по объему цена VWAP, цена открытия и т. д.) будут извлекаться всякий раз, когда происходит определенное число сделок, например 1000 тиков. Это позволяет нам синхронизировать отбор с косвенным индикатором прибытия информации (скоростью, с которой тики порождаются).

В публикации Mandelbrot and Taylor [1967] авторы одними из первых осознали, что отбор как функция от числа сделок обладает желательными статистическими свойствами: «Ценовые изменения на фиксированном числе сделок могут иметь гауссово распределение. Ценовое изменение за определенный промежуток времени может подчиняться стабильному распределению Парето, чья дисперсия бесконечна. Поскольку число сделок в любой период времени является случайным, вышеуказанные утверждения не обязательно расходятся».

Начиная с работы Мандельброта и Тейлора многочисленные исследования подтвердили, что отбор как функция торговой деятельности позволяет нам достигать финансовых возвратов, более близких к одинаково распределенным взаимно независимым случайным величинам (см. Ane and Geman [2000]). Это важно, потому что многие статистические методы опираются на допущение, что наблюдения извлекаются из одинаково распределенного взаимно независимого гауссова случайного процесса. В интуитивном плане мы можем сделать вывод только из инвариантной случайной величины, и тиковые бары допускают оптимальный статистический вывод, в отличие от временны́х баров.

При построении тиковых баров необходимо учитывать выбросы. Многие биржи проводят аукцион на открытии и аукцион на закрытии. Это означает, что в течение определенного периода времени книга ордеров накапливает заявки на покупку и продажу без их сопоставления. При завершении аукциона крупная сделка публикуется по клиринговой цене с нестандартным размером. Эта аукционная сделка может быть эквивалентна тысячам тиков, хотя сообщается как один тик.

2.3.1.3. Объемные бары

Одна из проблем с тиковыми барами заключается в том, что фрагментация ордеров вносит некоторую произвольность в число тиков. Например, предположим, что у нас есть один ордер, сидящий на цене предложения размером 10. Если мы приобретем 10 лотов, то наш один ордер будет записан как один тик. Если вместо этого в предложении 10 ордеров размером 1, то наша одна покупка будет записана как 10 отдельных сделок. Вдобавок в рамках операционного удобства сопряжение платформенных протоколов может еще дальше дробить одно исполнение ордера на многочисленные искусственные частичные исполнения.

Объемные бары (то есть на основе объема) обходят эту проблему путем отбора всякий раз, когда было обменяно предварительно определенное количество единиц (акций, фьючерсных контрактов и т. д.) ценной бумаги. Например, мы могли бы делать отбор цены всякий раз, когда фьючерсный контракт обменивается в размере 1000 единиц, независимо от числа тиков.

Сегодня трудно это представить, но еще в 60-х годах поставщики редко публиковали данные об объеме, так как клиенты были в основном заинтересованы в тиковых ценах. После того как стал сообщаться и объем, в публикации Clark [1973] было показано, что отбор финансовых возвратов по объему достигает еще более качественных статистических свойств (то есть ближе к одинаково распределенному взаимно независимому гауссову распределению), чем отбор по тиковым барам. Еще одна причина, почему следует отдавать предпочтение объемным барам над временными барами или тиковыми барами, состоит в том, что предметом нескольких теорий рыночной микроструктуры является взаимодействие между ценами и объемом. Отбор как функция одной из этих величин является удобным артефактом для такого анализа, как мы узнаем из главы 19.

2.3.1.4. Долларовые бары

Долларовые бары формируются путем отбора наблюдения всякий раз, когда обменивается заранее определенная рыночная стоимость. Разумеется, ссылаясь на доллары, мы имеем в виду применение к валюте, в которой деноминирована ценная бумага, но никто не ссылается на евро-бары, фунт-бары или иена-бары (хотя голд-бары сошли бы в качестве забавного каламбура[1]).

Давайте я проиллюстрирую обоснование необходимости долларовых баров на нескольких примерах. Во-первых, предположим, что мы хотим проанализировать акции, которые показали повышение стоимости на 100 % в течение определенного периода времени. Продажа этих акций на сумму 1000 долларов в конце этого периода потребует выставить на торги половину числа акций, на которые потребовалось потратить 1000 долларов на начало периода. Другими словами, число торгуемых акций зависит от фактической обменной стоимости. Поэтому имеет смысл отбирать бары с точки зрения обменной стоимости доллара, а не тиков или объема, в особенности когда анализ сопряжен со значительными ценовыми колебаниями. Этот момент можно проверить опытным путем. Если вы вычисляете тиковые бары и объемные бары на фьючерсном контракте E-mini S&P 500[2] для заданного размера бара, то число баров в день будет сильно варьироваться в течение многих лет. Этот диапазон и скорость вариации будут сокращены после того, как вы вычислите число

[1] Gold bar (*англ.*) — золотой слиток. — *Примеч. пер.*

[2] E-Mini S&P 500, часто сокращаемый до «E-mini» и обозначамый символом товарного тикера ES, — это фьючерсный контракт на фондовом рынке, который торгуется на электронной торговой платформе Chicago Globex Чикагской товарной биржи (Chicago Mercantile Exchange). — *Примеч. науч. ред.*

долларовых баров в день на протяжении многих лет, для постоянного размера бара. На рис. 2.1 показано экспоненциально взвешенное среднее число баров в день при применении фиксированного размера бара к тиковому, объемному и долларовому методам отбора.

Второй аргумент, который делает долларовые бары интереснее, чем временные, тиковые или объемные бары, состоит в том, что в результате корпоративных действий число выпущенных акций часто меняется несколько раз в течение жизни ценной бумаги. Даже после корректировки на дробления и обратные дробления есть другие действия, которые будут влиять на число тиков и объемов, например, выпуск новых акций или выкуп существующих (очень распространенная практика со времен Великой рецессии 2008 года). Перед лицом этих действий долларовые бары, как правило, устойчивы. Тем не менее вы можете сделать отбор долларовых баров, где размер бара не поддерживается постоянным во временно́й динамике. Вместо этого размер бара может динамически корректироваться в зависимости от свободно плавающей рыночной капитализации компании (в случае акций) или суммы выпущенного долга (в случае ценных бумаг с фиксированным доходом).

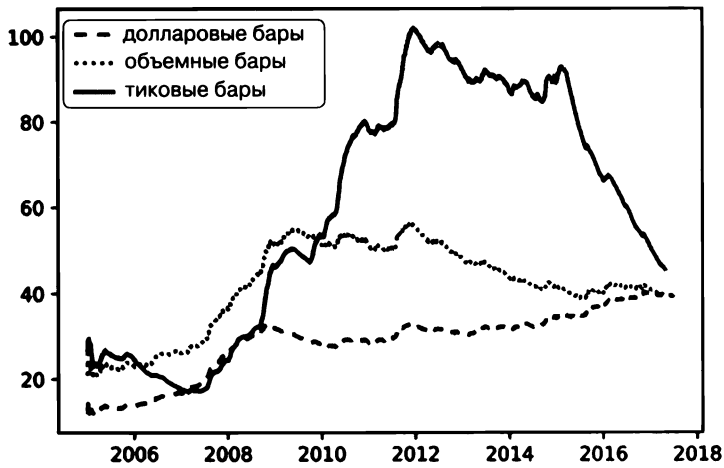


Рис. 2.1. Среднесуточная частота тиковых, объемных и долларовых баров

2.3.2. Информационные гистограммы

Цель информационно управляемых баров — более частый отбор при поступлении новой информации на рынок. В этом контексте слово «информация» употребляется в смысле рыночной микроструктуры. Как мы увидим в главе 19, теории рыночной микроструктуры придают особое значение устойчивости несбалансированных ориентированных (по знаку) объемов, поскольку этот феномен связан с присутствием информированных трейдеров. Синхронизируя отбор с прибытием информированных трейдеров, мы можем принимать решения до того, как цены достигнут нового

равновесного уровня. В этом разделе мы рассмотрим, как использовать различные индексы поступления информации для отбора баров.

2.3.2.1. Тиковые дисбалансные бары

Рассмотрим последовательность тиков $\{(p_t, v_t)\}_{t=1,\ldots,T}$, где p_t — это цена, связанная с тиком t, а v_t — это объем, связанный с тиком t. Так называемое правило тика определяет последовательность $\{b_t\}_{t=1,\ldots,T}$, где

$$b_t \begin{cases} b_{t-1}, \text{ если } \Delta p_t = 0 \\ \dfrac{|\Delta p_t|}{\Delta p_t}, \text{ если } \Delta p_t \neq 0 \end{cases}$$

при $b_t \in \{-1, 1\}$, а граничное условие b_0 устанавливается равным в соответствии с терминальным значением b_t из непосредственно предшествующего бара. Идея тиковых дисбалансных баров (tick imbalance bar, TIB) заключается в отборе баров всякий раз, когда тиковая несбалансированность превышает наши ожидания. Мы хотим определить тиковый индекс T так, чтобы накопление ориентированных по знаку тиков (ориентированных по тиковому правилу) превышало заданный порог. Далее обсудим процедуру определения T.

Для начала мы определяем диспропорцию тика по времени T как

$$\theta_T = \sum_{t=1}^{T} b_t .$$

Затем мы вычисляем математическое ожидание θ_t в начале бара, $E_0[\theta_T] = E_0[T]\,P[b_t = 1] - P[b_t = -1])$, где $E_0[T]$ — это ожидаемый размер тикового бара, $P[b_t = 1]$ — безусловная вероятность того, что тик классифицируется как покупка, а $P[b_t = -1]$ — безусловная вероятность того, что тик классифицируется как продажа. Поскольку $P[b_t = 1] + P[b_t = -1] = 1$, тогда $E_0[\theta_T] = E_0[T](2P[b_t = 1] - 1)$. На практике мы можем оценить $E_0[T]$ как экспоненциально взвешенное скользящее среднее значение T из предыдущих баров, а $(2P[b_t = 1] - 1)$ — как экспоненциально взвешенное скользящее среднее значение b_t из предыдущих баров.

Далее мы определяем тиковый дисбалансный бар (TIB) как T^* — сплошное подмножество тиков, такое, что выполняется следующее условие:

$$T^* = \arg_T \min\left\{|\theta_T| \geq E_0[T]\,|2P[b_t = 1] - 1|\right\},$$

где размер ожидаемой несбалансированности выводится по формуле $|2P[b_t = 1] - 1|$. Когда θ_T несбалансировано больше, чем ожидалось, низкое T удовлетворит этим условиям. Соответственно, тиковые дисбалансные бары возникают чаще в рамках присутствия информированной торговли (ассиметричной информации, которая запускает одностороннюю торговлю). По сути дела, мы можем рассматривать

тиковые дисбалансные бары как корзины сделок, которые содержат одинаковое количество информации (вне зависимости от торгуемых объемов, цен или тиков).

2.3.2.2. Объемные/долларовые дисбалансные бары

Идея объемных дисбалансных баров (volume imbalance bar, VIB) и долларовых дисбалансных баров (dollar imbalance bar, DIB) заключается в расширении идеи тиковых дисбалансных баров. Мы хотели бы отбирать бары, когда объемная или долларовая несбалансированности расходятся с нашими ожиданиями. Основываясь на тех же понятиях тикового правила и граничного условия b_0, которые мы обсуждали для тиковых дисбалансных баров, мы установим процедуру определения индекса следующей выборки, T.

Во-первых, мы определяем несбалансированность в момент времени T как

$$\theta_T = \sum_{t=1}^{T} b_t v_t,$$

где v_t может представлять либо число торгуемых ценных бумаг (VIB), либо сумму обменянных долларов (DIB). Ваш выбор v_t определяет, будете ли вы делать отбор согласно первому или последнему из перечисленных двух.

Во-вторых, мы вычисляем математическое ожидание θ_T в начале бара

$$\mathrm{E}_0[\theta_T] = \mathrm{E}_0\left[\sum_{t|b_t=1}^{T} v_t\right] - \mathrm{E}_0\left[\sum_{t|b_t=-1}^{T} v_t\right] = \mathrm{E}_0[T](\mathrm{P}[b_t=1]\mathrm{E}_0[v_t \mid b_t=1] - \\ - \mathrm{P}[b_t=-1]\mathrm{E}_0[v_t \mid b_t=-1]).$$

Представим, что $v^+ = \mathrm{P}[b_t = 1]\mathrm{E}_0[v_t|b_t = 1]$, $v^- = \mathrm{P}[b_t = -1]\mathrm{E}_0[v_t|b_t = -1]$, исходя из этого, $E_0[T]^{-1}\,\mathrm{E}_0[\Sigma_t v_t] = E_0[v_t] = v^+ + v^-$. Вы можете представить v^+ и v^- как разложение начального ожидания v_t на компоненту, вносимую покупками, и компоненту, вносимую продажами. Тогда

$$\mathrm{E}_0[\theta_T] = \mathrm{E}_0[T](v^+ - v^-) = \mathrm{E}_0[T](2v^+ - \mathrm{E}_0[v_t]).$$

На практике мы можем оценить $\mathrm{E}_0[T]$ как экспоненциально взвешенное скользящее среднее значений T из предыдущих баров и $(2v^+ - \mathrm{E}_0[v_t])$ как экспоненциально взвешенное скользящее среднее значений $b_t v_t$ из предыдущих баров.

В-третьих, мы определяем объемный дисбалансный бар или долларовый дисбалансный бар как T^* — сплошное подмножество тиков, такое, что выполняется следующее условие:

$$T^* = \arg_T \min\{|\theta_T| \geq \mathrm{E}_0[T]|2v^+ - \mathrm{E}_0[v_t]|\},$$

где размер ожидаемой несбалансированности выводится по формуле $|2v^+ - \mathrm{E}_0[v_t]|$. Когда θ_T несбалансированно больше, чем ожидалось, низкий T удовлетворит этим

условиям. Это информационный аналог объемного и долларового баров, и, как и его предшественники, он решает те же проблемы, связанные с выбросами и фрагментацией тиков. Более того, он также решает вопрос корпоративных действий, поскольку вышеупомянутая процедура не зависит от постоянного размера бара. Вместо этого размер бара регулируется динамически.

2.3.2.3. Тиковые интервальные бары

Тиковые дисбалансные бары, объемные дисбалансные бары и долларовые дисбалансные бары отслеживают несбалансированность потока ордеров, измеряемого в терминах тиков, объемов и стоимостей обменянных долларов. Крупные трейдеры будут сметать книгу ордеров, использовать айсберговые ордера[1] либо нарезать родительский ордер на несколько дочерних, каждый из которых оставляет след в виде интервалов в последовательности $\{b_t\}_{t=1,\ldots,T}$. По этой причине может быть полезно отслеживать *последовательность* покупок в совокупном объеме и делать выборки, когда эта последовательность отклоняется от наших ожиданий.

Для начала определим длину текущей операции как

$$\theta_T = \max\left\{\sum_{t|b_t=1}^{T} b_t, -\sum_{t|b_t=-1}^{T} b_t\right\}.$$

Затем мы рассчитаем математическое ожидание θ_T в начале бара

$$\mathrm{E}_0[\theta_T] = \mathrm{E}_0[T]\max\{\mathrm{P}[b_t = 1],\ 1 - \mathrm{P}[b_t = 1]\}.$$

На практике мы можем оценить $\mathrm{E}_0[T]$ как экспоненциально взвешенное скользящее среднее значений T из предыдущих баров, и $\mathrm{P}[b_t = 1]$ как экспоненциально взвешенное скользящее среднее доли покупных тиков из предыдущих баров.

В-третьих, мы определяем тиковый интервальный бар (tick runs bar, TRB) как T^* — сплошное подмножество тиков, такое, что выполняется следующее условие:

$$T^* = \arg_T \min\{\theta_T \geq \mathrm{E}_0[T]\max\{\mathrm{P}[b_t = 1],\ 1 - \mathrm{P}[b_t = 1]\}\},$$

где ожидаемое число тиков из интервалов выводится по формуле $\{\mathrm{P}[b_t = 1],\ 1 - \mathrm{P}[b_t = -1]\}$. Когда θ_T показывает интервалов больше, чем ожидалось, низкое T удовлетворит этим условиям. Обратите внимание, что в этом определении интервалов мы допускаем разрывы в последовательности. То есть вместо того, чтобы

[1] Айсберговый ордер (iceberg order) — это тип ордера, размещаемого на публичной бирже. Общая сумма ордера делится на видимую часть, которая сообщается другим участникам рынка, и скрытую часть, о которой ничего не сообщается. — *Примеч. науч. ред.*

измерять длину самой длинной последовательности, мы подсчитываем число тиков с каждой стороны, не сдвигая их (несбалансированность отсутствует). В контексте формирования баров это оказывается более полезным определением, чем измерение длин последовательностей.

2.3.2.4. Объемные и долларовые интервальные бары

Объемные интервальные бары (volume runs bar, VRB) и долларовые интервальные бары (dollar runs bar, DRB) расширяют приведенное выше определение интервалов соответственно на объемы и обменянные доллары. Интуитивная идея заключается в том, что мы хотим делать отбор баров всякий раз, когда объемы или доллары, торгуемые одной стороной, превышают наши ожидания для бара. Следуя нашей привычной номенклатуре для тикового правила, нам необходимо определить индекс T последнего наблюдения в баре.

Во-первых, мы определяем объемы или доллары, связанные с интервалом как

$$\theta_T = \max\left\{\sum_{t|b_t=1}^{T} b_t v_t, -\sum_{t|b_t=-1}^{T} b_t v_t\right\},$$

где v_t может представлять либо число торгуемых ценных бумаг (VRB), либо сумму обменянных долларов (DRB). Ваш выбор v_t определяет, будете ли вы делать отбор согласно первому или последнему из перечисленных двух.

Во-вторых, мы вычисляем математическое ожидание θ_T в начале бара

$$E_0[\theta_T] = E_0[T]\max\{P[b_t = 1]E_0[v_t \mid b_t = 1],\ (1 - P[b_t = 1])E_0[v_t \mid b_t = -1]\}.$$

На практике мы можем оценить $E_0[T]$ в качестве экспоненциально взвешенного скользящего среднего T значений из предыдущих баров, $P[b_t = 1]$ как экспоненциально взвешенное скользящее среднее доли покупных тиков из предыдущих баров, $E_0[v_t|b_t = 1]$ как экспоненциально взвешенное скользящее среднее покупных объемов из предыдущих баров и $E_0[v_t|b_t = -1]$ как экспоненциально взвешенное скользящее среднее продажных объемов из предыдущих баров.

В-третьих, мы определяем объемный интервальный бар как T^* — сплошное подмножество тиков, такое, что выполняется следующее условие:

$$T^* = \arg_T \min\{\theta_T \geq E_0[T]\max\{P[b_t = 1]E_0[v_t \mid b_t = 1],\ (1 - P[b_t = 1])E_0[v_t \mid b_t = -1]\}\},$$

где ожидаемый объем из интервалов выводится по формуле $\max\{P[b_t = 1]E_0[v_t|b_t = 1], (1 - P[b_t = 1])E_0[v_t|b_t = -1]\}$. Когда θ_T показывает интервалов больше, чем ожидалось, либо объем из отрезков больше, чем ожидалось, низкое T удовлетворит этим условиям.

2.4. Работа с мультипродуктовыми рядами

Иногда мы заинтересованы в моделировании временны́х рядов инструментов, где веса должны динамически корректироваться во временно́й динамике. В других случаях мы должны иметь дело с продуктами, которые выплачивают нерегулярные купоны или дивиденды либо которые подвержены корпоративным действиям. События, которые меняют характер исследуемых временны́х рядов, должны рассматриваться должным образом, иначе мы непреднамеренно внесем структурный сдвиг, который дезориентирует наши исследовательские усилия (подробнее об этом в главе 17). Эта проблема возникает в разных обличьях: когда мы моделируем спреды с изменяющимися весами, либо корзины ценных бумаг, в которые необходимо реинвестировать дивиденды/купоны, либо корзины, которые должны быть сбалансированы, либо когда компоненты индекса изменяются, либо когда мы должны заменить истекший/созревший контракт/ценную бумагу другим и т. д.

Примером могут служить фьючерсы. По моему опыту, люди тратят излишние усилия во время работы с фьючерсами главным образом потому, что не знают, как правильно обходиться с переносом фьючерсного контракта. То же самое можно сказать о стратегиях, основанных на спредах фьючерсов либо корзинах акций или облигаций. В следующем разделе я покажу вам один из способов смоделировать корзину ценных бумаг, как если бы это был один денежный продукт. Я называю это «трюком биржевого инвестиционного фонда», или просто трюком ETF[1], потому что цель состоит в том, чтобы трансформировать любую сложную совокупность данных из нескольких продуктов в единую совокупность данных, которая напоминает инструмент биржевого инвестиционного фонда с полным возвратом инвестиций. В чем здесь польза? Причина в том, что ваш программный код всегда может исходить из того, что вы торгуете только фактическими продуктами (денежными инструментами без срока действия), независимо от сложности и состава лежащего в основе ряда.

2.4.1. Трюк ETF

Предположим, мы хотим разработать стратегию, которая торгует спредом фьючерсного контракта. Несколько неприятностей возникают при работе именно со спредом, а не с прямым инструментом. Во-первых, этот спред характеризуется вектором весов, который меняется с течением времени. В результате, спред как таковой может сходиться, даже если цены не меняются. Когда это происходит, торгующая этим финансовым рядом модель будет дезориентирована и будет

[1] Биржевой инвестиционный фонд (exchange-traded fund, ETF) — это торгуемый на рынке финансовый актив, который отслеживает фондовый индекс, товар, облигации или корзину активов. Акции ETF торгуются как обычные акции на бирже. Цена акций ETF меняется в течение дня по мере их покупки и продажи. — *Примеч. науч. ред.*

ошибочно полагать, что PnL[1], то есть чистая пересчитанная по текущим рыночным ценам стоимость прибыли и убытков, является результатом этого вызванного весами схождения. Во-вторых, спреды могут приобретать отрицательные значения, поскольку они не представляют цену. Это часто может вызывать проблему, так как большинство моделей отталкиваются от положительных цен. В-третьих, времена торговли не будут точно выровнены для всех составляющих, поэтому спред не всегда торгуется на последних опубликованных уровнях либо с нулевым риском задержки. Кроме того, необходимо учитывать издержки на исполнение, такие как пересечение спреда между ценой спроса и ценой предложения[2].

Одним из способов избежать этих проблем является создание временного ряда, который отражает стоимость $1, инвестированную в спред. Изменения в данном ряде будут отражать изменения прибылях и убытках (PnL), ряд будет строго положительным (в худшем случае бесконечно малым), и этот недостаток реализации будет учтен. Этот ряд будет использован для моделирования, генерирования сигналов и торговли, как если бы это был инструмент ETF.

Предположим, что у нас есть история баров, полученная с помощью одного из методов, описанных в разделе 2.3. Эти бары содержат следующие столбцы с данными об инструменте:

- $o_{i,t}$ — это первичная цена открытия инструмента $i = 1, ..., I$ при баре $t = 1, ..., T$.
- $p_{i,t}$ — это первичная цена закрытия инструмента $i = 1, ..., I$ при баре $t = 1, ..., T$.
- $\varphi_{i,t}$ — это долларовое значение одной из меток инструмента $i = 1, ..., I$ при баре $t = 1, ..., T$. Здесь учитывается курс обмена валют.
- $v_{i,t}$ — это объем инструмента $i = 1, ..., I$ при гистограмме $t = 1, ..., T$.
- $d_{i,t}$ — это валютный актив, дивиденд или купон, выплачиваемый инструментом i при гистограмме t. Эта величина может также использоваться для начисления маржинальных расходов либо издержек фондирования,

где все инструменты $i = 1, ..., I$ торговались в баре $t = 1, ..., T$. Другими словами, даже если некоторые инструменты не торговались на протяжении всего временно́го интервала $[t - 1, t]$, по крайней мере, они торговались во времена $t - 1$ и t (рынки были открыты, и в эти моменты могли исполнять ордера). Для корзины фьючерсов, характеризующейся вектором размещений ω_t, ребалансированным (или

[1] Аббревиатура PnL в развернутом написании: net mark-to-market value of profits and losses, то есть чистая стоимость прибылей и убытков по текущим рыночным ценам. — *Примеч. науч. ред.*

[2] Спред между ценой спроса и ценой предложения (bid-ask spread) — это сумма, на которую цена предложения (ask) превышает цену спроса (bid) на ценную бумагу на рынке. Спред между ценами спроса и предложения представляет собой, по сути, разницу между самой высокой ценой, которую покупатель готов заплатить за актив, и самой низкой ценой, которую продавец готов принять, чтобы продать его. — *Примеч. науч. ред.*

пересчитанным) на барах $B \subseteq \{1, ..., T\}$, стоимость одного вложенного доллара K_t выводится как

$$h_{i,t} = \left\{ \begin{array}{c} \dfrac{\omega_{i,t} K_t}{o_{i,t+1} \varphi_{i,t} \sum_{i=1}^{I} |\omega_{i,t}|} \\ h_{i,t-1} \end{array} \right\}, \text{если } t \in B,$$

таким образом

$$\delta_{i,t} = \begin{cases} p_{i,t} - o_{i,t} \\ \Delta p_{i,t} \end{cases}, \text{если } (t-1) \in B,$$

таким образом

$$K_t = K_{t-1} + \sum_{i=1}^{I} h_{i,t-1} \varphi_{i,t} (\delta_{i,t} + d_{i,t}).$$

и $K_0 = 1$ в начальных активах в управлении (AUM)[1]. Переменная $h_{i,t}$ представляет содержимое (число ценных бумаг либо контрактов, находящихся во владении) инструмента i в момент времени t. Переменная $\delta_{i,t}$ — это изменение рыночной стоимости инструмента между $t - 1$ и t. Обратите внимание, что прибыли или убытки реинвестируются всякий раз, когда $t \in B$, тем самым предотвращая отрицательные цены. Дивиденды $d_{i,t}$ уже встроены в K_t, поэтому отсутствует необходимость в том, чтобы стратегия о них знала. $\omega_{i,t} (\sum_{i=1}^{I} |\omega_{i.t}|)^{-1}$ в $h_{i,t}$ предназначено для снижения кредитного плеча размещений. В случае фьючерсного ряда мы можем не знать $p_{i,t}$ нового контракта в момент t перенесения на новый срок, поэтому в качестве ближайшего по времени мы используем $o_{i,t+1}$.

Пусть τ_i равно транзакционной издержке, связанной с торговлей однодолларовым инструментом i, например $\tau_i = 1E{-}4$ (один базисный пункт). Для каждого наблюдаемого бара t стратегия должна знать три дополнительные переменные:

1. **Ребалансировочные стоимости** (rebalance costs): переменная стоимость $\{c_t\}$, ассоциированная с ребалансировкой размещения, равна

$$c_t = \sum_{i=1}^{I} (|h_{i,t} - 1| p_{i,t} + |h_{i,t}| o_{i,t+1}) \varphi_{i,t} \tau_i, \forall t \in B.$$

   Мы не встраиваем c_t в K_t, иначе шортирование спреда будет генерировать фиктивные прибыли при ребалансировке размещения. В своем исходном коде вы можете рассматривать $\{c_t\}$ как (отрицательный) дивиденд.

[1] Активы в управлении (assets under management, AUM) служат мерой общей рыночной стоимости всех ценных бумаг, которыми финансовое учреждение, такое как паевой инвестиционный фонд, венчурная компания или брокерский дом, управляет от имени своих клиентов и себя. — *Примеч. науч. ред.*

2. **Спред между ценой спроса и ценой предложения** (bid-ask spread): стоимость $\{\tilde{c}_t\}$ покупки или продажи одной единицы этого виртуального инструмента ETF равна $\tilde{c}_t = \sum_{i=1}^{I} \left| h_{i,t} - 1 \right| p_{i,t} \varphi_{i,t} \tau_i$. Когда единица покупается или продается, стратегия должна начислить эту стоимость $\tilde{c}_t$, которая эквивалентна пересечению спреда между ценой спроса и ценой предложения данного виртуального инструмента ETF.

3. **Объем** (volume): торгуемый объем $\{v_t\}$ определяется наименее активным членом корзины. Пусть $v_{i,t}$ — это объем, торгуемый инструментом i на баре t. Число торгуемых корзинных единиц $v_t = \min_i \left\{ \frac{v_{i,t}}{\left| h_{i,t-1} \right|} \right\}$.

Функции транзакционных издержек не обязательно линейны, и эти нелинейные издержки могут быть просимулированы стратегией на основе приведенной выше информации. Благодаря трюку ETF мы можем смоделировать корзину фьючерсов (или один фьючерс), как если бы это был один денежный продукт без срока действия.

2.4.2. Веса метода главных компонент (МГК)

В публикациях Lopez de Prado and Leinweber [2012] и Bailey and Lopez de Prado [2012] заинтересованный читатель найдет ряд практических способов вычисления хеджирующих весов. Для полноты картины рассмотрим один из способов получения вектора $\{\omega_t\}$, использованного в предыдущем разделе. Рассмотрим многомерный одинаково распределенный взаимно независимый гауссов процесс, характеризующийся вектором средних μ размером $N \times 1$ и ковариационной матрицей V размера $N \times N$. Этот стохастический процесс описывает инвариантную случайную величину, такую как финансовые возвраты на акции, изменения отдачи от облигаций или изменения волатильности опционов, для портфеля из N инструментов. Мы хотели бы вычислить вектор размещений ω, соответствующий определенному распределению рисков по всем главным компонентам V.

Во-первых, мы выполняем спектральное разложение $VW = W\Lambda$, где столбцы в W переупорядочены так, что элементы диагонали Λ были отсортированы по убыванию. Во-вторых, при заданном векторе размещений ω мы можем вычислить портфельный риск как $\sigma^2 = \omega' V \omega = \omega' W \Lambda W' \omega = \beta' \Lambda \beta = (\Lambda^{1/2}\beta)'(\Lambda^{1/2}\beta)$, где β представляет проекцию ω на ортогональный базис. В-третьих, Λ — это диагональная матрица, поэтому $\sigma^2 = \sum_{n=1}^{N} \beta_n^2 \Lambda_{n,n}$, и риск, относимый к n-й компоненте, равняется $R_n = \beta_n^2 \Lambda_{n,n} \sigma^{-2} = [W'\omega]_n^2 \Lambda_{n,n} \sigma^{-2}$, где $R' 1_N = 1$, а 1_N — это вектор из N единиц. Вы можете интерпретировать $\{R_n\}_{n=1,\ldots,N}$ как распределение рисков по ортогональным компонентам.

В-четвертых, мы хотели бы вычислить вектор ω, который обеспечивает определяемое пользователем распределение R риска. Мы отталкиваемся от формулы

$\beta = \left\{ \sigma \sqrt{\frac{R_n}{\Lambda_{n,n}}} \right\}_{n=1,\ldots,N}$, которая выражает распределение по новому (ортогональному) базису. В-пятых, размещение в старом базисе задается формулой $\omega = W\beta$. Масштабирование ω не затрагивает σ и сохраняет постоянство распределения риска. На рис. 2.2 показан вклад риска в расчете на главную компоненту для обратно-дисперсного размещения. Почти все главные компоненты вносят свой вклад в риск, включая компоненты с наибольшей дисперсией (компоненты 1 и 2). В отличие от этого, для портфеля PCA свой вклад в риск вносит только компонента с наименьшей дисперсией.

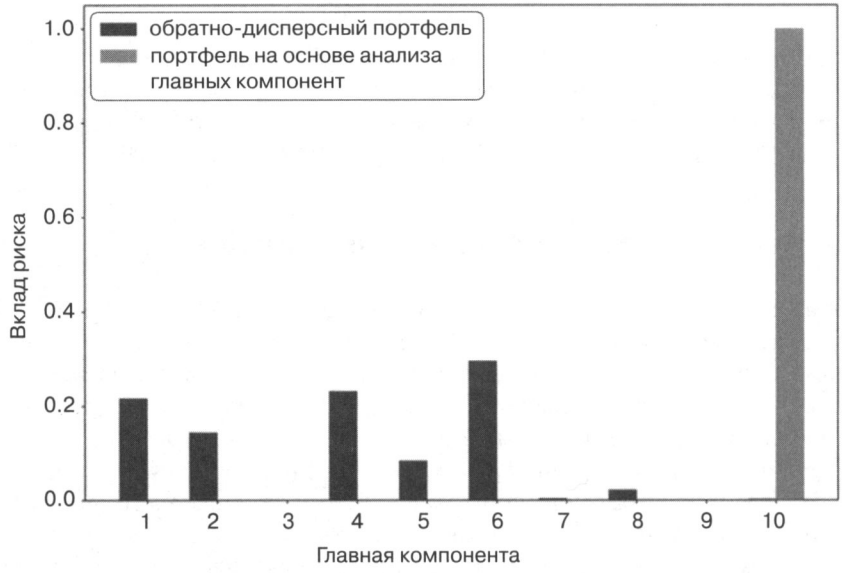


Рис. 2.2. Вклад в риск в расчете на главную компоненту

Листинг 2.1 реализует этот метод, где определяемое пользователем распределение R риска передается через аргумент `riskDist` (необязательно `None`). Если распределение `riskDist` равно `None`, то исходный код будет считать, что весь риск должен быть размещен в главной компоненте с наименьшим собственным значением, и веса будут последним собственным вектором, прошкалированным для соответствия цели по риску σ (`riskTarget`).

Листинг 2.1. Веса PCA из распределения риска R

```
def pcaWeights(cov,riskDist=None,riskTarget=1.):
    # Подчиняясь распределению riskAlloc, соответствовать значению riskTarget
    eVal,eVec=np.linalg.eigh(cov) # должна быть эрмитовой
```

```
indices=eVal.argsort()[::-1]  # аргументы для сортировки eVal по убыванию
eVal,eVec=eVal[indices],eVec[:,indices]
if riskDist is None:
    riskDist=np.zeros(cov.shape[0])
    riskDist[-1]=1.
loads=riskTarget*(riskDist/eVal)**.5
wghts=np.dot(eVec,np.reshape(loads,(-1,1)))
#ctr=(loads/riskTarget)**2*eVal # верифицировать riskDist
return wghts
```

2.4.3. Перенесение одного фьючерсного контракта

Трюк ETF может работать с перенесением на другой срок одного фьючерсного контракта, как частный случай одноногого (1-legged) спреда. Вместе с тем, при работе с одним фьючерсным контрактом эквивалентный и более прямой подход заключается в формировании временно́го ряда кумулятивных скачков в цене (гэпов), возникающих при перенесениях, и вычитании этого временно́го ряда из ценового ряда. Листинг 2.2 показывает возможную реализацию этой логики, используя серию тиковых баров, загруженных из Bloomberg и сохраненных в таблице HDF5. Смысл полей таблицы Bloomberg выглядит следующим образом:

- `FUT_CUR_GEN_TICKER`: идентифицирует контракт, связанный с этой ценой. Его значение меняется вместе с каждым перенесением.
- `PX_OPEN`: цена открытия, связанная с этим баром.
- `PX_LAST`: цена закрытия, связанная с этим баром.
- `VWAP`: средневзвешенная по объему цена, связанная с этим баром.

Аргумент `matchEnd` в функции `rollGaps` определяет, следует ли фьючерсный ряд переносить вперед (`matchEnd=False`) или назад (`matchEnd=True`)[1]. При переносе вперед цена в начале переносимого ряда совпадает с ценой в начале сырого ряда. При переносе назад цена в конце переносимого ряда совпадает с ценой в конце сырого ряда.

[1] Перенос вперед (roll forward) означает продление срока действия фьючерсного контракта путем закрытия первоначального контракта и открытия нового контракта с более долгим сроком на тот же базовый актив по текущей рыночной цене. Позволяет трейдеру поддерживать позицию после первоначального истечения контракта, так как фьючерсные контракты имеют конечные даты истечения. Обычно проводится незадолго до истечения первоначального контракта и требует урегулирования прибыли или убытка по первоначальному контракту. Перенос назад (roll backward) — это противоположная операция, которая означает укорочение срока действия фьючерсного контракта путем выхода из одной позиции и входа в новую позицию с более близким сроком действия. — *Примеч. науч. ред.*

Листинг 2.2. Формирование временного гэпового ряда и его вычитание из цен

```
def getRolledSeries(pathIn,key):
    series=pd.read_hdf(pathIn,key='bars/ES_10k')
    series['Time']=pd.to_datetime(series['Time'],format='%Y%m%d%H%M%S%f')
    series=series.set_index('Time')
    gaps=rollGaps(series)
    for fld in ['Close','VWAP']: series[fld]-=gaps
    return series
#---------------------------------------
def rollGaps(series,dictio={'Instrument':'FUT_CUR_GEN_TICKER','Open':
            'PX_OPEN', \
    'Close':'PX_LAST'},matchEnd=True):
    # Вычислить скачки/гэпы на каждом переносе,
    # между предыдущим закрытием и следующим открытием
    rollDates=series[dictio['Instrument']].drop_duplicates(keep='first').index
    gaps=series[dictio['Close']]*0
    iloc=list(series.index)
    iloc=[iloc.index(i)-1 for i in rollDates] # индекс дней перед переносом
    gaps.loc[rollDates[1:]]=series[dictio['Open']].loc[rollDates[1:]]- \
        series[dictio['Close']].iloc[iloc[1:]].values
    gaps=gaps.cumsum()
    if matchEnd: gaps-=gaps.iloc[-1] # перенести назад
    return gaps
```

Перенесенные цены используются для симулирования пересчитанных по текущим рыночным ценам стоимостей портфеля и прибыли и убытков. Однако сырые цены по-прежнему должны использоваться для выставления размеров позиций[1] и определения потребности в капитале. Имейте в виду, что перенесенные цены действительно могут стать отрицательными, в особенности во фьючерсных контрактах, которые были проданы с контанго[2]. Для того чтобы это увидеть, выполните код из листинга 2.2 на временном ряде фьючерсного контракта на хлопок #2 либо фьючерсного контракта на природный газ.

Мы хотим работать с неотрицательными перенесенными рядами, в этом случае мы можем получить ценовые ряды первой долларовой инвестиции следующим образом: 1) вычислить временной ряд перенесенных фьючерсных цен, 2) вычислить финансовый возврат (r), как изменение перенесенной цены, разделенной на предыдущую сырую цену, и 3) сформировать ценовой ряд с использованием этих возвратов (то есть `(1+p).cumprod()`). Листинг 2.3 иллюстрирует эту логику.

---

[1] Позиция — это сумма ценных бумаг, товара или валюты, которыми владеет физическое лицо, дилер, учреждение или другое налогооблагаемое лицо. Они бывают двух типов: короткие позиции, которые заимствуются, а затем продаются, и длинные позиции, которыми владеют и которые затем продаются. В зависимости от рыночных тенденций, движений и колебаний позиция может быть прибыльной или убыточной. Пересчет стоимости позиции для отражения ее фактической текущей стоимости на открытом рынке в отрасли называется «маркировкой по рынку». — *Примеч. науч. ред.*

[2] Контанго (contango, CGO) — надбавка в цене, взимаемая продавцом за отсрочку расчета по сделке. — *Примеч. науч. ред.*

Листинг 2.3. Неотрицательный ряд перенесенных цен

```
raw=pd.read_csv(filePath,index_col=0,parse_dates=True)
gaps=rollGaps(raw,dictio={'Instrument':'Symbol','Open':'Open',
     'Close':'Close'})
rolled=raw.copy(deep=True)
for fld in ['Open','Close']: rolled[fld]-=gaps
rolled['Returns']=rolled['Close'].diff()/raw['Close'].shift(1)
rolled['rPrices']=(1+rolled['Returns']).cumprod()
```

2.5. Отбор признаков

До сих пор мы учились создавать непрерывную, однородную и структурированную совокупность данных из коллекции неструктурированных финансовых данных. Хотя вы могли бы попытаться применить алгоритм МО к такой совокупности данных, в целом это не очень хорошая идея по нескольким причинам. Во-первых, несколько алгоритмов плохо масштабируются вместе с размером выборки, например опорно-векторные машины (SVM). Во-вторых, алгоритмы МО достигают наивысшей точности, когда они пытаются обучиться на релевантных примерах. Предположим, вы хотите предсказать, будет ли следующий 5 %-ный абсолютный возврат положительным (5 %-ное ралли) или отрицательным (5 %-ная активная распродажа). В любое случайное время точность такого предсказания будет низкой. Но если мы попросим классификатор предсказать знак следующего 5 %-ного абсолютного финансового возврата после определенных каталитических условий, то мы с большей вероятностью найдем информативные признаки, которые помогут нам достичь более точного предсказания. В этом разделе мы обсудим способы отбора баров для получения признаковой матрицы с релевантными тренировочными примерами.

2.5.1. Отбор с целью сокращения

Как мы уже упоминали ранее, одна из причин отбора признаков из структурированной совокупности данных заключается в сокращении объема данных, используемых для подгонки алгоритма. Эта операция также называется *понижающим отбором*, или отбором с пониженной частотой (downsampling). Она часто выполняется либо путем последовательного отбора с постоянным размером шага (линейно-пространственного отбора), либо путем случайного отбора с использованием равномерного распределения (равномерного отбора).

Основным преимуществом линейно-пространственного отбора является его простота. Недостатки заключаются в том, что размер шага является произвольным, и что исходы могут варьироваться в зависимости от посеянного бара. Равномерный отбор устраняет эти нежелательные явления, извлекая выборки равномерно по всему множеству баров в целом. Тем не менее оба метода подвергаются критике за то, что выборка не обязательно содержит подмножество наиболее релевантных наблюдений с точки зрения их предсказательной силы или информационного наполнения.

2.5.2. Событийно-управляемый отбор

Портфельные менеджеры обычно делают ставку после того, как происходит какое-либо событие, например, структурный сдвиг (глава 17), извлечен сигнал (глава 18) или возникают микроструктурные явления (глава 19). Эти события могут быть связаны с выходом некой макроэкономической статистики, всплеском волатильности, значительным отходом в спреде от его равновесного уровня и т. д. Мы можем охарактеризовать событие как значимое и дать алгоритму МО узнать, существует ли точная предсказательная функция в этих обстоятельствах. Возможно, ответ будет отрицательным, и в этом случае мы переопределим, что именно представляет собой событие, либо попробуем еще раз с альтернативными признаками. В целях иллюстрации рассмотрим один полезный метод событийного отбора.

2.5.2.1. Фильтр CUSUM

Фильтр на основе кумулятивных сумм (cumulative sum, CUSUM) — это статистический метод проверки качества, предназначенный для обнаружения сдвига в среднем значении измеренного числа в сторону от целевого значения. Рассмотрим одинаково распределенные взаимно независимые наблюдения $\{y_t\}_{t=1,\ldots,T}$, возникающие из локального стационарного процесса. Мы определяем кумулятивные суммы как

$$S_t = \max\{0, S_{t-1} + y_t - \mathrm{E}_{t-1}[y_t]\}$$

с граничным условием $S_0 = 0$. Данная процедура будет рекомендовать действие при первом t, удовлетворяющем $S_t \geq h$ для некого порога h (размер фильтра). Заметим, что $S_t = 0$ всякий раз, когда $y_t \leq \mathrm{E}_{t-1}[y_t] - S_{t-1}$. Этот нулевой нижний предел означает, что мы пропустим некоторые понижательные отклонения, которые в противном случае сделали бы S_t отрицательным. Причина в том, что данный фильтр настроен для определения последовательности повышательных дивергенций от любого нуля уровня сброса. В частности, порог срабатывает, когда

$$S_t \geq h \Leftrightarrow \exists \tau \in [1, t] \left| \sum_{i=\tau}^{t} (y_i - \mathrm{E}_{i-1}[y_t]) \geq h. \right.$$

Эта концепция накатов может быть расширена для включения откатов, давая в итоге симметричный фильтр CUSUM:

$$S_t^+ = \max\left\{0, S_{t-1}^+ + y_t - \mathrm{E}_{t-1}[y_t]\right\},\ S_0^+ = 0$$
$$S_t^- = \min\left\{0, S_{t-1}^- + y_t - \mathrm{E}_{t-1}[y_t]\right\},\ S_0^- = 0$$
$$S_t = \max\left\{S_t^+, -S_t^-\right\}.$$

В публикации Lam and Yam [1997] предлагается инвестиционная стратегия, в соответствии с которой сигналы на покупку и продажу генерируются поочередно,

когда наблюдается абсолютный финансовый возврат h относительно предыдущего максимума или минимума. Авторы этой работы демонстрируют, что такая стратегия эквивалентна так называемой «фильтрующей торговой стратегии», изученной в публикации Fama and Blume [1966]. Наше применение фильтра CUSUM отличается: мы будем отбирать бар t, если и только если $S_t \geq h$, и в этот момент S_t сбрасывается. Листинг 2.4 показывает реализацию симметричного фильтра CUSUM, где $E_{t-1}[y_t] = y_{t-1}$.

Листинг 2.4. Симметричный фильтр CUSUM

```
def getTEvents(gRaw,h):
    tEvents,sPos,sNeg=[],0,0
    diff=gRaw.diff()
    for i in diff.index[1:]:
        sPos,sNeg=max(0,sPos+diff.loc[i]),min(0,sNeg+diff.loc[i])
        if sNeg<-h:
            sNeg=0;tEvents.append(i)
        elif sPos>h:
            sPos=0;tEvents.append(i)
    return pd.DatetimeIndex(tEvents)
```

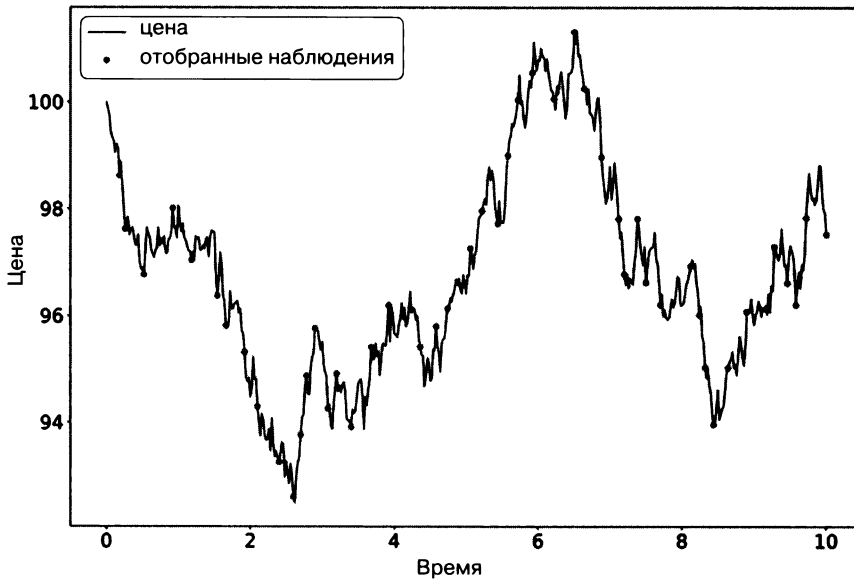


Рис. 2.3. Отбор из ценового ряда на основе фильтра CUSUM

Функция `getTEvents` получает два аргумента: сырой временной ряд `gRaw`, который мы хотим отфильтровать, и порог `h`. Один практический аспект, который делает фильтры CUSUM привлекательными, состоит в том, что сырой временной ряд `gRaw` не вызывает множественные события, если он колеблется вокруг порогового уровня, что является дефектом, от которого страдают популярные рыночные сигналы,

такие как полосы Боллинджера. Для запуска события сырому временному ряду `gRaw` потребуется полный отрезок длиной `h`. Рисунок 2.3 иллюстрирует образцы, взятые фильтром CUSUM на ценовом ряде.

Переменная S_t может быть основана на любом из признаков, которые мы обсудим в главах 17–19, таких как статистические показатели структурного сдвига, энтропия или меры рыночной микроструктуры. Например, мы могли бы объявлять событие всякий раз, когда метод SADF (дополненный супремумом метод проверки Дики—Фуллера) достаточно отходит от предыдущего уровня сброса (который будет определен в главе 17). Как только мы получим это подмножество событийно управляемых баров, то дадим алгоритму определить, является ли появление таких событий действенной оперативной информацией.

Упражнения

2.1. На ряде тиковых данных фьючерсного контракта E-mini S&P 500:

(а) Сформируйте тиковые, объемные и долларовые бары. Для работы с переносом используйте трюк ETF.

(б) Подсчитайте число баров, производимых тиковыми, объемными и долларовыми барами на еженедельной основе. Постройте график временного ряда этого числа баров. Какой тип баров обеспечивает наиболее стабильное недельное число? Почему?

(в) Вычислите внутрирядовую корреляцию финансовых возвратов для трех типов баров. Какой барный метод имеет самую низкую внутрирядовую корреляцию?

(г) Разбейте барные ряды на ежемесячные подмножества. Вычислите дисперсию финансовых возвратов для каждого подмножества каждого типа баров. Вычислите дисперсию этих дисперсий. Какой метод демонстрирует наименьшую дисперсию дисперсий?

(д) Примените проверку нормальности Харке—Бера на финансовых возвратах из трех типов баров. Каким методом достигается наименьший проверочный статистический показатель?

2.2. На ряде тиковых данных фьючерсного контракта E-mini S&P 500 вычислите долларовые бары и долларовые дисбалансные бары. Какой тип баров демонстрирует бóльшую внутрядовую корреляцию? Почему?

2.3. На рядах тиковых данных фьючерсного контракта E-mini S&P 500 и Eurostoxx 50[1]:

[1] EURO STOXX 50 — фондовый индекс ценных бумаг в еврозоне, разработанный поставщиком индекса STOXX, принадлежащим Deutsche Börse Group. — *Примеч. науч. ред.*

(а) Примените раздел 2.4.2 для вычисления вектора $\{\hat{\omega}_t\}$, используемого в трюке ETF. (Подсказка: вам понадобятся значения валютного контракта (FX) для пары EUR/USD в даты переноса.)

(б) Выведите временной ряд спреда S&P 500/Eurostoxx 50.

(в) С помощью проверки ADF подтвердите, что этот ряд является стационарным.

2.4. Сформируйте долларовые бары фьючерсного контракта E-mini S&P 500:

(а) Вычислите полосы Боллинджера шириной 5 % вокруг скользящей средней. Подсчитайте число раз, когда цены пересекают полосы (изнутри полос наружу).

(б) Теперь с помощью фильтра CUSUM выполните отбор из этих баров, где $\{y_t\}$ — это финансовые возвраты и $h = 0{,}05$. Сколько образцов у вас получилось?

(в) Вычислите скользящее среднеквадратическое отклонение двухвыборочного ряда. Какая из выборок наименее гетероскедастична? В чем причина таких результатов?

2.5. Используя бары из упражнения 2.4:

(а) Выполните отбор из баров с помощью фильтра CUSUM, где $\{y_t\}$ — это абсолютные финансовые возвраты, а $h = 0{,}05$.

(б) Вычислите скользящее среднеквадратическое отклонение отобранных баров.

(в) Сравните этот результат с результатами упражнения 2.4. Какая процедура позволила получить наименее гетероскедастичную выборку? Почему?

3 Маркировка

3.1. Актуальность

В главе 2 мы обсудили вопрос создания матрицы X финансовых признаков из неструктурированной совокупности данных. Из этой матрицы X неконтролируемые обучающиеся алгоритмы (то есть без учителя) могут заучивать закономерности, например, содержит ли она иерархические кластеры. С другой стороны, контролируемые обучающиеся алгоритмы (то есть с учителем) требуют, чтобы строки в X были связаны с массивом маркеров, меток или значений y, благодаря чему эти маркеры, метки или значения могут быть предсказаны на ранее не встречавшихся образцах признаков. В этой главе мы обсудим способы маркировки финансовых данных.

3.2. Метод фиксированного временного горизонта

Что касается финансов, то практически во всех публикациях по МО наблюдения маркируются с использованием метода фиксированного временно́го горизонта. Этот метод можно описать следующим образом. Рассмотрим признаковую матрицу X с I строками, $\{X_i\}_{i=1,\ldots,I}$, извлеченными из нескольких баров с индексом $t = 1, \ldots T$, где $I \leq T$. В главе 2, раздел 2.5, обсуждались методы отбора, которые производят множество признаков $\{X_i\}_{i=1,\ldots,I}$. Наблюдению X_i назначается метка $y_i \in \{-1, 0, 1\}$,

$$y_t = \begin{cases} -1 & \text{если } r_{t_{i,0}t_{i,0}+h} < -\tau \\ 0 & \text{если } |r_{t_{i,0}t_{i,0}+h}| \leq \tau \\ 1 & \text{если } r_{t_{i,0}t_{i,0}+h} > \tau, \end{cases}$$

где τ — это предварительно определенный постоянный порог; $t_{i,0}$ — индекс бара сразу после того, как происходит X_i; $t_{i,0} + h$ — это индекс h-го бара после $t_{i,0}$ и $r_{t_{i,0},t_{i,0}+h}$ — ценовой финансовый возврат на барном горизонте h:

$$r_{t_{i,0}t_{i,0}+h} = \frac{P_{t_{i,0}+h}}{P_{t_{i,0}}} - 1.$$

Поскольку литература почти всегда работает с временными барами, h подразумевает фиксированный временной горизонт. В приложении В перечислены несколько исследований по машинному обучению, из которых публикация Dixon и соавт. [2016] является недавним примером этого метода маркировки. Несмотря на его популярность, есть несколько причин избегать такого подхода в большинстве случаев. Во-первых, как мы видели в главе 2, временные бары не демонстрируют хорошие статистические свойства. Во-вторых, один и тот же порог T применяется независимо от наблюдаемой волатильности. Предположим, что $\tau = 1E{-}2$, где иногда мы маркируем наблюдение как $y_i = 1$ при условии реализованной волатильности бара $\sigma_{t_{i,0}} = 1E{-}4$ (например, во время ночной сессии), а иногда $\sigma_{t_{i,0}} = 1E{-}2$ (например, возле открытия). Подавляющее большинство меток будет равно 0, даже если финансовый возврат $r_{t_{i,0},t_{i,0}+h}$ был предсказуем и статистически значим.

Другими словами, очень распространенной ошибкой является маркировка наблюдений в соответствии с фиксированным порогом на временны́х барах. Вот несколько более качественных альтернатив. Во-первых, маркировать на переменном пороге $\sigma_{t_{i,0}}$, оцениваемом с использованием скользящего экспоненциально взвешенного среднеквадратического отклонения финансовых возвратов. Во-вторых, использовать объемные или долларовые бары, так как их волатильность гораздо ближе к постоянной (гомоскедастичности). Но даже эти два улучшения упускают ключевой недостаток метода фиксированного временно́го горизонта: траекторию, по которой следуют цены. Каждая инвестиционная стратегия имеет лимиты остановки убытков (стоп-лоссов), независимо от того, были ли они самостоятельно назначены портфельным менеджером, введены отделом рисков или запущены маржин коллом[1]. Просто нереально построить стратегию, которая бы получала прибыль от позиций, которые останавливались бы биржей принудительно. То, что практически ни одна публикация это не объясняет во время маркировки наблюдений, говорит вам кое-что о текущем состоянии инвестиционной литературы.

[1] Маржин колл (margin call) — это сигнал уведомления инвестора о приближении к критическому уровню, который не обязывает вносить средства для поддержания кредита и покрытия дальнейших возможных убытков. В отличие от него, стоп-аут (stop out) — это более низкий, чем маржин колл, уровень, помогающий избежать потери финансовых средств, предоставленных через кредитное плечо. В этом случае, если количество убытков продолжает увеличиваться, то сделка, или даже несколько открытых позиций, закрывается принудительно, причем основные убытки в этом случае несет только инвестор. — *Примеч. науч. ред.*

3.3. Вычисление динамических порогов

Как отмечалось в предыдущем разделе, на практике мы хотим задать лимиты взятия прибыли (тейк-профит) и остановки убытка (стоп-лосс), которые являются функцией от участвующих в ставке рисков. В противном случае иногда мы будем ставить слишком завышенную цель ($\tau \gg \sigma_{t_{i,0}}$), а иногда слишком заниженную цель ($\tau \ll \sigma_{t_{i,0}}$), учитывая преобладающую волатильность.

Листинг 3.1 вычисляет суточную волатильность во внутридневных оценочных точках, применяя промежуток `span0` дней к экспоненциально взвешенному скользящему стандартному отклонению. Для получения подробной информацией по функции `pandas.Series.ewm` обратитесь к документации по библиотеке pandas.

Листинг 3.1. Оценки суточной волатильности

```
def getDailyVol(close,span0=100):
    # суточный объем, переиндексированный по close
    df0=close.index.searchsorted(close.index-pd.Timedelta(days=1))
    df0=df0[df0>0]
    df0=pd.Series(close.index[df0-1], index=close.index[close.shape[0]-
        df0.shape[0]:])
    df0=close.loc[df0.index]/close.loc[df0.values].values-1 # суточные
                                                            # возвраты
    df0=df0.ewm(span=span0).std()
    return df0
```

Мы можем использовать результат этой функции для задания стандартных лимитов взятия прибыли и остановки убытка до конца этой главы.

3.4. Тройной барьерный метод

Здесь я представлю альтернативный метод маркировки, которого я не нашел в литературе. Если вы являетесь профессиональным инвестором, то, думаю, согласитесь, что он имеет больше смысла. Я называю это тройным барьерным методом, потому что он маркирует наблюдение в соответствии с тем, какой барьер из трех был затронут первым. Во-первых, мы устанавливаем два горизонтальных барьера и один вертикальный барьер. Два горизонтальных барьера определяются границами взятия прибыли и остановки убытка, которые являются динамической функцией оценочной волатильности (реализованной или предполагаемой). Третий барьер определяется числом баров, прошедших с момента открытия позиции (лимит экспирации). Если верхний барьер задет первым, то мы маркируем наблюдение как 1. Если нижний барьер задет первым, то мы маркируем наблюдение как -1. Если вертикальный барьер задет первым, то у нас есть два варианта: знак финансового возврата либо 0. Я лично предпочитаю первое как реализацию прибыли

или убытка в пределах лимитов, но вы должны разведать, работает ли 0 лучше в ваших конкретных задачах.

Возможно, вы заметили, что тройной барьерный метод зависит от траектории. Для того чтобы промаркировать наблюдение, мы должны принять в расчет всю траекторию, охватывающую $[t_{i,0}, t_{i,0} + h]$, где h определяет вертикальный барьер (лимит окончания срока действия, или экспирации). Обозначим через $t_{i,1}$ время первого касания барьера и финансовый возврат, связанный с наблюдаемым признаком, — через $r_{t_{i,0},t_{i,1}}$. Для ясности, $t_{i,1} \leq t_{i,0} + h$ и горизонтальные барьеры не обязательно симметричны.

Листинг 3.2 реализует тройной барьерный метод. Данная функция получает четыре аргумента:

- `close`: ценовой ряд библиотеки pandas;
- `events`: кадр данных библиотеки pandas со столбцами:
  - `t1`: временной штамп вертикального барьера. Когда его значение равно `np.nan`, вертикальный барьер будет отсутствовать;
  - `trgt`: единичная ширина горизонтальных барьеров.
- `ptSL`: список двух неотрицательных вещественных значений:
  - `ptSL [0]`: сомножитель, который умножает `trgt` для установки ширины верхнего барьера. Если равен 0, то верхний барьер отсутствует;
  - `ptSL[1]`: сомножитель, который умножает `trgt` для установки ширины нижнего барьера. Если равен 0, то нижний барьер отсутствует;
- `molecule`: список с подмножеством индексов событий, который будет обрабатываться одиночным потоком. Его использование станет понятным позже в этой главе.

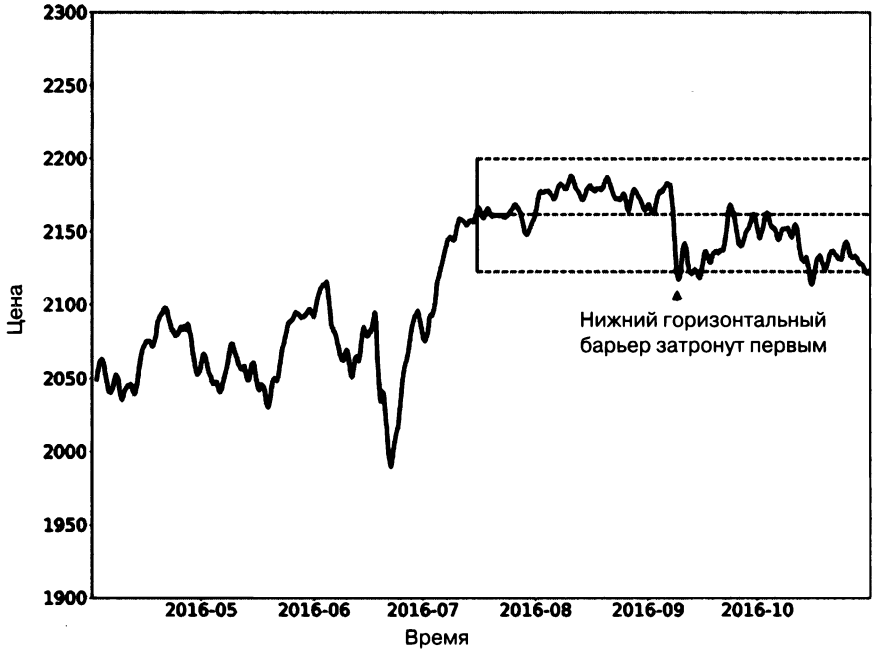
Листинг 3.2. Тройной барьерный метод

```
def applyPtSlOnT1(close,events,ptSl,molecule):
   # применить стоп-лосс/тейк-профит, если он возникает перед t1
   # (концом события)
   events_=events.loc[molecule]
   out=events_[['t1']].copy(deep=True)
   if ptSl[0]>0:pt=ptSl[0]*events_['trgt']
   else:pt=pd.Series(index=events.index) # NaNs
   если ptSl[1]>0:sl=-ptSl[1]*events_['trgt']
   else:sl=pd.Series(index=events.index) # NaNs
   for loc,t1 в событиях_['t1'].fillna(close.index[-1]).iteritems():
       df0=close[loc:t1] # пути цен
       df0=(df0/close[loc]-1)*events_.at[loc,'side']  # путь прибыли
       out.loc[loc,'sl']=df0[df0<sl[loc]].index.min() # самый ранний стоп-лосс
       out.loc[loc,'pt']=df0[df0>pt[loc]].index.min() # самый ранний
                                                      # тейк-профит
  return out
```

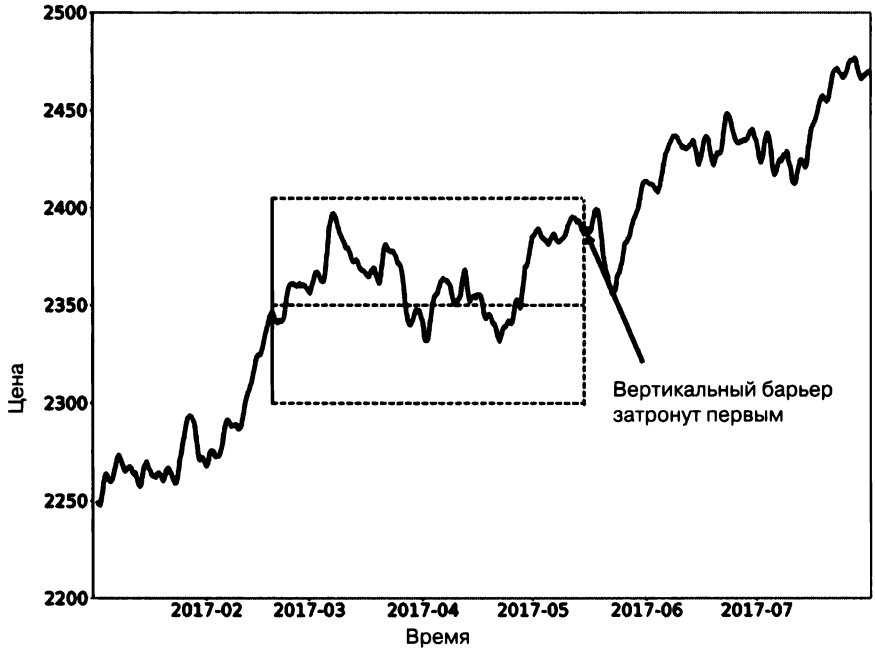
Выходом из этой функции является кадр данных библиотеки pandas, содержащий временные штампы (если таковые имеются), в которых было касание каждого барьера. Как видно из предыдущего описания, данный метод учитывает возможность отключения каждого из трех барьеров. Обозначим барьерную конфигурацию тройкой `[pt, sl, t1]`, где `0` означает, что барьер неактивен, и `1` — что барьер активен. В итоге получим восемь возможных конфигураций:

- Три полезные конфигурации:
  - [1, 1, 1]: это стандартная конфигурация, где мы определяем три условия выхода за барьер. Мы хотели бы реализовать прибыль, но у нас есть максимальная терпимость к убыткам и периоду владения.
  - [0, 1, 1]: в этой конфигурации мы хотели бы выйти после нескольких баров, если только мы не будем остановлены принудительно.
  - [1, 1, 0]: здесь мы хотели бы взять прибыль при условии, что мы не будем остановлены принудительно. Это несколько нереалистично в том плане, что мы готовы занимать эту позицию столько, сколько потребуется.
- Три менее вероятные конфигурации:
  - [0, 0, 1]: эта конфигурация эквивалентна методу фиксированного временного горизонта. Она может быть полезна применительно к объемным, долларовым или информационно-управляемым барам, и многочисленные предсказания обновляются внутри горизонта.
  - [1, 0, 1]: позиция держится во владении до тех пор, пока не будет сделана прибыль либо не будет превышен максимальный период владения без учета промежуточных нереализованных убытков.
  - [1, 0, 0]: позиция держится во владении до тех пор, пока не будет сделана прибыль. Она могла бы означать ситуацию блокировки в проигрышной позиции.
- Две алогичные конфигурации:
  - [0, 1, 0]: это бесцельная конфигурация, где мы держим позицию во владении до тех пор, пока не будем остановлены принудительно.
  - [0, 0, 0]: барьеры отсутствуют. Позиция заблокирована навсегда, и ни одна метка не генерируется.

На рис. 3.1 показаны две альтернативные конфигурации тройного барьерного метода. Слева расположена конфигурация [1, 1, 0], где первый затронутый барьер — нижний горизонтальный. Справа конфигурация [1, 1, 1], где первый затронутый барьер — вертикальный.



а



б

Рис. 3.1. Две альтернативные конфигурации тройного барьерного метода

3.5. Выяснение стороны и размера ставки

В этом разделе мы обсудим вопрос маркировки примеров для того, чтобы обучающийся алгоритм мог узнать сторону и размер ставки[1]. Мы заинтересованы в том, чтобы знать сторону ставки, когда у нас нет базовой модели для установки знака нашей позиции (длинная или короткая). При таких обстоятельствах мы не можем провести различие между барьером взятия прибыли и барьером остановки убытка, поскольку для этого требуется знать сторону ставки. Выяснение стороны подразумевает, что либо нет горизонтальных барьеров, либо горизонтальные барьеры должны быть симметричными.

Листинг 3.3 реализует функцию `getEvents`, которая находит время первого касания барьера. Данная функция получает следующие ниже аргументы:

- `close`: ценовой ряд библиотеки pandas.
- `tEvents`: индекс DateTimeIndex библиотеки pandas, содержащий временные штампы, которые делают посев каждого тройного барьера. Это те самые временные штампы, которые отбираются процедурами отбора, описанными в разделе 2.5 главы 2.
- `ptSl`: неотрицательное вещественное, задающее ширину двух барьеров. Значение 0 означает, что соответствующий горизонтальный барьер (взятие прибыли и/или остановка убытка) будет отключен.
- `t1`: ряд библиотеки pandas с временными штампами вертикальных барьеров. Мы передаем `False`, когда хотим отключить вертикальные барьеры.
- `trgt`: ряд библиотеки pandas с целями, выраженный в виде абсолютных финансовых возвратов.
- `minRet`: минимальный целевой финансовый возврат, необходимый для выполнения тройного барьерного поиска.
- `numThreads`: число потоков, одновременно используемых функцией.

Листинг 3.3. Получение времени первого касания

```
def getEvents(close,tEvents,ptSl,trgt,minRet,numThreads,t1=False):
    #1) получить цель
    trgt=trgt.loc[tEvents]
    trgt=trgt[trgt>minRet] # minRet
    #2) получить t1 (максимальный период владения)
    if t1 is False: t1=pd.Series(pd.NaT,index=tEvents)
    #3) сформировать объект событий events, применить стоп-лосс на t1
    side_=pd.Series(1.,index=trgt.index)
```

[1] Термин «сторона ставки» (the side of the bet) отражает то, в какую сторону пойдет движение цены — в длинную (вверх) или короткую (вниз). Этот термин имеет синоним «позиция», то есть соответственно длинная или короткая позиция. — *Примеч. науч. ред.*

```
events=pd.concat({'t1':t1,'trgt':trgt,'side':side_}, \
    axis=1).dropna(subset=['trgt'])
df0=mpPandasObj(func=applyPtSlOnT1,pdObj=('molecule',events.index), \
    numThreads=numThreads,close=close,events=events,ptSl=[ptSl,ptSl])
events['t1']=df0.dropna(how='all').min(axis=1) # pd.min игнорирует
                                               # значения nan
events=events.drop('side',axis=1)
return events
```

Предположим, что $I = 1E6$ и $h = 1E3$, тогда число оцениваемых условий на одном инструменте достигает миллиарда. Многие задачи машинного обучения — вычислительно дорогостоящи, если только вы не знакомы с многопоточностью, и эта задача одна из таких. Именно здесь в игру вступают параллельные вычисления. В главе 20 обсуждается несколько функций мультиобработки, которые мы будем использовать во всей книге.

Функция `mpPandasObj` вызывает механизм мультиобработки, который подробно описан в главе 20. На данный момент вам просто нужно знать, что эта функция будет выполнять `applyPtsiOnT1` параллельно. Функция `applyPtsiOnT1` возвращает временные штампы, в которых происходит касание каждого барьера (если вообще оно происходит). Тогда время первого касания — это самое раннее время среди трех, возвращенных функцией `applyPtsiOnT1`. Так как мы должны узнать сторону ставки, мы передали в качестве аргумента `ptSl = [ptSl, ptSl]` и произвольно установили сторону всегда равной длинной (горизонтальные барьеры симметричны, поэтому для определения времени первого касания сторона не имеет значения). Вывод этой функции — кадр данных библиотеки pandas со столбцами:

- `t1`: временной штамп первого касания барьера;
- `trgt`: цель, которая была использована для генерирования горизонтальных барьеров.

Листинг 3.4 показывает один из способов определения вертикального барьера. Для каждого индекса в `tEvents` он находит временной штамп следующего ценового бара в числе дней `numDays` или сразу после. Этот вертикальный барьер может быть передан в функцию `getEvents` как необязательный аргумент `t1`.

Листинг 3.4. Добавление вертикального барьера

```
t1=close.index.searchsorted(tEvents+pd.Timedelta(days=numDays))
t1=t1[t1<close.shape[0]]
t1=pd.Series(close.index[t1],index=tEvents[:t1.shape[0]]) # NaNs в конце
```

Наконец, мы можем промаркировать наблюдения с помощью функции `getBins`, определенной в листинге 3.5. Ее аргументами являются кадр данных событий `events`, который мы только что обсуждали, и ценовой ряд библиотеки pandas `close`. На выходе получается кадр данных со столбцами:

- `ret`: финансовый возврат, реализованный в момент первого касания барьера;

- bin: метка, {−1, 0, 1}, в качестве знаковой функции sign исхода. Данную функцию можно легко скорректировать, чтобы маркировать как 0 те события, когда вертикальный барьер был задет первым, что мы оставляем в качестве упражнения.

Листинг 3.5. Маркировка стороны и размера ставки

```
def getBins(events,close):
    #1) цены выровнены с событиями events
    events_=events.dropna(subset=['t1'])
    px=events_.index.union(events_['t1'].values).drop_duplicates()
    px=close.reindex(px,method='bfill')
    #2) создать объект
    out=pd.DataFrame(index=events_.index)
    out['ret']=px.loc[events_['t1'].values].values/px.loc[events_.index]-1
    out['bin']=np.sign(out['ret'])
    return out
```

3.6. Метамаркировка

Предположим, у вас есть модель для установки стороны ставки (длинной или короткой). Вам осталось узнать размер этой ставки, который подразумевает возможность отсутствия ставки вообще (нулевой размер). Практики сталкиваются с этой ситуацией регулярно. Мы часто знаем, хотим ли мы купить или продать продукт, и единственный оставшийся вопрос заключается в том, каким количеством денег мы должны рискнуть в такой ставке. Мы не хотим, чтобы алгоритм МО обучался узнавать сторону ставки просто для того, чтобы сообщать нам, какой размер является подходящим. В этой связи, вероятно, вас не удивит, если вы услышите, что ни в одной книге или публикации эта распространенная задача до сих пор не обсуждалась. К счастью, на этом страдания заканчиваются. Я называю эту задачу метамаркировкой, потому что мы хотим построить вторичную модель МО, которая учится использовать первичную экзогенную, внешне обусловленную модель.

Для того чтобы обрабатывать метамаркировку, вместо написания совершенно новой функции `getEvents` мы внесем несколько корректировок в предыдущий исходный код. Во-первых, мы принимаем новый необязательный аргумент `side` (по умолчанию `None`), который содержит сторону наших ставок, как было решено первичной моделью. Когда аргумент `side` равен `None`, данная функция понимает, что в игру вступает метамаркировка. Во-вторых, поскольку теперь мы знаем сторону, мы можем эффективно различать взятие прибыли и остановку убытка. Горизонтальные барьеры не обязательно должны быть симметричными, как в разделе 3.5. Аргумент `ptSl` — это список из двух неотрицательных вещественных значений, где `ptSl[0]` представляет собой сомножитель, который умножает `trgt` для установки ширины верхнего барьера, и `ptSl[1]` — сомножитель, который умножает `trgt` для определения ширины нижнего барьера. Если значение равно 0, то соответствующий барьер отключается. Листинг 3.6 реализует эти улучшения.

Листинг 3.6. Расширение функции getEvents для встраивания метамаркировки

```
def getEvents(close,tEvents,ptSl,trgt,minRet,numThreads,t1=False,side=None):
    #1) получить цель
    trgt=trgt.loc[tEvents]
    trgt=trgt[trgt>minRet] # minRet
    #2) получить t1 (максимальный период владения)
    if t1 is False: t1=pd.Series(pd.NaT,index=tEvents)
    #3) сформировать объект событий events, применить стоп-лосс к t1
    if side is None: side_,ptSl_=pd.Series(1.,index=trgt.
                     index),[ptSl[0],ptSl[0]]
    else: side_,ptSl_=side.loc[trgt.index],ptSl[:2]
    events=pd.concat({'t1':t1,'trgt':trgt,'side':side_}, \
        axis=1).dropna(subset=['trgt'])
    df0=mpPandasObj(func=applyPtSlOnT1,pdObj=('molecule',events.index), \
        numThreads=numThreads,close=inst['Close'],events=events,ptSl=ptSl_)
    events['t1']=df0.dropna(how='all').min(axis=1) # pd.min игнорирует
                                                   # значение nan
    if side is None: events=events.drop('side',axis=1)
    return events
```

Чтобы функция `getBins` обрабатывала метамаркировку, мы должны ее расширить схожим образом. Код из листинга 3.7 реализует необходимые изменения.

Листинг 3.7. Расширение функции getBins для встраивания метамаркировки

```
def getBins(events,close):
    '''
    Вычислить исход события (включая информацию о стороне, если
предоставлена).
    events является кадром данных, где:
    -events.index - время начала события
    -events['t1'] - время окончания события
    -events['trgt'] - цель события
    -events['side'] (необязательно) предполагает сторону позиции алгоритма
    Случай 1: ('side' не находится в events):
        интервал в (-1,1) <- метка по движению цены
    Случай 2: ('side' находится в events):
        интервал в (0,1) <- метка по прибыли и убыткам pnl (метамаркировка)
    '''
    #1) цены выровнены с событиями events
    events_=events.dropna(subset=['t1'])
    px=events_.index.union(events_['t1'].values).drop_duplicates()
    px=close.reindex(px,method='bfill')
    #2) создать объект
    out=pd.DataFrame(index=events_.index)
    out['ret']=px.loc[events_['t1'].values].values/px.loc[events_.index]-1
    if 'side' in events_: out['ret']*=events_['side']    # метамаркировка
    out['bin']=np.sign(out['ret'])
    if 'side' in events_: out.loc[out['ret']<=0,'bin']=0 # метамаркировка
    return out
```

Теперь вероятные значения разметки для `out['bin']` лежат в пределах {0,1}, в противовес предыдущим подходящим величинам {–1,0,1}. Алгоритм МО учится выбирать между решением открыть позицию и бездействием — сугубо бинарное прогнозирование. Если прогнозируемая маркировка имеет значение 1, мы можем использовать вероятность второго прогноза для определения объема средств, причем тип позиции будет заранее установлен основной моделью.

3.7. Как использовать метамаркировку

Задачи бинарной классификации представляют собой компромисс между ошибками 1-го рода (ложные утверждения) и ошибками 2-го рода (ложные отрицания). В общем случае, увеличение частоты истинных утверждений бинарного классификатора будет приводить к увеличению его частоты ложных утверждений. Кривая операционной характеристики приемника (receiver operating characteristic, ROC) бинарного классификатора измеряет стоимость увеличения частоты истинных утверждений с точки зрения принятия более высоких частот ложных утверждений.

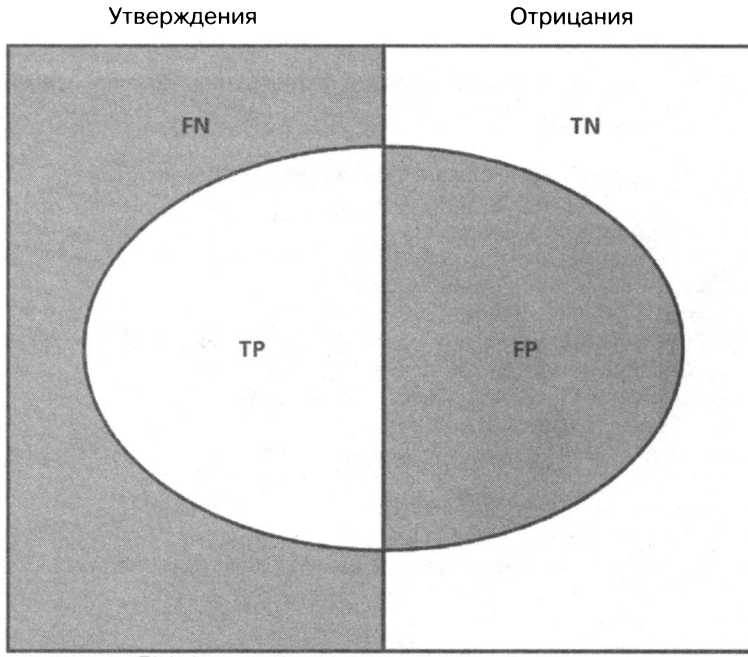


Рис. 3.2. Визуализация «матрицы ошибок». FN (false negatives) — ложные отрицания, TP (true positives) — истинные утверждения, TN (true negatives) — истинные отрицания, FP (false positives) — ложные утверждения

На рис. 3.2 показана так называемая матрица ошибок. На множестве наблюдений есть элементы, которые показывают условие (утверждения, левый прямоугольник), и элементы, которые не показывают условие (отрицания, правый прямоугольник). Бинарный классификатор предсказывает, что некоторые элементы показывают условие (эллипс), где область TP содержит истинные утверждения и область TN содержит истинные отрицания. Это приводит к двум видам ошибок: ложным утверждениям (FP) и ложным отрицаниям (FN). «Точность» — это соотношение между площадью TP и площадью эллипса. «Полнота» — это отношение между областью TP и областью в левом прямоугольнике. Данное понятие полноты (так называемой частоты истинных утверждений) находится в контексте классификационных задач, аналогично «силе» в контексте проверки статистических гипотез. «Правильность» — это сумма областей TP и TN, деленная на совокупное множество элементов (квадрат). В общем случае, уменьшение области FP происходит за счет увеличения области FN, потому что более высокая точность обычно означает меньшее число распознаваний, следовательно, более низкую полноту. Тем не менее существует некое сочетание точности и полноты, которое максимизирует суммарную эффективность классификатора. Балльная оценка F1 служит мерой эффективности классификатора в качестве гармонического среднего между точностью и полнотой (подробнее об этом в главе 14).

Метамаркировка особенно полезна, когда вы хотите достичь более высоких оценок F1. Во-первых, мы строим модель, которая достигает высокой полноты, даже если точность не особенно высока. Во-вторых, мы поправляем низкую точность, применяя метамаркировку к утвердительным исходам, предсказанным первичной моделью.

Метамаркировка увеличит вашу оценку F1, отфильтровав ложные утверждения, где большинство утвердительных исходов уже были идентифицированы первичной моделью. Иными словами, роль вторичного алгоритма МО заключается в определении того, является ли утверждение первичной (экзогенной) модели истинным или ложным. Его цель не в том, чтобы выдавать возможность делать ставки, а в том, чтобы определять, следует ли нам действовать либо следует пропустить предоставленную возможность.

Метамаркировка является очень мощным инструментом, который стоит иметь в своем арсенале по четырем дополнительным причинам. Во-первых, алгоритмы МО часто критикуются как черные ящики (см. главу 1). Метамаркировка позволяет построить систему МО поверх белого ящика (подобно фундаментальной модели, основанной на экономической теории). Эта способность трансформировать фундаментальную модель в модель МО должна сделать метамаркировку особенно полезной для «квантоментальных» фирм. Во-вторых, когда вы применяете метамаркировку, эффекты от переподгонки ограничены, потому что алгоритм будет решать не сторону вашей ставки, а только размер. В-третьих, отделяя предсказание стороны ставки от предсказания ее размера, метамаркировка позволяет создавать сложные стратегические структуры. Например, учтите, что признаки, служащие

драйвером для ралли, могут отличаться от признаков, служащих драйвером для активной распродажи. В этом случае вы, возможно, захотите разработать стратегию МО исключительно для длинных позиций, опираясь на рекомендации первичной модели по покупке, и обучающуюся стратегию исключительно для коротких позиций, опираясь на рекомендации совершенно другой первичной модели по продаже. В-четвертых, достижение высокой точности на малых ставках и низкой точности на больших ставках вас погубит. Правильное определение размера хороших возможностей так же важно, как и их выявление, поэтому имеет смысл разработать алгоритм МО, исключительно сосредоточенный на правильном получении этого критически важного решения (установление размера). Мы пересмотрим этот четвертый пункт в главе 10. По моему опыту, обучающиеся модели маркировки могут обеспечить более робастные и надежные результаты, чем стандартные модели маркировки.

3.8. Квантоментальный способ

Возможно, вам приводилось читать в прессе, что многие хеджевые фонды с восторгом принимают квантоментальный подход. Простой поиск в интернете покажет сообщения о том, что многие хеджевые фонды, в том числе некоторые из самых традиционных, инвестируют десятки миллионов долларов в технологии, предназначенные для объединения человеческого опыта с квантитативными методами. Оказывается, метамаркировка — это именно то, чего эти люди ждали. Давайте посмотрим почему.

Предположим, что у вас есть ряд, состоящий из признаков, которые, по вашему мнению, могут прогнозировать некоторые цены, вы просто не знаете как. Поскольку у вас нет модели, которая определяет сторону каждой ставки, вам нужно узнать ее сторону и размер. Вы применяете то, что вы узнали в разделе 3.5, и производите некие метки, основываясь на тройном барьерном методе с симметричными горизонтальными барьерами. Теперь вы готовы выполнить подгонку своего алгоритма на тренировочном подмножестве и оценить точность своих прогнозов на тестовом подмножестве. В качестве альтернативы можно сделать следующее:

1. Использовать свои прогнозы из первичной модели и генерировать метаметки. Помните, что горизонтальные барьеры не обязательно должны быть симметричными.
2. Снова выполнить подгонку модели на том же тренировочном подмножестве, но на этот раз с использованием только что сгенерированных метаметок.
3. Совместить «стороны» из первой обучающейся модели с «размерами» из второй обучающейся модели.

Вы всегда можете добавить метамаркировочный слой в любую первичную модель, будь то алгоритм МО, эконометрическое уравнение, техническое торговое правило, фундаментальный анализ и т. д. Сюда относятся и прогнозы, генерируемые челове-

ком исключительно на основе его интуиции. В этом случае метамаркировка поможет нам выяснить, когда мы должны исполнять или отклонять ставку дискреционного портфельного менеджера. Признаки, используемые таким обучающимся алгоритмом маркировки, могут варьироваться от рыночной информации до биометрической статистики и психологических оценок. Например, алгоритм МО маркировки может обнаружить, что дискреционные портфельные менеджеры склонны делать особенно хорошие ставки, когда имеется структурный сдвиг (глава 17), поскольку они могут быстрее схватывать изменение рыночного режима. И наоборот, он может обнаружить, что дискреционные портфельные менеджеры, находящиеся в состоянии стресса, что проявляется при уменьшении числа часов для сна, во время усталости, изменении в весе и т. д., склонны делать неточные предсказания[1]. Многие профессии требуют регулярных психологических экзаменов, и алгоритм МО маркировки может обнаруживать, что эти балльные оценки также релевантны для того, чтобы оценивать нашу текущую степень уверенности в предсказаниях портфельного менеджера. Вполне возможно, ни один из этих факторов на дискреционных портфельных менеджеров не влияет, и их мозги работают независимо от эмоционального состояния, как холодные вычислительные машины. Смею предположить, что это не так, и поэтому метамаркировка должна стать существенным методом МО для каждого дискреционного хеджевого фонда. В ближайшем будущем каждый дискреционный хеджевый фонд станет квантоментальной фирмой, и метамаркировка предлагает им четкий путь для этого перехода.

3.9. Исключение ненужных меток

Некоторые классификаторы МО плохо работают, когда классы слишком несбалансированны. В этих условиях предпочтительнее отказаться от крайне редких меток и сосредоточиться на более распространенных исходах. Листинг 3.8 представляет процедуру, которая рекурсивно отбрасывает наблюдения, связанные с чрезвычайно редкими метками. Функция `dropLabels` рекурсивно устраняет те наблюдения, связанные с классами, которые появляются меньше, чем доля `minPct` случаев, если только не осталось всего два класса.

Листинг 3.8. Устранение малозаселенных меток

```
def dropLabels(events,minPtc=.05):
    # применить веса, устранить метки с недостаточным числом примеров
    while True:
        df0=events['bin'].value_counts(normalize=True)
        if df0.min()>minPct or df0.shape[0]<3: break
        print 'dropped label',df0.argmin(),df0.min()
        events=events[events['bin']!=df0.argmin()]
    return events
```

[1] Вы, вероятно, знаете по крайней мере один крупный хедж-фонд, который круглосуточно отслеживает эмоциональное состояние своих аналитиков.

Кстати, еще одна причина, по которой вы, возможно, захотите удалять ненужные метки, — это известный дефект библиотеки sklearn: https://github.com/scikit-learn/scikit-learn/issues/8566. Такого рода дефекты являются следствием очень фундаментальных допущений, принятых в реализации библиотеки sklearn, и их устранение далеко не тривиально. В данном конкретном случае ошибка связана с решением разработчиков библиотеки sklearn работать со стандартными массивами библиотеки numpy, а не со структурированными массивами или объектами библиотеки pandas. Маловероятно, что к моменту прочтения вами этой главы или в ближайшем будущем будет найдено его исправление. В последующих главах мы рассмотрим способы обхода подобных ошибок реализации путем создания собственных классов и расширения функциональности библиотеки sklearn.

Упражнения

3.1. Сформируйте долларовые бары для фьючерсного контракта E-mini S&P 500:

- **(а)** Примените симметричный фильтр CUSUM (глава 2, раздел 2.5.2.1), где порогом является среднеквадратическое отклонение суточных финансовых возвратов (листинг 3.1).
- **(б)** Используйте листинг 3.4 на ряде `t1` библиотеки pandas, где `numDays = 1`.
- **(в)** На этих отобранных признаках примените тройной барьерный метод, где `ptSl=[1,1]`, а `t1` — это ряд, созданный вами в пункте 3.1.б.
- **(г)** Примените функцию `getBins` для генерирования меток.

3.2. Из упражнения 3.1 используйте листинг 3.8 для устранения редких меток.

3.3. Скорректируйте функцию `getBins` (листинг 3.5) так, чтобы она возвращала 0 всякий раз, когда вертикальный барьер затрагивается первым.

3.4. Разработайте стратегию следования за трендом, опираясь на популярный статистический показатель технического анализа (к примеру, пересечение скользящих средних). Для каждого наблюдения модель предлагает сторону, но не размер ставки.

- **(а)** Получите метаметки для `ptSl=[1,2]` и `t1`, где `numDays=1`. В качестве `trgt` используйте суточное среднеквадратическое отклонение, как вычислено в листинге 3.1.
- **(б)** Натренируйте случайный лес принимать решение, торговать или нет. Важно: решением является торговать или нет, {0, 1}, поскольку лежащая в основе модель (пересекающиеся скользящие средние) приняла решение о стороне ставки {−1, 1}.

3.5. Разработайте стратегию взврата к среднему значению на основе полос Боллинджера. Для каждого наблюдения модель предлагает сторону, но не размер ставки.

(а) Получите метаметки для `ptSl = [0,2]` и `t1`, где `numDays = 1`. В качестве `trgt` используйте суточное среднеквадратическое отклонение, как вычислено в листинге 3.1.

(б) Натренируйте случайный лес принимать решение, торговать или нет. В качестве признаков используйте: волатильность, внутрирядовую корреляцию и пересекающиеся скользящие средние из упражнения 3.2.

(в) Какова точность предсказаний от первичной модели (то есть если вторичная модель не фильтрует ставки)? Каковы точность, полнота и оценка F1?

(г) Какова точность предсказаний от вторичной модели? Каковы точность, полнота и балльная оценка F1?

4 Веса выборки

4.1. Актуальность

В главе 3 представлен ряд новых методов маркировки финансовых наблюдений. Мы ввели два новых понятия, тройной барьерный метод и метамаркировку, и объяснили, как они могут быть полезны в финансовых приложениях, включая квантоментальные инвестиционные стратегии. В этой главе вы узнаете, как использовать веса выборки для решения другой проблемы, повсеместно встречающейся в финансовых приложениях, а именно что наблюдения не генерируются одинаково распределенными взаимно независимыми случайными процессами. Подавляющая часть литературы по машинному обучению основывается на допущении об одинаковой распределенности и взаимной независимости, и одна из причин, почему многие приложения машинного обучения оказываются безуспешными в финансах, заключается в том, что эти допущения нереалистичны в случае финансовых временных рядов.

4.2. Накладывающиеся исходы

В главе 3 мы назначили метку y_i наблюдаемому признаку X_i, где y_i — это функция ценовых баров, возникших на интервале $[t_{i,0}, t_{i,1}]$. Если $t_{i,1} > t_{j,0}$, а $i < j$, тогда y_i и y_j будут оба зависеть от общего финансового возврата $r_{t_{j,0},\min\{t_{i,1},t_{j,1}\}}$, то есть возврата на интервале $[t_{j,0}, \min\{t_{i,1}, t_{j,1}\}]$. Из этого следует, что ряд меток, $\{y_i\}_{i=1,\ldots,I}$ не является одинаково распределенным и взаимно независимым всякий раз, когда существует наложение между любыми двумя исходами подряд, $\exists i | t_{i,1} > t_{i+1,0}$.

Предположим, что мы обходим эту проблему, ограничивая горизонт ставки следующим образом: $t_{i,1} \leq t_{i+1,0}$. В этом случае наложение отсутствует, так как исход каждого признака определяется до или в начале следующего наблюдаемого признака. Это ведет к грубым моделям, в которых частота отбора признаков будет ограничена горизонтом, используемым для определения исхода. С одной стороны, если бы мы хотели исследовать исходы, которые длились месяц, то признаки должны были бы отбираться с частотой до месяца. С другой стороны, если бы мы увеличили частоту отбора, скажем, до суточной, то мы были бы вынуждены сократить горизонт исхода до одного дня. Более того, если бы мы хотели применить зависящий

от траектории метод маркировки, такой как тройной барьерный метод, то частота отбора была бы подчинена касанию первого барьера. Независимо от того, что вы делаете, ограничение горизонта исхода для устранения наложений является ужасным решением. Мы должны допустить $t_{i,1} > t_{i+1,0}$, что возвращает нас к описанной ранее проблеме наложения исходов.

Такая ситуация характерна для финансовых приложений. Большинство нефинансовых исследователей машинного обучения могут допускать, что наблюдения взяты из одинаково распределенных взаимно независимых процессов. Например, вы можете получить пробы крови у большого числа пациентов и измерить в них уровень холестерина. Разумеется, различные базовые общие факторы изменят среднее значение и среднеквадратическое отклонение распределения холестерина, но образцы по-прежнему будут независимыми: имеется одно наблюдение на испытуемого. Предположим, вы берете эти пробы крови, а кто-то в вашей лаборатории разливает кровь из каждой пробирки в следующие девять пробирок справа от них. То есть пробирка 10 содержит кровь пациента 10, а также кровь пациентов с 1 по 9. Пробирка 11 содержит кровь пациента 11, а также кровь пациентов со 2 по 10 и т. д. Теперь вам нужно определить признаки, предсказывающие высокий уровень холестерина (диета, физические нагрузки, возраст и др.), не зная наверняка уровня холестерина каждого пациента. С эквивалентной задачей мы сталкиваемся в финансовом машинном обучении, только с дополнительным препятствием в том, что характер разлива является недетерминированным и неизвестным. Что касается приложений машинного обучения, то финансы — это не самонастраивающийся предмет plug-and-play. Любой, кто скажет вам иначе, потратит ваше время и деньги.

Существует несколько способов атаковать проблему меток, которые не являются одинаково распределенными и взаимно независимыми, и в этой главе мы ее рассмотрим, сконструировав схемы отбора и взвешивания, которые корректируют чрезмерное влияние накладывающихся исходов.

4.3. Число одновременных меток

Две метки y_i и y_j — одновременны в t, когда обе являются функцией хотя бы одного общего финансового возврата, $r_{t-1,t} = \frac{p_t}{p_{t-1}} - 1$. Наложение не обязательно должно быть идеальным, в смысле, что обе метки охватывают один и тот же временно́й интервал. В этом разделе мы собираемся вычислить число меток, которое является функцией данного финансового возврата, $r_{t-1,t}$. Во-первых, для каждой временно́й точки $t = 1, ..., T$ мы формируем двоичный массив $\{1_{t,i}\}_{i=1,...,I}$, где $1_{t,i} \in \{0, 1\}$. Переменная $1_{t,i} = 1$, если и только если $[t_{i,0}, t_{i,1}]$ накладывается на $[t-1, t]$, в противном случае $1_{t,i} = 0$. Напомним, что охват у меток $\{[t_{i,0}, t_{i,1}]\}_{i=1,...,I}$ определяется объектом `t1`, представленным в главе 3. Во-вторых, мы вычисляем число меток, одновременных в t, $c_t = \sum_{i=1}^{I} 1_{t,i}$. Листинг 4.1 иллюстрирует реализацию этой логики.

Листинг 4.1. Оценивание уникальности метки

```
def mpNumCoEvents(closeIdx,t1,molecule):
    '''
    Вычислить число одновременных событий в расчете на бар.
    +molecule[0] – это дата первого события, на котором будет вычислен вес
    +molecule[-1] – это дата последнего события, на котором будет
    вычислен вес
    Любое событие, которое начинается перед t1[molecule].max(), влияет
    на число.
    '''
    #1) найти события, которые охватывают период [molecule[0],molecule[-1]]
    # незакрытые события по-прежнему должны влиять на другие веса
    t1=t1.fillna(closeIdx[-1])
    # события, которые заканчиваются в molecule[0] или после
    t1=t1[t1>=molecule[0]]
    # события, которые начинаются в t1[molecule].max() или перед
    t1=t1.loc[:t1[molecule].max()]
    #2) подсчитать события, охватывающие бар
    iloc=closeIdx.searchsorted(np.array([t1.index[0],t1.max()]))
    count=pd.Series(0,index=closeIdx[iloc[0]:iloc[1]+1])
    for tIn,tOut in t1.iteritems(): count.loc[tIn:tOut]+=1.
    return count.loc[molecule[0]:t1[molecule].max()]
```

4.4. Средняя уникальность метки

В этом разделе мы оценим уникальность метки (без наложения) как ее среднюю уникальность за период ее существования. Во-первых, уникальность метки i в момент времени t равна $u_{t,i}= 1_{t,i}c_t^{-1}$. Во-вторых, средняя уникальность метки i является средним $u_{t,i}$ за период ее существования, $\bar{u}_i = \left(\sum_{t=1}^{T} u_{t,i}\right)\left(\sum_{t=1}^{T} 1_{t,i}\right)^{-1}$. Эта средняя уникальность также может быть интерпретирована как обратная гармоническому среднему c_t за период существования события. На рис. 4.1 показана гистограмма значений уникальности, полученных из объекта `t1`. Листинг 4.2 реализует эти расчеты.

Листинг 4.2. Оценивание средней уникальности метки

```
def mpSampleTW(t1,numCoEvents,molecule):
    # Получить среднюю уникальность за период существования события
    wght=pd.Series(index=molecule)
    for tIn,tOut in t1.loc[wght.index].iteritems():
        wght.loc[tIn]=(1./numCoEvents.loc[tIn:tOut]).mean()
    return wght
#-----------------------------------
numCoEvents=mpPandasObj(mpNumCoEvents,('molecule',events.index),numThreads, \
    closeIdx=close.index,t1=events['t1'])
numCoEvents=numCoEvents.loc[~numCoEvents.index.duplicated(keep='last')]
numCoEvents=numCoEvents.reindex(close.index).fillna(0)
out['tW']=mpPandasObj(mpSampleTW,('molecule',events.index),numThreads, \
    t1=events['t1'],numCoEvents=numCoEvents)
```

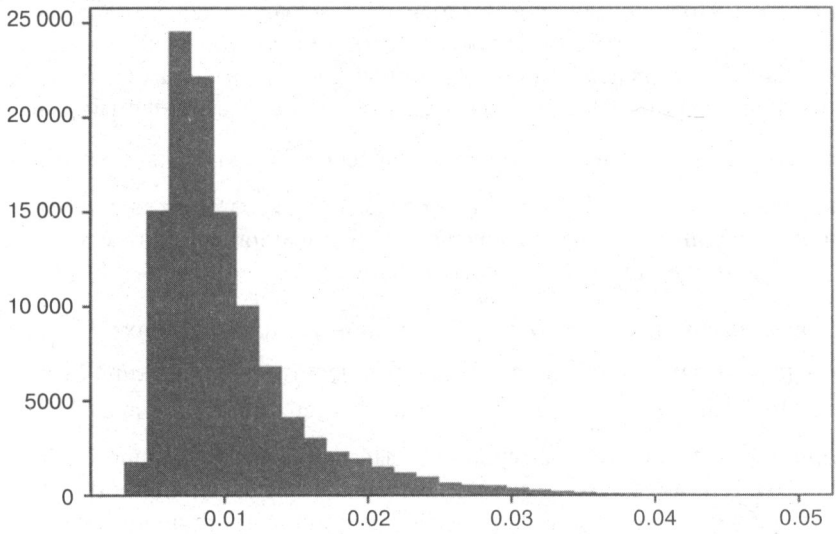


Рис. 4.1. Гистограмма значений уникальности

Обратите внимание, что мы снова используем функцию `mpPandasObj`, которая ускоряет процесс вычислений за счет многопроцессорной обработки данных (см. главу 20). Вычисление средней уникальности метки i, $\bar{u}_i$ требует информации, которая доступна только в будущем, `events['t1']`. Это не является проблемой, поскольку $\{\bar{u}_i\}_{i=1,\ldots,I}$ используется в наборе данных для обучения в сочетании с информацией о метке. Подобные $\{\bar{u}_i\}_{i=1,\ldots,I}$ не используются для прогнозирования метки, поэтому утечка информации невозможна. Эта процедура позволяет нам назначить результат уникальности между 0 и 1 для каждого наблюдаемого признака относительно пересекающихся исходов.

4.5. Бэггинг классификаторов и уникальности

Вероятность не отобрать конкретный элемент i после I взятий образцов с возвратом[1] на множестве из I элементов равна $(1 - I^{-1})^I$. По мере роста размера выборки эта вероятность сходится к асимптотическому значению $\lim_{I\to\infty}(1 - I^{-1})^I = e^{-1}$. Из этого следует, что число извлеченных уникальных наблюдений должно равняться $(1 - e^{-1}) \approx \frac{2}{3}$.

[1] Взятие образцов с возвратом/возмещением или без возврата/возмещения (with или without replacement) — это варианты отбора, когда элемент возвращается или не возвращается в исходное множество перед тем, как из него взять следующий образец. — *Примеч. науч. ред.*

Предположим, что максимальное число ненакладывающихся исходов равно $K \leq I$. Следуя тому же аргументу, вероятность не отобрать конкретный элемент i после I взятий образцов с возвратом на множестве из I элементов равна $(1 - K^{-1})^I$. По мере роста размера выборки эта вероятность может быть аппроксимирована как $(1-I^{-1})^{I\frac{K}{I}} \approx e^{-\frac{K}{I}}$. Это означает, что число извлеченных уникальных наблюдений должно равняться $1 - e^{-\frac{K}{I}} \leq 1 - e^{-1}$. Из этого следует, что принятие неправильного допущения об одинаковой распределенности и взаимной независимости извлечений образцов приводит к избыточному отбору.

При отборе с возвратом (то есть бутстрапировании[1]) на наблюдениях с $I^{-1}\sum_{i=1}^{I}\bar{u}_i \ll 1$ все более вероятным становится то, что внутрипакетные наблюдения будут (1) избыточны друг для друга и (2) очень похожи на внепакетные наблюдения[2].

Избыточность выборок отрицательно сказывается на эффективности бутстрапа (см. главу 6). К примеру, в случае применения алгоритма случайного леса все деревья в лесу будут представлять собой схожие копии одного подогнанного дерева поиска решений. Поскольку при случайной выборке примеры, вошедшие в нее, очень напоминают примеры, не вошедшие в выборку, точность последних серьезно страдает. Мы обратимся ко второй ситуации чуть позже, в главе 7, когда речь пойдет о кросс-валидации при наблюдениях вне HOP-данных. А пока мы сконцентрируемся на первом примере, то есть на бэггинге при наблюдениях $I^{-1}\sum_{i=1}^{I}\bar{u}_i \ll 1$.

Одним из решений может быть отказ от пересекающихся исходов перед выполнением бутстрапа. Поскольку пересечения не идеальны, отказ от наблюдения из-за частичного пересечения может вылиться в критическую потерю информации. Я не рекомендую пользоваться этим способом.

Второе и более качественное решение — использовать среднюю уникальность, $I^{-1}\sum_{i=1}^{I}\bar{u}_i$ с целью сокращения чрезмерного влияния исходов, содержащих избыточную информацию. Соответственно, мы могли бы отбирать только часть `out['tW'].mean()` наблюдений, или небольшую их часть. В библиотеке sklearn

---

[1] Процесс бутстрапирования можно концептуально представить как многократное взятие исходных образцов с их возвратом в совокупность данных с целью получения синтетической выборки. — *Примеч. науч. ред.*

[2] Бэггинг (bagging, bootstrap aggregating, пакетирование) — это общая методика формирования набора моделей путем взятия бутстраповских выборок из данных. Синонимы: агрегирование бутстраповских образцов, бутстрап-агрегирование. В бэггинге предикторы строятся путем взятия бутстраповских образцов из тренировочного подмножества, после чего они агрегируются для формирования бэггированного (пакетного) предиктора. Образцы, использованные для построения предиктора, называются внутрипакетными (in-bag), а образцы, которые для этого не использовались, — внепакетными (out-of-bag). — *Примеч. науч. ред.*

класс `sklearn.ensemble.BaggingClassifier` принимает аргумент `max_samples`, который может быть установлен равным `max_samples=out['tW'].mean()`. Благодаря этому мы добиваемся того, что внутрипакетные наблюдения не отбираются с частотой, намного превышающей их уникальность. Случайные леса эту функциональность `max_samples` не предлагают, однако решение состоит в том, чтобы собрать в пакет большое число деревьев решений. Мы обсудим это решение далее в главе 6.

4.5.1. Последовательное бутстрапирование

Третьим и более качественным решением является последовательное взятие бутстраповских образцов, где образцы извлекаются в соответствии с изменяющейся вероятностью, которая делает поправку на избыточность. В публикации Rao и соавт. [1997] предлагается последовательный повторный отбор с возвратом до тех пор, пока не появятся K отличающихся исходных наблюдений. Хотя их схема и представляет интерес, она не в полной мере относится к нашей финансовой проблеме накладывающихся исходов. В последующих разделах мы введем альтернативный метод, который непосредственно решает данную проблему.

Во-первых, наблюдение X_i берется из равномерного распределения $i \sim U[1, I]$, то есть вероятность взятия какого-либо конкретного значения i изначально равняется $\delta_i^{(1)} = I^{-1}$. В случае взятия второго образца мы хотим уменьшить вероятность взятия наблюдения X_j с сильно накладывающимся исходом. Напомним, что бутстрапирование допускает отбор образцов с повторами, поэтому можно по-прежнему снова взять наблюдение X_i, но мы хотим уменьшить его вероятность, так как между X_i и им самим существует наложение (по сути дела, идеальное наложение). Обозначим через φ последовательность взятых на данный момент наблюдений, которая может включать повторы. Пока мы знаем, что $\varphi^{(1)} = \{i\}$. Уникальность j в момент времени t равна $u_{t,j}^{(2)} = 1_{t,j}(1 + \Sigma_{k \in \varphi^{(1)}} 1_{t,k})^{-1}$, поскольку эта уникальность вытекает из добавления альтернативных j в существующую последовательность взятых образцов $\varphi^{(1)}$. Средняя уникальность для j — это среднее значение $u_{t,j}^{(2)}$ за период существования j, $\bar{u}_j^{(2)} = \left(\sum_{t=1}^{T} u_{t,j}\right)\left(\sum_{t=1}^{T} 1_{t,j}\right)^{-1}$. Теперь мы можем взять второе наблюдение, опираясь на обновленные вероятности $\left\{\delta_j^{(2)}\right\}_{j=1,\ldots,I}$,

$$\delta_j^{(2)} = \bar{u}_j^{(2)}\left(\sum_{k=1}^{I} \bar{u}_K^{(2)}\right)^{-1}.$$

где $\left\{\delta_j^{(2)}\right\}_{j=1,\ldots,I}$ масштабируется так, чтобы в сумме давать 1, $\sum_{j=1}^{I} \delta_j^{(2)} = 1$. Теперь мы можем взять второй образец, обновить $\varphi^{(2)}$ и переоценить $\left\{\delta_j^{(3)}\right\}_{j=1,\ldots,I}$. Данный процесс повторяется до тех пор, пока не будет взято I образцов. Эта последовательная бутстраповская схема имеет то преимущество, что наложения (даже повторы) по-прежнему возможны, но с меньшей вероятностью. Последовательный бутстраповский отбор будет намного ближе к одинаковой распределенности и взаимной

независимости, чем метод стандартного бутстраповского отбора. Это можно перепроверить, измерив увеличение $I^{-1}\sum_{i=1}^{I}\bar{u}_i$ относительно стандартного бутстраповского метода.

4.5.2. Реализация последовательного бутстрапирования

Листинг 4.3 получает индикаторную матрицу из двух аргументов: индекса баров (`barIx`) и ряда `t1` библиотеки pandas, который мы использовали несколько раз в главе 3. Напомним, что ряд `t1` определяется индексом, содержащим время, в которое признаки наблюдались, и массивом значений, содержащим время, в которое метка была определена. На выходе из этой функции получается бинарная матрица, указывающая на то, какие (ценовые) бары влияют на метку по каждому наблюдению.

Листинг 4.3. Построение индикаторной матрицы

```
import pandas as pd,numpy as np
#---------------------------------------
def getIndMatrix(barIx,t1):
    # Получить индикаторную матрицу
    indM=pd.DataFrame(0,index=barIx,columns=range(t1.shape[0]))
    for i,(t0,t1) in enumerate(t1.iteritems()): indM.loc[t0:t1,i]=1.
    return indM
```

Листинг 4.4 возвращает среднюю уникальность каждого наблюдаемого признака. На вход подается индикаторная матрица, построенная функцией `getIndMatrix`.

Листинг 4.4. Вычисление средних уникальностей

```
def getAvgUniqueness(indM):
    # Средняя уникальность из индикаторной матрицы
    c=indM.sum(axis=1)   # одновременность
    u=indM.div(c,axis=0) # уникальность
    avgU=u[u>0].mean()   # средняя уникальность
    return avgU
```

Листинг 4.5 дает нам индекс признаков, отобранных последовательным бутстрапированием. На вход подаются индикаторная матрица (`indM`) и необязательная длина выборки (`sLength`) с принятым по умолчанию значением извлечений образцов, равным числу строк в матрице `indM`.

Листинг 4.5. Вернуть выборку из последовательного бутстрапирования

```
def seqBootstrap(indM,sLength=None):
    # Сгенерировать выборку посредством последовательного бутстрапирования
    if sLength is None: sLength=indM.shape[1]
    phi=[]
```

```
while len(phi)<sLength:
    avgU=pd.Series()
    for i in indM:
        indM_=indM[phi+[i]] # сократить indM
        avgU.loc[i]=getAvgUniqueness(indM_).iloc[-1]
    prob=avgU/avgU.sum() # извлечь вероятность
    phi+=[np.random.choice(indM.columns,p=prob)]
return phi
```

4.5.3. Числовой пример

Рассмотрим множество меток $\{y_i\}_{i=1,2,3}$, где метка y_1 — это функция финансового возврата $r_{0,3}$, метка y_2 — функция финансового возврата $r_{2,4}$, и метка y_3 — функция финансового дохода $r_{4,6}$. Наложения исходов характеризуются приведенной ниже индикаторной матрицей $\{1_{t,i}\}$:

$$\{1_{t,i}\} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}.$$

Процедура начинается с $\varphi^{(0)} = \emptyset$ и равномерного распределения вероятности, $\delta_i = \frac{1}{3}$, $\forall i = 1, 2, 3$. Предположим, что мы случайно вынимаем число из {1, 2, 3}, и у нас получается число 2. Перед тем как вынуть из {1, 2, 3} во второй раз (напомним, что бутстраповские образцы вынимаются с возвратами), нам нужно скорректировать вероятности. Множество наблюдений, сделанных к настоящему моменту, равняется $\varphi^{(1)} = \{2\}$. Средняя уникальность для первого признака равна $\bar{u}_1^{(2)} = \left(1 + 1 + \frac{1}{2}\right)\frac{1}{3} = \frac{5}{6} < 1$, и для второго признака равна $\bar{u}_2^{(2)} = \left(\frac{1}{2} + \frac{1}{2}\right)\frac{1}{2} = \frac{1}{2} < 1$. Вероятность для второго вынутого образца равняется $\delta^{(2)} = \left\{\frac{5}{14}, \frac{3}{14}, \frac{6}{14}\right\}$. Здесь стоит отметить два момента: 1) наименьшая вероятность относится к признаку, который был выбран в первой выемке, поскольку этим показывается самое высокое наложение; и 2) между двумя возможными выемками за пределами $\varphi^{(1)}$ наибольшая вероятность относится к $\delta_3^{(2)}$, поскольку данная метка не имеет наложение на $\varphi^{(1)}$. Предположим, что во второй раз было вынуто число 3. Мы оставляем в качестве упражнения обновление вероятностей $\delta^{(3)}$ для третьей и последней выемки. Листинг 4.6 выполняет последовательное бутстрапирование на индикаторной матрице $\{1_{t,i}\}$ в данном примере.

Листинг 4.6. Пример последовательного бутстрапирования

```
def main():
    t1=pd.Series([2,3,5],index=[0,2,4]) # t0,t1 для каждого наблюдения
                                        # признака
    barIx=range(t1.max()+1) # индекс баров
    indM=getIndMatrix(barIx,t1)
    phi=np.random.choice(indM.columns,size=indM.shape[1])
    print phi
    print 'Стандартная уникальность:',getAvgUniqueness(indM[phi]).mean()
    phi=seqBootstrap(indM)
    print phi
    print 'Последовательная уникальность:',getAvgUniqueness(indM[phi]).mean()
    return
```

4.5.4. Эксперименты Монте-Карло[1]

Мы можем оценить эффективность алгоритма последовательного бутстрапирования экспериментальными методами. Листинг 4.7 содержит функцию, которая генерирует случайный ряд `t1` для определенного числа наблюдений `numObs` (I). Каждое наблюдение производится по случайному числу, берущемуся из равномерного распределения, с границами 0 и `numBars`, где `numBars` — это число баров (T). Число охватываемых наблюдением баров определяется путем извлечения случайного числа из равномерного распределения с границами 0 и `maxH`.

Листинг 4.7. Генерирование случайного ряда t1

```
def getRndT1(numObs,numBars,maxH):
    # случайный ряд t1
    t1=pd.Series()
    for i in xrange(numObs):
        ix=np.random.randint(0,numBars)
        val=ix+np.random.randint(1,maxH)
        t1.loc[ix]=val
    return t1.sort_index()
```

Листинг 4.8 берет этот случайный ряд `t1` и выводит предполагаемую индикаторную матрицу `indM`. Затем эта матрица подвергается двум процедурам. В первой мы получаем среднюю уникальность из стандартного бутстраповского отбора (случайный отбор с возвратом). Во второй мы получаем среднюю уникальность, применяя наш алгоритм последовательного бутстрапирования. Результаты представляются в виде словаря.

---

[1] Методы Монте-Карло (ММК) — группа численных методов для изучения случайных процессов. — *Примеч. ред.*

Листинг 4.8. Уникальность из стандартного и последовательного бутстраповского отбора

```
def auxMC(numObs,numBars,maxH):
    # Параллелизованная вспомогательная функция
    t1=getRndT1(numObs,numBars,maxH)
    barIx=range(t1.max()+1)
    indM=getIndMatrix(barIx,t1)
    phi=np.random.choice(indM.columns,size=indM.shape[1])
    stdU=getAvgUniqueness(indM[phi]).mean()
    phi=seqBootstrap(indM)
    seqU=getAvgUniqueness(indM[phi]).mean()
    return {'stdU':stdU,'seqU':seqU}
```

Приведенные выше операции должны повторяться в течение большого числа итераций. Листинг 4.9 реализует этот эксперимент Монте-Карло, используя методы мультиобработки, описанные в главе 20. Например, 24-ядерному серверу потребуется около 6 часов для выполнения 1E6 итераций экспериментов Монте-Карло, где `numObs=10`, `numBars=100` и `maxH=5`. Без параллелизации аналогичный эксперимент Монте-Карло занял бы около 6 дней.

Листинг 4.9. Многопотоковые эксперименты Монте-Карло

```
import pandas as pd,numpy as np
from mpEngine import processJobs,processJobs_
#-------------------------------------
def mainMC(numObs=10,numBars=100,maxH=5,numIters=1E6,numThreads=24):
    # Эксперименты Монте-Карло
    jobs=[]
    for i in xrange(int(numIters)):
        job={'func':auxMC,'numObs':numObs,'numBars':numBars,'maxH':maxH}
        jobs.append(job)
    if numThreads==1: out=processJobs_(jobs)
    else: out=processJobs(jobs,numThreads=numThreads)
    print pd.DataFrame(out).describe()
    return
```

На рис. 4.2 показана гистограмма уникальности из стандартных бутстраповских образцов (слева) и последовательных бутстраповских образцов (справа). Медиана средней уникальности для стандартного метода равна 0.6, а медиана средней уникальности для последовательного метода — 0.7. Дисперсионно-аналитическая проверка ANOVA на разности средних возвращает исчезающе малую вероятность. С точки зрения статистики, образцы из последовательного бутстраповского метода имеют ожидаемую уникальность, которая превышает стандартный бутстраповский метод, на любом разумном уровне достоверности.

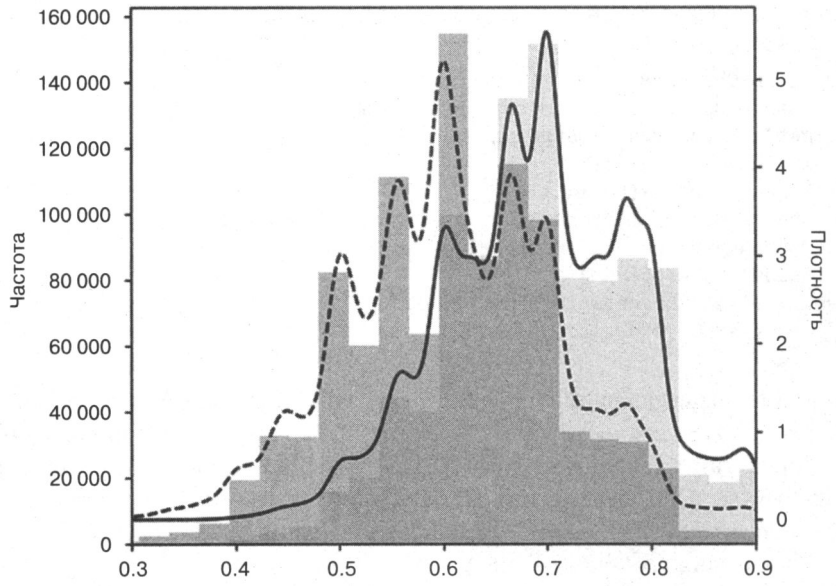


Рис. 4.2. Эксперименты Монте-Карло со стандартным и последовательным бутстраповским отбором

4.6. Атрибутирование финансовых возвратов

В предыдущем разделе мы познакомились с методом отбора бутстраповских образцов, более близких к одинаковой распределенности и взаимной независимости. В этом разделе мы представим метод взвешивания этих образцов с целью тренировки алгоритма МО. Сильно накладывающиеся исходы будут иметь непропорционально высокий вес, если их считать равными ненакладывающимся исходам. В то же время меткам, связанным с бóльшими абсолютными финансовыми возвратами, следует уделять больше внимания, чем меткам, связанным с незначительными абсолютными финансовыми возвратами. Короче говоря, нам нужно взвешивать наблюдения с помощью некой функции уникальности и абсолютного финансового возврата.

Когда метки являются функцией знака финансового возврата ($\{-1, 1\}$ для стандартной маркировки или $\{0, 1\}$ для метамаркировки), веса выборки можно определять в терминах суммы атрибутированных (отнесенных) финансовых возвратов за период существования события, $[t_{i,0}, t_{i,1}]$,

$$\tilde{w}_i = \left| \sum_{t=t_{i,0}}^{t_{i,1}} \frac{r_{t-1,t}}{c_t} \right|$$

$$w_i = \tilde{w}_i I \left(\sum_{j=1}^{I} \tilde{w}_j \right)^{-1},$$

следовательно, $\sum_{i=1}^{I} w_i = I$. Мы прошкалировали эти веса так, чтобы они в сумме давали I, так как библиотеки (включая `sklearn`) обычно определяют алгоритмические параметры, исходя из принятого по умолчанию веса, равного 1.

Обоснование этого метода заключается в том, что мы хотим взвешивать наблюдение как функцию абсолютных логарифмических финансовых возвратов, которые могут быть отнесены уникально только к нему. Однако этот метод не будет работать, если есть «нейтральный» случай (финансовый возврат ниже порога). В этом случае более низким финансовым возвратам следует назначать более высокие веса, а не обратные. «Нейтральный» случай не нужен, так как он может вытекать из предсказания «–1» или «1» с низкой достоверностью. Это одна из нескольких причин, по которой я бы посоветовал вам устранять «нейтральные» случаи. Листинг 4.10 реализует данный метод.

Листинг 4.10. Определение веса выборки путем атрибутирования абсолютного финансового возврата

```
def mpSampleW(t1,numCoEvents,close,molecule):
    # Получить вес выборки путем отнесения финансового возврата
    ret=np.log(close).diff() # логарифмические возвраты, чтобы сделать их
                             # аддитивными
    wght=pd.Series(index=molecule)
    for tIn,tOut in t1.loc[wght.index].iteritems():
        wght.loc[tIn]=(ret.loc[tIn:tOut]/numCoEvents.loc[tIn:tOut]).sum()
    return wght.abs()
#---------------------------------------
out['w']=mpPandasObj(mpSampleW,('molecule',events.index),numThreads, \
    t1=events['t1'],numCoEvents=numCoEvents,close=close)
out['w']*=out.shape[0]/out['w'].sum()
```

4.7. Временной спад, или эрозия

Рынки — это адаптивные системы (Lo [2017]). По мере развития рынков более старые примеры менее релевантны, чем более новые. Следовательно, мы, как правило, хотели бы, чтобы веса выборки убывали по мере поступления новых наблюдений. Пусть $d[x] \geq 0, \forall x \in \left[0, \sum_{i=1}^{I} \bar{u}_i\right]$ равно факторам временной эрозии, которые будут умножать веса выборки, полученные в предыдущем разделе. Конечный вес не имеет эрозии, $d\left[\sum_{i=1}^{I} \bar{u}_i\right] = 1$, и все остальные веса будут корректироваться относительно него. Пусть $c \in (-1, 1]$ равно определяемому пользователем параметру, задающему функцию убывания следующим образом: для $c \in [0, 1]$, тогда $d[1] = c$, с линейным убыванием; для $c \in (-1, 0)$ будет $d\left[-c\sum_{i=1}^{I} \bar{u}_i\right] = 0$, с линейным убыванием между $\left[-c\sum_{i=1}^{I} \bar{u}_i, \sum_{i=1}^{I} \bar{u}_i\right]$ и $d[x] = 0 \forall \leq -c\sum_{i=1}^{I} \bar{u}_i$. Для кусочно-линей-

ной функции $d = \max\{0, a + bx\}$ такие требования удовлетворяются следующими граничными условиями:

1. $d = a + b\sum_{i=1}^{I} \bar{u}_i = 1 \Rightarrow a = 1 - b\sum_{i=1}^{I} \bar{u}_i$.
2. В зависимости от c:

   (а) $d = a + b0 = c \Rightarrow b = (1-c)\left(\sum_{i=1}^{I} \bar{u}_i\right)^{-1}, \forall c \in [0, 1]$.

   (б) $d = a - bc \quad \sum_{i=1}^{I} \bar{u}_i = 0 \Rightarrow b = \left[(c+1)\sum_{i=1}^{I} \bar{u}_i\right]^{-1}, \forall c \in (-1, 0)$.

Листинг 4.11 реализует эту форму факторов временной эрозии. Обратите внимание, что время не следует воспринимать как хронологическое. В этой реализации эрозия происходит в соответствии с кумулятивной уникальностью, $x \in \left[0, \sum_{i=1}^{I} \bar{u}_i\right]$, потому что хронологическая эрозия слишком быстро сокращает веса в присутствии избыточных наблюдений.

Листинг 4.11. Реализация факторов временной эрозии

```
def getTimeDecay(tW,clfLastW=1.):
    # применить кусочно-линейную эрозию к наблюдаемой уникальности (tW)
    # новейшее наблюдение получает вес weight=1, старейшее получает
    # weight=clfLastW
    clfW=tW.sort_index().cumsum()
    if clfLastW>=0: slope=(1.-clfLastW)/clfW.iloc[-1]
    else: slope=1./((clfLastW+1)*clfW.iloc[-1])
    const=1.-slope*clfW.iloc[-1]
    clfW=const+slope*clfW
    clfW[clfW<0]=0
    print const,slope
    return clfW
```

Давайте обсудим несколько интересных случаев:

- $c = 1$ означает отсутствие временного спада.
- $0 < c < 1$ означает, что веса убывают линейно во временно́й динамике, но каждое наблюдение по-прежнему получает строго положительный вес, независимо от возраста.
- $c = 0$ означает, что по мере старения веса сходятся линейно к нулю.
- $c < 0$ означает, что самая старая часть cT наблюдений получает нулевой вес (то есть они стираются из памяти).

На рис. 4.3 показаны эрозийные веса, `out['w']*df`, после применения факторов эрозии для $c \in \{1, .75, .5, 0, -.25, -.5\}$. Несмотря на то что этот процесс не всегда полезен в практическом плане, путем установки $c > 1$ данная процедура позволяет генерировать веса, которые по мере старения увеличиваются.

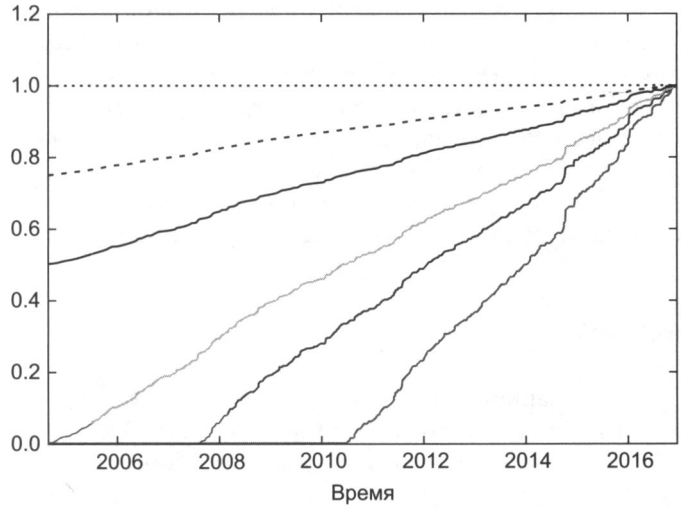


Рис. 4.3. Кусочно-линейные факторы временной эрозии

4.8. Веса классов

В дополнение к весам выборки часто полезно применять веса классов. Веса классов — это веса, которые служат для исправления недопредставленных меток. Это особенно важно в классификационных задачах, где наиболее важные классы встречаются редко (King and Zeng [2001]). Например, предположим, что вы хотите предсказать кризис ликвидности, такой как молниеносный обвал фондового рынка США 6 мая 2010 года. По сравнению с миллионами наблюдений, которые происходят между ними, такие события редки. Если мы не назначим более высокие веса образцам, связанным с этими редкими метками, то алгоритм МО будет максимизировать точность наиболее распространенных меток, а молниеносные обвалы будут считаться не редкими событиями, а выбросами.

В библиотеках машинного обучения для работы с весами классов обычно реализуется своя функциональность. Например, библиотека sklearn штрафует ошибки в образцах класса `class[j]`, $j = 1, ..., J$ весом `class_weight[j]`, а не 1. Соответственно, более высокие веса класса на метке j заставят алгоритм достичь более высокой точности на j. Когда веса классов не дают в сумме j, то эффект эквивалентен изменению регуляризационного параметра классификатора.

В финансовых приложениях стандартные метки классификационного алгоритма равны $\{-1, 1\}$, где нулевой (или нейтральный) случай будет подразумеваться предсказанием с вероятностью лишь немногим выше 0,5 и ниже некоторого нейтрального порога. Нет причин отдавать предпочтение точности одного класса перед другим, и поэтому хорошим значением по умолчанию является использование аргумента `class_weight='balanced'`. Этот вариант значения аргумента выполняет

перевесовку наблюдений, тем самым симулируя появление всех классов с одинаковой частотой. В контексте бэггированных классификаторов можно рассмотреть аргумент `class_weight='balanced_subsample'`, который означает, что аргумент `class_weight='balanced'` будет применен к внутрипакетным бутстрапированным образцам, а не ко всей совокупности данных в целом. Для получения более подробной информации полезно почитать исходный код, реализующий `class_weight` в библиотеке sklearn. Пожалуйста, обратите внимание на этот зарегистрированный дефект: https://github.com/scikit-learn/scikit-learn/issues/4324.

Упражнения

4.1. В главе 3 мы обозначили через `t1` ряд библиотеки pandas с временными штампами, когда было касание первого барьера и индексом был временной штамп наблюдения. Это было результатом функции `getEvents`.

(а) Вычислите ряд `t1` на долларовых барах, полученных из тиковых данных фьючерса E-mini S&P 500.

(б) Примените функцию `mpNumCoEvents` для вычисления числа накладывающихся исходов в каждый момент времени.

(в) Постройте график, в котором временной ряд числа одновременных меток будет расположен на основной оси, а временной ряд экспоненциально взвешенного скользящего среднеквадратического отклонения финансовых возвратов — на вторичной оси.

(г) Постройте диаграмму рассеяния числа одновременных меток (ось x) и экспоненциально взвешенного скользящего среднеквадратического отклонения финансовых возвратов (ось y). Можете ли вы оценить их взаимосвязь?

4.2. С помощью функции `mpSampleTW` вычислите среднюю уникальность каждой метки. Какова внутрирядовая корреляция первого порядка, AR(1), этого временного ряда? Является ли она статистически значимой? Почему?

4.3. Выполните подгонку случайного леса к совокупности финансовых данных, где $I^{-1}\sum_{i=1}^{I}\bar{u}_i \ll 1$.

(а) Каково среднее внепакетной правильности?

(б) Каково среднее правильности k-блочной перекрестной проверки (без перетасовки) на той же совокупности данных?

(в) Почему внепакетная правильность намного выше, чем перекрестно-проверочная точность? Какая из них более правильная/менее смещенная? Что является источником этого смещения?

4.4. Модифицируйте исходный код в разделе 4.7, чтобы применить экспоненциальный фактор временной эрозии.

4.5. Рассмотрите возможность применения метаметок к событиям, определяемым моделью следования за трендом. Предположим, что две трети меток равны 0, а одна треть равна 1.

(а) Что произойдет, если вы выполните подгонку классификатора без уравновешивания весов классов?

(б) Метка 1 означает истинное утверждение, и метка 0 означает ложное утверждение. Применяя сбалансированные веса классов, мы заставляем классификатор уделять больше внимания истинным утверждениям и меньше внимания ложным утверждениям. Почему это имеет смысл?

(в) Каково распределение предсказанных меток перед применением уравновешенных весов классов и после их применения?

4.6. Обновите вероятности взятия образцов для финального взятия из раздела 4.5.3.

4.7. Предположим, что при взятии второго образца (из раздела 4.5.3) число 2 выбрано снова. Каковы будут обновленные вероятности для третьего взятия образца?

5 Дробно-дифференцированные признаки

5.1. Актуальность

Общеизвестно, что вследствие действия арбитражных сил финансовые ряды демонстрируют низкие соотношения сигнал/шум (Lopez de Prado [2015]). Что еще хуже, стандартные стационарные преобразования, такие как целочисленное дифференцирование, еще больше уменьшают этот сигнал, удаляя память. Ценовые ряды имеют память, потому что каждое значение зависит от длинной истории предыдущих уровней. Напротив, целочисленные дифференцированные ряды, как и финансовые возвраты, имеют отсечение памяти в том смысле, что история полностью игнорируется за конечным окном выборки. После того как преобразования стационарности стерли всю память из данных, статистики прибегают к сложным математическим методам для извлечения остаточного сигнала, каким бы он ни был. Неудивительно, что применение этих сложных методов на ряде со стираемой памятью, по всей видимости, приводит к ложным открытиям. В этой главе мы вводим метод преобразования данных, который обеспечивает стационарность данных, сохраняя при этом как можно больше памяти.

5.2. Дилемма «стационарность или память»

В финансах часто встречаются нестационарные временные ряды. Эти ряды делает нестационарными наличие памяти, то есть долгая история предыдущих уровней, которые сдвигают среднее значение ряда во временной динамике. Для того чтобы провести инференциально-статистический анализ, исследователям необходимо работать с инвариантными процессами, такими как финансовые возвраты от цен (или изменения в логарифмических ценах), изменения в отдаче от инвестиций или изменения волатильности. Эти преобразования данных делают ряды стационарными за счет удаления всей памяти из исходного ряда (Alexander [2001], глава 11). Хотя стационарность является необходимым свойством для целей статистического вывода, в обработке сигналов редко возникает потреб-

ность в стирании всей памяти, так как эта память является основой для предсказательной силы модели. Например, для того чтобы сформировать прогноз, равновесным (стационарным) моделям требуется некоторая память для оценки того, насколько далеко процесс ценообразования отошел от долгосрочного математического ожидания. Дилемма заключается в том, что финансовые возвраты стационарны, но без памяти, а цены имеют память, но они не стационарны. Возникает вопрос: каким является минимальный объем дифференцирования, который делает ценовой ряд стационарным, сохраняя при этом как можно больше памяти? Соответственно, мы хотели бы обобщить понятие финансовых возвратов для того, чтобы учитывать *стационарные ряды, где стирается не вся память*. В рамках этой системы финансовые возвраты являются лишь одним из видов (и в большинстве случаев субоптимальным) ценовой трансформации среди многих других возможностей.

Частью важности коинтеграционных методов является их способность моделировать ряды с памятью. Но почему особый случай нулевой дифференциации дает наилучшие исходы? Нулевое дифференцирование так же произвольно, как и одношаговое дифференцирование. Между этими двумя крайностями имеется широкая область (полнодифференцированные ряды, с одной стороны, и нульдифференцированные ряды — с другой), которые могут быть разведаны с помощью дробного дифференцирования с целью разработки высокопредсказательной модели МО.

Контролируемые обучающиеся алгоритмы обычно требуют стационарных признаков. Причина в том, что нам нужно увязать ранее не встречавшееся (не промаркированное) наблюдение с коллекцией промаркированных примеров и вывести из них метку этого нового наблюдения. Если признаки не стационарны, то мы не можем увязать новое наблюдение с большим числом известных примеров. Однако стационарность не обеспечивает предсказательную силу. Стационарность является необходимым, но недостаточным условием для высокой результативности алгоритма МО. Проблема в том, что существует компромисс между стационарностью и памятью. Мы всегда можем сделать ряд более стационарным через дифференцирование, но это будет стоить стирания некоторой памяти, что нанесет поражение прогнозной цели алгоритма МО. В этой главе мы рассмотрим один из способов решения этой дилеммы.

5.3. Обзор публикаций

Практически вся литература по финансовым временным рядам основывается на предпосылке превращения нестационарных рядов в стационарные посредством целочисленного преобразования (для примера см. Hamilton [1994]). В связи с этим возникает два вопроса: 1) почему целочисленное дифференцирование (подобное тому, которое используется для вычисления финансовых возвратов от логарифмических цен) является оптимальным? 2) является ли чрезмерное дифференциро-

вание одной из причин, почему литература настолько смещена в пользу гипотезы эффективных рынков?

Понятие дробного дифференцирования, применяемое к предсказательному анализу временных рядов, восходит, по крайней мере, к публикации Hosking [1981]. В этой статье было обобщено семейство процессов ARIMA[1], позволившее порядку дифференцирования принимать дробные значения. Это было полезно, потому что дробно дифференцированные процессы проявляют долгосрочную устойчивость и антиустойчивость, таким образом повышая предсказательную силу по сравнению со стандартным подходом ARIMA. В той же публикации автор утверждает: «за исключением беглого упоминания Грейнджером (Granger, 1978), дробное дифференцирование, по-видимому, ранее не упоминалось в связи с анализом временных рядов».

После публикации Хоскинга литература по этому вопросу была удивительно скудной, составив до восьми журнальных статей, написанных всего девятью авторами: Хоскингом, Йохансеном, Нильсеном, МакКинноном, Дженсеном, Джонсом, Попиелем, Кавальером и Тейлором (Hosking, Johansen, Nielsen, MacKinnon, Jensen, Jones, Popiel, Cavaliere и Taylor). Обратитесь к ссылкам за подробностями. Большинство этих работ относятся к техническим вопросам, таким как быстрые алгоритмы вычисления дробного дифференцирования в непрерывных стохастических процессах (например, Jensen and Nielsen [2014]).

Дифференцирование стохастического процесса является дорогостоящей вычислительной операцией. В этой главе мы рассмотрим практический, альтернативный и новый подходы к восстановлению стационарности: обобщим разностный оператор на нецелочисленные шаги.

5.4. Метод

Рассмотрим оператор обратного сдвига B, примененный к матрице вещественных признаков $\{X_t\}$, где $B^k X_t = X_{t-k}$ для любого целого $k \geq 0$. К примеру, $(1 - B)^2 = 1 - 2B + B^2$, где $B^2 X_t = X_{t-2}$, отсюда $(1 - B)^2 X_t = X_t - 2X_{t-1} + X_{t-2}$. Заметим, что $(x + y)^n = \sum_{k=0}^{n} \binom{n}{k} x^k y^{n-k} = \sum_{k=0}^{n} \binom{n}{k} x^{n-k} y^k$, где n — положительное целое число. Для вещественного числа d, $(1 + x)^d = \sum_{k=0}^{\infty} \binom{d}{k} x^k$, биномиальный ряд.

В дробной модели показатель степени d может быть вещественным числом со следующим формальным разложением биномиального ряда:

[1] ARIMA (autoregressive integrated moving average) — авторегрессионное интегрированное скользящее среднее. — *Примеч. науч. ред.*

$$(1-B)^d = \sum_{k=0}^{\infty}\binom{d}{k}(-B)^k = \sum_{k=0}^{\infty}\frac{\prod_{i=0}^{k-1}(d-i)}{k!}(-B)^k$$
$$= \sum_{k=0}^{\infty}(-B)^k\prod_{i=0}^{k-1}\frac{d-i}{k-i}$$
$$= 1 - dB + \frac{d(d-1)}{2!}B^2 - \frac{d(d-1)(d-2)}{3!}B^3 + \ldots$$

5.4.1. Долгая память

Давайте посмотрим, как реальный (нецелочисленный) положительный d сохраняет память. Этот арифметический ряд состоит из скалярного произведения

$$\tilde{X}_t = \sum_{k=0}^{\infty}\omega_k X_{t-k}$$

с весовым коэффициентом ω

$$\omega = \left\{1, -d, \frac{d(d-1)}{2!}, \frac{d(d-1)(d-2)}{3!}, \ldots, (-1)^k\prod_{i=0}^{k-1}\frac{d-i}{k!}, \ldots\right\}$$

и значениями X

$$X = \{X_t, X_{t-1}, X_{t-2}, X_{t-3}, \ldots, X_{t-k}, \ldots\}.$$

Когда d — положительное целое число, $\Pi_{i=0}^{k-1}\frac{d-i}{k!} = 0, \forall k > d$, и память за этой точкой отменяется. Например, $d = 1$ используется для вычисления финансовых возвратов, где $\Pi_{i=0}^{k-1}\frac{d-i}{k!} = 0, \forall k > d$ и $\omega = \{1, -1, 0, 0, \ldots\}$.

5.4.2. Итеративное оценивание

С учетом последовательности весовых коэффициентов ω мы можем представить, что для $k = 0, \ldots, \infty$, при $\omega_0 = 1$, веса могут быть сгенерированы итеративно в следующем виде:

$$\omega_k = -\omega_{k-1}\frac{d-k+1}{k}.$$

На рис. 5.1 показана последовательность весов, используемых для вычисления каждого значения дробно дифференцированного ряда. Легенда сообщает значение d, используемое для генерирования каждой последовательности, ось x указывает на значение k, а ось y показывает на значение ω_k. Например, для $d = 0$ все веса равны 0, за исключением $\omega_0 = 1$. Это тот случай, когда дифференцированный ряд совпадает с исходным. Для $d = 1$ все веса равны 0, за исключением $\omega_0 = 1$ и $\omega_1 = -1$. Это стандартное целочисленное дифференцирование первого порядка, которое используется для получения логарифмически-ценовых финансовых возвратов.

В любом месте между этими двумя случаями все веса после $\omega_0 = 1$ отрицательны и больше -1.

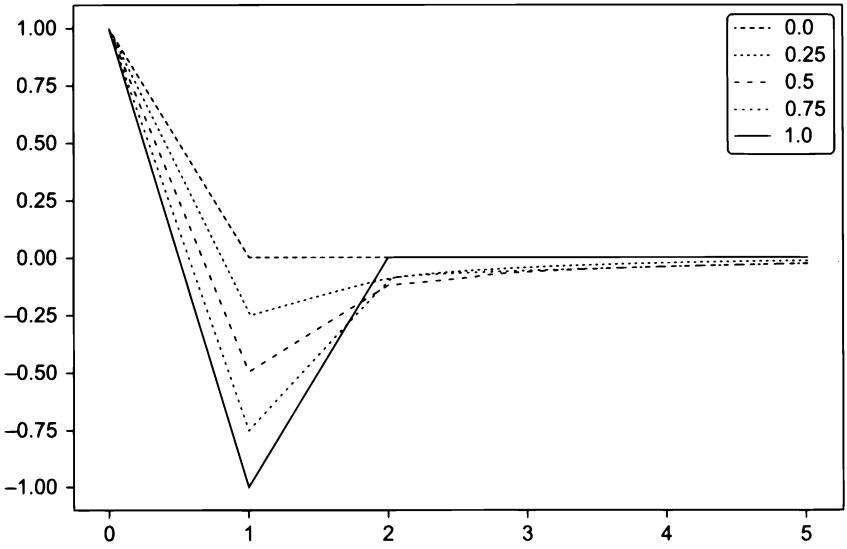


Рис. 5.1. ω_k (ось y) по мере увеличения k (ось x). Каждая линия относится к определенному значению $d \in [0, 1]$, с приростом 0.1

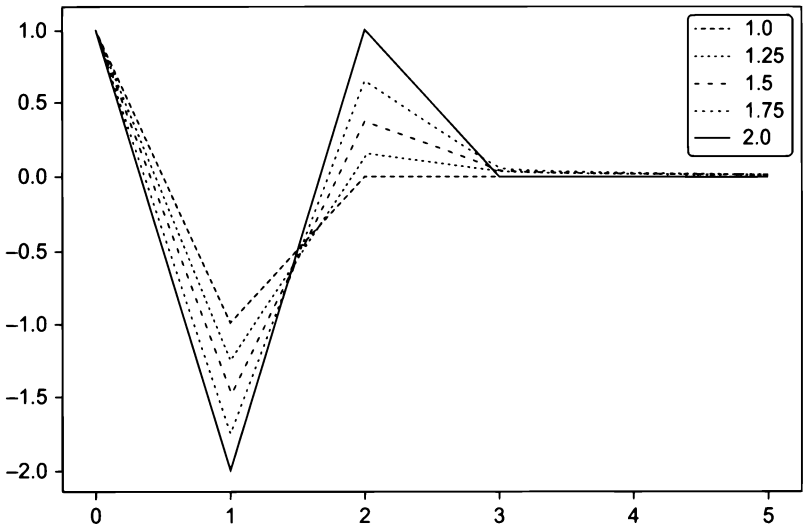


Рис. 5.2. ω_k (ось y) по мере увеличения k (ось x). Каждая линия относится к определенному значению $d \in [1, 2]$, с приростом 0.1

На рис. 5.2 показана последовательность весов, где $d \in [1, 2]$ с шагом 0.1. Для $d > 1$ мы наблюдаем $\omega_1 < -1$ и $\omega_k > 0$, $\forall k \geq 2$.

Листинг 5.1 содержит исходный код, используемый для генерирования приведенных графиков.

Листинг 5.1. Взвешивающая функция

```
def getWeights(d,size):
    # thres>0 устраняет незначительные веса
    w=[1.]
    for k in range(1,size):
        w_=-w[-1]/k*(d-k+1)
        w.append(w_)
    w=np.array(w[::-1]).reshape(-1,1)
    return w
#--------------------------------------
def plotWeights(dRange,nPlots,size):
    w=pd.DataFrame()
    for d in np.linspace(dRange[0],dRange[1],nPlots):
        w_=getWeights(d,size=size)
        w_=pd.DataFrame(w_,index=range(w_.shape[0])[::-1],columns=[d])
        w=w.join(w_,how='outer')
    ax=w.plot()
    ax.legend(loc='upper left');mpl.show()
    return
#--------------------------------------
if __name__=='__main__':
    plotWeights(dRange=[0,1],nPlots=11,size=6)
    plotWeights(dRange=[1,2],nPlots=11,size=6)
```

5.4.3. Сходимость

Рассмотрим сходимость весов. Из приведенного выше результата видно, что для $k > d$, если $\omega_{k-1} \neq 0$, то $\left|\frac{\omega_k}{\omega_{k-1}}\right| = \left|\frac{d-k+1}{k}\right| < 1$ и $\omega_k = 0$ в противном случае. Следовательно, веса асимптотически сходятся к нулю, как бесконечное произведение факторов внутри единичного круга. Кроме того, для положительных d и $k < d + 1$ мы имеем $\frac{d-k+1}{k} \geq 0$, что заставляет начальные веса чередоваться по знаку. Для нецелочисленного d, как только $k \geq d + 1$, ω_k будет отрицательным, если $\text{int}[d]$ — четное, и положительным в противном случае. Подводя итог, $\lim_{k\to\infty} \omega_k = 0^-$ (стремится к нулю слева), когда $\text{int}[d]$ — нечетное, и $\lim_{k\to\infty} \omega_k = 0^+$ (сходится к нулю справа), когда $\text{int}[d]$ — четное. В частном случае $d \in (0, 1)$ это означает, что $-1 < \omega_k < 0$, $\forall k > 0$. Такое чередование весовых признаков необходимо для того, чтобы сделать $\{\tilde{X}_t\}_{t=1,\ldots,T}$

стационарным, поскольку память ослабевает или нейтрализуется в долгосрочной перспективе.

5.5. Применение

В этом разделе мы рассмотрим две альтернативные реализации дробного дифференцирования: стандартный метод «расширяющегося окна» и новый метод, который я называю «дробным дифференцированием с окном фиксированной ширины» fracdiff (fixed-width window fracdiff, FFD).

5.5.1. Расширяющееся окно

Давайте обсудим, каким образом дробно дифференцируется (конечный) временной ряд на практике. Предположим, что у нас имеется временной ряд с T вещественными наблюдениями, $\{X_t\}$, $t = 1, ..., T$. Из-за ограничений данных дробно дифференцированное значение $\tilde{X}_T$ не может быть вычислено на бесконечном ряду весов. Например, последняя точка $\tilde{X}_T$ будет использовать веса $\{\omega_k\}$, $k = 0, ..., T - 1$, а $\tilde{X}_{T-l}$ будет использовать веса $\{\omega_k\}$, $k = 0, ..., T - l - 1$. Это означает, что начальные точки будут иметь разный объем памяти по сравнению с конечными точками. Для каждого l можно определить относительную потерю веса $\lambda_l = \frac{\sum_{j=T-l}^{T}|\omega_j|}{\sum_{i=0}^{T-1}|\omega_i|}$. Учитывая уровень допуска $\tau \in [0, 1]$, мы можем определить значение l^* такое, что $\lambda_{l^*} \leq \tau$ и $\lambda_{l^*+1} > \tau$. Это значение l^* соответствует первым результатам $\{\tilde{X}_t\}_{t=1,...,l^*}$, где потеря веса превышает допустимый порог, $\lambda_t > \tau$ (например, $\tau = 0.01$).

Из нашего предыдущего обсуждения ясно, что λ_{l^*} зависит от скорости схождения $\{\omega_k\}$, которая, в свою очередь, зависит от $d \in [0, 1]$. Для $d = 1$, $\omega_k = 0$, $\forall k > 1$ и $\lambda_1 = 0$, $\forall l > 1$, следовательно, достаточно отбросить $\tilde{X}_1$. По мере того как $d \to 0^+$, l^* увеличивается, и более крупная часть начального $\{\tilde{X}_t\}_{t=1,...,l^*}$ должна быть отброшена, чтобы оставить потерю веса $\lambda_{l^*} \leq \tau$. На рис. 5.3 строится график сделочных баров фьючерса E-mini S&P 500 размера 1E4, скорректированного вперед, дробно дифференцированного, с параметрами ($d = .4$, $\tau = 1$) вверху и параметрами ($d = .4$, $\tau = 1E-2$) внизу.

Отрицательный дрейф на обоих графиках вызван отрицательными весами, которые добавляются в исходные наблюдения по мере расширения окна. Когда мы не контролируем потерю веса, отрицательный дрейф является экстремальным, до такой степени, что видна только эта тенденция. Отрицательный дрейф несколько более умеренный в правом графике, после поправки на потерю веса, однако он по-прежнему существен, потому что значения $\{\tilde{X}_t\}_{t=l^*+1,...,T}$ вычисляются на расширяющемся окне. Эту проблему можно исправить с помощью окна фиксированной ширины, реализованного в листинге 5.2.

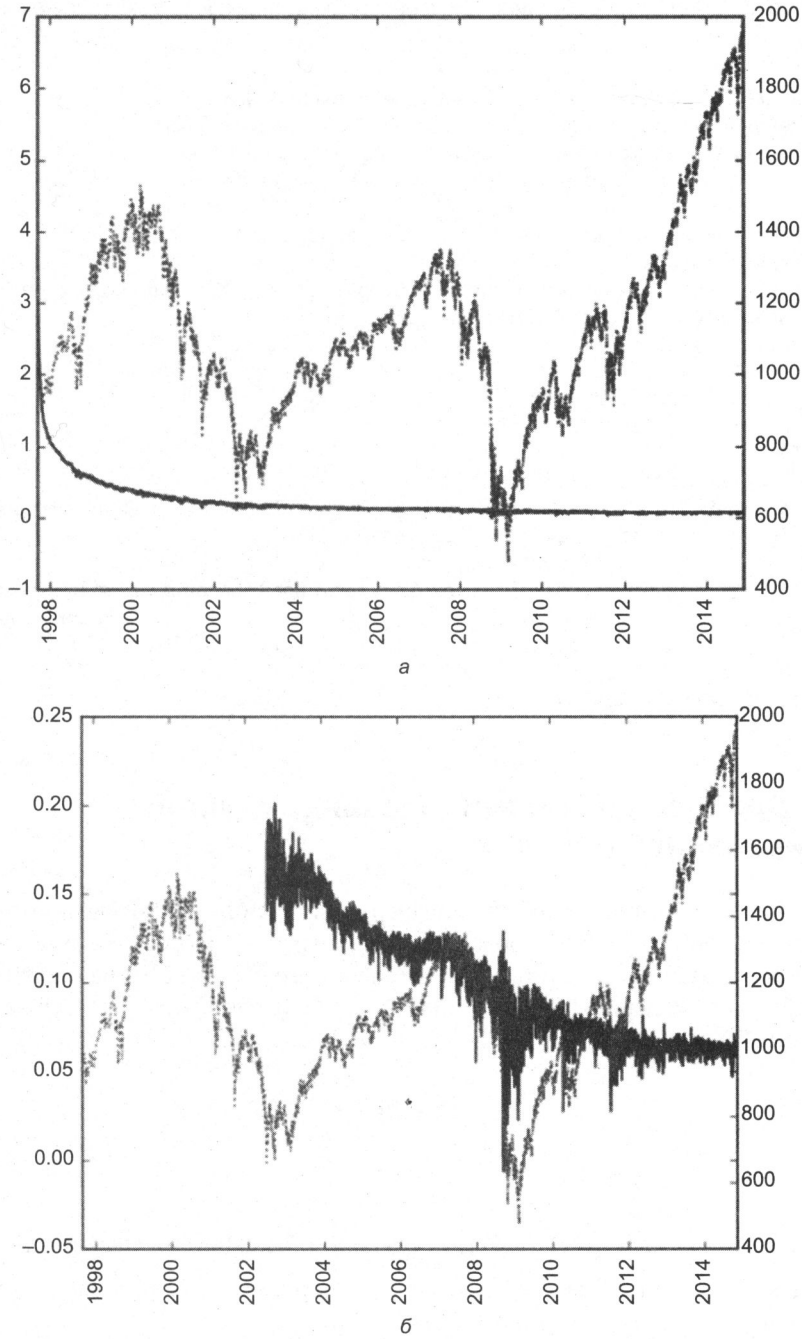


Рис. 5.3. Фракционная дифференциация без поправки на потерю веса (верхний график) и с поправкой на потерю веса с расширением окна (нижний график)

Листинг 5.2. Стандартное дробное дифференцирование (расширяющееся окно)

```
def fracDiff(series,d,thres=.01):
    '''
    Увеличение размера окна, с обработкой значений NaN
    Примечание 1: для порога thres=1 ничего не пропускается.
    Примечание 2: d может быть любым положительным дробным,
                  не обязательно ограниченным [0,1].
    '''
    #1) вычислить веса для самого длинного ряда
    w=getWeights(d,series.shape[0])
    #2) определить начальные расчеты, которые должны быть пропущены,
    # основываясь на пороге потери веса
    w_=np.cumsum(abs(w))
    w_/=w_[-1]
    skip=w_[w_>thres].shape[0]
    #3) применить веса к значениям
    df={}
    for name in series.columns:
        seriesF,df_=series[[name]].fillna(method='ffill').dropna(),pd.Series()
        for iloc in range(skip,seriesF.shape[0]):
            loc=seriesF.index[iloc]
            if not np.isfinite(series.loc[loc,name]): continue # исключить
                                                               # значения NA
            df_[loc]=np.dot(w[-(iloc+1):,:].T,seriesF.loc[:loc])[0,0]
        df[name]=df_.copy(deep=True)
    df=pd.concat(df,axis=1)
    return df
```

5.5.2. Дробное дифференцирование с окном фиксированной ширины

В качестве альтернативы дробное дифференцирование может быть вычислено с использованием окна фиксированной ширины, то есть отбрасывая веса после того, как их модуль ($|\omega_k|$) опустится ниже заданного порогового значения (τ). Это равносильно нахождению первого l^* такого, что $|\omega_{l^*}| \geq \tau$ и $|\omega_{l^*+1}| \leq \tau$, установив новую переменную $\tilde{\omega}_k$

$$\tilde{\omega}_k = \begin{cases} \omega_k & \text{если } k \leq l^* \\ 0 & \text{если } k > l^* \end{cases}$$

и $\tilde{X}_t = \sum_{k=0}^{l^*} \tilde{\omega}_k X_{t-k}$, для $t = T - l^* + 1, ..., T$. На рис. 5.4 строится график сделочных баров фьючерса E-mini S&P 500 размера 1E4, скорректированного вперед, дробно дифференцированного ($d = .4$, $\tau = 1\text{E}{-5}$).

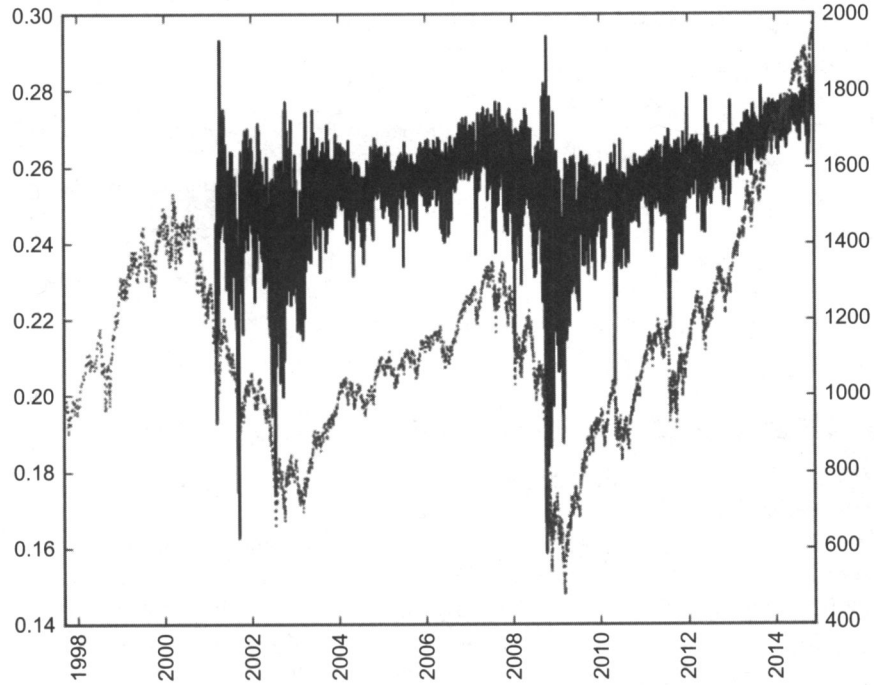


Рис. 5.4. Дробное дифференцирование с поправкой на потерю веса с окном фиксированной ширины

Эта процедура имеет то преимущество, что один и тот же вектор весов используется во всех оценках $\{\tilde{X}_t\}_{t=l^*,\ldots,T}$, что позволяет избежать отрицательного дрейфа, вызванного добавленными весами расширяющегося окна. Результатом является бездрейфовое сочетание уровня плюс шум, как и ожидалось. Распределение больше не гауссово, в результате асимметрии и избыточного эксцесса, которые приходят вместе с памятью, однако оно стационарно. Листинг 5.3 представляет реализацию этой идеи.

Листинг 5.3. Новый метод дробного дифференцирования с окном фиксированной ширины

```
def getWeights_FFD(d,thres):
    w,k=[1.],1
    while True:
        w_=-w[-1]/k*(d-k+1)
        if abs(w_)<thres: break
        w.append(w_);k+=1
    return np.array(w[::-1]).reshape(-1,1)
#--------------------------------------
```

```
def fracDiff_FFD(series,d,thres=1e-5):
    # Окно с постоянной шириной (новое решение)
    w,width,df=getWeights_FFD(d,thres),len(w)-1,{}
    for name in series.columns:
        seriesF,df_=series[[name]].fillna(method='ffill').dropna(),pd.Series()
        for iloc1 in range(width,seriesF.shape[0]):
            loc0,loc1=seriesF.index[iloc1-width],seriesF.index[iloc1]
            if not np.isfinite(series.loc[loc1,name]): continue # исключить
                                                                # значения NA
            df_[loc1]=np.dot(w.T,seriesF.loc[loc0:loc1])[0,0]
        df[name]=df_.copy(deep=True)
    df=pd.concat(df,axis=1)
    return df
```

5.6. Стационарность с максимальным сохранением памяти

Рассмотрим ряд $\{X_t\}_{t=1,\ldots,T}$. Применив метод дробного дифференцирования fracdiff с окном фиксированной ширины (FFD) к этому ряду, мы можем вычислить минимальный коэффициент d^* такой, что результирующий дробно-дифференцированный ряд $\{X_t\}_{t=l^*,\ldots,T}$ является стационарным. Этот коэффициент d^* квантифицирует объем памяти, который необходимо удалить для достижения стационарности. Если $\{X_t\}_{t=l^*,\ldots,T}$ уже стационарен, то $d^* = 0$. Если $\{X_t\}_{t=l^*,\ldots,T}$ содержит единичный корень, то $d^* < 1$. Если $\{X_t\}_{t=l^*,\ldots,T}$ проявляет взрывное поведение (как в пузыре), то $d^* > 1$. Особый интерес представляет случай $0 < d^* \ll 1$, когда исходный ряд «слегка нестационарен». В этом случае, хотя дифференцирование и необходимо, полное целочисленное дифференцирование удаляет избыточную память (и предсказательную силу).

Рисунок 5.5 иллюстрирует эту идею. На правой оси y он строит статистический показатель ADF, вычисленный на логарифмических ценах фьючерса E-mini S&P 500, перенесенных вперед с использованием трюка ETF (см. главу 2), отобранных с понижением до суточной частоты, возвращаясь к началу контракта. На оси x график показывает значение d, используемое для генерирования ряда, на котором был вычислен показатель ADF. Оригинальный ряд имеет показатель ADF, равный −0.3387, в то время как ряд с финансовыми возвратами имеет показатель ADF, равный −46.9114. При 95 %-ном уровне достоверности критическое значение статистической проверки составляет −2.8623. Статистический показатель ADF пересекает этот порог в области $d = 0.35$. Левая ось y строит корреляцию между исходным рядом ($d = 0$) и дифференцированным рядом при различных значениях d. При $d = 0.35$ корреляция по-прежнему очень высока, на уровне 0.995. Это подтверждает, что представленная в этой главе процедура была успешной в достижении стационарности, не отказываясь от слишком большого объема памяти. Напротив, корреляция между исходным рядом и рядом с финансовыми возвратами

составляет всего 0.03, таким образом показывая, что стандартное целочисленное дифференцирование почти полностью стирает память ряда.

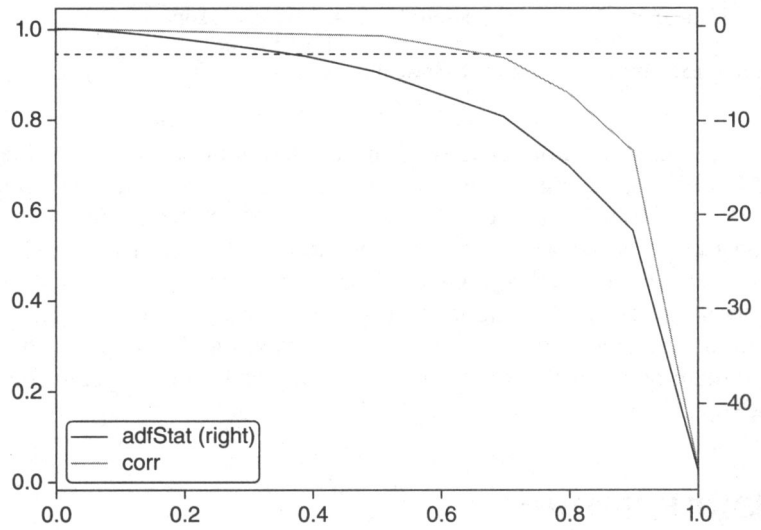


Рис. 5.5. Проверочный статистический показатель ADF как функция от d, на логарифмических ценах фьючерса E-mini S&P 500

Практически все финансовые публикации пытаются восстановить стационарность, применяя целочисленное дифференцирование $d = 1 \gg 0.35$, а значит, в большинстве исследований ряд чрезмерно дифференцирован, то есть из него удаляется гораздо больше памяти, чем было необходимо для удовлетворения стандартных эконометрических допущений. Листинг 5.4 содержит исходный код, используемый для получения этих результатов.

Листинг 5.4. Нахождение минимального значения d, которое успешно проходит статистическую проверку ADF

```
def plotMinFFD():
    from statsmodels.tsa.stattools import adfuller
    path,instName='./','ES1_Index_Method12'
    out=pd.DataFrame(columns=['adfStat','pVal','lags','nObs','95%
                              conf','corr'])
    df0=pd.read_csv(path+instName+'.csv',index_col=0,parse_dates=True)
    for d in np.linspace(0,1,11):
        df1=np.log(df0[['Close']]).resample('1D').last() # понизить до
                                                         # суточных наблюдений
        df2=fracDiff_FFD(df1,d,thres=.01)
        corr=np.corrcoef(df1.loc[df2.index,'Close'],df2['Close'])[0,1]
        df2=adfuller(df2['Close'],maxlag=1,regression='c',autolag=None)
```

```
        out.loc[d]=list(df2[:4])+[df2[4]['5%']]+[corr] # с критическим
                                                       # значением
    out.to_csv(path+instName+'_testMinFFD.csv')
    out[['adfStat','corr']].plot(secondary_y='adfStat')
    mpl.axhline(out['95% conf'].mean(),linewidth=1,color=
                                'r',linestyle='dotted')
    mpl.savefig(path+instName+'_testMinFFD.png')
    return
```

Пример с фьючерсным контрактом E-mini отнюдь не является исключением. В табл. 5.1 приведены статистические показатели проверки ADF после применения метода FFD(d) на различных значениях d для 87 наиболее ликвидных фьючерсов в мире. Во всех случаях стандартный $d = 1$, используемый для вычисления финансовых возвратов, подразумевает чрезмерное дифференцирование. Фактически во всех случаях стационарность достигается при $d < 0.6$. В некоторых случаях, таких как товарные фьючерсы на апельсиновый сок (JO1 Comdty) или живой крупный рогатый скот (LC1 Comdty), в дифференцировании вообще не было необходимости.

5.7. Заключение

Подводя итоги, стоит отметить, что большинство эконометрических процедур анализа следуют одной из двух парадигм:

1. Бокс—Дженкинс: финансовые возвраты стационарны, но без памяти.
2. Энгель—Грейнджер: логарифмы цен имеют память, но они нестационарны. Коинтеграция — это прием, который побуждает регрессию работать на нестационарных рядах для сохранения памяти. Однако число коинтегрированных переменных ограничено, и коинтегрирующие векторы печально известны своей нестабильностью.

Напротив, представленный в этой главе подход на основе метода дробного дифференцирования с окном фиксированной ширины (FFD) показывает, что для того чтобы получить стационарность, нет необходимости отказываться от всей памяти. И нет необходимости в коинтеграционном подходе, что касается прогнозирования МО. Как только вы познакомитесь с методом FFD, это позволит вам достигать стационарности, не отказываясь от памяти (или предсказательной силы).

На практике я предлагаю вам поэкспериментировать со следующим преобразованием ваших признаков: сначала вычислите кумулятивную сумму временного ряда. Это будет гарантировать, что необходим некоторый порядок дифференцирования. Во-вторых, вычислите ряд FFD(d) для различных $d \in [0, 1]$. В-третьих, определите минимум d такой, что p-значение статистического показателя проверки ADF на FFD(d) опускается ниже 5 %. В-четвертых, используйте этот ряд FFD(d) в качестве предсказательного признака.

Таблица 5.1. Статистический показатель проверки ADF на методе FFD(d) для нескольких наиболее ликвидных фьючерсных контрактов

	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
AD1 Curncy	−1.7253	−1.8665	−2.2801	−2.9743	−3.9590	−5.4450	−7.7387	−10.3412	−15.7255	−22.5170	−43.8281
BO1 Comdty	−0.7039	−1.0021	−1.5848	−2.4038	−3.4284	−4.8916	−7.0604	−9.5089	−14.4065	−20.4393	−38.0683
BP1 Curncy	−1.0573	−1.4963	−2.3223	−3.4641	−4.8976	−6.9157	−9.8833	−13.1575	−19.4238	−26.6320	−43.3284
BTS1 Comdty	−1.7987	−2.1428	−2.7600	−3.7019	−4.8522	−6.2412	−7.8115	−9.4645	−11.0334	-12.4470	−13.6410
BZ1 Index	−1.6569	−1.8766	−2.3948	−3.2145	−4.2821	−5.9431	−8.3329	−10.9046	−15.7006	−20.7224	−29.9510
C 1 Comdty	−1.7870	−2.1273	−2.9539	−4.1642	−5.7307	−7.9577	−11.1798	−14.6946	−20.9925	−27.6602	−39.3576
CC1 Comdty	−2.3743	−2.9503	−4.1694	−5.8997	−8.0868	−10.9871	−14.8206	−18.6154	−24.1738	−29.0285	−34.8580
CD1 Curncy	−1.6304	−2.0557	2.2784	3.8380	−5.2341	−7.3172	−10.3738	−13.8263	−20.2897	−27.6242	−43.6794
CF1 Index	−1.5539	−1.9387	−2.7421	−3.9235	−5.5085	−7.7585	−11.0571	−14.6829	−21.4877	−28.9810	−44.5059
CL1 Comdty	−0.3795	−0.7164	−1.3359	−2.2018	−3.2603	−4.7499	−6.9504	−9.4531	−14.4936	−20.8392	−41.1169
CN1 Comdty	−0.8798	−0.8711	−1.1020	−1.4626	−1.9732	−2.7508	−3.9217	−5.2944	−8.4257	−12.7300	−42.1411
CO1 Comdty	−0.5124	−0.8468	−1.4247	−2.2402	−3.2566	−4.7022	−6.8601	−9.2836	−14.1511	−20.2313	−39.2207
CT1 Comdty	−1.7604	−2.0728	−2.7529	−3.7853	−5.1397	−7.1123	−10.0137	−13.1851	−19.0603	−25.4513	−37.5703
DM1 Index	−0.1929	−0.5718	−1.2414	−2.1127	−3.1765	−4.6695	−6.8852	−9.4219	−14.6726	−21.5411	−49.2663
DU1 Comdty	−0.3365	−0.4572	−0.7647	−1.1447	−1.6132	−2.2759	−3.3389	−4.5689	−7.2101	−10.9025	−42.9012
DX1 Curncy	−1.5768	−1.9458	−2.7358	−3.8423	−5.3101	−7.3507	−10.3569	−13.6451	−19.5832	−25.8907	−37.2623
EC1 Comdty	−0.2727	−0.6650	−1.3359	−2.2112	−3.3112	−4.8320	−7.0777	−9.6299	−14.8258	−21.4634	−44.6452
EC1 Curncy	−1.4733	−1.9344	−2.8507	−4.1588	−5.8240	−8.1834	−11.6278	−15.4095	−22.4317	−30.1482	−45.6373
ED1 Comdty	−0.4084	−0.5350	−0.7948	−1.1772	−1.6633	−2.3818	−3.4601	−4.7041	−7.4373	−11.3175	−46.4487
EE1 Curncy	−1.2100	−1.6378	−2.4216	−3.5470	−4.9821	−7.0166	−9.9962	−13.2920	−19.5047	−26.5158	−41.4672
EO1 Comdty	−0.7903	−0.8917	−1.0551	−1.3465	−1.7302	−2.3500	−3.3068	−4.5136	−7.0157	−10.6463	−45.2100
EO1 Index	−0.6561	−1.0567	−1.7409	−2.6774	−3.8543	−5.5096	−7.9133	−10.5674	−15.6442	−21.3066	−35.1397
ER1 Comdty	−0.1970	−0.3442	−0.6334	−1.0363	−1.5327	−2.2378	−3.2819	−4.4647	−7.1031	−10.7389	−40.0407

Таблица 5.1 (*окончание*)

	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
ES1 Index	−0.3387	−0.7206	−1.3324	−2.2252	−3.2733	−4.7976	−7.0436	−9.6095	−14.8624	−21.6177	−46.9114
FA1 Index	−0.5292	−0.8526	−1.4250	−2.2359	−3.2500	−4.6902	−6.8272	−9.2410	−14.1664	−20.3733	−41.9705
FC1 Comdty	−1.8846	−2.1853	−2.8808	−3.8546	−5.1483	−7.0226	−9.6889	−12.5679	−17.8160	−23.0530	−31.6503
FV1 Comdty	−0.7257	−0.8515	−1.0596	−1.4304	−1.8312	−2.5302	−3.6296	−4.9499	−7.8292	−12.0467	−49.1508
G1 Comdty	0.2326	0.0026	−0.4686	−1.0590	−1.7453	−2.6761	−4.0336	−5.5624	−8.8575	−13.3277	−42.9177
GC1 Comdty	−2.2221	−2.3544	−2.7467	−3.4140	−4.4861	−6.0632	−8.4803	−11.2152	−16.7111	−23.1750	−39.0715
GX1 Index	−1.5418	−1.7749	−2.4666	−3.4417	−4.7321	−6.6155	−9.3667	−12.5240	−18.6291	−25.8116	−43.3610
HG1 Comdty	−1.7372	−2.1495	−2.8323	−3.9090	−5.3257	−7.3805	−10.4121	−13.7669	−19.8902	−26.5819	−39.3267
HI1 Index	−1.8289	−2.0432	−2.6203	−3.5233	−4.7514	−6.5743	−9.2733	−12.3722	−18.5308	−25.9762	−45.3396
HO1 Comdty	−1.6024	−1.9941	−2.6619	−3.7131	−5.1772	−7.2468	−10.3326	−13.6745	−19.9728	−26.9772	−40.9824
IB1 Index	−2.3912	−2.8254	−3.5813	−4.8774	−6.5884	−9.0665	−12.7381	−16.6706	−23.6752	−30.7986	−43.0687
IK1 Comdty	−1.7373	−2.3000	−2.7764	−3.7101	−4.8686	−6.3504	−8.2195	−9.8636	−11.7882	−13.3983	−14.8391
IR1 Comdty	−2.0622	−2.4188	−3.1736	−4.3178	−5.8119	−7.9816	−11.2102	−14.7956	−21.6158	−29.4555	−46.2683
JA1 Comdty	−2.4701	−2.7292	−3.3925	−4.4658	−5.9236	−8.0270	−11.2082	−14.7198	−21.2681	−28.4380	−42.1937
JB1 Comdty	−0.2081	−0.4319	−0.8490	−1.4289	−2.1160	−3.0932	−4.5740	−6.3061	−9.9454	−15.0151	−47.6037
JE1 Curncy	−0.9268	−1.2078	−1.7565	−2.5398	−3.5545	−5.0270	−7.2096	−9.6808	−14.6271	−20.7168	−37.6954
JG1 Comdty	−1.7468	−1.8071	−2.0654	−2.5447	−3.2237	−4.3418	−6.0690	−8.0537	−12.3908	−18.1881	−44.2884
JO1 Comdty	−3.0052	−3.3099	−4.2639	−5.7291	−7.5686	−10.1683	−13.7068	−17.3054	−22.7853	−27.7011	−33.4658
JY1 Curncy	−1.2616	−1.5891	−2.2042	−3.1407	−4.3715	−6.1600	−8.8261	−11.8449	−17.8275	−25.0700	−44.8394
KC1 Comdty	−0.7786	−1.1172	−1.7723	−2.7185	−3.8875	−5.5651	−8.0217	−10.7422	−15.9423	−21.8651	−35.3354
L 1 Comdty	−0.0805	−0.2228	−0.6144	−1.0751	−1.6335	−2.4186	−3.5676	−4.8749	−7.7528	−11.7669	−44.0349

На 95 %-ном уровне достоверности критическое значение статистической проверки ADF равняется −2.8623. Все ряды логарифмических цен достигают стационарности при $d < 0.6$, а подавляющее большинство стационарны при $d < 0.3$.

Упражнения

5.1. Сгенерируйте временной ряд из одинаково распределенного взаимно независимого случайного гауссова процесса. У вас получится не обладающий памятью стационарный ряд.

(а) Вычислите статический показатель проверки ADF в этом ряду. Каким будет p-значение?

(б) Вычислите кумулятивную сумму наблюдений. У вас получится не обладающий памятью нестационарный ряд.

i) Каким будет порядок интегрирования этого кумулятивного ряда?

ii) Вычислите статический показатель проверки ADF в этом ряду. Каким будет p-значение?

(в) Продифференцируйте ряд дважды. Каким будет p-значение этого избыточно продифференцированного ряда?

5.2. Сгенерируйте временной ряд, который подчиняется синусоидальной функции. У вас получится обладающий памятью стационарный ряд.

(а) Вычислите статический показатель проверки ADF в этом ряду. Каким будет p-значение?

(б) Сдвиньте каждое наблюдение на одинаковую положительную величину. Вычислите кумулятивную сумму наблюдений. У вас получится обладающий памятью нестационарный ряд.

i) Рассчитайте результаты ADF-теста этих рядов. Определите значение p.

ii) Примените дробное дифференцирование с расширяющимся окном, с $\tau = 1E{-}2$. При каком минимальном значении d вы получите p-значение ниже 5 %?

iii) Примените дробное дифференцирование с фиксированным окном (FFD), с $\tau = 1E{-}5$. При каком минимальном значении d вы получите p-значение ниже 5 %?

5.3. Возьмите ряд из упражнения 5.2.б.

(а) Выполните подгонку этого ряда к синусоидальной функции. Каким будет R-квадрат?

(б) Примените FFD($d = 1$). Выполните подгонку этого ряда к синусоидальной функции. Каким будет R-квадрат?

(в) Какое значение d максимизирует R-квадрат синусоидальной подгонки на FFD(d)? Почему?

5.4. Возьмите долларовый барный ряд на фьючерсном контракте E-mini S&P 500. Используя исходный код листинга 5.3, для некоторых $d \in [0, 2]$ вычислите `fracDiff_FFD(fracDiff_FFD(series,d),-d)`. Что вы получите? Почему?

5.5. Возьмите долларовый барный ряд на фьючерсном контракте E-mini S&P 500.

(а) Сформируйте новый ряд как кумулятивную сумму логарифмов цен.

(б) Примените FFD с $\tau = 1E{-}5$. Определите, для какого минимального $d \in [0, 2]$ новый ряд будет стационарным.

(в) Вычислите корреляцию дробно дифференцированного ряда с исходным (нетрансформированным) рядом.

(г) Примените коинтеграционную проверку Энгеля—Грейнджера на оригинальном и дробно дифференцированном рядах.

(д) Примените проверку нормальности Харке—Бера на дробно дифференцированном ряде.

5.6. Возьмите дробно дифференцированный ряд из упражнения 5.5.

(а) Примените фильтр CUSUM (глава 2), где h — это удвоенное среднеквадратическое отклонение ряда.

(б) Используйте отфильтрованные временные штампы для отбора из признаковой матрицы. Используйте в качестве одного из признаков значение дробного дифференцирования fracdiff.

(в) Сформируйте метки с помощью тройного барьерного метода с симметричными горизонтальными барьерами из удвоенного суточного среднеквадратического отклонения и вертикальным барьером из 5 дней.

(г) Выполните подгонку бэггированного классификатора деревьев решения, где:

i) наблюдаемые признаки бутстрапируются с помощью последовательного метода из главы 4;

ii) на каждой бутстрапированной выборке веса выборки определяются с помощью методов из главы 4.

Часть 2
МОДЕЛИРОВАНИЕ

6 Ансамблевые методы

6.1. Актуальность

В этой главе мы обсудим два самых популярных ансамблевых метода машинного обучения[1]. В справочниках и сносках вы найдете книги и статьи, знакомящие с этими методами. Как и везде в этой книге, предполагается, что эти подходы вы уже использовали. Цель этой главы — объяснить, что делает их эффективными и как избежать распространенных ошибок, которые приводят к их неправильному использованию в финансах.

6.2. Три источника ошибок

Модели МО обычно страдают от трех ошибок[2].

1. **Смещение**: данная ошибка вызвана нереалистичными допущениями. Когда смещение высоко, это означает, что алгоритм МО не смог распознать важные связи между признаками и исходами. В этой ситуации говорят, что алгоритм «недоподогнан».
2. **Дисперсия**: данная ошибка вызвана чувствительностью к малым изменениям в тренировочном подмножестве. Когда дисперсия высока, это означает, что алгоритм переподогнан к тренировочному подмножеству, и поэтому даже минимальные изменения в тренировочном подмножестве могут дать ужасно разнящиеся предсказания. Вместо того чтобы моделировать общие закономерности в тренировочном подмножестве, алгоритм ошибочно принимает шум за сигнал.

---

[1] Для ознакомления с ансамблевыми методами, пожалуйста, посетите: http://scikit-learn.org/stable/modules/ensemble.html.

[2] Я обычно не цитирую «Википедию», однако по этому вопросу пользователь может найти некоторые иллюстрации в этой статье полезными: https://en.wikipedia.org/wiki/Bias%E2%80%93variance_tradeoff (https://wikipedia.org/wiki/Дилемма_смещения-дисперсии).

3. **Шум**: Данная ошибка вызвана дисперсией наблюдаемых значений, как, например, непредсказуемые изменения или ошибки замеров. Это неустранимая ошибка, которая не может быть объяснена ни одной моделью.

Рассмотрим тренировочное подмножество наблюдений $\{x_i\}_{i=1,\ldots,n}$ и вещественные исходы $\{y_i\}_{i=1,\ldots,n}$. Предположим, что существует функция $f[x]$ такая, что $y = f[x] + \varepsilon$, где ε — это белый шум с $E[\varepsilon_i] = 0$ и $E[\varepsilon_i^2] = \sigma_\varepsilon^2$. Мы хотели бы оценить функцию $\hat{f}[x]$, которая ближе всего соответствует $f[x]$ в смысле минимизации дисперсии ошибки оценивания $E(y_i - \hat{f}[x_i])^2]$ (среднеквадратическая ошибка не может равняться нулю из-за шума, который представлен как σ_ε^2). Эта среднеквадратическая ошибка может быть разложена на

$$E\left[(y_i - \hat{f}[x_i])^2\right] = \left(\underbrace{E[\hat{f}[x_i] - f[x_i]]}_{\text{смещение}}\right)^2 + \underbrace{V[\hat{f}[x_i]]}_{\text{дисперсия}} + \underbrace{\sigma_\varepsilon^2}_{\text{шум}}.$$

Ансамблевый метод — это метод, который совмещает множество слабых учеников, которые основаны на одном и том же обучающемся алгоритме, с целью создания (более сильного) ученика, чья результативность лучше, чем у любого из отдельно взятых учеников. Ансамблевые методы помогают уменьшить смещение и/или дисперсию.

6.3. Агрегация бутстрапов

Бэггирование (пакетирование), или агрегирование бутстраповских выборок, — это эффективный способ сокращения дисперсии в прогнозах. Оно работает следующим образом: во-первых, надо сгенерировать N тренировочных подмножеств данных с помощью случайного отбора *с возвратом*. Во-вторых, выполнить подгонку N оценщиков[1], по одному на каждое тренировочное подмножество. Эти оценщики подгоняются независимо друг от друга, следовательно, модели могут быть подогнаны параллельно. В-третьих, ансамблевый прогноз — это *простое* среднее арифметическое индивидуальных прогнозов из N моделей. В случае категориальных переменных вероятность того, что наблюдение принадлежит классу, определяется долей оценщиков, классифицирующих это наблюдение как член этого класса (мажоритарным голосованием, то есть большинством голосов). Когда базовый оценщик может делать прогнозы с вероятностью предсказания, бэггированный классификатор может получать среднее значение вероятностей.

[1] Оценщик (estimator) — это правило (формула) для расчета оценки заданной величины на основе наблюдаемых данных: таким образом, различаются правило (оценщик), интересующая величина (эстиманд) и его результат (оценка). См. https://en.wikipedia.org/wiki/Estimator. — *Примеч. науч. ред.*

Если для вычисления внепакетной точности вы используете класс `BaggingClassifier` библиотеки sklearn, то вы должны знать об этом дефекте: https://github.com/scikit-learn/scikit-learn/issues/8933. Один из обходных путей состоит в переименовании меток в целочисленном последовательном порядке.

6.3.1. Сокращение дисперсии

Главное преимущество бэггирования заключается в том, что оно уменьшает дисперсию прогнозов, тем самым помогая решать проблему переподгонки. Дисперсия в бэггированном предсказании ($\varphi_i[c]$) является функцией числа бэгированных оценщиков (N), средней дисперсии предсказания, выполняемого одним оценщиком ($\bar{\sigma}$), и средней корреляции между их прогнозами ($\bar{\rho}$):

$$V\left[\frac{1}{N}\sum_{i=1}^{N}\varphi_i[c]\right]=\frac{1}{N^2}\sum_{i=1}^{N}\left(\sum_{j=1}^{N}\sigma_{i,j}\right)=\frac{1}{N^2}\sum_{i=1}^{N}\left(\sigma_i^2+\sum_{j\neq i}^{N}\sigma_i\sigma_j\rho_{i,j}\right)=$$

$$=\frac{1}{N^2}\sum_{i=1}^{N}\left(\bar{\sigma}^2+\underbrace{\sum_{j\neq i}^{N}\bar{\sigma}^2\bar{\rho}}_{\substack{(N-1)\bar{\sigma}^2\bar{\rho}\\ \text{для фикси-}\\ \text{рованного } i}}\right)=\frac{\bar{\sigma}^2+(N-1)\bar{\sigma}^2\bar{\rho}}{N}=$$

$$=\bar{\sigma}^2\left(\bar{\rho}+\frac{1-\bar{\rho}}{N}\right),$$

где $\sigma_{i,j}$ — это ковариация[1] предсказаний по оценщикам i, j;

$$\sum_{i=1}^{N}\bar{\sigma}^2=\sum_{i=1}^{N}\sigma_i^2\Leftrightarrow\bar{\sigma}^2=N^{-1}\sum_{i=1}^{N}\sigma_i^2;$$

$$\text{и}\ \sum_{j\neq i}^{N}\bar{\sigma}^2\bar{\rho}=\sum_{j\neq i}^{N}\sigma_i\sigma_j\rho_{i,j}\Leftrightarrow\bar{\rho}=(\bar{\sigma}^2N(N-1))^{-1}\sum_{j\neq i}^{N}\sigma_i\sigma_j\rho_{i,j}.$$

Приведенное выше уравнение показывает, что бэггирование эффективно только в той мере, в какой $\bar{\rho}<1$; в то время как $\bar{\rho}\to1\Rightarrow V[\frac{1}{N}\sum_{i=1}^{N}\varphi_i[c]]\to\bar{\sigma}^2$. Одна из целей последовательного бутстрапирования (глава 4) заключается в том, чтобы производить отборы образцов как можно более независимыми, тем самым уменьшая $\bar{\rho}$, что должно снизить дисперсию бэггированных классификаторов. На рис. 6.1 по-

[1] Ковариация (covariance) по исходному названию и по определению — это совместная дисперсия, или содисперсия. Следовательно, и название соответствующей матрицы будет «содисперсная матрица», а не ковариационная. Такой унифицирующий подход помогает не потеряться в терминологии. — *Примеч. науч. ред.*

строена диаграмма среднеквадратического отклонения бэггированного предсказания как функции $N \in [5, 30]$, $\bar{\rho} \in [0, 1]$ и $\bar{\sigma} = 1$.

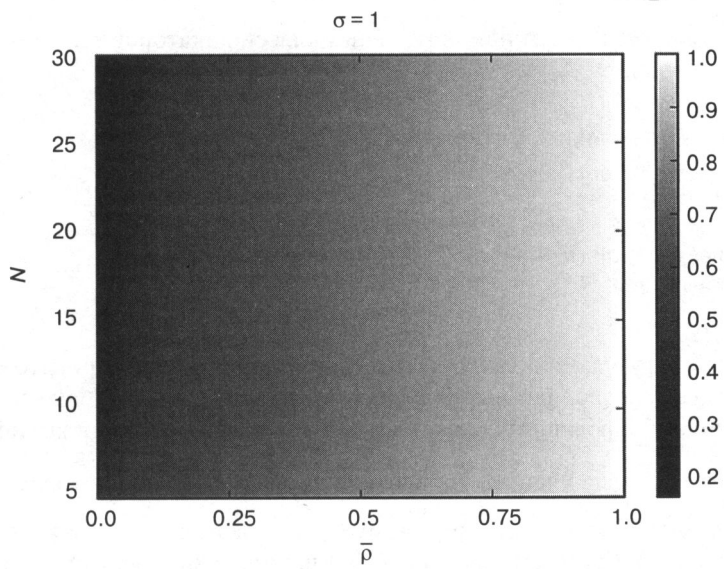


Рис. 6.1. Стандартное отклонение бэггированного предсказания

6.3.2. Улучшенная точность

Рассмотрим бэггированный классификатор, который делает предсказание на k классах мажоритарным голосованием среди N независимых классификаторов. Мы можем обозначить предсказания как {0,1}, где 1 означает правильное предсказание. Точность классификатора — это вероятность p промаркировать предсказание как 1. В среднем мы получим N_p предсказаний, промаркированных как 1, с дисперсией $Np(1 - p)$. Мажоритарное голосование делает правильное предсказание, когда наблюдается самый прогнозируемый класс. Например, для $N = 10$ и $k = 3$ бэггированный классификатор сделал правильное предсказание, когда наблюдался класс A, и отданные голоса были $[A, B, C] = [4, 3, 3]$. Однако бэггированный классификатор сделал неправильное предсказание, когда наблюдался класс A, и отданные голоса были $[A, B, C] = [4, 1, 5]$. Достаточным условием является то, что сумма этих меток равна $X > \frac{N}{2}$. Необходимым (но недостаточным) условием является то, что $X > \frac{N}{k}$, которое происходит с вероятностью

$$\mathrm{P}\left[X > \frac{N}{k}\right] = 1 - \mathrm{P}\left[X \leq \frac{N}{k}\right] = 1 - \sum_{i=0}^{\left\lfloor N/k \right\rfloor} \binom{N}{i} p^i (1-p)^{N-i}.$$

Из этого вытекает, что для достаточно большого N, скажем $N > p(p - 1/k)^{-2}$, $p > \frac{1}{k} \Rightarrow$ $\Rightarrow P[X > \frac{N}{k}] > p$, следовательно, точность бэггированного классификатора превышает среднюю точность индивидуальных классификаторов. Листинг 6.1 реализует это вычисление.

Листинг 6.1. Правильность бэггированного классификатора

```
from scipy.misc import comb
N,p,k=100,1./3,3.
p_=0
for i in xrange(0,int(N/k)+1):
    p_+=comb(N,i)*p**i*(1-p)**(N-i)
print p,1-p_
```

Это сильный аргумент в пользу бэггирования любого классификатора в общем случае, когда это позволяют вычислительные возможности. Однако, в отличие от бустирования, бэггирование не может улучшить точность слабых классификаторов: если индивидуальные ученики являются слабыми классификаторами ($p \ll \frac{1}{k}$), мажоритарное голосование по-прежнему будет показывать слабую результативность (хотя и с более низкой дисперсией). Рисунок 6.2 иллюстрирует эти факты. Поскольку легче добиться $\bar{\rho} \ll 1$, чем $p > \frac{1}{k}$, бэггирование имеет больше шансов быть успешным в сокращении дисперсии, чем в сокращении смещения.

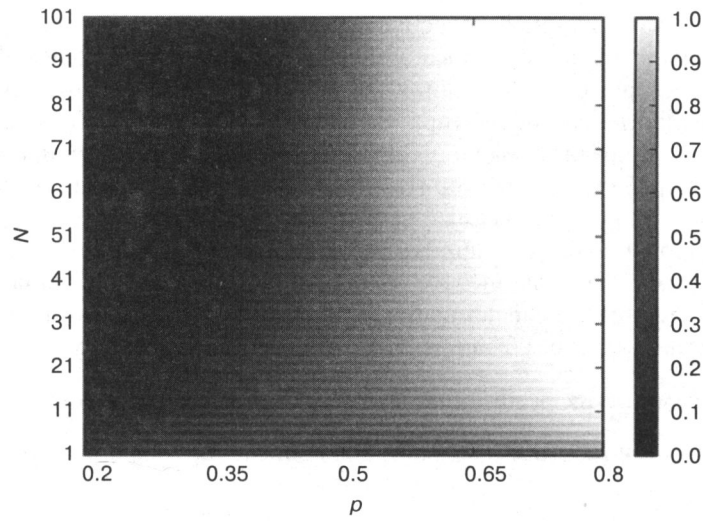


Рис. 6.2. Правильность бэггированного классификатора как функция точности индивидуального оценщика (P), числа оценщиков (N) и $k = 2$

Для получения подробного анализа данной темы читателю рекомендуется обратиться к теореме присяжных Кондорсе. Хотя эта теорема получена для целей мажоритарного голосования в политической науке, проблема, к которой обращается эта теорема, имеет общие черты с описанной выше.

6.3.3. Избыточность наблюдений

В главе 4 мы исследовали одну из причин, по которой финансовые наблюдения нельзя считать одинаково распределенными и взаимно независимыми. Избыточные наблюдения оказывают два пагубных воздействия на бэггирование. Во-первых, образцы, изымаемые с возвратом, с большей вероятностью будут практически идентичными, даже если они не имеют общих наблюдений. Это делает $\bar{\rho} \approx 1$, и бэггирование не сократит дисперсию, независимо от N. Например, если каждое наблюдение в t промаркировано в соответствии с финансовым возвратом между t и $t + 100$, то мы должны отобрать 1 % наблюдений в расчете на бэггированного оценщика, но не более. В главе 4, раздел 4.5, рекомендуются три альтернативных решения, одно из которых состояло в установке `max_samples=out['tW'].mean()` в реализации класса бэггированного классификатора в библиотеке sklearn. Еще одним (более качественным) решением было применение метода последовательного бутстраповского отбора.

Второе пагубное влияние избыточности наблюдений заключается в том, что будет взвинчена внепакетная точность. Это происходит из-за того, что случайный отбор с возвратом образцов помещает в тренировочное подмножество образцы, которые очень похожи на те, что вне пакета. В таком случае правильная стратифицированная k-блочная перекрестная проверка без перетасовки перед разбиением покажет гораздо меньшую точность на тестовом подмножестве, чем та, которая была оценена вне пакета. По этой причине при использовании этого класса библиотеки sklearn рекомендуется установить `stratifiedKFold(n_splits=k, shuffle=False)`, перекрестно проверить бэггированный классификатор и проигнорировать результаты внепакетной точности. Низкое число k предпочтительнее высокого, так как чрезмерное разбиение снова поместит в тестовое подмножество образцы, слишком похожие на те, которые используются в тренировочном подмножестве.

6.4. Случайный лес

Деревья решений общеизвестны тем, что они склонны к переподгонке, что увеличивает дисперсию прогнозов[1]. Для того чтобы обратиться к решению этой проблемы, был разработан метод случайного леса (random forest, RF) для порождения ансамблевых прогнозов с более низкой дисперсией.

[1] Для получения интуитивно понятного объяснения случайного леса перейдите по следующей ссылке: https://quantdare.com/random-forest-many-is-better-than-one/.

Случайный лес имеет некоторые общие сходства с бэггированием в смысле тренировки индивидуальных оценщиков независимо друг от друга на бутстрапированных подмножествах данных. Ключевое отличие от бэггирования заключается в том, что в случайные леса встраивается второй уровень случайности: во время оптимизации каждого узлового дробления будет оцениваться только случайная подвыборка (без возврата) атрибутов с целью дальнейшего декоррелирования оценщиков.

Как и бэггирование, случайный лес уменьшает дисперсию прогнозов без переподгонки (напомним, что до тех пор, пока $\bar{\rho} < 1$). Второе преимущество состоит в том, что случайный лес оценивает важность признаков, которую мы обсудим подробно в главе 8. Третье преимущество заключается в том, что случайный лес предоставляет оценки внепакетной точности, однако в финансовых приложениях они, скорее всего, будут взвинчены (как описано в разделе 6.3.3). Но как и бэггирование, случайный лес не обязательно будет демонстрировать более низкое смещение, чем индивидуальные деревья решений.

Если большое число образцов избыточно (не являются одинаково распределенными и взаимно независимыми), все равно будет иметь место переподгонка: случайный отбор с возвратом построит большое число практически идентичных деревьев ($\bar{\rho} \approx 1$), где каждое дерево решений переподогнано (недостаток, благодаря которому деревья решений печально известны). В отличие от бэггирования, случайный лес всегда задает размер бутстрапированных выборок в соответствии с размером тренировочного подмножества данных. Давайте рассмотрим, как мы можем решить эту проблему переподгонки случайных лесов в библиотеке sklearn. В целях иллюстрации я буду обращаться к классам библиотеки sklearn; однако эти решения могут быть применены к любой реализации:

1. Установить для параметра `max_features` меньшее значение, чтобы добиться расхождения между деревьями.
2. Ранняя остановка: установить параметр регуляризации `min_weight_fraction_leaf` равным достаточно большому значению (например, 5 %), для того чтобы внепакетная точность сходилась к вневыборочной (k-блочной) правильности.
3. Использовать оценщик `BaggingClassifier` на базовом оценщике `DecisionTreeClassifier`, где `max_samples` установлен равным средней уникальности (`avgU`) между выборками.
   - `clf=DecisionTreeClassifier(criterion='entropy', max_features='auto', class_weight='balanced')`
   - `bc=BaggingClassifier(base_estimator=clf, n_estimators=1000, max_samples=avgU, max_features=1.)`
4. Использовать оценщик `BaggingClassifier` на базовом оценщике `RandomForestClassifier`, где `max_samples` установлен равным средней уникальности (`avgU`) между выборками.
   - `clf=RandomForestClassifier(n_estimators=1, criterion='entropy', bootstrap=False, class_weight='balanced_subsample')`

- `bc=BaggingClassifier(base_estimator=clf, n_estimators=1000, max_samples=avgU, max_features=1.)`

5. Модифицировать класс случайного леса для замены стандартного бутстрапирования на последовательное бутстрапирование.

Резюмируя, листинг 6.2 демонстрирует три альтернативных способа настройки случайного леса, используя разные классы.

Листинг 6.2. Три способа настройки случайного леса

```
clf0=RandomForestClassifier(n_estimators=1000, class_weight='balanced_
                            subsample', criterion='entropy')
clf1=DecisionTreeClassifier(criterion='entropy', max_features='auto',
                            class_weight='balanced')
clf1=BaggingClassifier(base_estimator=clf1, n_estimators=1000,
                            max_samples=avgU)
clf2=RandomForestClassifier(n_estimators=1, criterion='entropy',
                            bootstrap=False, class_weight='balanced_subsample')
clf2=BaggingClassifier(base_estimator=clf2, n_estimators=1000,
                            max_samples=avgU, max_features=1.)
```

При подгонке деревьев решений поворот признакового пространства в направлении, совпадающем с осями, как правило, сокращает число необходимых дереву уровней. По этой причине я предлагаю вам выполнять подгонку случайного дерева на PCA признаков, так как это может ускорить вычисления и немного сократить переподгонку (подробнее об этом в главе 8). Кроме того, как описано в главе 4, раздел 4.8, аргумент `class_weight='balanced_subsample'` поможет не допустить, чтобы деревья неправильно классифицировали миноритарные классы.

6.5. Бустирование

Kearns and Valiant [1989] были одними из первых, кто задались вопросом, можно ли объединить слабых оценщиков, для того чтобы достичь реализации высокоточного оценщика. Вскоре после этого Schapire [1990] продемонстрировал утвердительный ответ на этот вопрос, используя процедуру, которую мы сегодня называем бустированием (boosting, форсирование, усиление). В общих чертах она работает следующим образом: во-первых, сгенерировать одно тренировочное подмножество путем случайного отбора с возвратом в соответствии с некими весами выборки (инициализируемым равномерными весами). Во-вторых, выполнить подгонку одного оценщика с помощью этого тренировочного подмножества. В-третьих, если одиночный оценщик достигает точности, превышающей порог приемлемости (например, в бинарном классификаторе она равна 50 %, чтобы классификатор работал лучше, чем случайное гадание), то оценщик остается, в противном случае он отбрасывается. В-четвертых, придать больший вес неправильно классифицированным наблюдениям и меньший вес правильно классифицированным наблюдениям. В-пятых, повторять предыдущие шаги до тех пор, пока не будут получены

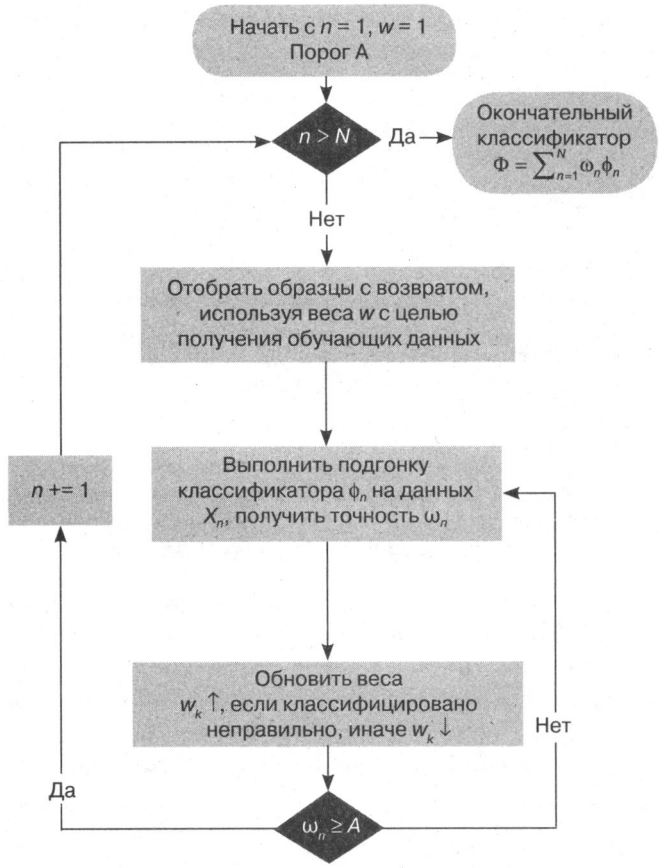


Рис. 6.3. Поток принятия решений в алгоритме AdaBoost

N оценщиков. В-шестых, ансамблевый прогноз — это средневзвешенное значение индивидуальных прогнозов из *N* моделей, где веса определяются точностью индивидуальных оценщиков. Существует ряд бустированных алгоритмов, из которых адаптивное бустирование AdaBoost является одним из самых популярных (Geron [2017]). Рисунок 6.3 резюмирует поток принятия решений в стандартной реализации алгоритма AdaBoost.

6.6. Бэггинг vs бустинг в финансах

Из приведенного выше описания несколько аспектов делают бустирование совершенно отличающимся от бэггирования[1]:

[1] Для получения визуального объяснения разницы между бэггированием и бустированием посетите: https://quantdare.com/what-is-the-difference-between-bagging-and-boosting/.

- Подгонка индивидуальных классификаторов выполняется последовательно.
- Слаборезультативные классификаторы отклоняются.
- На каждой итерации наблюдения взвешиваются по-разному.
- Ансамблевый прогноз представляет собой средневзвешенное значение индивидуальных учеников.

Главное преимущество бустирования состоит в том, что оно сокращает как дисперсию, так и смещение в прогнозах. Тем не менее исправление смещения происходит за счет большего риска переподгонки. Можно утверждать, что в финансовых приложениях бэггирование, как правило, предпочтительнее бустирования. Бэггирование решает проблему переподгонки, в то время как бустирование решает проблему недоподгонки. Переподгонка часто является более серьезной проблемой, чем недоподгонка, так как подогнать алгоритм МО слишком плотно к финансовым данным совсем не трудно из-за низкого соотношения сигнал/шум. Более того, бэггирование поддается параллелизации, тогда как бустирование обычно требует последовательного выполнения.

6.7. Бэггинг для масштабируемости

Как известно, некоторые популярные алгоритмы МО масштабируются не очень хорошо в зависимости от размера выборки. Метод опорных векторов (support vector machines, SVM) является ярким примером. Если вы попытаетесь выполнить подгонку оценщика SVM на миллионе наблюдений, то может потребоваться продолжительное время, пока алгоритм не сойдется. И даже после того, как он сошелся, нет никакой гарантии, что решение является глобальным оптимумом или что он не будет переподогнан.

Один из практических подходов заключается в построении бэггированного алгоритма, где базовый оценщик принадлежит классу, который плохо масштабируется вместе с размером выборки, например SVM. При определении этого базового оценщика мы введем жесткое условие ранней остановки. Например, в реализации опорно-векторных машин (SVM) в библиотеке sklearn вы могли бы задать низкое значение для параметра `max_iter`, например 1E5 итераций. По умолчанию принято значение `max_iter=-1`, которое сообщает оценщику продолжать выполнение итераций до тех пор, пока ошибки не упадут ниже уровня допуска. С другой стороны, вы можете повысить уровень допуска с помощью параметра `tol`, который по умолчанию имеет значение `tol=iE-3`. Любой из этих двух параметров приведет к ранней остановке. Вы можете останавливать другие алгоритмы рано с помощью эквивалентных параметров, таких как число уровней в случайном лесе (`max_depth`) или минимальная взвешенная доля итоговой суммы весов (всех входных выборок), обязанных находиться на листовом узле (`min_weight_fraction_leaf`).

Учитывая, что бэггированные алгоритмы могут быть параллелизованы, мы трансформируем большую последовательную задачу в ряд более мелких, которые вы-

полняются одновременно. Конечно, ранняя остановка увеличит дисперсию результатов от индивидуальных базовых оценщиков; однако это увеличение может быть более чем компенсировано уменьшением дисперсии, связанным с бэггированным алгоритмом. Вы можете контролировать это сокращение, добавляя новые независимые базовые оценщики. Применяемое таким образом, бэггирование позволит получать быстрые и робастные оценки на очень крупных совокупностях данных.

Упражнения

6.1. Почему бэггирование основано на случайном отборе с возвратом? Будет ли бэггирование все равно сокращать дисперсию прогноза, если отбор образцов будет без возврата?

6.2. Предположим, что ваше тренировочное подмножество основано на сильно накладывающихся метках (то есть с низкой уникальностью, как определено в главе 4).

(а) Будет ли из-за этого процедура бэггирования подверженной переподгонке или просто неэффективной? Почему?

(б) Является ли внепакетная правильность, как правило, надежной в финансовых приложениях? Почему?

6.3. Постройте ансамбль оценщиков, где базовым оценщиком является дерево решений.

(а) Чем этот ансамбль отличается от случайного леса?

(б) Используя библиотеку sklearn, создайте бэггированный классификатор, который ведет себя как случайный лес. Какие параметры вы должны были настроить и как?

6.4. Рассмотрим отношения между алгоритмом случайного леса, количеством деревьев в его составе и количеством использованных признаков:

(а) Могли бы вы обрисовать связь между минимальным числом деревьев, необходимых в случайном лесе, и числом задействованных признаков?

(б) Может ли число деревьев быть слишком малым для числа используемых признаков?

(в) Может ли число деревьев быть слишком большим для числа имеющихся наблюдений?

6.5. Как внепакетная правильность отличается от стратифицированной k-блочной (с перетасовкой) прекрестно-проверочной правильности?

7 Перекрестная проверка в финансах

7.1. Актуальность

Целью перекрестной проверки (cross-validation, CV) является определение ошибки обобщения алгоритма МО с целью предотвращения переподгонки. Перекрестная проверка является еще одним примером, когда стандартные методы машинного обучения оказываются безуспешными при применении к финансовым задачам. Произойдет переподгонка, и перекрестная проверка не сможет ее обнаружить. По сути дела, перекрестная проверка будет вносить свой вклад в переподгонку посредством регулировки гиперпараметров. В этой главе мы узнаем, почему стандартная перекрестная проверка оказывается безуспешной в финансах и что можно с этим сделать.

7.2. Цель перекрестной проверки

Одна из целей машинного обучения — узнать общую структуру данных, для того чтобы мы могли делать предсказания относительно будущих, ранее не встречавшихся признаков. Когда мы тестируем алгоритм МО на той же совокупности данных, которая использовалась для его тренировки, неудивительно, что мы достигаем впечатляющих результатов. Когда алгоритмы МО используются неправильно, они ничем не отличаются от алгоритмов сжатия файлов с потерями: они могут резюмировать данные с предельной точностью, но с нулевой предсказательной силой.

Перекрестная проверка дробит наблюдения, извлеченные из одинаково распределенного взаимно независимого случайного процесса, на два подмножества: тренировочное и тестовое подмножества. Каждое наблюдение в полной совокупности данных принадлежит одному и только одному подмножеству. Это сделано для предотвращения утечки из одного подмножества в другое, так как это нарушило бы цель тестирования на ранее не встречавшихся данных. Более подробную ин-

формацию можно найти в книгах и статьях, перечисленных в разделе справочных материалов.

Существует ряд альтернативных перекрестно-проверочных схем, из которых одной из самых популярных является k-блочная перекрестная проверка. Рисунок 7.1 иллюстрирует k обучающих и тестовых подразделов (то есть блоков), выполняемых k-блочной перекрестной проверкой, где $k = 5$. В этой схеме:

1. Набор данных разбит на k-подгруппы.
2. При $i = 1, ..., k$

   (а) алгоритм МО обучается на всех подгруппах, за исключением i;

   (б) обученный алгоритм МО тестируется на i.

Рис. 7.1. Обучающие и тестовые подразделы в схеме 5-блочной перекрестной проверки

Результатом k-блочной перекрестной проверки является массив $k \times l$ перекрестно проверенных метрических показателей результативности. Например, в бинарном классификаторе считается, что модель чему-то научилась, если перекрестно проверенная точность превышает $1/2$, так как это та точность, которой мы достигнем, подбрасывая правильную монету.

В финансах перекрестная проверка, как правило, используется в двух ситуациях: разработка модели (к примеру, регулировка гиперпараметров) и тестирование. Бэктестирование является сложной темой, которую мы подробно обсудим в главах 10–16. В этой главе мы сосредоточимся на перекрестной проверке для разработки модели.

7.3. Почему перекрестная проверка по k блокам оказывается безуспешной в финансах

К настоящему времени вы, возможно, прочитали немало статей в области финансов, которые представляют подтверждающие данные k-блочной перекрестной проверки о том, что алгоритм МО имеет хорошую результативность. К сожалению, почти наверняка эти результаты неверны. Одна из причин, почему k-блочная перекрестная проверка оказывается безуспешной в финансах, заключается в том, что наблюдения не могут быть взяты из одинаково распределенного взаимно независимого случайного процесса. Вторая причина безуспешности перекрестной проверки заключается в том, что в процессе разработки модели тестовое подмножество используется несколько раз, что приводит к множественному тестированию и систематическому смещению при отборе образцов. Мы вновь рассмотрим эту вторую причину безуспешности в главах 11–13. Пока же давайте займемся исключительно первой причиной безуспешности.

Утечка происходит, когда тренировочное подмножество содержит информацию, которая также появляется в тестовом подмножестве. Рассмотрим внутрирядово коррелированный признак X, связанный с метками Y, которые формируются на накладывающихся данных:

- Из-за внутрирядовой корреляции $X_t \approx X_{t+1}$.
- Из-за того, что метки проистекают из накладывающихся точек данных, $Y_t \approx Y_{t+1}$.

Размещение t и $t + 1$ в разных подмножествах вызывает утечку информации. Когда классификатор сначала тренируют на (X_t, Y_t), а затем его просят предсказать $\mathrm{E}[Y_{t+1}|X_{t+1}]$, основываясь на наблюдаемом X_{t+1}, этот классификатор, вероятнее всего, достигнет $Y_{t+1} = \mathrm{E}[Y_{t+1}|X_{t+1}]$, даже если X является нерелевантным признаком.

Если X является предсказательным признаком, то утечка повысит результативность и без того ценной стратегии. Проблемой является утечка в присутствии нерелевантных признаков, так как оно приводит к ложным открытиям. Существует как минимум два способа сокращения вероятности утечки:

1. Устранять из тренировочного подмножества любое наблюдение i, где Y_i — это функция информации, используемая для определения Y_j, и j принадлежит тестовому подмножеству.

   (a) К примеру, Y_i и Y_j не должны охватывать накладывающиеся периоды (см. главу 4 по поводу обсуждения уникальности выборки).

2. Избегать переподгонки классификатора. Благодаря этому, даже если произойдет утечка, классификатор не сможет извлечь из этого выгоду. Использовать:

(а) Раннюю остановку базовых оценщиков (см. главу 6).

(б) Бэггирование классификаторов, при этом следя за излишним отбором на избыточных примерах, с тем чтобы индивидуальные классификаторы были как можно более многообразными:

i) установить `max_samples` равным средней уникальности;

ii) применить последовательное бутстрапирование (см. главу 4).

Рассмотрим случай, когда X_i и X_j формируются на накладывающейся информации, где i принадлежит тренировочному подмножеству и j принадлежит тестовому подмножеству. Является ли этот случай утечкой информации? Не обязательно, если Y_i и Y_j независимы. Для того чтобы утечка имела место, должно быть $(X_i, Y_i) \approx (X_j, Y_j)$, и не достаточно, что $X_i \approx X_j$ или даже $Y_i \approx Y_j$.

7.4. Решение: прочищенная *k*-блочная перекрестная проверка

Одним из способов сокращения утечки является удаление из тренировочного подмножества всех наблюдений, метки которых накладывались во времени с метками, включенными в тестовое подмножество. Я называю этот процесс «прочисткой» (purging). Вдобавок, поскольку финансовые признаки часто включают в себя ряды, которые демонстрируют внутрирядовую корреляцию (например, процессы ARMA[1]), мы должны исключить из тренировочного подмножества наблюдения, которые немедленно следуют за наблюдением в тестовом подмножестве. Я называю этот процесс «эмбарго».

7.4.1. Очищение набора данных для обучения

Предположим, что у нас есть тестовое наблюдение, чья метка Y_j определяется на основе информации Φ_j. Для того чтобы предотвратить тип утечки, описанный в предыдущем разделе, мы хотели бы удалить из тренировочного подмножества любое наблюдение, чья метка Y_j определяется на основе информации Φ_j такой, что $\Phi_i \cap \Phi_j = \emptyset$.

В частности, мы определим, что между двумя наблюдениями i и j существует информационное наложение всякий раз, когда Y_i и Y_j одновременны (см. главу 4, раздел 4.3) в том смысле, что обе метки зависят от взятия по крайней мере одного общего случайного образца. Например, рассмотрим метку Y_j, которая является функцией наблюдений в закрытом интервале $t \in [t_{j,0}, t_{j,1}]$, $Y_j = f[[t_{j,0}, t_{j,1}]]$ (с некото-

[1] ARMA (autoregressive moving-average) — авторегрессионное скользящее среднее. Модель ARMA обобщает две более простые модели временных рядов — авторегрессионную модель (AR) и модель на основе скользящего среднего (MA). — *Примеч. науч. ред.*

рым злоупотреблением математической записи). Например, в контексте тройного барьерного метода маркировки (глава 3) это означает, что метка является знаком финансового возврата, охватывающего ценовые бары с индексами $t_{j,0}$ и $t_{j,1}$, то есть $\text{sgn}[r_{t_{j,0},t_{j,1}}]$. Метка $Y_j = f[[t_{i,0}, t_{i,1}]]$ накладывается с Y_j, если выполняется любое из трех достаточных условий:

1. $t_{j,0} \leq t_{i,0} \leq t_{j,1}$.
2. $t_{j,0} \leq t_{i,1} \leq t_{j,1}$.
3. $t_{i,0} \leq t_{j,0} \leq t_{j,1} \leq t_{i,1}$.

Листинг 7.1 реализует это удаление наблюдений из тренировочного подмножества. Если тестовое подмножество является сплошным, в том смысле что между первым и последним тестовым наблюдением не встречается никаких тренировочных наблюдений, то прочистка может быть ускорена: объект `testTimes` может быть рядом библиотеки pandas с одним-единственным элементом, охватывающим все тестовое подмножество.

Листинг 7.1. Прочистка наблюдения из тренировочного подмножества

```
def getTrainTimes(t1,testTimes):
    '''
    С учетом testTimes найти времена тренировочных наблюдений.
    - t1.index: время, когда наблюдение началось.
    - t1.value: время, когда наблюдение закончилось.
    - testTimes: времена тестовых наблюдений.
    '''
    trn=t1.copy(deep=True)
    for i,j in testTimes.iteritems():
        df0=trn[(i<=trn.index)&(trn.index<=j)].index # тренировка начинается
                                                     # внутри теста
        df1=trn[(i<=trn)&(trn<=j)].index # тренировка заканчивается внутри
                                         # теста
        df2=trn[(trn.index<=i)&(j<=trn)].index # тренировка охватывает тест
        trn=trn.drop(df0.union(df1).union(df2))
    return trn
```

Когда имеет место утечка, результативность улучшается просто за счет увеличения $k \to T$, где T — это число баров. Причина в том, что чем больше число тестовых подразделов, тем больше число накладывающихся наблюдений в тренировочном подмножестве. Во многих случаях для того, чтобы предотвратить утечку, будет достаточно прочистки: результативность будет улучшаться по мере увеличения k, потому что мы чаще позволяем модели перекалибровываться. Но за пределами определенного значения k^* результативность не улучшится, что указывает на то, что бэктест не извлекает прибыли из утечек. На рис. 7.2 показан график с одним разделом k-блочной перекрестной проверки. Тестовое подмножество окружено двумя тренировочными подмножествами, генерирующими два наложения, которые должны быть удалены, для того чтобы предотвратить утечку.

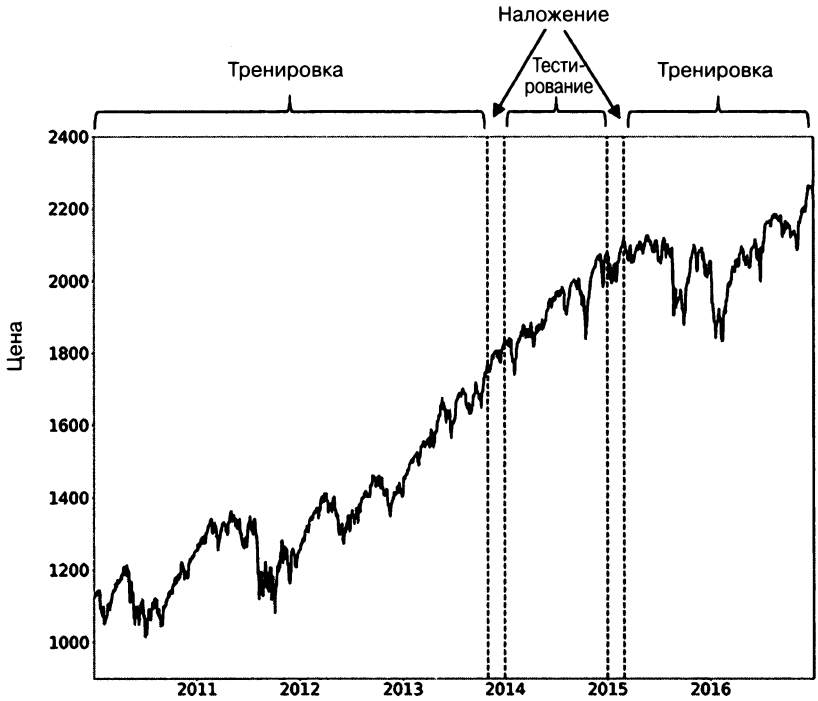


Рис. 7.2. Удаление наложений в тренировочном подмножестве

7.4.2. Эмбарго

В тех случаях, когда прочистка не в состоянии предотвратить все утечки, мы можем ввести эмбарго на тренировочные наблюдения *после* каждого тестового подмножества. Эмбарго не должно влиять на тренировочные наблюдения перед тестовым подмножеством, поскольку тренировочные метки $Y_i = f[[t_{i,0}, t_{j,1}]]$, где $t_{j,1} < t_{j,0}$ (тренировка заканчивается до начала тестирования), содержат информацию, которая была доступна во время тестирования $t_{j,0}$. Другими словами, мы имеем дело только с тренировочными метками $Y_i = f[[t_{I,0}, t_{j,1}]]$, которые имеют место сразу после теста, $t_{j,1} \leq t_{i,0} \leq t_{j,1} + h$. Мы можем реализовать этот эмбарговый период h, установив $Y_i = f[[t_{j,0}, t_{j,1} + h]]$ перед прочисткой. Малое значение $h \approx .01T$ часто вполне достаточно для того, чтобы предотвратить все утечки, что может быть подтверждено, перепроверив, что результативность не улучшается бесконечно за счет увеличения $k \rightarrow T$. Рисунок 7.3 иллюстрирует наложение эмбарго на тренировочные наблюдения сразу после тестового подмножества. Листинг 7.2 реализует логику эмбарго.

Листинг 7.2. Наложение эмбарго на тренировочные наблюдения

```
def getEmbargoTimes(times,pctEmbargo):
    # Получить время эмбарго для каждого бара
    step=int(times.shape[0]*pctEmbargo)
```

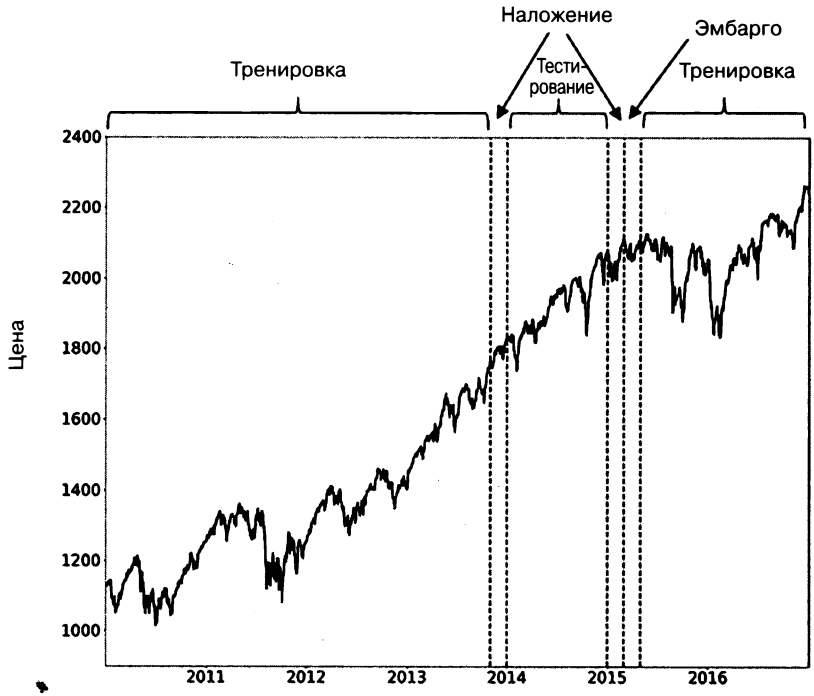


Рис. 7.3. Эмбарго посттестовых тренировочных наблюдений

```
    if step==0:
        mbrg=pd.Series(times,index=times)
    else:
        mbrg=pd.Series(times[step:],index=times[:-step])
        mbrg=mbrg.append(pd.Series(times[-1],index=times[-step:]))
    return mbrg
#-------------------------------------
testTimes=pd.Series(mbrg[dt1],index=[dt0]) # задействовать эмбарго перед
                                           # прочисткой
trainTimes=getTrainTimes(t1,testTimes)
testTimes=t1.loc[dt0:dt1].index
```

7.4.3. Класс прочищенной *k*-блочной перекрестной проверки

В предыдущих разделах мы обсуждали способ создания тренировочных и тестовых подразделов, когда метки накладываются. Он предусматривает введение понятия прочистки и эмбарго в конкретном контексте разработки модели. В общем случае нам нужно проводить прочистку и накладывать эмбарго на накладывающиеся тренировочные наблюдения всякий раз, когда мы производим дробление тренировочного и тестового подмножеств, будь то подгонка гиперпараметров,

бэктестирование или оценивание результативности. Листинг 7.3 расширяет класс `KFold` библиотеки scikit-learn, с тем чтобы учесть возможность утечки тестовой информации в тренировочное подмножество.

Листинг 7.3. Перекрестно-проверочный класс для случаев, когда наблюдения накладываются

```
class PurgedKFold(_BaseKFold):
    '''
    Расширить класс KFold для работы с метками,
    которые охватывают интервалы
    Тренировочный образец удаляется из наблюдений,
    накладывающихся на интервалы с тестовой меткой
    Тестовое подмножество считается сплошным (shuffle=False),
    без тренировочных образцов внутри
    '''
    def __init__(self,n_splits=3,t1=None,pctEmbargo=0.):
        if not isinstance(t1,pd.Series):
            raise ValueError('Метка по датам должна быть pd.Series')
        super(PurgedKFold,self).__init__(n_splits,shuffle=False,random_
              state=None)
        self.t1=t1
        self.pctEmbargo=pctEmbargo
    def split(self,X,y=None,groups=None):
        if (X.index==self.t1.index).sum()!=len(self.t1):
            raise ValueError('X и ThruDateValues должны иметь одинаковый
                             индекс')
        indices=np.arange(X.shape[0])
        mbrg=int(X.shape[0]*self.pctEmbargo)
        test_starts=[(i[0],i[-1]+1) for i in \
            np.array_split(np.arange(X.shape[0]),self.n_splits)]
        for i,j in test_starts:
            t0=self.t1.index[i] # начать тестовое подмножество
            test_indices=indices[i:j]
            maxT1Idx=self.t1.index.searchsorted(self.t1[test_indices].max())
            train_indices=self.t1.index.searchsorted(self.t1[self.t1<=t0].
                          index)
            if maxT1Idx<X.shape[0]: # правильный тренировочный образец
                                    # (с эмбарго)
                train_indices=np.concatenate((train_indices,indices
                                              [maxT1Idx+mbrg:]))
            yield train_indices,test_indices
```

7.5. Дефекты реализации перекрестной проверки в библиотеке sklearn

Можно подумать, что нечто столь же важное, как перекрестная проверка, будет прекрасно реализовано в одной из самых популярных библиотек машинного обучения. К сожалению, это не так, и это одна из причин, почему вы должны всегда

читать весь исходный код, который выполняете, и это сильный плюс в пользу открытого исходного кода. Одно из преимуществ открытого исходного кода состоит в том, что вы можете все перепроверить и скорректировать его для ваших нужд. Листинг 7.4 устраняет два известных дефекта библиотеки sklearn:

1. Оценивающие функции не знают массива `classes_`, как следствие опоры библиотеки sklearn на массивы библиотеки numpy, а не на ряды библиотеки pandas: https://github.com/scikitlearn/scikit-learn/issues/6231.
2. Вспомогательная функция `cross_val_score` будет давать разные результаты, так как она передает веса в метод подгонки, а не в метод `log_loss`: https://github.com/scikitlearn/scikit-learn/issues/9144.

Листинг 7.4. Использование класса PurgedKFold

```
def cvScore(clf,X,y,sample_weight,scoring='neg_log_loss',t1=None,cv=None,
            cvGen=None,
            pctEmbargo=None):
    if scoring not in ['neg_log_loss','accuracy']:
        raise Exception('неправильный оценивающий метод.')
    from sklearn.metrics import log_loss,accuracy_score
    from clfSequential import PurgedKFold
    if cvGen is None:
        cvGen=PurgedKFold(n_splits=cv,t1=t1,pctEmbargo=pctEmbargo) # прочищен
    score=[]
    for train,test in cvGen.split(X=X):
        fit=clf.fit(X=X.iloc[train,:],y=y.iloc[train],
                    sample_weight=sample_weight.iloc[train].values)
        if scoring=='neg_log_loss':
            prob=fit.predict_proba(X.iloc[test,:])
            score_=-log_loss(y.iloc[test],prob,
                sample_weight=sample_weight.iloc[test].values,labels=clf.
                                classes_)
        else:
            pred=fit.predict(X.iloc[test,:])
            score_=accuracy_score(y.iloc[test],pred,sample_weight= \
                sample_weight.iloc[test].values)
        score.append(score_)
    return np.array(score)
```

Пожалуйста, учтите, что до того времени, когда исправление этих дефектов будет согласовано, реализовано, протестировано и выпущено, может пройти много времени. До тех пор следует использовать функцию `cvScore` в листинге 7.4 и избегать выполнения функции `cross_val_score`.

Упражнения

7.1. Почему перетасовка совокупности данных перед проведением k-блочной перекрестной проверки вообще плохая идея в финансах? Какова цель пере-

тасовки? Почему перетасовка противоречит цели k-блочной перекрестной проверки в финансовых наборах данных?

7.2. Возьмите пару матриц (X, y), представляющих наблюдаемые признаки и метки. Они могут быть одной из совокупностей данных, полученных из упражнений в главе 3.

(а) Выведите результативность из 10-блочной перекрестной проверки классификатора на основе случайного леса на матрицах (X, y) без перетасовки.

(б) Выведите результативность из 10-блочной перекрестной проверки классификатора на основе случайного леса на матрицах (X, y) с перетасовкой.

(в) Почему результаты такие разные?

(г) Как перетасовка приводит к утечкам?

7.3. Возьмите ту же пару матриц (X, y) из упражнения 2.

(а) Выведите результативность из 10-блочной перекрестной проверки классификатора на основе случайного леса на матрицах (X, y) с 1 %-ным эмбарго.

(б) Почему результативность ниже?

(в) Почему этот результат более реалистичен?

7.4. В этой главе мы сосредоточились на одной из причин, почему k-блочная перекрестная проверка оказывается безуспешной в финансовых приложениях, а именно на том факте, что некоторая информация из тестового подмножества просачивается в тренировочное подмножество. Можете ли вы подумать и указать вторую причину безуспешности перекрестной проверки?

7.5. Предположим, что вы пробуете тысячу конфигураций одной и той же инвестиционной стратегии и выполняете перекрестную проверку на каждой из них. Некоторые результаты гарантированно выглядят хорошо, просто по счастливой случайности. Если вы опубликуете только эти утвердительные результаты и скроете остальные, то ваша аудитория не сможет сделать вывод, что эти результаты являются ложными утверждениями, статистической случайностью. Это явление называется систематическим смещением при отборе.

(а) Можете ли вы представить себе процедуру, которая это предотвращает?

(б) Что делать, если мы разделим совокупность данных на три подмножества: тренировочное, контрольное и тестовое? Контрольное подмножества используется для оценивания натренированных параметров, и тестирование выполняется только на одной конфигурации, выбранной в контрольной фазе. В каком случае эта процедура по-прежнему будет безуспешной?

(в) Как можно избежать ошибки выборки?

8 Важность признаков

8.1. Актуальность

Одна из самых распространенных ошибок в финансовых исследованиях — брать немного данных, прогонять их через алгоритм МО, бэктестировать предсказания и повторять эту последовательность до тех пор, пока не появится красивый результат бэктеста. Академические журналы наполнены такими псевдооткрытиями, и даже крупные хеджевые фонды постоянно попадают в эту ловушку. Неважно, является ли бэктест прямым вневыборочным. Тот факт, что мы повторяем тест снова и снова на одних и тех же данных, скорее всего, приведет к ложному открытию. Эта методологическая ошибка настолько печально известна среди статистиков, что они считают ее научным мошенничеством, и Американская статистическая ассоциация предупреждает об этом в своих этических руководящих принципах (American Statistical Association [2016], дискуссия № 4). Обычно требуется около 20 таких итераций, для того чтобы открыть (ложную) инвестиционную стратегию с учетом стандартного уровня значимости (частоты ложных утверждений) 5 %. В этой главе мы рассмотрим причины, почему такой подход является пустой тратой времени и денег и как важность признаков предлагает альтернативу.

8.2. Значимость признаков

Поразительным аспектом финансовой индустрии является то, что очень многие опытные портфельные менеджеры (в том числе многие с квантитативной подготовкой) не понимают, как легко можно довести бэктест до переподгонки. Ответ на вопрос, как правильно проводить бэктест, не является предметом данной главы; мы рассмотрим эту чрезвычайно важную тему в главах 11–15. Цель этой главы — объяснить одну из процедур анализа, которая должна быть выполнена *до* проведения бэктеста.

Предположим, что вам дана пара матриц (X, y), которые содержат соответственно признаки и метки для конкретного финансового инструмента. Мы можем выполнить подгонку классификатора на (X, y) и оценить ошибку обобщения посредством

прочищенной k-блочной перекрестной проверки (CV), как мы видели в главе 7. Предположим, что мы добьемся хорошей результативности. Следующий естественный вопрос — попытаться понять, какие признаки вносили свой вклад в эту результативность. Вполне возможно, мы могли бы добавить несколько признаков, которые усиливают сигнал, ответственный за предсказательную силу классификатора. Или же мы могли бы устранить несколько признаков, которые только добавляют шум в систему. Примечательно, что понимание важности признаков раскрывает пресловутый черный ящик. Мы можем получить представление о закономерностях, выявленных классификатором, если мы поймем, какой источник информации для него необходим. Это одна из причин, почему черноящичная мантра скептиками машинного обучения несколько преувеличена. Да, алгоритм научился без нас руководить процессом в черном ящике (в этом и есть весь смысл машинного обучения!), но это не означает, что мы не можем (или не должны) взглянуть на то, что именно алгоритм нашел. Охотники не вслепую же едят все, что их умные собаки им приносят, разве не так?

После того как мы нашли, какие признаки важны, мы можем узнать больше, проведя ряд экспериментов. Важны ли эти признаки все время или только в определенных средах? Что запускает изменения в важности во временной динамике? Можно ли предсказать эти переключения режимов? Релевантны ли эти важные признаки для других родственных финансовых инструментов? Релевантны ли они для других классов активов? Каковы наиболее релевантные признаки во всех финансовых инструментах? Каково подмножество признаков с наивысшей ранговой корреляцией во всем инвестиционном универсуме? Это гораздо более качественный способ исследования стратегий, чем глупый цикл бэктестирования. Позвольте мне изложить этот принцип как один из самых важных уроков, которые, я надеюсь, вы вынесете из этой книги:

ЛИСТИНГ 8.1. ПЕРВЫЙ ЗАКОН БЭКТЕСТИНГА МАРКОСА — ИГНОРИРУЙТЕ ЕГО НА СВОЙ СТРАХ И РИСК

«Бэктестинг — это не исследовательский инструмент. Им является важность признаков».

— Маркос Лопез де Прадо
Машинное обучение: алгоритмы для бизнеса (2018)

8.3. Важность признаков и эффекты замещения

Я считаю полезным различать методы анализа важности признаков на основе того, влияют ли на них эффекты замещения или нет. В этом контексте эффект замещения имеет место, когда оценочная важность одного признака сокращается

за счет присутствия других связанных признаков. Эффекты замещения являются аналогом из машинного обучения того, что статистика и эконометрия называют мультиколлинеарностью. Одним из способов устранения эффектов линейного замещения является применение анализа главных компонент (PCA) на сырых признаках, а затем выполнение анализа важности признаков на ортогональных признаках. См. Belsley и соавт. [1980], Goldberger [1991, pp. 245–253] и Hill и соавт. [2001] для получения более подробной информации.

8.3.1. Среднее снижение в примесности

Среднее снижение в примесности (mean decrease impurity, MDI)[1] — это быстрый, объяснительно-значимый (внутривыборочный) метод, специфичный для древовидных классификаторов, таких как случайный лес. В каждом узле каждого дерева решений отобранный признак дробит полученное подмножество таким образом, чтобы снизить примесность. Следовательно, для каждого дерева решений мы можем вывести, сколько совокупного снижения примесности может быть назначено каждому признаку. И учитывая, что у нас есть лес деревьев, мы можем усреднить эти значения по всем оценщикам и соответствующим образом ранжировать признаки. См. публикацию Louppe и соавт. [2013] для получения детального описания. При работе со средним снижением в примесности MDI необходимо учитывать несколько важных соображений:

1. Маскирующие эффекты имеют место, когда некоторые признаки систематически игнорируются древовидными классификаторами в пользу других. Для того чтобы избежать их, при использовании класса случайного леса библиотеки sklearn установите `max_features=int(1)`. Благодаря этому учитывается только один случайный признак в расчете на каждый уровень:
   а) каждому признаку дается шанс (на некоторых случайных уровнях некоторых случайных деревьев) снизить примесность;
   б) проверить, чтобы признаки с нулевой важностью не усреднялись, так как единственная причина для 0 состоит в том, что признак не был выбран случайно. Заменить эти значения на `np.nan`.
2. Данная процедура явно внутривыборочная. Каждый признак будет иметь некоторое значение, даже если они не имеют предсказательной силы вообще.
3. Анализ MDI не может быть обобщен на другие недревовидные классификаторы.
4. В силу своей конструкции анализ MDI имеет хорошее свойство в том, что важности признаков в сумме дают 1 и важность каждого признака ограничена между 0 и 1.

[1] См. https://medium.com/the-artificial-impostor/feature-importance-measures-for-tree-models-part-i-47f187c1a2c3. — *Примеч. науч. ред.*

5. Данный метод не рассматривает эффекты замещения в присутствии коррелированных признаков. Среднее снижение в примесности разбавляет важность замещающих признаков из-за их взаимозаменяемости: важность двух идентичных признаков будет снижена вдвое, так как они выбираются случайно с равной вероятностью.

6. В публикации Strobl и соавт. [2007] экспериментально показано, что метод среднего снижения в примесности смещен в сторону некоторых предикторных переменных. В публикации White and Liu [1994] утверждается, что в случае одиночных деревьев решений это смещение обусловлено несправедливым преимуществом, оказываемым популярными примесными функциями предикторам с большим числом категорий.

Класс `RandomForest` библиотеки sklearn реализует среднее снижение в примесности MDI как принятую по умолчанию оценку важности признака. Этот выбор, вероятно, мотивирован способностью вычислять среднее снижение в примесности на лету с минимальными вычислительными затратами[1]. Листинг 8.2 иллюстрирует реализацию метода MDI с учетом перечисленных выше соображений.

Листинг 8.2. Важность признака на основе анализа MDI

```
def featImpMDI(fit,featNames):
    # важность признаков на основе внутривыборочного сокращения среднего
    # в примесности
    df0={i:tree.feature_importances_ for i,tree in enumerate(fit.estimators_)}
    df0=pd.DataFrame.from_dict(df0,orient='index')
    df0.columns=featNames
    df0=df0.replace(0,np.nan) # потому что max_features=1
    imp=pd.concat({'mean':df0.mean(),'std':df0.std()*df0.shape[0]**-.5},axis=1)
    imp/=imp['mean'].sum()
    return imp
```

8.3.2. Среднее снижение точности

Среднее снижение точности (mean decrease accuracy, MDA) — это медленный, предсказательно-значимый (вневыборочный) метод. Во-первых, он выполняет подгонку классификатора; во-вторых, он выводит свою результативность вне выборки в соответствии с некой балльной оценкой результативности (точность, отрицательная логарифмическая потеря и т. д.); в-третьих, он переставляет каждый столбец признаковой матрицы (X), по одному столбцу за раз, выводя результативность вне выборки после перестановки каждого столбца. Важность признака является функцией от потери результативности, вызванной перестановкой столбца. В этой связи вот несколько релевантных соображений:

1. Данный метод может быть применен к любому классификатору, а не только к древовидным классификаторам.

[1] http://blog.datadive.net/selecting-good-features-part-iii-random-forests/.

2. Анализ MDA не ограничен точностью как единственной балльной оценкой результативности. Например, в контексте метамаркировочных приложений вместо точности мы можем предпочесть выставлять балльную оценку классификатору с помощью оценки F1 (см. пояснение в главе 14, раздел 14.8). Это одна из причин, почему более подходящим названием была бы «перестановочная важность». Когда оценивающая функция не соответствует метрическому пространству, результаты анализа MDA должны использоваться в качестве ранговой оценки.

3. Как и анализ MDI, данная процедура также подвержена эффектам замещения в присутствии коррелированных признаков. При заданных двух идентичных признаках анализ MDA всегда считает один из них избыточным для другого. К сожалению, анализ MDA покажет оба признака совершенно нерелевантными, даже если они критически значимы.

4. В отличие от анализа MDI, существует возможность, что анализ MDA заключит, что все признаки неважны. Причина в том, что анализ MDA основывается на вневыборочной результативности.

5. Перекрестная проверка должна быть прочищена и подвергнута эмбарго по причинам, изложенным в главе 7.

Листинг 8.3 реализует важность признаков на основе MDA с весами выборки, с прочищенной k-блочной перекрестной проверкой и балльным оцениванием с помощью отрицательной логарифмической потери либо точности. Он измеряет важность на основе MDA как функцию улучшения (с перестановкой и без перестановки признака), относительно максимально возможной балльной оценки (отрицательной логарифмической потери, равной 0, либо точности, равной 1). Обратите внимание, что в некоторых случаях улучшение может быть отрицательным, имея в виду, что признак фактически вреден для предсказательной силы обучающегося алгоритма.

Листинг 8.3. Важность признаков на основе анализа MDA

```
def featImpMDA(clf,X,y,cv,sample_weight,t1,pctEmbargo,scoring='neg_log_loss'):
    # важность признаков на основе вневыборочной оценивающей функции
    if scoring not in ['neg_log_loss','accuracy']:
        raise Exception('wrong scoring method.')
    from sklearn.metrics import log_loss,accuracy_score
    cvGen=PurgedKFold(n_splits=cv,t1=t1,pctEmbargo=pctEmbargo) # прочищенная cv
    scr0,scr1=pd.Series(),pd.DataFrame(columns=X.columns)
    for i,(train,test) in enumerate(cvGen.split(X=X)):
        X0,y0,w0=X.iloc[train,:],y.iloc[train],sample_weight.iloc[train]
        X1,y1,w1=X.iloc[test,:],y.iloc[test],sample_weight.iloc[test]
        fit=clf.fit(X=X0,y=y0,sample_weight=w0.values)
        if scoring=='neg_log_loss':
            prob=fit.predict_proba(X1)
            scr0.loc[i]=-log_loss(y1,prob,sample_weight=w1.values,
                                  labels=clf.classes_)
        else:
```

```
            pred=fit.predict(X1)
            scr0.loc[i]=accuracy_score(y1,pred,sample_weight=w1.values)
        for j in X.columns:
            X1_=X1.copy(deep=True)
            np.random.shuffle(X1_[j].values) # перестановка одного столбца
            if scoring=='neg_log_loss':
                prob=fit.predict_proba(X1_)
                scr1.loc[i,j]=-log_loss(y1,prob,sample_weight=w1.values,
                                        labels=clf.classes_)
            else:
                pred=fit.predict(X1_)
                scr1.loc[i,j]=accuracy_score(y1,pred,sample_weight=w1.values)
    imp=(-scr1).add(scr0,axis=0)
    if scoring=='neg_log_loss': imp=imp/-scr1
    else: imp=imp/(1.-scr1)
    imp=pd.concat({'mean':imp.mean(),'std':imp.std()*imp.shape[0]**-.5},axis=1)
    return imp,scr0.mean()
```

8.4. Важность признаков без эффектов замещения

Эффекты замещения могут привести нас к отказу от важных признаков, которые оказываются избыточными. Это, как правило, не проблема в контексте предсказания, но это может привести нас к неправильным заключениям, когда мы пытаемся понять, что делать: улучшить модель либо ее упростить. По этой причине следующий ниже метод однопризнаковой важности может быть хорошим дополнением к анализам MDI и MDA.

8.4.1. Однопризнаковая важность

Однопризнаковая важность (single feature importance, SFI) — это перекрестно-проверочный предсказательно-значимый (вневыборочный) метод. Он вычисляет балльную оценку вневыборочной результативности для каждого признака изолированно. Вот несколько соображений:

1. Данный метод может быть применен к любому классификатору, а не только к древовидным классификаторам.
2. Анализ SFI не ограничивается точностью как единственной балльной оценкой результативности.
3. В отличие от анализов MDI и MDA, никаких эффектов замещения не происходит, так как одновременно учитывается только один признак.
4. Как и анализ MDA, он может заключить, что все признаки неважны, потому что результативность оценивается посредством вневыборочной перекрестной проверки.

Главное ограничение анализа SFI заключается в том, что классификатор с двумя признаками может показывать более высокую результативность, чем бэггирование двух однопризнаковых классификаторов. Например: 1) признак B может быть полезен только в сочетании с признаком A; или 2) признак B может быть полезен при объяснении дроблений из признака A, даже если только признак B является неточным. Другими словами, совместные эффекты и иерархическая важность в анализе SFI теряются. Одним из вариантов могло бы быть вычисление балльной оценки вневыборочной результативности из подмножеств признаков, но этот расчет станет трудноразрешимым по мере рассмотрения большего числа признаков. Листинг 8.4 демонстрирует одну из возможных реализаций метода SFI. Обсуждение функции `cvScore` можно найти в главе 7.

Листинг 8.4. Реализация метода SFI

```
def auxFeatImpSFI(featNames,clf,trnsX,cont,scoring,cvGen):
    imp=pd.DataFrame(columns=['mean','std'])
    for featName in featNames:
        df0=cvScore(clf,X=trnsX[[featName]],y=cont['bin'],
                    sample_weight=cont['w'],
                    scoring=scoring,cvGen=cvGen)
        imp.loc[featName,'mean']=df0.mean()
        imp.loc[featName,'std']=df0.std()*df0.shape[0]**-.5
    return imp
```

8.4.2. Ортогональные признаки

Как указывалось в разделе 8.3, эффекты замещения ослабляют важность признаков, измеряемых анализом MDI, и значительно недооценивают важность признаков, измеряемых анализом MDA. Частичным решением является ортогонализирование признаков перед применением анализов MDI и MDA. Процедура ортогонализации, такая как анализ главных компонент (PCA), не предотвращает все эффекты замещения, но, по крайней мере, она должна смягчать влияние линейных эффектов замещения.

Рассмотрим матрицу $\{X_{t,n}\}$ стационарных признаков с наблюдениями $t = 1, ..., T$ и переменными $n = 1, ..., N$. Во-первых, мы вычисляем матрицу Z стандартизованных признаков, таких, что $Z_{t,n} = \sigma_n^{-1}(X_{t,n} - \mu_n)$, где μ_n — это среднее из $\{X_{t,n}\}_{t=1,...,T}$, а σ_n — среднеквадратическое отклонение $\{X_{t,n}\}_{t=1,...,T}$. Во-вторых, мы вычисляем собственные значения Λ и собственные векторы W такие, что $Z'ZW = W\Lambda$, где Λ — это диагональная $N \times N$-матрица, в которой главные записи отсортированы в убывающем порядке, и W — ортонормированная $N \times N$-матрица. В-третьих, мы получаем ортогональные признаки как $P = ZW$. Мы можем перепроверить ортогональность признаков, отметив, что $P'P = W'Z'ZW = W'W\Lambda W'W = \Lambda$.

Диагонализация выполняется не на X, а на Z, по двум причинам: 1) центрирование данных гарантирует, что первая главная компонента правильно ориентирована в главном направлении наблюдений. Это эквивалентно добавлению пересечения

в линейной регрессии; 2) перешкалирование данных заставляет анализ PCA сосредоточиться на объяснении корреляций, а не дисперсий. Без перешкалирования в первых главных компонентах будут преобладать столбцы X с наибольшей дисперсией, и мы узнаем совсем не много о структуре или взаимосвязи между переменными.

Кода из листинга 8.5 вычисляет наименьшее число ортогональных признаков, объясняющих по меньшей мере 95 % дисперсии Z.

Листинг 8.5. Вычисление ортогональных признаков

```
def get_eVec(dot,varThres):
    # вычислить eVec из матрицы скалярных произведений, сократить размерность
    eVal,eVec=np.linalg.eigh(dot)
    idx=eVal.argsort()[::-1] # аргументы для сортировки eVal по убыванию
    eVal,eVec=eVal[idx],eVec[:,idx]
    #2) только положительные eVals
    eVal=pd.Series(eVal,index=['PC_'+str(i+1) for i in range(eVal.shape[0])])
    eVec=pd.DataFrame(eVec,index=dot.index,columns=eVal.index)
    eVec=eVec.loc[:,eVal.index]
    #3) сократить размерность, из главных компонент
    cumVar=eVal.cumsum()/eVal.sum()
    dim=cumVar.values.searchsorted(varThres)
    eVal,eVec=eVal.iloc[:dim+1],eVec.iloc[:,:dim+1]
    return eVal,eVec
#---------------------------------------
def orthoFeats(dfX,varThres=.95):
    # с учетом кадра данных dfX признаков вычислить ортопризнаки dfP
    dfZ=dfX.sub(dfX.mean(),axis=1).div(dfX.std(),axis=1) # стандартизировать
    dot=pd.DataFrame(np.dot(dfZ.T,dfZ),index=dfX.columns,columns=dfX.columns)
    eVal,eVec=get_eVec(dot,varThres)
    dfP=np.dot(dfZ,eVec)
    return dfP
```

Помимо принятия мер к эффектам замещения, работа с ортогональными признаками дает два дополнительных преимущества: 1) ортогонализацию можно также использовать для сокращения размерности признаковой матрицы X, устраняя признаки, связанные с малыми собственными значениями. Это обычно ускоряет схождение машинно-обучающихся алгоритмов; 2) анализ проводится на признаках, предназначенных для объяснения структуры данных.

Позвольте мне подчеркнуть последний момент. Повсеместно распространенной проблемой является риск переподгонки. Алгоритмы МО всегда найдут закономерность, даже если эта закономерность является статистической случайностью. Вы всегда должны скептически относиться к предположительно важным признакам, определенным любым методом, в том числе анализом MDI, MDA и SFI. Теперь

предположим, что вы выводите ортогональные признаки, используя анализ главных компонент (PCA). Ваш анализ главных компонент определил, что некоторые признаки «главнее», чем другие, без какого-либо знания о метках (неконтролируемое обучение). То есть анализ главных компонент ранжировал признаки без какой-либо возможной переподгонки в классификационном смысле. Когда ваш анализ MDI, MDA или SFI отбирает в качестве наиболее важных (используя информацию о метках) те же признаки, которые анализ главных компонент выбрал в качестве главных (игнорируя информацию о метках), это представляет собой подтверждающее свидетельство того, что идентифицированная алгоритмом закономерность не совсем переподогнана. Если бы признаки были совершенно случайными, то ранжирование по главным компонентам не соответствовало бы ранжированию по важности признаков. Рисунок 8.1 иллюстрирует диаграмму рассеяния собственных значений, связанных с собственным вектором (ось x), в паре с MDI признака, связанного с собственным вектором (ось y). Корреляция Пирсона составляет 0.8491 (p-значение ниже 1E–150), свидетельствуя о том, что анализ главных компонент идентифицировал информативные признаки и ранжировал их правильно без переподгонки.

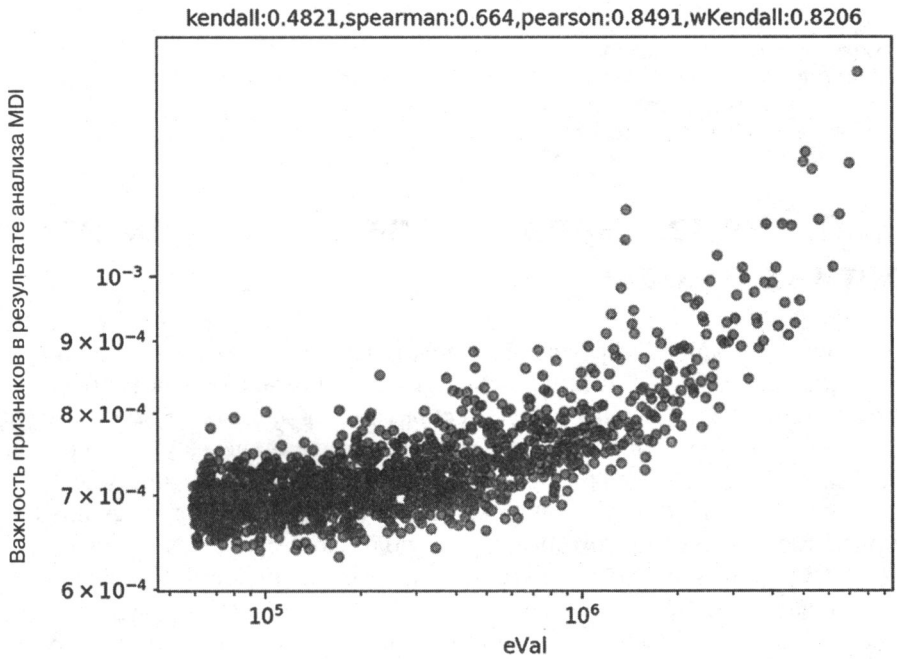


Рис. 8.1. Диаграмма рассеяния собственных значений (ось x) и уровней MDI (ось y) в двойной логарифмической шкале

Я считаю полезным вычислять взвешенный тау Кендалла[1] между важностями признаков и связанными с ними собственными значениями (или, что то же самое, их обратный ранг PCA). Чем ближе это значение к 1, тем сильнее согласованность между ранжированием по главным компонентам и ранжированием по важности признаков. Один из аргументов в пользу предпочтения взвешенного тау Кендалла над стандартным заключается в том, что мы хотим приоритизировать согласование ранга среди наиболее важных признаков. Нас не особо заботит ранговое согласование среди нерелевантных (вероятно шумных) признаков. Гиперболический взвешенный тау Кендалла для выборки на рис. 8.1 равен 0.8206.

Листинг 8.6 показывает, как вычислить эту корреляцию с помощью библиотеки scipy. В этом примере сортировка признаков по убыванию важности дает нам ранговую последовательность PCA, очень близкую к возрастающему списку. Поскольку функция `weightedtau` дает больший вес более высоким значениям, мы вычисляем корреляцию на обратном ранжировании PCA, `pcRank**-1`. Полученный взвешенный тау Кендалла относительно высок, на уровне 0.8133.

Листинг 8.6. Вычисление взвешенного тау Кендалла между важностью признаков и обратным ранжированием PCA

```
>>> import numpy as np
>>> from scipy.stats import weightedtau
>>> featImp=np.array([.55,.33,.07,.05]) # важность признаков
>>> pcRank=np.array([1,2,4,3]) # ранг PCA
>>> weightedtau(featImp,pcRank**-1.)[0]
```

8.5. Параллелизованная важность признаков против стековой

Существует по крайней мере два исследовательских подхода к выведению важности признаков. Во-первых, для каждой ценной бумаги i в инвестиционном универсуме $i = 1, ..., I$ мы формируем совокупность данных (X_i, y_i) и получаем важность признаков параллельно. Например, обозначим через $\lambda_{i,j,k}$ важность признака j на инструменте i по критерию k. Тогда мы можем агрегировать все результаты по всему универсуму, получив комбинированную $\Lambda_{j,k}$ важность признака j по критерию k. Признаки, которые важны по широкому спектру инструментов, с большей вероятностью будут связаны с базовым явлением, в особенности когда эти важности признаков демонстрируют высокоранговую корреляцию по всем критериям. Возможно, стоит углубленно изучить теоретический механизм, который делает эти признаки предсказательными. Главным преимуществом этого подхода является то, что он вычислительно быстр, так как может быть параллелизован. Недостатком является то, что из-за эффектов замещения важ-

[1] Коэффициент ранговой корреляции Кендалла. — *Примеч. науч. ред.*

ные признаки могут обменивать свои ранги по всем инструментам, увеличивая дисперсию оценочного $\lambda_{i,j,k}$. Этот недостаток становится относительно незначительным, если мы усредняем $\lambda_{i,j,k}$ по всем инструментам для достаточно крупного инвестиционного универсума.

Второй альтернативой является то, что я называю стековой укладкой признаков. Она состоит в стековой укладке всех совокупностей данных $\{(\tilde{X}_i, y_i)\}_{i=1,\ldots,I}$ в одну объединенную совокупность данных (X, y), где $\tilde{X}_i$ — это трансформированный экземпляр X_i (например, стандартизированный в скользящем за предыдущий период окне). Предназначение этой трансформации — обеспечить некоторую распределительную однородность, $\tilde{X}_i \sim X$. При таком подходе классификатор должен выучить, какие признаки важнее по всем инструментам одновременно, как если бы весь инвестиционный универсум был фактически единым инструментом. Стековая укладка признаков дарит несколько преимуществ: 1) классификатор будет подогнан на гораздо более крупном массиве данных, чем тот, который используется с параллелизованным (первым) подходом; 2) важность выводится напрямую, и для объединения результатов не требуется никакой взвешивающей схемы; 3) заключения более общие и менее смещены выбросами или переподгонкой; и 4) поскольку значимые балльные оценки не усредняются по всем инструментам, эффекты замещения не вызывают ослабление этих балльных оценок.

Я обычно предпочитаю стековую укладку признаков не только для важности признаков, но и всякий раз, когда классификатор может быть подогнан на множестве инструментов, в том числе с целью модельного предсказания. Это сокращает вероятность переподгонки оценщика к конкретному инструменту или малой совокупности данных. Главным недостатком стековой укладки является то, что она может потреблять много памяти и ресурсов, однако здесь пригодится хорошее знание методов высокопроизводительных вычислений (главы 20–22).

8.6. Эксперименты с синтетическими данными

В этом разделе мы протестируем, как эти методы выведения важности признаков реагируют на синтетические данные. Мы сгенерируем совокупность данных (X, y), состоящую из трех типов признаков:

1. Информативные: это признаки, которые используются для определения метки.
2. Избыточные: это случайные линейные комбинации информативных признаков. Они будут вызывать эффекты замещения.
3. Шумные: это признаки, которые не имеют никакого отношения к определению метки наблюдения.

Листинг 8.7 показывает, как генерировать синтетическую совокупность из 40 признаков, где 10 — информативные, 10 — избыточные и 20 — шумные, на 10 000 наблюдений. Для получения более подробной информации о том, как библиотека sklearn генерирует синтетические совокупности данных, посетите: http://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_classification.html.

Листинг 8.7. Создание синтетической совокупности данных

```
def getTestData(n_features=40,n_informative=10,n_redundant=10,n_
samples=10000):
    # сгенерировать случайную совокупность данных для классификационной задачи
    from sklearn.datasets import make_classification
    trnsX,cont=make_classification(n_samples=n_samples,n_features=n_features,
        n_informative=n_informative,n_redundant=n_redundant,random_state=0,
        shuffle=False)
    df0=pd.DatetimeIndex(periods=n_samples,freq=pd.tseries.offsets.BDay(),
                         end=pd.datetime.today())
    trnsX,cont=pd.DataFrame(trnsX,index=df0),
                           pd.Series(cont,index=df0).to_frame('bin')
    df0=['I_'+str(i) for i in xrange(n_informative)]+
        ['R_'+str(i) for i in xrange(n_redundant)]
    df0+=['N_'+str(i) for i in xrange(n_features-len(df0))]
    trnsX.columns=df0
    cont['w']=1./cont.shape[0]
    cont['t1']=pd.Series(cont.index,index=cont.index)
    return trnsX,cont
```

Учитывая, что мы точно знаем, какой признак принадлежит каждому классу, мы можем оценить, работают ли эти три метода выведения важности признаков, как было задумано. Теперь нам нужна функция, которая может выполнять каждый анализ на той же самой совокупности данных. Листинг 8.8 достигает этого, используя в качестве классификатора по умолчанию бэггированные деревья решений (глава 6).

Листинг 8.8. Вызов функции выведения значимости признаков для любого метода

```
def featImportance(trnsX,cont,n_estimators=1000,cv=10,
                   max_samples=1.,numThreads=24,
                   pctEmbargo=0,scoring='accuracy',method='SFI',
                   minWLeaf=0.,**kargs):
    # важность признаков из случайного леса
    from sklearn.tree import DecisionTreeClassifier
    from sklearn.ensemble import BaggingClassifier
    from mpEngine import mpPandasObj
    n_jobs=(-1 if numThreads>1 else 1) # выполнять 1 поток с ht_helper
                                       # в dirac1
    #1) подготовить классификатор, cv. max_features=1 для предотвращения
    # маскировки
    clf=DecisionTreeClassifier(criterion='entropy',max_features=1,
        class_weight='balanced',min_weight_fraction_leaf=minWLeaf)
```

```
    clf=BaggingClassifier(base_estimator=clf,n_estimators=n_estimators,
        max_features=1.,max_samples=max_samples,oob_score=True,n_jobs=n_jobs)
    fit=clf.fit(X=trnsX,y=cont['bin'],sample_weight=cont['w'].values)
    oob=fit.oob_score_
    if method=='MDI':
        imp=featImpMDI(fit,featNames=trnsX.columns)
        oos=cvScore(clf,X=trnsX,y=cont['bin'],cv=cv,sample_weight=cont['w'],
                    t1=cont['t1'],pctEmbargo=pctEmbargo,scoring=scoring).
                       mean()
    elif method=='MDA':
        imp,oos=featImpMDA(clf,X=trnsX,y=cont['bin'],cv=cv,
                sample_weight=cont['w'],
                           t1=cont['t1'],pctEmbargo=pctEmbargo,scoring=scoring)
    elif method=='SFI':
        cvGen=PurgedKFold(n_splits=cv,t1=cont['t1'],pctEmbargo=pctEmbargo)
        oos=cvScore(clf,X=trnsX,y=cont['bin'],sample_weight=cont['w'],
                    scoring=scoring,
            cvGen=cvGen).mean()
        clf.n_jobs=1 # paralellize auxFeatImpSFI rather than clf
        imp=mpPandasObj(auxFeatImpSFI,('featNames',trnsX.columns),numThreads,
            clf=clf,trnsX=trnsX,cont=cont,scoring=scoring,cvGen=cvGen)
    return imp,oob,oos
```

Наконец, нам нужна главная функция для вызова всех компонентов, от генерирования данных и анализа важности признаков до сбора и обработки результатов. Эти задачи выполняются листингом 8.9.

Листинг 8.9. Вызов всех компонентов

```
def testFunc(n_features=40,n_informative=10,n_redundant=10,n_estimators=1000,
             n_samples=10000,cv=10):
    # Протестировать результативность методов выведения важности признаков
    # на синтетических данных
    # Число шумных признаков = n_features - n_informative - n_redundant
    trnsX,cont=getTestData(n_features,n_informative,n_redundant,n_samples)
    dict0={'minWLeaf':[0.],'scoring':['accuracy'],'method':['MDI','MDA','SFI'],
           'max_samples':[1.]}
    jobs,out=(dict(izip(dict0,i)) for i in product(*dict0.values())),[]
    kargs={'pathOut':'./testFunc/','n_estimators':n_estimators,
           'tag':'testFunc','cv':cv}
    for job in jobs:
        job['simNum']=job['method']+'_'+job['scoring']+'_'+'%.2f'%job
                          ['minWLeaf']+ \
                      '_'+str(job['max_samples'])
        print job['simNum']
        kargs.update(job)
        imp,oob,oos=featImportance(trnsX=trnsX,cont=cont,**kargs)
        plotFeatImportance(imp=imp,oob=oob,oos=oos,**kargs)
        df0=imp[['mean']]/imp['mean'].abs().sum()
        df0['type']=[i[0] for i in df0.index]
        df0=df0.groupby('type')['mean'].sum().to_dict()
        df0.update({'oob':oob,'oos':oos});df0.update(job)
```

```
        out.append(df0)
    out=pd.DataFrame(out).sort_values(['method','scoring','minWLeaf',
                                       'max_samples'])
    out=out['method','scoring','minWLeaf','max_samples','I','R','N','oob',
            'oos']
    out.to_csv(kargs['pathOut']+'stats.csv')
    return
```

Для эстетически настроенных читателей листинг 8.10 обеспечивает неплохую компоновку для построения графика важностей признаков.

Листинг 8.10. Функция построения графика важностей признаков

```
def plotFeatImportance(pathOut,imp,oob,oos,method,tag=0,simNum=0,**kargs):
    # построить график средневажностных баров со стандартным отклонением
    mpl.figure(figsize=(10,imp.shape[0]/5.))
    imp=imp.sort_values('mean',ascending=True)
    ax=imp['mean'].plot(kind='barh',color='b',alpha=.25,xerr=imp['std'],
    error_kw={'ecolor':'r'})
    if method=='MDI':
        mpl.xlim([0,imp.sum(axis=1).max()])
        mpl.axvline(1./imp.shape[0],linewidth=1,color='r',linestyle='dotted')
    ax.get_yaxis().set_visible(False)
    for i,j in zip(ax.patches,imp.index):ax.text(i.get_width()/2,
                  i.get_y()+i.get_height()/2,j,ha='center',va='center',
                  color='black')
    mpl.title('tag='+tag+' | simNum='+str(simNum)+' | oob='+str(round(oob,4))+
              ' | oos='+str(round(oos,4)))
    mpl.savefig(pathOut+'featImportance_'+str(simNum)+'.png',dpi=100)
    mpl.clf();mpl.close()
    return
```

Рисунок 8.2 показывает результаты анализа MDI. Для каждого признака горизонтальная полоса указывает среднее снижение в примесности (MDI) во всех деревьях принятия решений, а горизонтальная линия — среднеквадратическое отклонение этого среднего. Поскольку значения MDI в сумме дают 1, если все признаки одинаково важны, каждое значение будет составлять 1/40. Вертикальная пунктирная линия маркирует этот порог в 1/40, отделяя признаки, важность которых превышает то, что можно было бы ожидать от неразличимых признаков. Как вы можете видеть, анализ MDI делает очень хорошую работу с точки зрения размещения всех информативных и избыточных признаков над красной пунктирной линией, за исключением R_5, который не прошел отбор с небольшим отрывом. Эффекты замещения приводят к тому, что некоторые информативные или избыточные признаки ранжируются лучше, чем другие, что и ожидалось.

Рисунок 8.3 показывает, что анализ MDA также проделал хорошую работу. Результаты согласуются с результатами анализа MDI в том смысле, что все информированные и избыточные признаки ранжируются лучше, чем шумные признаки,

за исключением R_6, вероятно, из-за эффекта замещения. Одним из не столь положительных аспектов анализа MDA является то, что среднеквадратическое отклонение от средних несколько выше, хотя это можно было бы решить путем увеличения числа разделов в прочищенной k-блочной перекрестной проверке, скажем, от 10 до 100 (ценой 10-кратного увеличения времени вычисления без параллелизации).

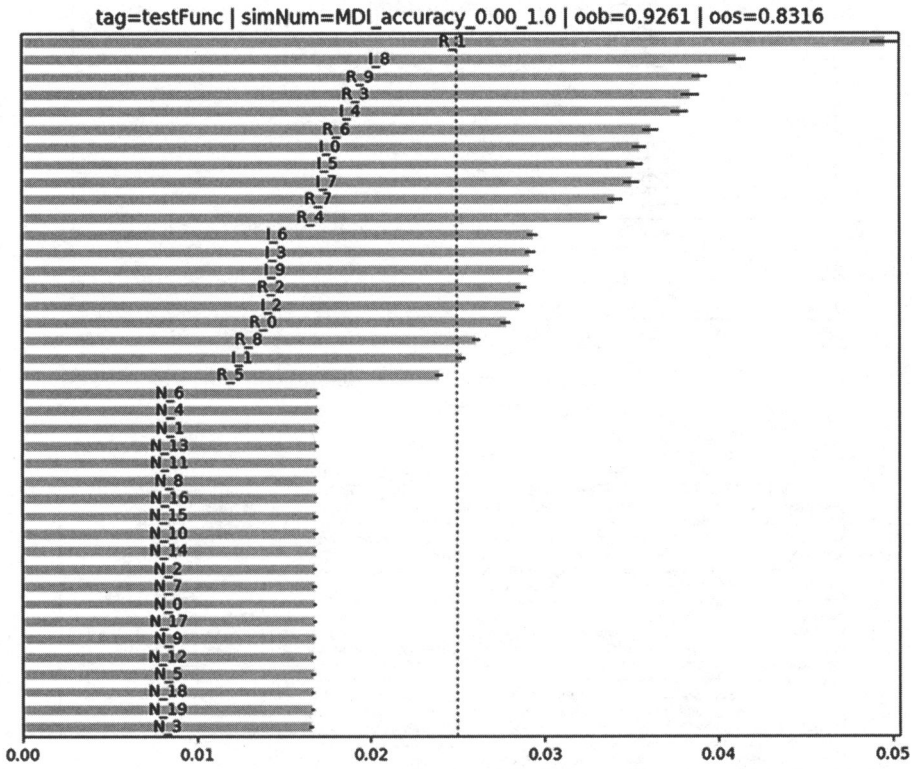


Рис. 8.2. Важность признаков на основе анализа MDI, вычисленная на синтетических данных

Рисунок 8.4 показывает, что анализ SFI также делает достойную работу; однако несколько важных признаков ранжируются хуже, чем шум (I_6, I_2, I_9, I_1, I_3, R_5), вероятно, из-за совместных эффектов.

Метки являются функцией сочетания признаков, и попытка спрогнозировать их независимо пропускает совместные эффекты. Тем не менее анализ SFI полезен в качестве дополнения к процедурам анализа MDI и MDA, именно потому, что оба типа анализа страдают от проблем разного рода.

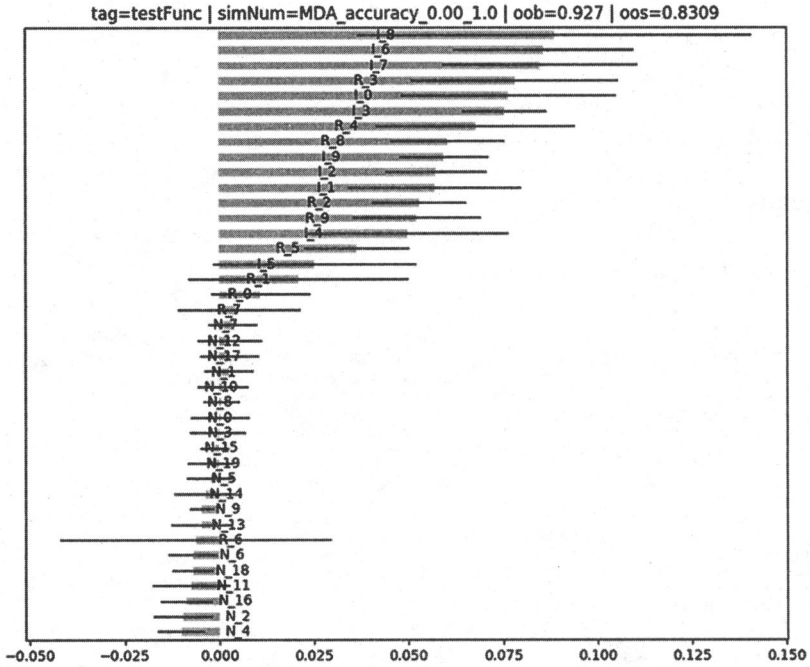


Рис. 8.3. Важность признаков на основе анализа MDA, вычисленная на синтетических данных

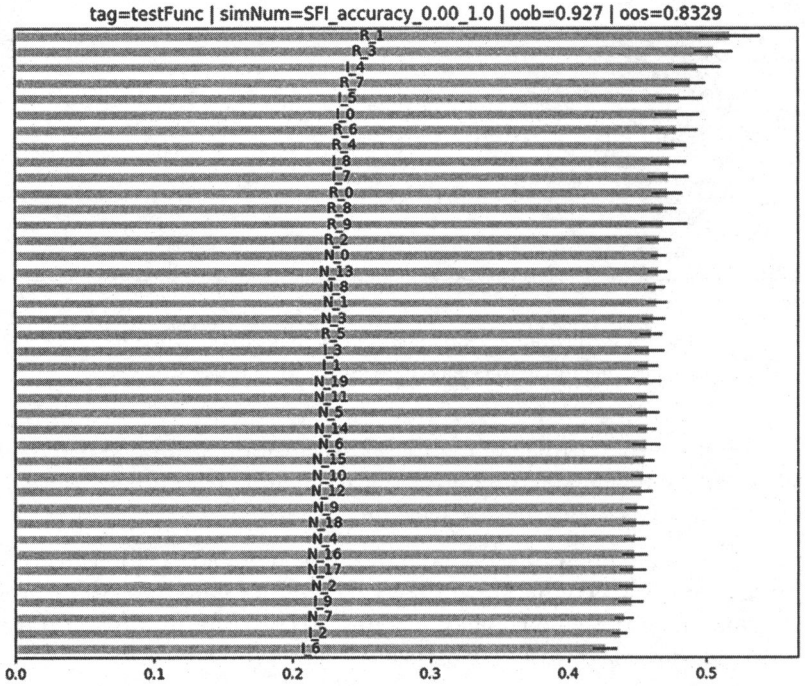


Рис. 8.4. Важность признаков на основе анализа SFI, вычисленная на синтетических данных

Упражнения

8.1. Используя исходный код, представленный в разделе 8.6:

(а) Сгенерируйте совокупность данных (X, y).

(б) Примените трансформацию из анализа PCA на матрице X, получив результат, который мы обозначим через $\dot{X}$.

(в) Рассчитайте ССКП-, ССПДС- и ЗОП-значимости признаков на основе $(\dot{X}, y)$, где основной классификатор — это алгоритм случайного леса.

(г) Согласуются ли эти три метода? Какие признаки важны? Почему?

8.2. Из упражнения 8.1 сгенерируйте новую совокупность данных $(\ddot{X}, y)$, где $\ddot{X}$ — это признаковое объединение X и $\dot{X}$.

(а) Вычислите важности признаков на основе анализа MDI, MDA и SFI на $(\ddot{X}, y)$, где базовый оценщик — это случайный лес.

(б) Нашли ли эти три метода общее решение относительно важных признаков? Почему?

8.3. Возьмите результаты из упражнения 8.2.

(а) Отбросьте самые важные признаки в соответствии с каждым методом, в результате чего получится признаковая матрица $\ddot{X}$.

(б) Вычислите важности признаков на основе анализа MDI, MDA и SFI на $(\ddot{X}, y)$, где базовый оценщик — это случайный лес.

(в) Отмечаете ли вы значительные изменения в ранжированиях важности признаков относительно результатов упражнения 8.2?

8.4. Используя исходный код, представленный в разделе 8.6:

(а) Сгенерируйте совокупность данных (X, y) из 1E6 наблюдений, в котором 5 признаков — информативные, 5 — избыточные и 10 — шумные.

(б) Разделите (X, y) на 10 подмножеств данных $\{(X_i, y_i)\}_{i=1,\ldots,10}$, каждое из 1E5 наблюдений.

(в) Вычислите параллелизованную важность признаков (раздел 8.5) на каждом из 10 подмножеств данных $\{(X_i, y_i)\}_{i=1,\ldots,10}$.

(г) Вычислите стековую важность признаков на объединенной совокупности данных (X, y).

(д) Что вызывает несоответствие между ними? Какой из них более надежный?

8.5. Повторите все расчеты анализа MDI из упражнений 1–4, но на этот раз допустите маскирующий эффект. Это означает, что не следует устанавливать `max_features=int(1)` в листинге 8.2. Чем отличаются результаты вследствие этого изменения? Почему?

9 Регулировка гиперпараметров с помощью перекрестной проверки

9.1. Актуальность

Регулировка гиперпараметров является важным шагом в подгонке обучающегося алгоритма. При ее ненадлежащем выполнении алгоритм, скорее всего, будет переподогнан, и живое исполнение разочарует. В литературе по машинному обучению особое внимание уделяется выполнению перекрестной проверки любого отрегулированного гиперпараметра. Как мы видели в главе 7, перекрестная проверка в финансах является особенно сложной задачей, где решения из других областей, вероятно, окажутся безуспешными. В данной главе мы обсудим способы регулировки гиперпараметров с помощью метода прочищенной k-блочной перекрестной проверки. В разделе справочных материалов перечислены исследования, предлагающие альтернативные методы, которые могут быть полезны в конкретных задачах.

9.2. Перекрестная проверка с помощью решеточного поиска

Перекрестная проверка с помощью решеточного поиска выполняет исчерпывающий поиск такого параметрического сочетания, которое будет максимизировать перекрестно-проверочную результативность в соответствии с некой определяемой пользователем оценивающей функцией[1]. Когда мы почти ничего не знаем о лежащей в основе структуре данных, это будет разумным первым подходом. В библиотеке scikit-learn эта логика реализована в функции `GridSearchCV`, которая в качестве аргумента принимает генератор перекрестной проверки. По причинам, описанным в главе 7, нам нужно передать наш класс `PurgedKFold` (листинг 7.3) для

[1] Решеточный поиск (grid search), или поиск в решетке гиперпараметров, предусматривает, что для каждого гиперпараметра заранее подбирается список значений, которые могут оказаться для него хорошими, затем пишется вложенный цикл for, который пробует все комбинации этих значений с целью найти их контрольные точности и отслеживает те, которые показывают наилучшую результативность. — *Примеч. науч. ред.*

недопущения того, чтобы функция GridSearchCV выполнила переподгонку обучающегося оценщика к информации, полученной в результате утечки.

Листинг 9.1. Решеточный поиск с помощью прочищенной *k*-блочной перекрестной проверки

```
def clfHyperFit(feat,lbl,t1,pipe_clf,param_grid,cv=3,bagging=[0,None,1.],
                n_jobs=-1,pctEmbargo=0,**fit_params):
    if set(lbl.values)=={0,1}: scoring='f1' # f1 для метамаркировки
    else: scoring='neg_log_loss' # симметричное по отношению ко всем случаям
    #1) гиперпараметрический поиск на тренировочных данных
    inner_cv=PurgedKFold(n_splits=cv,t1=t1,pctEmbargo=pctEmbargo) # прочищено
    gs=GridSearchCV(estimator=pipe_clf,param_grid=param_grid,
                    scoring=scoring,cv=inner_cv,n_jobs=n_jobs,iid=False)
    gs=gs.fit(feat,lbl,**fit_params).best_estimator_ # конвейер
    #2) выполнить подгонку проверенной модели на всей совокупности данных
    if bagging[1]>0:
        gs=BaggingClassifier(base_estimator=MyPipeline(gs.steps),
                             n_estimators=int(bagging[0]),
                             max_samples=float(bagging[1]),
                             max_features=float(bagging[2]),n_jobs=n_jobs)
        gs=gs.fit(feat,lbl,sample_weight=fit_params \
                  [gs.base_estimator.steps[-1][0]+'__sample_weight'])
        gs=Pipeline([('bag',gs)])
    return gs
```

Листинг 9.1 показывает функцию clfHyperFit, которая реализует прочищенный решеточный поиск GridSearchCV. Аргумент fit_params может использоваться для передачи аргумента sample_weight, а аргумент param_grid содержит значения, которые будут объединены в решетку. Вдобавок эта функция допускает бэггирование отрегулированного оценщика. Бэггирование оценщика, как правило, является хорошей идеей по причинам, описанным в главе 6, и в вышеуказанной функции для этой цели встроена соответствующая логика.

Я советую вам использовать аргумент scoring='f1' в контексте приложений метамаркировки по следующей причине. Предположим, что у нас есть выборка с очень большим числом отрицательных (то есть с метками '0') случаев. Классификатор, который предсказывает, что все случаи являются отрицательными, добьется высокой точности 'accuracy' или отрицательной логарифмической потери 'neg_log_loss', хотя он не научился на признаках, как проводить различие между случаями. По сути дела, такая модель достигает нулевой полноты и неопределенной точности (см. главу 3, раздел 3.7). Балльная оценка 'f1' делает поправку на взвинчивание результативности, оценивая классификатор с точки зрения точности и полноты (см. главу 14, раздел 14.8).

Что касается других (не метамаркировочных) приложений, то вполне нормально использовать балльные оценки 'accuracy' или 'neg_log_loss', потому что мы в равной степени заинтересованы в предсказании всех случаев. Обратите внимание, что перемаркировка случаев не влияет на 'accuracy' или 'neg_log_loss', но влияет на f1.

В этом примере показано одно досадное ограничение конвейеров библиотеки sklearn: их метод `fit` не ожидает на входе аргумент `sample_weight`. Вместо этого он ожидает именованный аргумент `fit_params`. Этот дефект был зарегистрирован на GitHub; однако на его исправление может уйти продолжительное время, так как оно предусматривает переписывание исходного кода и тестирование обширной функциональности. Не стесняйтесь использовать обходной путь в листинге 9.2. Он создает новый класс, именуемый `MyPipeline`, который наследует все методы из конвейера библиотеки sklearn. Он перезаписывает унаследованный метод `fit` новым, который обрабатывает аргумент `sample_weight`, после чего перенаправляет его в родительский класс.

Листинг 9.2. Расширенный класс Pipeline

```
class MyPipeline(Pipeline):
    def fit(self,X,y,sample_weight=None,**fit_params):
        if sample_weight is not None:
            fit_params[self.steps[-1][0]+'__sample_weight']=sample_weight
    return super(MyPipeline,self).fit(X,y,**fit_params)
```

Если с этим приемом расширения классов вы не знакомы, то можете прочитать вот этот вводный пост на Stackoverflow: http://stackoverflow.com/questions/576169/understanding-python-super-with-init-methods.

9.3. Перекрестная проверка с помощью рандомизированного поиска

Для обучающихся алгоритмов с большим числом параметров перекрестная проверка с помощью решеточного поиска становится вычислительно неразрешимой. В этом случае альтернативой с хорошими статистическими свойствами является отбор каждого параметра из распределения (Begstra и соавт. [2011, 2012]). Данный подход имеет два преимущества. Во-первых, мы можем контролировать число комбинаций, которые будем искать, независимо от размерности задачи (эквивалент вычислительного бюджета). Во-вторых, наличие параметров, которые относительно нерелевантны с точки зрения результативности, не приведет к существенному увеличению времени поиска, как в случае с перекрестной проверкой с помощью решеточного поиска.

Вместо того чтобы писать новую функцию для работы с рандомизированным поиском, `RandomizedSearchCV`, давайте расширим листинг 9.1, встроив еще одну альтернативу для этой цели. Возможная реализация приведена в листинге 9.3.

Листинг 9.3. Прочищенная k-блочная перекрестная проверка с помощью рандомизированного поиска

```
def clfHyperFit(feat,lbl,t1,pipe_clf,param_grid,cv=3,bagging=[0,None,1.],
                rndSearchIter=0,n_jobs=-1,pctEmbargo=0,**fit_params):
```

```
    if set(lbl.values)=={0,1}: scoring='f1' # f1 для метамаркировки
    else: scoring='neg_log_loss' # симметричное по отношению ко всем случаям
    #1) гиперпараметрический поиск на тренировочных данных
    inner_cv=PurgedKFold(n_splits=cv,t1=t1,pctEmbargo=pctEmbargo) # прочищено
    if rndSearchIter==0:
        gs=GridSearchCV(estimator=pipe_clf,param_grid=param_grid,
                        scoring=scoring,cv=inner_cv,n_jobs=n_jobs,iid=False)
    else:
        gs=RandomizedSearchCV(estimator=pipe_clf,param_distributions= \
                              param_grid,scoring=scoring,cv=inner_cv,n_
                              jobs=n_jobs, iid=False,n_iter=rndSearchIter)
        gs=gs.fit(feat,lbl,**fit_params).best_estimator_ # конвейер
    #2) выполнить подгонку проверенной модели на всей совокупности данных
    if bagging[1]>0:
        gs=BaggingClassifier(base_estimator=MyPipeline(gs.steps),
                             n_estimators=int(bagging[0]),
                             max_samples=float(bagging[1]),
                             max_features=float(bagging[2]),n_jobs=n_jobs)
        gs=gs.fit(feat,lbl,sample_weight=fit_params \
                  [gs.base_estimator.steps[-1][0]+'__sample_weight'])
        gs=Pipeline([('bag',gs)])
    return gs
```

9.3.1. Логарифмически равномерное распределение

Обычно некоторые алгоритмы МО принимают только неотрицательные гиперпараметры. Так обстоит дело с некоторыми очень популярными параметрами, такими как `C` в опорно-векторном классификаторе (SVC) и `gamma` в ядре RBF[1]. Мы могли бы взять случайные числа из равномерного распределения, ограниченного между 0 и некоторым крупным значением, скажем 100. Это будет означать, что 99 % значений ожидаемо будет больше 1. Это не обязательно самый эффективный способ разведать допустимый участок параметров, функции которых не откликаются линейно. Например, опорно-векторный классификатор (SVC) может одинаково откликаться на увеличение `C` от 0.01 до 1, как и на увеличение `C` от 1 до 100[2]. Поэтому отбор образцов `C` из $U[0, 100]$ (равномерного) распределения будет неэффективным. В этих случаях более эффективным выглядит получение значений из распределения, где логарифм этих взятий будет распределен равномерно. Я называю это «логарифмически равномерным распределением», и поскольку я не смог найти это распределение в литературе, я должен определить его правильно.

Случайная величина x подчиняется логарифмически равномерному распределению между $a > 0$ и $b > a$, если и только если $\log[x] \sim U[\log[a], \log[b]]$. Это распределение имеет кумулятивную функцию распределения (CDF)

[1] Параметр C — это параметр регуляризации в опорно-векторном классификаторе (support-vector classifier, SVC), и gamma — ширина ядра радиально-базисной функции (radial basis function, RBF). См. http://scikit-learn.org/stable/modules/metrics.html.

[2] См. http://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html.

$$F(x)=\begin{cases}\dfrac{\log[x]-\log[a]}{\log[b]-\log[a]} & \text{для } a\le x\le b\\ 0 & \text{для } x<a\\ 1 & \text{для } x>b\end{cases}$$

Из него мы выводим функцию плотности вероятности (PDF):

$$f(x)=\begin{cases}\dfrac{1}{x\log[b/a]} & \text{для } a\le x\le b\\ 0 & \text{для } x<a\\ 0 & \text{для } x>b\end{cases}$$

Обратите внимание, что кумулятивная функция распределения инвариантна относительно основания логарифма, поскольку $\dfrac{\log\left[\dfrac{x}{a}\right]}{\log\left[\dfrac{b}{a}\right]}=\dfrac{\log_c\left[\dfrac{x}{a}\right]}{\log_c\left[\dfrac{b}{a}\right]}$ для любого основания c, следовательно, случайная величина не является функцией от c. Листинг 9.4 реализует (и тестирует) в модуле `scipy.stats` случайную величину, где $[a, b] = [1E{-}3, 1E3]$, отсюда $\log[x] \sim U[\log[1E{-}3], \log[1E3]]$. Рисунок 9.1 иллюстрирует равномерность образцов в логарифмической шкале.

Листинг 9.4. Класс logUniform_gen

```
import numpy as np,pandas as pd,matplotlib.pyplot as mpl
from scipy.stats import rv_continuous,kstest
#--------------------------------------
class logUniform_gen(rv_continuous):
    # случайные числа, логарифмически равномерно распределенные между 1 и e
    def _cdf(self,x):
        return np.log(x/self.a)/np.log(self.b/self.a)
def logUniform(a=1,b=np.exp(1)):
    return logUniform_gen(a=a,b=b,name='logUniform')
#--------------------------------------
a,b,size=1E-3,1E3,10000
vals=logUniform(a=a,b=b).rvs(size=size)
print kstest(rvs=np.log(vals),cdf='uniform',args=(np.log(a),
            np.log(b/a)),N=size)
print pd.Series(vals).describe()
mpl.subplot(121)
pd.Series(np.log(vals)).hist()
mpl.subplot(122)
pd.Series(vals).hist()
mpl.show()
```

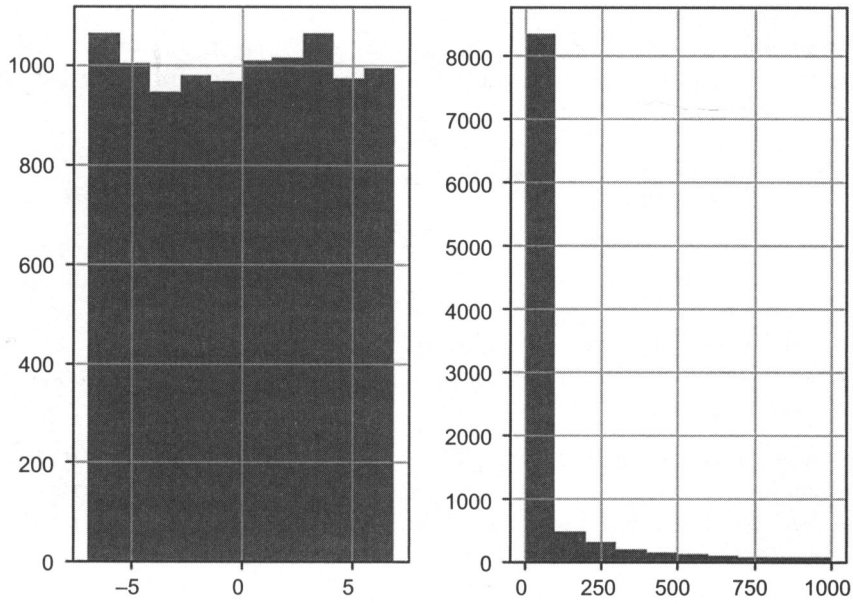


Рис. 9.1. Результаты тестирования класса logUniform_gen

9.4. Балльное оценивание и регулировка гиперпараметров

Листинги 9.1 и 9.3 устанавливают аргумент `scoring='fi'` для метамаркировочных приложений. В случае других приложений вместо стандартного `scoring='accuracy'` они устанавливают аргумент `scoring='neg_log_loss'`. Хотя точность (`accuracy`) имеет более интуитивно понятную интерпретацию, я предлагаю при регулировке гиперпараметров для инвестиционной стратегии использовать отрицательную логарифмическую потерю (`neg_log_loss`). Давайте я объясню свои рассуждения.

Предположим, что ваша обучающаяся инвестиционная стратегия с высокой вероятностью предсказывает, что вам следует приобрести некую ценную бумагу. Вы входите в большую длинную позицию, как функцию от степени достоверности стратегии. Если предсказание было ошибочным и рынок активно распродает, то вы потеряете много денег. И тем не менее точность одинаково объясняет и ошибочное высоковероятностное предсказание покупки, и ошибочное низковероятностное предсказание покупки. Более того, точность может нивелировать высоковероятностный промах низковероятностным попаданием.

Инвестиционные стратегии получают прибыль от предсказания правильной метки с высокой достоверностью. Выгоды от хороших низкодостоверных предсказаний недостаточны для компенсации убытков от плохих высокодостоверных предсказаний. По этой причине точность не обеспечивает реалистичную оценку результативности

классификатора. И наоборот, логарифмическая потеря[1] (так называемая перекрестно-энтропийная потеря) вычисляет логарифмическую вероятность классификатора при заданной истинной метке, которая учитывает вероятности в предсказаниях. Логарифмическая потеря может оцениваться следующим образом:

$$L[Y, P] = -\log[\text{Prob}[Y|P]] = -N^{-1}\sum_{n=0}^{N-1}\sum_{k=0}^{K-1} y_{n,k}\log[p_{n,k}],$$

где

- $p_{n,k}$ — это вероятность, связанная с предсказанием n метки k;
- Y — это двоичная индикаторная матрица «1-из-K» такая, что $y_{n,k} = 1$, когда наблюдению n была назначена метка k из K возможных меток, либо 0 в противном случае.

Предположим, что классификатор предсказывает две единицы, причем истинные метки равны 1 и 0. Первое предсказание является попаданием, а второе предсказание является промахом, при этом точность составляет 50 %. На рис. 9.2 показана перекрестно-энтропийная потеря, когда эти предсказания поступают из вероятностей в диапазоне [0.5, 0.9]. Можно заметить, что в правой части рисунка

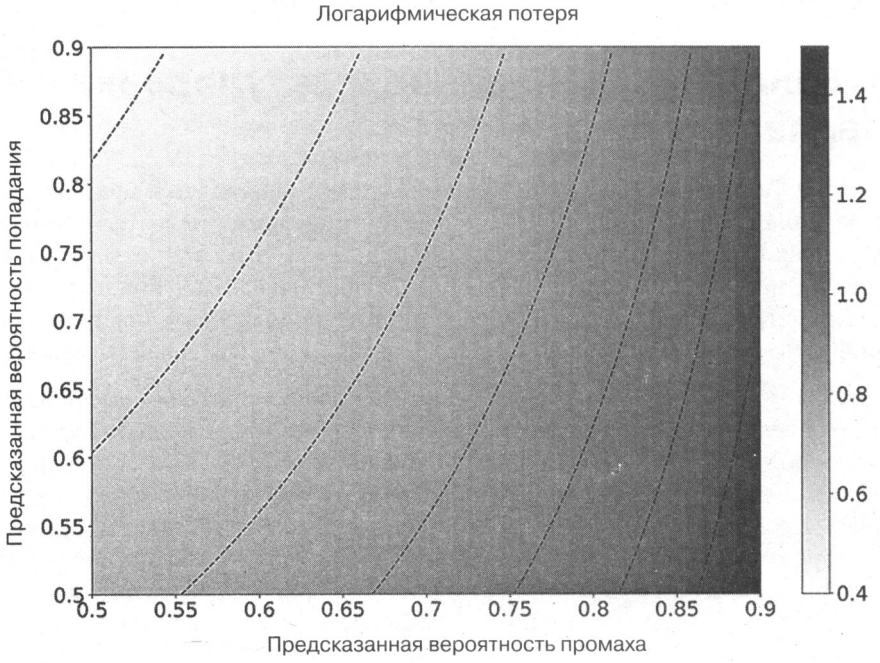


Рис. 9.2. Логарифмическая потеря как функция предсказанных вероятностей попадания и промаха

[1] См. http://scikit-learn.org/stable/modules/model_evaluation.html#log-loss.

логарифмическая потеря велика из-за промахов с большой вероятностью, хотя точность во всех случаях составляет 50 %.

Существует вторая причина предпочесть перекрестно-энтропийную потерю точности. Перекрестная проверка оценивает классификатор, применяя веса выборки (см. главу 7, раздел 7.5). Как вы помните из главы 4, веса наблюдений определялись как функция абсолютного финансового возврата у наблюдения. Из этого вытекает, что взвешенная по выборке перекрестно-энтропийная потеря оценивает результативность классификатора с точки зрения переменных, участвующих в калькуляции PnL (прибыли и убытков, пересчитанных по текущим рыночным ценам): она использует правильную метку для стороны позиции, вероятность для размера позиции и вес выборки для финансового возврата/исхода наблюдения. Именно этот метрический показатель результативности машинного обучения, а не точность, является правильным для гиперпараметрической регулировки финансовых приложений.

Когда мы используем логарифмическую потерю в качестве оценивающего статистического показателя, мы часто предпочитаем менять ее знак, отсюда и ссылка на «neg log loss» (на отрицательную логарифмическую потерю). Причина этого изменения — чисто косметическая, обусловленная интуицией: высокое значение отрицательной логарифмической потери предпочтительнее низкого, так же как и с точностью. При использовании аргумента `neg_log_loss` имейте в виду следующий дефект библиотеки sklearn: https://github.com/scikit-learn/scikit-learn/issues/9144. Для того чтобы обойти этот дефект, вы должны использовать функцию `cvScore`, представленную в главе 7.

Упражнения

9.1. Используя функцию `getTestData` из главы 8, сформируйте синтетическую совокупность данных из 10 000 наблюдений с 10 признаками, из которых пять — информативные и пять — шумные.

(а) Примените решеточный поиск `GridSearchCV` на 10-блочной перекрестной проверке, для того чтобы найти оптимальные гиперпараметры `C` и `gamma` на опорно-векторном классификаторе (SVC) с ядром RBF, где `param_grid={'C':[1E-2,1E-1,1,10,100],'gamma':[1E-2,1E-1,1,10,100]}` и оценивающей функцией является отрицательная логарифмическая потеря `neg_log_loss`.

(б) Сколько узлов получилось в сетке?

(в) Сколько подгонок потребовалось выполнить, чтобы найти оптимальное решение?

(г) Сколько времени ушло на поиск этого решения?

(д) Как получить оптимальный результат?

(е) Какова перекрестно-проверочная оценка оптимального сочетания параметров?

(ж) Как передавать веса выборки в опорно-векторный классификатор SVC?

9.2. Используя совокупность данных из упражнения 9.1:

(а) Примените `RandomizedSearchCV` на 10-блочной перекрестной проверке, для того чтобы найти оптимальные гиперпараметры `C` и `gamma` на опорно-векторном классификаторе (SVC) с ядром RBF, где `param_distributions= 'C':logUniform(a = 1E-2,b = 1E2),'gamma':logUniform(a = 1E-2,b = = 1E2)},n_iter=25`, а оценивающей функцией является `neg_log_loss`.

(б) Сколько времени ушло на поиск этого решения?

(в) Является ли оптимальное сочетание параметров аналогичным найденному в упражнении 9.1?

(г) Какова перекрестно-проверочная оценка оптимального сочетания параметров? Как она соотносится с перекрестно-проверочной оценкой из упражнения 9.1?

9.3. Из упражнения 9.1:

(а) Вычислите коэффициент Шарпа результирующих внутривыборочных прогнозов из пункта 9.1.а (см. главу 14 и определение понятия «коэффициент Шарпа»).

(б) Повторите пункт 9.1.а, используя скоринг-функцию accuracy. Высчитайте прогнозы в пределах выборки, выведенные из гипернастроенных параметров.

(в) Какой метод оценки привел к более высокому коэффициенту Шарпа?

9.4. Из упражнения 9.2:

(а) Рассчитайте коэффициент Шарпа получившихся прогнозов в пределах выборки, из пункта 9.2.а.

(б) Повторите пункт 9.1.а, в этот раз с точностью `accuracy` в качестве оценивающей функции. Вычислите внутривыборочные прогнозы, исходя из отрегулированных гиперпараметров.

(в) Какой метод оценки привел к более высокому коэффициенту Шарпа?

9.5. Прочитайте определение логарифмической функции потерь, $L[Y, P]$.

(а) Почему оценивающая функция `neg_log_loss` определяется как отрицательная логарифмическая потеря $-L[Y, P]$?

(б) Каким будет результат максимизации логарифмической потери вместо отрицательной логарифмической потери?

9.6. Рассмотрим инвестиционную стратегию, которая устанавливает размер ставок одинаково, независимо от достоверности в прогнозе. Какова в этом случае будет более подходящая оценивающая функция для регулировки гиперпараметров, точности или перекрестно-энтропийной потери?

Часть 3
БЭКТЕСТИРОВАНИЕ

10 Выставление размера ставки

10.1. Актуальность

Между стратегическими играми и инвестированием существуют увлекательные параллели. Некоторые лучшие портфельные менеджеры, с которыми я работал, — отличные игроки в покер, возможно, даже больше, чем шахматисты. Одна из причин — выставление размеров ставок, для которых техасский холдем[1] обеспечивает отличный аналог и тренировочную площадку. Ваш алгоритм МО может достигать высокой точности, но если вы не выставляете размеры ваших ставок как положено, то ваша инвестиционная стратегия неизбежно потеряет деньги. В этой главе мы рассмотрим несколько подходов к выставлению размеров ставок из прогнозов, полученных алгоритмами.

10.2. Независимые от стратегии подходы к выставлению размеров

Рассмотрим две стратегии на одинаковом инструменте. Пусть $m_{i,t} \in [-1, 1]$ равно ставочному размеру стратегии i в момент времени t, где $m_{i,t} = -1$ обозначает полную короткую позицию и $m_{i,t} = 1$ обозначает полную длинную позицию. Предположим, что одна стратегия произвела последовательность ставочных размеров $[m_{1,1}, m_{1,2}, m_{1,3}] = [.5, 1, 0]$, так как рыночная цена шла в последовательности $[p_1, p_2, p_3] = [1, .5, 1.25]$, где p_t — это цена в момент времени t. Другая стратегия произвела последовательность $[m_{2,1}, m_{2,2}, m_{2,3}] = [1, .5, 0]$, так как она была вынуждена сократить размер своей ставки, как только рынок двинулся против начальной полной позиции. Обе стратегии дали прогнозы, которые оказались правильными (цена выросла на 25 % между p_1 и p_3), однако первая стратегия заработала деньги (0.5), а вторая стратегия их потеряла (−.125).

---

[1] Техасский холдем (Texas hold 'em) — самая популярная разновидность покера, игра с двумя карманными и пятью общими картами, используемыми всеми игроками при составлении комбинаций. — *Примеч. науч. ред.*

Мы бы предпочли выставить размер позиции таким образом, чтобы оставить немного наличности и дать возможность торговому сигналу усилиться перед тем, как он ослабеет. Одним из вариантов является вычисление ряда $c_t = c_{t,l} - c_{t,s}$, где $c_{t,l}$ — это число одновременных ставок в момент времени t и $c_{t,s}$ — число одновременных коротких ставок в момент времени t. Эта одновременность ставок выводится для каждой стороны, подобно тому как мы вычисляли одновременность меток в главе 4 (вспомните объект `t1` с накладывающимися временами существования). Мы выполняем подгонку смеси двух гауссиан на $\{c_t\}$, применяя метод, подобный описанному в публикации Lopez de Prado и Foreman [2014]. Тогда размер ставки выводится как

$$m_t = \begin{cases} \dfrac{F[c_t] - F[0]}{1 - F[0]} & \text{если } c_t \geq 0 \\ \dfrac{F[c_t] - F[0]}{F[0]} & \text{если } c_t < 0. \end{cases}$$

где $F[x]$ — это кумулятивная функция распределения (CDF) подогнанной смеси двух гауссиан для значения x. Например, мы можем выставить размер ставки, равный 0.9, когда вероятность наблюдения сигнала большего значения равна 0.1. Чем сильнее сигнал, тем меньше вероятность того, что сигнал станет еще сильнее, следовательно, тем больше размер ставки.

Второе решение — следовать бюджетному подходу. Мы вычисляем максимальное число (либо некоторый другой квантиль) одновременных длинных ставок, $\max_i\{c_{i,l}\}$, и максимальное число одновременных коротких ставок, $\max_i\{c_{i,s}\}$. Затем мы выводим размер ставки как $m_t = c_{t,l}\dfrac{1}{\max_i\{c_{i,l}\}} - c_{t,s}\dfrac{1}{\max_i\{c_{i,s}\}}$, где $c_{t,l}$ — это число одновременных длинных ставок в момент времени t и $c_{t,s}$ — число одновременных коротких ставок в момент времени t. Цель состоит в том, чтобы максимальная позиция не достигалась до того, как будет запущен последний одновременный сигнал.

Третий подход состоит в применении метамаркировки, как мы объяснили в главе 3. Мы выполняем подгонку классификатора, такого как опорно-векторный классификатор (SVC) или классификатор на основе случайного леса (RF), для того чтобы определить вероятность неправильного отнесения к классу и использовать эту вероятность для получения размера ставки[1]. Данный подход имеет несколько преимуществ. Во-первых, алгоритм МО, который решает размеры ставок, не зависит от первичной модели, что позволяет встраивать признаки,

[1] В разделе справочных материалов приводится ряд статей, объясняющих, как эти вероятности получаются. Обычно в эти вероятности встраивается информация о качестве подгонки либо достоверности в предсказании. См. публикацию Wu и соавт. [2004] и посетите http://scikit-learn.org/stable/modules/svm.html#scores-andprobabilities.

предсказывающие ложные утверждения (см. главу 3). Во-вторых, предсказанная вероятность может быть напрямую транслирована в размер ставки. Давайте посмотрим, как.

10.3. Выставление размера из предсказанных вероятностей

Обозначим через $p[x]$ вероятность появления метки x. Для двух возможных исходов, $x \in \{-1, 1\}$, мы хотели бы проверить нулевую гипотезу $H_0 : p[x=1] = \frac{1}{2}$. Мы вычисляем проверочный статистический показатель[1]

$$z = \frac{p[x=1] - \frac{1}{2}}{\sqrt{p[x=1](1-p[x=1])}} = \frac{2p[x=1]-1}{2\sqrt{p[x=1](1-p[x=1])}} \sim Z,$$

причем $z \in (-\infty, +\infty)$, где Z — это стандартное нормальное распределение. Мы выводим размер ставки как $m = 2Z[z] - 1$, где $m \in [-1, 1]$ и $Z[.]$ — это кумулятивная функция распределения (CDF) Z.

В случае более чем двух возможных исходов мы используем метод «один против остальных». Пусть $X = \{-1, ..., 0, ..., 1\}$ равно различным меткам, связанным с размерами ставок, и $x \in X$ — пресказанная метка. Другими словами, метка идентифицируется связанным с ней размером ставки. Для каждой метки $i = 1, ..., \|X\|$ мы оцениваем вероятность p_i, причем $\sum_{i=1}^{\|X\|} p_i = 1$. Мы определяем $\tilde{p} = \max_i\{p_i\}$ как вероятность x, и мы хотели бы проверить нулевую гипотезу H_0 : $\tilde{p} = \frac{1}{\|X\|}$[2]. Мы вычисляем проверочный статистический показатель $z = \frac{\tilde{p} - \frac{1}{\|X\|}}{\sqrt{\tilde{p}(1-\tilde{p})}} \sim Z$, с $z \in [0, +\infty)$. Мы выводим размер ставки как $m = \underbrace{x}_{\in[0,1]}(2Z[z]-1)$, где $m \in [-1, 1]$, а $Z[z]$ регулирует размер для предсказания x (где сторона вытекает из x).

На рис. 10.1 построен график размера ставки как функции проверочного статистического показателя. Листинг 10.1 реализует трансляцию из вероятностей в размер

[1] Проверочный статистический показатель (test statistic) — это метрический показатель целевой разницы или эффекта, используемый в качестве критерия при проверке статистической гипотезы. В данном случае используется z-оценка, получаемая после стандартизации. — *Примеч. науч. ред.*

[2] Неопределенность абсолютна, когда все исходы равновероятны.

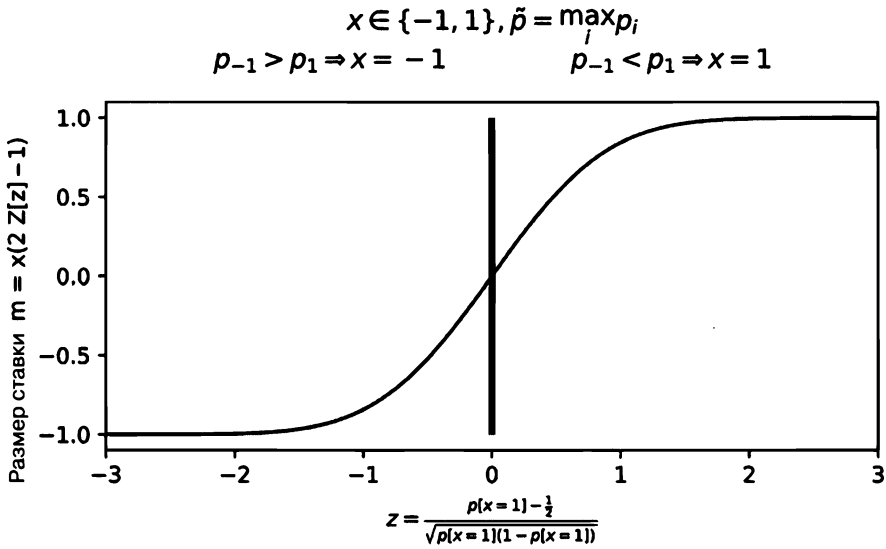


Рис. 10.1. Размер ставки из предсказанных вероятностей

ставки. Он обрабатывает возможность того, что предсказание исходит от метамаркировочного оценщика, а также от стандартного маркировочного оценщика. На шаге № 2 он также усредняет активные ставки и дискретизирует конечное значение, которое мы объясним в последующих разделах.

Листинг 10.1. Трансляция из вероятностей в размер ставки

```
def getSignal(events,stepSize,prob,pred,numClasses,numThreads,**kargs):
    # получить сигналы от предсказаний
    if prob.shape[0]==0: return pd.Series()
    #1) сгенерировать предсказания из многоклассовой классификации
    # (один против остальных, OvR)
    signal0=(prob-1./numClasses)/(prob*(1.-prob))**.5 # t-значение OvR
    signal0=pred*(2*norm.cdf(signal0)-1) # сигнал=сторона * размер
    if 'side' in events: signal0*=events.loc[signal0.index,'side'] # метамар-
                                                                   # кировка
    #2) вычислить средний сигнал среди одновременно открытых
    df0=signal0.to_frame('signal').join(events[['t1']],how='left')
    df0=avgActiveSignals(df0,numThreads)
    signal1=discreteSignal(signal0=df0,stepSize=stepSize)
    return signal1
```

10.4. Усреднение активных ставок

Каждая ставка связана с периодом владения, охватывающим период с момента ее возникновения до момента первого касания барьера, `t1` (см. главу 3). Один

из возможных подходов заключается в переопределении старой ставки по мере поступления новой ставки; однако это может привести к чрезмерному обороту. Более разумным подходом является усреднение всех размеров по всем ставкам, все еще активным в данный момент времени. Листинг 10.2 иллюстрирует одну из возможных реализаций этой идеи.

Листинг 10.2. Ставки усредняются, пока они активны

```
def avgActiveSignals(signals,numThreads):
    # вычислить средний сигнал среди активных
    # 1) временные точки, где сигналы изменяются (сигнал начинается либо
    # заканчивается)
    tPnts=set(signals['t1'].dropna().values)
    tPnts=tPnts.union(signals.index.values)
    tPnts=list(tPnts);tPnts.sort()
    out=mpPandasObj(mpAvgActiveSignals,('molecule',tPnts),numThreads,
                                        signals=signals)
    return out
#-----------------------------------
def mpAvgActiveSignals(signals,molecule):
    '''
    Во время loc средний сигнал среди все еще активных.
    Сигнал активен, если:
    а) выпущен перед временем или во время loc и
    б) время loc до времени окончания сигнала либо
       время окончания пока неизвестно (NaT).
    '''
    out=pd.Series()
    for loc in molecule:
        df0=(signals.index.values<=loc)&((loc<signals['t1'])|pd.
isnull(signals['t1']))
        act=signals[df0].index
        if len(act)>0: out[loc]=signals.loc[act,'signal'].mean()
        else: out[loc]=0 # ни один сигнал в настоящее время не активен
    return out
```

10.5. Дискретизация размера

Усреднение сокращает часть избыточного оборота, но все же вполне вероятно, что вместе с каждым предсказанием будут запускаться малые сделки. Поскольку этот джиттер вызывает ненужную избыточную торговлю, я предлагаю вам дискретизировать размер ставки как $m^{*} = \text{round}\left[\frac{m}{d}\right]d$, где $d \in (0, 1]$ определяет степень дискретизации. На рис. 10.2 показана дискретизация размера ставки. Листинг 10.3 реализует эту идею.

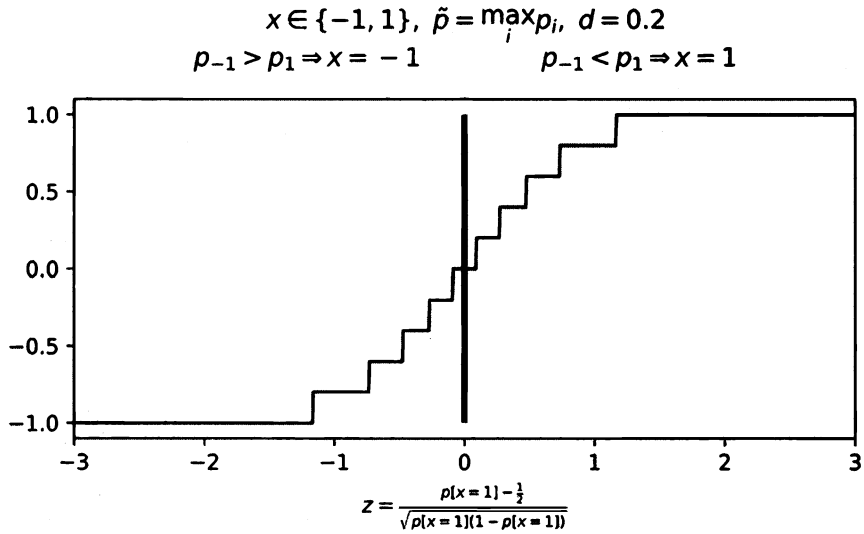


Рис. 10.2. Дискретизация размера ставки, $d = 0.2$

Листинг 10.3. Дискретизация размера ставки с целью предотвращения избыточной торговли

```
def discreteSignal(signal0,stepSize):
    # дискретизировать сигнал
    signal1=(signal0/stepSize).round()*stepSize # дискретизировать
    signal1[signal1>1]=1 # верхний предел
    signal1[signal1<-1]=-1 # нижний предел
    return signal1
```

10.6. Динамические размеры ставок и лимитные цены

Вспомните тройной барьерный метод маркировки, представленный в главе 3. Бар i формируется в момент времени $t_{i,0}$, в этот момент мы прогнозируем первый барьер, который будет затронут. Из этого предсказания вытекает прогнозируемая цена $E_{t_{j,0}}[p_{t_{i,1}}]$, совместимая с настройками барьеров. За период, прошедший до появления исхода, $t \in [t_{i,0}, t_{i,1}]$, цена p_t колеблется и могут быть сформированы дополнительные прогнозы, $E_{t_{j,0}}[p_{t_{i,1}}]$, где $j \in [i+1, I]$, а $t_{j,0} \leq t_{i,1}$. В разделах 10.4 и 10.5 мы обсудили методы усреднения активных ставок и дискретизации размера ставки по мере формирования новых прогнозов. В этом разделе мы представим подход к корректировке размеров ставок по мере колебания рыночной цены p_t и прогнозной цены f_i, и мы выведем лимитную цену ордера.

Пусть q_t равно текущей позиции, Q равно максимальному абсолютному размеру позиции и $\hat{q}_{i,t}$ равно размеру целевой позиции, связанной с прогнозом f_i, так, что

$$\hat{q}_{i,t} = \text{int}[m[\omega, f_i - p_t]Q];$$

$$m[\omega, x] = \frac{x}{\sqrt{\omega + x^2}},$$

где $m[\omega, x]$ — это размер ставки, $x = f_i - p_t$ — дивергенция между текущей рыночной ценой и прогнозом, ω — коэффициент, который регулирует ширину сигмоидальной функции, и int $[x]$ — целочисленное значение x. Отметим, что для вещественно-ценовой дивергенции x, $-1 < m[\omega, x] < 1$, целочисленное значение $\hat{q}_{i,t}$ ограничено $-Q < \hat{q}_{i,t} < Q$.

Размер целевой позиции $\hat{q}_{i,t}$ можно динамически корректировать по мере изменения p_t. В частности, по мере того как $p_t \to f_i$, мы получаем $\hat{q}_{i,t} \to 0$, потому что алгоритм хочет реализовать выигрыши. Из этого вытекает безубыточная лимитная цена p для ордерного размера $\hat{q}_{i,t} - q_t$, чтобы избежать реализации убытков. В частности

$$\overline{p} = \frac{1}{\left|\hat{q}_{i,t} - q_t\right|} \sum_{j=|q_t|+\text{sgn}[\hat{q}_{i,t}-q_t]}^{|\hat{q}_{i,t}|} L\left[f_i, \omega, \frac{j}{Q}\right],$$

где $L[f_i, \omega, m]$ — это обратная функция от $m[\omega, f_i - p_t]$ с учетом p_t.

$$L[f_i, \omega, m] = f_i - m\sqrt{\frac{\omega}{1-m^2}}.$$

Нам не нужно беспокоиться о случае $m^2 = 1$, потому что $|\hat{q}_{i,t}| < 1$. Поскольку эта функция монотонна, алгоритм не может реализовать убытки в виде $p_t \to f_i$.

Теперь давайте откалибруем ω. При заданной определяемой пользователем пары (x, m^*) такой, что $x = f_i - p_t$ и $m^* = m[\omega, x]$, обратная функция от $m[\omega, x]$ по ω равна

$$\omega = x^2(m^{*-2} - 1).$$

Листинг 10.4 реализует алгоритм, вычисляющий динамический размер позиции и лимитные цены как функции от p_t и f_t. Во-первых, мы калибруем сигмоидальную функцию, чтобы она возвращала размер ставки $m^* = .95$ для ценовой дивергенции $x = 10$. Во-вторых, мы вычисляем целевую позицию $\hat{q}_{i,t}$ для максимальной позиции $Q = 100$, $f_i = 115$ и $p_t = 100$. Если вы попробуете $f_i = 110$, то получите $\hat{q}_{i,t} = 95$, в соответствии с калибровкой ω. В-третьих, лимитная цена для данного ордерного раз-

мера $\hat{q}_{i,t} - q_t = 97$ составляет $p_t < 112.3657 < f_i$, которая находится между текущей ценой и прогнозной ценой.

Листинг 10.4. Динамический размер позиции и лимитные цены

```
def betSize(w,x):
    return x*(w+x**2)**-.5
#------------------------------------
def getTPos(w,f,mP,maxPos):
    return int(betSize(w,f-mP)*maxPos)
#------------------------------------
def invPrice(f,w,m):
    return f-m*(w/(1-m**2))**.5
#------------------------------------
def limitPrice(tPos,pos,f,w,maxPos):
    sgn=(1 if tPos>=pos else -1)
    lP=0
    for j in xrange(abs(pos+sgn),abs(tPos+1)):
        lP+=invPrice(f,w,j/float(maxPos))
    lP/=tPos-pos
    return lP
#------------------------------------
def getW(x,m):
    # 0<alpha<1
    return x**2*(m**-2-1)
#------------------------------------
def main():
    pos,maxPos,mP,f,wParams=0,100,100,115,{'divergence':10,'m':.95}
    w=getW(wParams['divergence'],wParams['m']) # калибровать w
    tPos=getTPos(w,f,mP,maxPos) # получить tPos
    lP=limitPrice(tPos,pos,f,w,maxPos) # лимитная цена для ордера
    return
#------------------------------------
if __name__=='__main__':main()
```

В качестве альтернативы сигмоидальной функции мы могли бы использовать степенную функцию $\tilde{m}[\omega, x] = \text{sgn}[x]|x|^{\omega}$, где $\omega \geq 0, x \in [-1, 1]$, что приводит к $\tilde{m}[\omega, x] \in [-1, 1]$. Эта альтернатива обеспечивает преимущества в том, что:

- $\tilde{m}[\omega,-1] = -1, \tilde{m}[\omega,1] = 1$.
- Кривизной можно манипулировать напрямую через ω.
- Для $\omega > 1$ функция переходит от вогнутой к выпуклой, а не наоборот, следовательно, функция почти плоская вокруг точки перегиба.

Мы оставляем вывод уравнений для степенной функции в качестве упражнения. На рис. 10.3 показаны размеры ставок (ось y) как функция от ценовой дивергенции $f - p_t$ (ось x) как для сигмоидальной, так и для степенной функции.

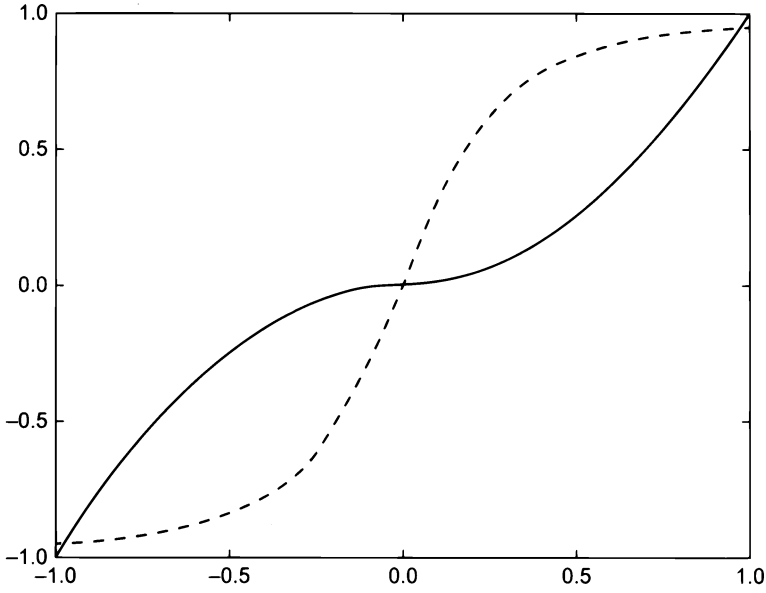


Рис. 10.3. $f[x] = \text{sgn}[x]\,|x|^2$ (от вогнутой до выпуклой) и $f[x] = x(.1 + x^2)^{.5}$ (от выпуклой до вогнутой)

Упражнения

10.1. Используя формулировку в разделе 10.3, постройте график размера ставки (m) как функции от максимальной предсказанной вероятности ($\tilde{p}$) при $\|X\| = 2, 3, ..., 10$.

10.2. Возьмите 10 000 случайных чисел из равномерного распределения с границами $U[.5, 1.]$.

(а) Вычислите размеры ставок m для $\|X\| = 2$.

(б) Назначьте 10 000 последовательных календарных дней размерам ставок.

(в) Возьмите 10 000 случайных чисел из равномерного распределения с границами $U[1, 25]$.

(г) Сформируйте ряд библиотеки pandas, проиндексированный по датам, из 10.2.б и со значениями, равными индексу, сдвинутому вперед на число дней в 10.2.в. Получится объект `t1`, подобный тем, которые мы использовали в главе 3.

(д) Вычислите результирующие средние активные ставки на основе раздела 10.4.

10.3. Используя объект `t1` из упражнения 10.2.г,

(а) Определите максимальное число одновременных длинных ставок $\bar{c}_l$.

(б) Определите максимальное число одновременных коротких ставок $\bar{c}_s$.

(в) Выведите размер ставки как $m_t = c_{t,l}\frac{1}{\bar{c}_l} - c_{t,s}\frac{1}{\bar{c}_s}$, где $c_{t,l}$ — это число одновременных длинных ставок в момент времени t, а $c_{t,s}$ — число одновременных коротких ставок в момент времени t.

10.4. Используя объект `t1` из упражнения 10.2.г,

(а) Вычислите ряд $c_t = c_{t,l} - c_{t,s}$, где $c_{t,l}$ — это число одновременных длинных ставок в момент времени t, а $c_{t,s}$ — число одновременных коротких ставок в момент времени t.

(б) Произведите подгонку комбинации двух гауссиан к $\{c_t\}$. Возможно, вы захотите использовать метод, описанный в публикации Lopez de Prado and Foreman [2014].

(в) Выведите размер ставки в виде $m_t = \begin{cases} \frac{F[c_t] - F[0]}{1 - F[0]} & \text{если } c_t \geq 0 \\ \frac{F[c_t] - F[0]}{F[0]} & \text{если } c_t < 0 \end{cases}$, где $F[x]$ — это кумулятивная функция распределения подогонанной комбинации двух гауссиан к значению x.

(г) Объясните, чем этот ряд $\{m_t\}$ отличается от ряда с размерами ставок, вычисленного в упражнении 10.3.

10.5. Повторите упражнение 10.1, где вы дискретизируете m через `stepSize=.01`, `stepSize=.05`, и `stepSize=.1`.

10.6. Перепишите уравнения из раздела 10.6 так, чтобы размер ставки определялся не сигмоидальной функцией, а степенной.

10.7. Измените код из листинга 10.4 таким образом, чтобы он реализовывал уравнения, полученные в упражнении 10.6.

11 Опасности бэктестирования

11.1. Актуальность

Тестирование на исторических данных, или бэктестирование (backtesting), является одним из наиболее важных и наименее изученных методов в арсенале кванта. Распространенное заблуждение рассматривать бэктестирование как исследовательский инструмент. Исследование и бэктестирование — это как выпивка за рулем. Не следует проводить исследования под воздействием бэктеста. Большинство бэктестов, опубликованных в журналах, имеют изъяны в результате систематической ошибки отбора на многочисленных тестах (Bailey, Borwein, Lopez de Prado and Zhu [2014]; Harvey и соавт. [2016]). Можно написать целую книгу с перечислением всего многообразия ошибок, которые люди делают во время бэктестирования. Вполне возможно, что я являюсь академическим автором наибольшего числа журнальных статей о бэктестировании[1] и метрических показателях инвестиционной результативности, и все же я не чувствую, что у меня хватит выносливости, чтобы собрать все многообразие ошибок, которые я встречал за последние 20 лет. Эта глава — не интенсивный курс по бэктестированию, а всего лишь краткий список из нескольких распространенных ошибок, которые допускают даже опытные специалисты.

11.2. Миссия невыполнима: безупречный бэктест

В самом узком определении бэктест — это историческое симулирование того, какую результативность стратегия показала бы, в случае если бы она выполнялась в течение прошлого периода времени. Как таковой он — гипотетический и отнюдь не является экспериментом. В физической лаборатории, в такой как Национальная лаборатория им. Лоуренса в Беркли, мы можем повторять эксперимент, учитывая средовые переменные, для того чтобы вывести точную причинно-следственную

[1] См. http://papers.ssrn.com/sol3/cf_dev/AbsByAuth.cfm?per_id=434076; http://www.QuantResearch.org/.

связь. В отличие от этого, бэктестирование — это не эксперимент, и оно ничего не доказывает. Бэктестирование ничего не гарантирует, даже не достигая пресловутого коэффициента Шарпа, если бы только мы могли пропутешествовать назад во времени на нашем модернизированном спортивном автомобиле DeLorean DMC-12 (Bailey and Lopez de Prado [2012]). Случайные выемки образцов будут другими. Прошлое не повторится.

Тогда в чем смысл бэктеста? Бэктест представляет собой проверку исправности на нескольких переменных, включая выставление размера сделок, оборот, устойчивость к издержкам и поведение в рамках заданного сценария. Хороший бэктест может быть очень полезен, но бэктестирование является крайне сложной процедурой. В 2014 году команда квантов Deutsche Bank во главе с Инь Ло опубликовала исследование под названием «Семь грехов квантитативного инвестирования» (Seven Sins of Quantitative Investing, Luo и соавт. [2014]). Это очень графический и доступный материал, который я бы посоветовал всем работающим в этом бизнесе внимательно прочитать. В нем данная команда упоминает обычных подозреваемых:

1. **Систематическое смещение по выживаемости** (survivorship bias): использование в качестве инвестиционного универсума текущего универсума и, следовательно, игнорирование того, что некоторые компании обанкротились, а ценные бумаги были исключены из списка.

2. **Систематическое смещение из-за забегания вперед** (look-ahead bias): использование информации, которая не была публичной в момент принятия симулируемого решения. Полная уверенность во временном штампе для каждой точки данных. Принятие в расчет дат публикации данных, задержек в их распространении и исправлений задним числом.

3. **Сторителлинг**: выдумывание истории *постфактум*, для того чтобы оправдать случайную закономерность.

4. **Извлечение данных и прочесывание данных**: тренировка модели на тестовом подмножестве.

5. **Транзакционные издержки**: просимулировать транзакционные издержки (стоимости) тяжело, потому что единственный способ быть уверенным в этой издержке состоял бы в том, чтобы взаимодействовать с торговой книгой (то есть выполнять фактическую торговлю).

6. **Выбросы**: базирование стратегии на нескольких экстремальных исходах, которые вполне могут никогда не повториться снова точно так же, как они наблюдались в прошлом.

7. **Шортирование**: для занятия короткой позиции на фактических (денежных) продуктах необходимо найти кредитора. Стоимость кредитования и доступная сумма, как правило, неизвестны и зависят от связей, оборотного капитала, относительного спроса и т. д.

Это лишь несколько основных ошибок, которые регулярно встречаются в большинстве статей, публикуемых в журналах. Другие распространенные ошибки включают вычислительную результативность с использованием нестандартного метода (глава 14), игнорирование скрытых рисков, ориентацию только на финансовые возвраты, игнорируя другие метрические показатели, смешивание корреляции с причинно-следственной связью, выбор нерепрезентативного временного периода, неспособность ожидать неожиданное, игнорирование лимита уровня принудительной остановки (стоп-аута) или маржин-коллов, игнорирование издержек фондирования и упущение практических аспектов (Sarfati [2015]). Существует еще ряд других, но в действительности нет смысла их перечислять из-за названия следующего раздела.

11.3. Даже если ваш бэктест безупречен, он, вероятнее всего, будет ошибочен

Поздравляю! Ваш бэктест безупречен в том смысле, что каждый может воспроизвести ваши результаты, и допущения настолько консервативны, что даже босс не может возразить. Вы заплатили за каждую сделку более чем в два раза больше, чем кто-либо может запросить. Вы выполнили тест через несколько часов после того, как информация была известна половине земного шара, при смехотворно низком уровне участия. Несмотря на все эти вопиющие издержки, ваш бэктест по-прежнему зарабатывает много денег. Тем не менее этот безупречный бэктест, вероятнее всего, ошибочен. Почему? Потому что только эксперт может произвести безупречный бэктест. Стать экспертом означает, что за предыдущие годы вы провели десятки тысяч бэктестов. Наконец, это не первый бэктест, который вы производите, поэтому мы должны учитывать возможность того, что он представляет собой ложное открытие, статистическую случайность, которая неизбежно возникает после выполнения нескольких тестов на одной и той же совокупности данных.

Безумная вещь касательно бэктестирования заключается в том, что чем лучше вы в нем разбираетесь, тем больше вероятность того, что всплывут ложные открытия. Новички падки на семь грехов, перечисленных в публикации Luo и соавт. [2014] (есть и другие, но кто их считает?). Профессионалы вполне могут произвести безупречные бэктесты и все равно попадутся на многократном тестировании, систематическом смещении при отборе либо переподгонке бэктеста (Bailey and Lopez de Prado [2014b]).

11.4. Бэктестирование — это не исследовательский инструмент

В главе 8 обсуждаются эффекты замещения, совместные эффекты, маскирование, методы анализа MDI, MDA, SFI, параллелизованные признаки, стековые при-

знаки и т. д. Даже если некоторые признаки очень важны, это не означает, что их можно монетизировать с помощью инвестиционной стратегии. И наоборот, есть много стратегий, которые будут казаться прибыльными, даже если они основаны на нерелевантных признаках. Важность признаков — это по-настоящему исследовательский инструмент, поскольку он помогает нам понять природу закономерностей, обнаруженных алгоритмом МО, независимо от их монетизации. Весьма существенно, что важность признаков выводится априорно, до симулирования исторической результативности.

Бэктестирование не является исследовательским инструментом. Оно дает нам очень мало понимания причины, почему конкретная стратегия могла бы заработать деньги. Так же как победитель лотереи, который может чувствовать, что он сделал нечто такое, чтобы заслужить свою удачу, всегда есть какая-то история постфактум (грех номер три у Луо). Авторы заявляют, что нашли сотни «альф» и «факторов», и для них всегда есть какое-то запутанное объяснение. Вместо этого то, что они нашли, — это лотерейные билеты, которые принесли выигрыш в последнем розыгрыше. Победитель получил по чеку, и эти цифры бесполезны для следующего раунда. Если ты не будешь доплачивать за лотерейные билеты, почему тебя так волнуют эти сотни альф? Эти авторы никогда не рассказывают нам обо всех проданных билетах, то есть о миллионах симуляций, которые потребовались для того, чтобы найти эти «счастливые» альфы.

Предназначение бэктестирования — отбрасывать плохие модели, а не улучшить их. Корректировка вашей модели на основе результатов бэктестирования — это пустая трата времени... и она опасна. Инвестируйте свое время и усилия в то, чтобы все компоненты были правильными, как мы обсуждали в другом месте книги: структурированные данные, маркировка, взвешивание, ансамбли, перекрестная проверка, важность признаков, размер ставок и т. д. Ко времени бэктестирования уже будет поздно. Никогда не бэктестируйте, пока модель не будет полностью определена. Если бэктестирование окажется безуспешным, начните все сначала. Если вы поступите именно так, то шансы найти ложное открытие существенно снизятся, но все равно они не будут равны нулю.

11.5. Несколько общих рекомендаций

Переподгонку бэктеста можно определить как систематическое смещение при отборе на многократных бэктестах. Переподгонка бэктеста имеет место, когда стратегия разработана так, чтобы показывать хорошую результативность на бэктесте, монетизируя случайные исторические закономерности. Поскольку эти случайные закономерности вряд ли повторятся в будущем, разработанная таким образом стратегия окажется безуспешной. Каждая бэктестированная стратегия в какой-то степени является результатом «систематического смещения при отборе»: единственные бэктесты, которыми делятся большинство людей, — это те, которые изображают якобы выигрышные инвестиционные стратегии.

Вопрос, как решить проблему бэктестовой переподгонки, возможно, является самым фундаментальным вопросом в квантитативном финансировании. Почему? Потому что, если бы был простой ответ на этот вопрос, то инвестиционные фирмы с уверенностью достигали бы высоких результатов, поскольку они инвестировали бы только в выигрышные бэктесты. Журналы тогда будут с достоверностью выявлять, может ли стратегия быть ложным утверждением. Финансы могут стать истинной наукой в попперовском и лакатозском смысле (Lopez de Prado [2017]). Что так затрудняет выявление переподгонки бэктеста, так это то, что вероятность ложных утверждений изменяется с каждым новым тестом, проведенным на одной и той же совокупности данных, и эта информация либо исследователю неизвестна, либо ею не делятся с инвесторами или рефери. Хотя нет простого способа предотвратить переподгонку, ряд шагов способны помочь уменьшить ее присутствие.

1. Разрабатывать модели для целых классов активов или инвестиционных универсумов, а не для конкретных ценных бумаг (глава 8). Инвесторы диверсифицируют, следовательно, они не делают ошибки X только на ценной бумаге Y. Если вы найдете ошибку X только на ценной бумаге Y, независимо от того, насколько на первый взгляд она прибыльна, скорее всего это ложное открытие.

2. Применять бэггирование (глава 6) как средство предотвращения переподгонки и уменьшения дисперсии предсказательной ошибки. Если бэггирование ухудшает результативность стратегии, то вполне вероятно, оно переподогнано к небольшому числу наблюдений или выбросам.

3. Не проводить бэктестирование до тех пор, пока все исследования не будут завершены (главы 1–10).

4. Записывать каждый бэктест, проводимый на совокупности данных, так чтобы вероятность бэктестовой переподгонки могла быть оценена на окончательном отобранном результате (см. Bailey, Borwein, Lopez de Prado and Zhu [2017a] и главу 14) и коэффициент Шарпа мог быть правильно дефлирован числом проведенных испытаний (Bailey and Lopez de Prado [2014b]).

5. Симулировать сценарии, а не историю (глава 12). Стандартный бэктест — это историческое симулирование, которое можно легко перенастроить. История — это просто одна из случайных траекторий, которая была реализована, и она могла быть совершенно другой. Ваша стратегия должна быть прибыльной при широком спектре сценариев, а не только на эпизодической исторической траектории. Труднее достичь переподгонки исхода тысячи сценариев, отвечающих на вопрос «а что если?».

6. Если бэктест оказался безуспешным в выявлении прибыльной стратегии, то начать с нуля. Не поддавайтесь искушению многократно использовать эти результаты. Следуйте второму закону тестирования на исторических данных.

ЛИСТИНГ 11.1. ВТОРОЙ ЗАКОН БЭКТЕСТИРОВАНИЯ МАРКОСА

«Проведение бэктестирования во время исследования равносильно выпивке за рулем. Не следует проводить исследования под воздействием бэктеста».

— Маркос Лопез де Прадо
Машинное обучение: алгоритмы для бизнеса (2018)

11.6. Выбор стратегии

В главе 7 мы обсудили, как присутствие внутрирядовой обусловленности в метках срывает стандартную k-блочную перекрестную проверку, потому что случайный отбор будет разбрызгивать избыточные наблюдения как в тренировочные, так и в тестовые подмножества. Мы должны найти другую (истинную вневыборочную) проверочную процедуру: процедуру, которая оценивает нашу модель на наблюдениях, которые с наименьшей вероятностью будут коррелированы/избыточны для тех, которые используются для тренировки модели. См. публикацию Arlot and Celisse [2010] с исследованием данного вопроса.

В библиотеке scikit-learn реализован прямой метод временных сгибов (walk-forward timefolds method). В рамках этого подхода тестирование продвигается вперед (в направлении времени) с целью предотвращения утечки. Это согласуется с тем, как проводятся исторические бэктесты (и торговля). Однако в присутствии долгосрочной внутрирядовой зависимости тестирование одного наблюдения вдали от конца тренировочного подмножества может оказаться недостаточным для предотвращения утечки информации. Мы вновь затронем этот вопрос в разделе 12.2 главы 12.

Одним из недостатков прямого метода является то, что он может быть легко переподогнан. Причина заключается в том, что без случайного отбора существует единственная траектория тестирования, которая может повторяться снова и снова до тех пор, пока не появится ложное утверждение. Как и в стандартной перекрестной проверке, для того чтобы избежать такого рода таргетирования результативности или бэктестовой оптимизации, избегая при этом утечки примеров, коррелированных с тренировочным подмножеством, в тестовое подмножество, требуется некоторая рандомизация. Далее мы представим перекрестно-проверочный метод отбора стратегии, основанный на оценивании вероятности бэктестовой переподгонки (probability of backtest overfitting, PBO). Объяснение перекрестно-проверочных методов бэктестирования мы оставляем для главы 12.

В публикации Bailey и соавт. [2017a] вероятность PBO оценивается посредством комбинаторно-симметричного перекрестно-проверочного (combinatorially symmetric cross-validation, CSCV) метода. Схематично данная процедура работает следующим образом.

Во-первых, мы формируем матрицу M, собирая ряд, состоящий из результативностей в результате N испытаний. В частности, каждый столбец $n = 1, ..., N$ представляет вектор PnL (пересчитанных по текущим рыночным ценами прибыли и убытков) на $t = 1, ..., T$ наблюдениях, связанных с конкретной модельной конфигурацией, опробованной исследователем. Следовательно, M является вещественной матрицей порядка $(T \times N)$. Единственными навязываемыми нами условиями является то, что: 1) M — это истинная матрица, то есть с одинаковым числом строк для каждого столбца, где наблюдения синхронны для каждой строки в N испытаниях, и 2) метрический показатель оценивания результативности, используемый для выбора «оптимальной» стратегии, может быть оценен на подвыборках каждого столбца. Например, если этот метрический показатель является коэффициентом Шарпа, то мы исходим из того, что допущение о нормальном распределения одинаково распределенных взаимно независимых случайных величин может храниться в разных срезах сообщаемой результативности. Если разные модельные конфигурации торгуют с разной частотой, то наблюдения агрегируются (отбираются с пониженной частотой), для того чтобы соответствовать общему индексу $t = 1, ..., T$.

Во-вторых, мы разделяем M по строкам на четное число S непересекающихся подматриц равных размерностей. Каждая из этих подматриц M_s, где $s = 1, ..., S$, имеет порядок $\frac{T}{S} \times N$.

В-третьих, мы формируем все сочетания C_S из M_s, берущиеся в группах размера $S/2$. В результате получаем общее число сочетаний

$$\binom{S}{S/2} = \binom{S-1}{S/2-1}\frac{S}{S/2} = ... = \prod_{i=0}^{S/2-1}\frac{S-i}{S/2-i}.$$

Например, если $S = 16$, то мы сформируем 12 780 сочетаний. Каждое сочетание $c \in C_S$ состоит из $S/2$ подматриц M_s.

Затем для каждого сочетания $c \in C_S$ мы:

1. Формируем *обучающее подмножество* J путем объединения $S/2$ подматриц M_s, которые составляют c. J — это матрица порядка $\left(\frac{T}{S}\frac{S}{2} \times N\right) = \left(\frac{T}{2} \times N\right)$.

2. Формируем *тестовое подмножество* $\bar{J}$ как дополнение J в M. Другими словами, $\bar{J}$ — это матрица $(\frac{T}{2} \times N)$, сформированная всеми строками M, которые не являются частью J.

3. Формируем вектор R статистических показателей результативности порядка N, где n-й элемент R сообщает результативность, связанную с n-м столбцом J (обучающее подмножество).

4. Определяем элемент n^* такой, что $R_n \leq R_{n^*}$, $\forall n = 1, ..., N$. Другими словами, $n^* = \arg\max_n\{R_n\}$.

5. Формируем вектор $\bar{R}$ статистических показателей результативности порядка N, где n-й элемент $\bar{R}$ сообщает результативность, связанную с n-м столбцом $\bar{J}$ (тестовое подмножество).

6. Определяем относительный ранг $\bar{R}_{n^*}$ внутри $\bar{R}$. Мы обозначаем данный относительный ранг как $\bar{\omega}_c$, где $\bar{\omega}_c \in (0, 1)$. Это относительный ранг вневыборочной результативности, связанной с испытанием, выбранным внутривыборочно. Если процедура оптимизации стратегии не переподогнана, то мы должны заметить, что $\bar{R}_{n^*}$ систематически превосходит по результативности $\bar{R}$ (вневыборочно), так же как $\bar{R}_{n^*}$ превосходил R (внутривыборочно).

7. Определяем логит[1] $\lambda_c = \log\left[\frac{\bar{\omega}_c}{1-\bar{\omega}_c}\right]$. Он представляет свойство, что $\lambda_c = 0$, когда $\bar{R}_{n^*}$ совпадает с медианой $\bar{R}$. Из высоких значений логита вытекает согласованность между внутри- и вневыборочной результативностью, что свидетельствует о низком уровне бэктестовой переподгонки.

В-пятых, вычисляем распределение вневыборочных рангов, собрав все λ_c для $c \in C_S$. Затем оценивается функция распределения вероятностей $f(\lambda)$ как относительная частота, с которой λ встречалась во всех C_S, с $\int_{-\infty}^{\infty} f(\lambda)d\lambda = 1$. Наконец, вероятность PBO оценивается как $PBO = \int_{-\infty}^{0} f(\lambda)d\lambda$, так как это вероятность, связанная с внутривыборочными оптимальными стратегиями, которые отстают по производительности от вневыборочных.

Ось x рис. 11.1 показывает коэффициент Шарпа внутривыборочно из лучшей отобранной стратегии. Ось y показывает коэффициент Шарпа вневыборочно для той же самой лучшей отобранной стратегии. Как можно отметить, имеется сильное и устойчивое затухание результативности, вызванное переподгонкой бэктеста. Как показано на рис. 11.2, применив приведенный выше алгоритм, мы можем получить вероятность PBO, связанную с этим процессом отбора стратегии.

Наблюдения в каждом подмножестве сохраняют исходную временную последовательность. Случайный отбор производится не на наблюдениях, а на относительно некоррелированных подмножествах. См. публикацию Bailey и соавт. [2017a] для получения экспериментального анализа точности данной методологии.

[1] Логит (logit), или логит-преобразование, — функция, которая увязывает вероятность принадлежности классу с диапазоном $\pm\infty$ (вместо диапазона 0...1). — *Примеч. науч. ред.*

Рис. 11.1. Лучший коэффициент Шарпа внутривыборочно (SR IS) против коэффициента Шарпа вневыборочно (SR OOS)

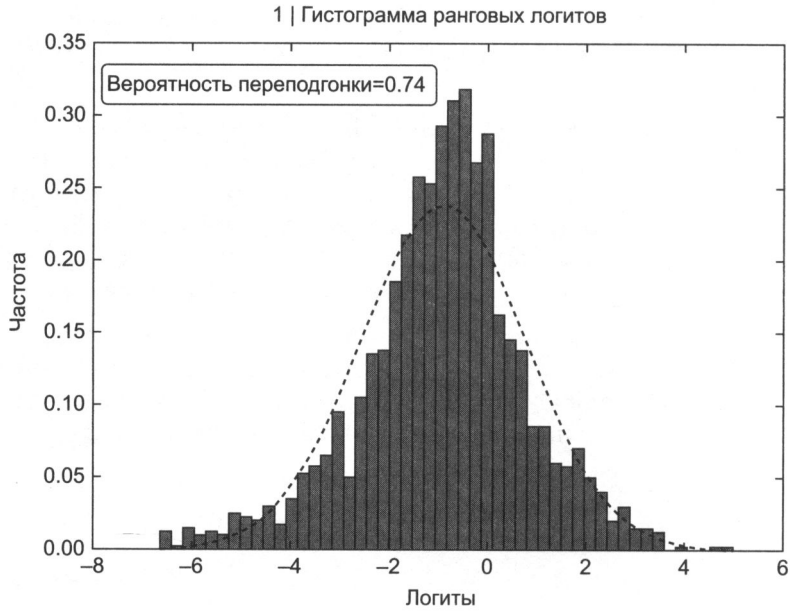


Рис. 11.2. Вероятность бэктестовой переподгонки, выведенная из распределения логитов

Упражнения

11.1. Аналитик выполняет подгонку классификатора на основе случайного леса (RF), где некоторые признаки включают сезонно скорректированные данные о занятости. Он выравнивает с январскими данными сезонно скорректированное значение января и т. д. Какой «грех» он допустил?

11.2. Аналитик разрабатывает алгоритм МО, где он генерирует сигнал, используя цены закрытия и исполняемый на закрытии. В чем заключается его «грех»?

11.3. Между суммарным доходом, полученным от аркад, и докторскими степенями в области компьютерных наук, присужденными в Соединенных Штатах, существует 98,51 %-ная корреляция. Поскольку число докторских степеней должно вырасти, следует ли нам инвестировать в аркадные компании? Если нет, то в чем заключается «грех»?

11.4. *The Wall Street Journal* сообщил, что сентябрь, если оглянуться на 20, 50 и 100 лет назад, является единственным месяцем года, который имеет отрицательную среднюю возвратность инвестиций в акции. Должны ли мы продавать акции в конце августа? Если нет, то в чем заключается «грех»?

11.5. Мы скачиваем коэффициенты цена/прибыль (P/E) из Bloomberg, ранжируем акции каждый месяц, продаем верхний квартиль и покупаем длинный квартиль. Результативность удивительная. В чем заключается «грех»?

12 Бэктестирование через кросс-валидацию

12.1. Актуальность

Бэктест оценивает вне выборки результативность инвестиционной стратегии, используя прошлые наблюдения. Эти прошлые наблюдения могут использоваться двумя способами: 1) в узком смысле для симулирования исторической результативности инвестиционной стратегии, как если бы она выполнялась в прошлом; и 2) в более широком смысле для симулирования сценариев, которых не было в прошлом. Первый (узкий) подход, именуемый прямым, настолько распространен, что, по сути, термин «бэктест» стал де-факто синонимом «исторического симулирования». Второй (более широкий) подход гораздо менее известен, и в этой главе мы представим несколько новых способов его реализации. У каждого подхода есть свои плюсы и минусы и каждому следует уделить пристальное внимание.

12.2. Прямой метод

Наиболее распространенным методом бэктестирования в литературе является прямой, или прямонаправленный, подход (walk-forward, WF). Прямой подход — это историческое симулирование того, какую результативность стратегия показывала бы в прошлом. Каждое стратегическое решение основывается на наблюдениях, предшествующих этому решению. Как мы видели в главе 11, проведение безупречного прямого симулирования является сложной задачей, которая требует экстремальных знаний источников данных, рыночной микроструктуры, рискового менеджмента, стандартов измерения результативности (например, глобального

стандарта инвестиционной результативности GIPS), многочисленных методов тестирования, экспериментальной математики и т. д. К сожалению, для проведения бэктеста нет универсального рецепта. Для того чтобы быть точным и репрезентативным, каждый бэктест должен быть кастомизирован для оценивания допущений, принятых в конкретной стратегии.

Прямой подход WF обладает двумя ключевыми преимуществами: 1) он имеет четкую историческую интерпретацию. Его результативность может быть согласована с бумажной торговлей; 2) история — это фильтрация; следовательно, использование предшествующих данных гарантирует, что тестовое подмножество является вневыборочным (нет никакой утечки), если только прочистка была реализована как положено (см. главу 7, раздел 7.4.1). Распространенной ошибкой является отыскивание утечки в прямых бэктестах, где индекс `t1.index` попадает внутрь тренировочного подмножества, а значения `t1.values` попадают в тестовое подмножество (см. главу 3). В прямых бэктестах не нужно накладывать эмбарго, поскольку тренировочное подмножество всегда предшествует тестовому подмножеству.

12.2.1. Ловушки прямого метода

Прямой метод страдает от трех основных недостатков: во-первых, тестируется один сценарий (историческая траектория или путь), который может быть легко переподогнан (Bailey и соавт. [2014]). Во-вторых, прямой метод не обязательно отражает будущую результативность, поскольку результаты могут быть смещены той или иной последовательностью точек данных. Сторонники прямого метода (WF) обычно выдвигают в качестве аргумента то, что предсказание прошлого приведет к чрезмерно оптимистичным оценкам результативности. И все же, очень часто подгонка превосходящей по результативности модели на реверсированной последовательности наблюдений приводит к отстающему по результативности прямому (WF) бэктесту. Истина в том, что достичь переподгонки прямого бэктеста так же легко, как и достичь переподгонки обратного бэктеста, и тот факт, что изменение последовательности наблюдений дает противоречивые результаты, является доказательством этой переподгонки. Если сторонники прямого метода были бы правы, мы должны были бы заметить, что обратные бэктесты систематически превосходят своих коллег по результативности. Это отнюдь не так, поэтому главный аргумент в пользу прямого метода довольно слаб.

Чтобы сделать этот второй недостаток понятнее, предположим, что у нас есть стратегия инвестирования основного капитала, которая пробэктестирована с помощью обратного метода на данных S&P 500 начиная с 1 января 2007 года. До 15 марта 2009 года смесь ралли и активных распродаж будет тренировать стратегию быть рыночно нейтральной с низкой достоверностью на каждой позиции. После этого длинное ралли будет доминировать в совокупности данных, и к 1 января 2017 года прогнозы на покупку будут преобладать над прогнозами на продажу. Результативность была бы совсем другой, если бы мы отыграли информацию в обратном направлении, с 1 января 2017 года по 1 января 2007 года (длинное ралли

с последующей активной распродажей). Путем эксплуатирования определенной последовательности стратегия, отобранная прямым методом, может привести нас к фиаско.

Третий недостаток прямого метода WF заключается в том, что начальные решения принимаются на меньшей порции общей выборки. Даже если установлен период разогрева, бóльшая часть информации используется лишь небольшой порцией решений. Рассмотрим стратегию с периодом разогрева, в которой используются t_0 наблюдений из T. Эта стратегия делает половину своих решений $\left(\frac{T-t_0}{2}\right)$ на среднем числе точек данных:

$$\left(\frac{T-t_0}{2}\right)^{-1}\left(t_0+\frac{T+t_0}{2}\right)\frac{T-t_0}{4}=\frac{1}{4}T+\frac{3}{4}t_0,$$

то есть только на $\frac{3}{4}\frac{t_0}{T}+\frac{1}{4}$ доле наблюдений. Хотя эта проблема уменьшается за счет увеличения времени разогрева, поступая так, мы также приходим к сокращению продолжительности бэктеста.

12.3. Перекрестно-проверочный метод

Инвесторы часто спрашивают, какую результативность стратегия будет показывать, если ее подвергнуть стрессовому сценарию, столь же непредсказуемому, как кризис 2008 года, или пузырь доткомов, или истерика «taper tantum»[1], или китайская паника 2015–2016 годов и т. д. Один из способов ответа на этот вопрос — разделить наблюдения на два подмножества: один с периодом, который мы хотим проверить (тестовое подмножество), и один с остальными (тренировочное подмножество). Например, классификатор будет тренироваться на периоде с 1 января 2009 года по 1 января 2017 года, а затем тестироваться на периоде с 1 января 2008 по 31 декабря 2008 года. Результативность, которую мы получим за 2008 год, не является исторически точной, поскольку классификатор был натренирован на данных, которые были доступны только после 2008 года. Но историческая точность не является целью теста. Целевая задача теста состояла в том, чтобы сделать ничего не ведающую о 2008 годе стратегию предметом стрессового сценария, такого как 2008 год.

Цель бэктестирования посредством перекрестной проверки не в том, чтобы получить исторически точную результативность, а в том, чтобы статистически вывести будущую результативность из ряда вневыборочных сценариев. Для каждого пе-

[1] Паника на финансовых рынках в 2013 году, вызванная анонсированным ФРС постепенным сворачиванием программы количественного смягчения (QE). — *Примеч. науч. ред.*

риода бэктеста мы симулируем результативность классификатора, который знал все, за исключением этого периода.

Преимущества

1. Тест не является результатом отдельного (исторического) сценария. Фактически, перекрестная проверка тестирует k альтернативных сценариев, из которых только один соответствует исторической последовательности.
2. Каждое решение принимается на подмножествах одинакового размера. Это делает результаты сопоставимыми по периодам с точки зрения объема информации, используемой для принятия этих решений.
3. Каждое наблюдение является частью одного и только одного тестового подмножества. Разогревающее подмножество отсутствует, тем самым обеспечивая как можно более продолжительную вневыборочную симуляцию.

Недостатки

1. Как и прямой метод, симулируется одиночная бэктестовая траектория (хотя и не историческая). Для каждого наблюдения генерируется один и только один прогноз.
2. Перекрестная проверка не имеет четкой исторической интерпретации. Результат на выходе симулирует не то, какую результативность стратегия показывала бы в прошлом, а то, какую результативность она сможет показать в будущем в рамках разных стрессовых сценариев (полезный результат сам по себе).
3. Поскольку тренировочное подмножество не идет позади тестового подмножества, возможна утечка. Необходимо соблюдать крайнюю осторожность для того, чтобы избежать утечки тестовой информации в тренировочное подмножество. См. главу 7 относительно обсуждения того, как прочистка и наложение эмбарго способны помочь предотвратить утечку информации в контексте перекрестной проверки.

12.4. Комбинаторный прочищенный перекрестно-проверочный метод

В этом разделе я представлю новый метод, который устраняет главный недостаток прямого и перекрестно-проверочного методов, а именно что эти схемы тестируют одну траекторию. Я называю его методом «комбинаторной прочищенной перекрестной проверки» (combinatorial purged cross-validation, CPCV). При заданном числе φ бэктестовых траекторий, выставленных исследователем в качестве ориентиров, метод CPCV генерирует точное число сочетаний тренировочного/тестового подмножеств, необходимых для генерирования этих траекторий, при этом удаляя тренировочные наблюдения, которые содержат утечку информации.

12.4.1. Комбинаторное дробление на подразделы

Рассмотрим T наблюдений, подразделенных на N групп без перемешивания, где группы $n = 1, ..., N - 1$ имеют размер $[T/N]$, N-я группа имеет размер $T - [T/N] \times (N - 1)$ и $[.]$ — округление вниз до ближайшего целого, или целочисленная функция. Для тестового подмножества размера k групп число возможных тренировочных/тестовых дроблений равно

$$\binom{N}{N-k} = \frac{\prod_{i=0}^{k-1}(N-i)}{k!}.$$

Поскольку каждое сочетание включает k тестовых групп, общее число тестовых групп равно $k\binom{N}{N-k}$. И так как мы вычислили все возможные сочетания, то эти тестовые группы равномерно распределены по всем N (каждая группа принадлежит одному и тому же числу тренировочных и тестовых подмножеств). Из этого вытекает, что из k-размерных тестовых подмножеств на N группах мы можем пробэктестировать общее число траекторий $\varphi[N, k]$:

$$\varphi[N,k] = \frac{k}{N}\binom{N}{N-k} = \frac{\prod_{i=0}^{k-1}(N-i)}{(k-1)!}.$$

На рис. 12.1 показана композиция тренировочных и тестовых дроблений на подразделы для $N = 6$ и $k = 2$. Существует $\binom{6}{4} = 15$ подразделов, индексированных как $S1, ..., S15$. Для каждого подраздела рисунок помечает крестиком (x) группы, включенные в тестовое подмножество, и оставляет без пометок группы, которые формируют тренировочное подмножество. Каждая группа формирует часть $\varphi[6, 2] = 5$ тестовых подмножеств, поэтому эта схема тренировочного/тестового дробления позволяет нам вычислить пять бэктестовых траекторий.

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	Paths
G1	x	x	x	x	x											5
G2	x					x	x	x	x							5
G3		x				x				x	x	x				5
G4			x				x			x			x	x		5
G5				x				x			x		x		x	5
G6					x				x			x		x	x	5

Рис. 12.1. Траектории, сгенерированные для $\varphi[6, 2] = 5$

На рис. 12.2 показано назначение каждой тестовой группы одной бэктестовой траектории. Например, траектория 1 является результатом сочетания прогнозов из ($G1$, $S1$), ($G2$, $S1$), ($G3$, $S2$), ($G4$, $S3$), ($G5$, $S4$) и ($G6$, $S5$). Траектория 2 является результатом сочетания прогнозов из ($G1$, $S2$), ($G2$, $S6$), ($G3$, $S6$), ($G4$, $S7$), ($G5$, $S8$) и ($G6$, $S9$) и т. д.

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	Paths
G1	1	2	3	4	5											5
G2	1					2	3	4	5							5
G3		1				2				3	4	5				5
G4			1				2			3			4	5		5
G5				1				2			3		4		5	5
G6					1				2			3		4	5	5

Рис. 12.2. Назначение тестовых групп каждой из пяти траекторий

Эти траектории генерируются путем тренировки классификатора на порции $\theta = 1 - k/N$ данных для каждого сочетания. Хотя теоретически можно тренировать на порции $\theta < 1/2$, на практике будем считать, что $k \leq N/2$. Порция данных в тренировочном подмножестве θ увеличивается вместе с $N \rightarrow T$, но уменьшается вместе с $k \rightarrow N/2$. Число траекторий $\varphi[N, k]$ увеличивается вместе с $N \rightarrow T$ и вместе с $k \rightarrow N/2$. В пределе наибольшее число траекторий достигается установкой $N = T$ и $k = N/2 = T/2$, за счет тренировки классификатора только на половине данных для каждого сочетания ($\theta = 1/2$).

12.4.2. Алгоритм бэктестирования на основе комбинаторной прочищенной перекрестной проверки

В главе 7 мы ввели понятия прочистки и наложения эмбарго в контексте перекрестной проверки. Теперь мы воспользуемся этими понятиями для бэктестирования посредством перекрестной проверки. Алгоритм бэктестирования на основе комбинаторной прочищенной перекрестной проверки (CPCV) протекает следующим образом:

1. Подразделить T наблюдений на N групп без перетасовки, где группы $n = 1, ..., N - 1$ имеют размер $\lfloor T/N \rfloor$ и N-я группа имеет размер $T - \lfloor T/N \rfloor (N - 1)$.

2. Вычислить все возможные тренировочные/тестовые дробления на подразделы, где для каждого подраздела $N - k$ групп составляют тренировочное подмножество и k групп составляют тестовое подмножество.

3. Для любой пары меток (y_i, y_j), где y_i принадлежит тренировочному подмножеству и y_j принадлежит тестовому подмножеству, применить класс `PurgedKFold` для прочистки y_i, если y_i охватывает период, используемый для определения метки y_j. Этот класс будет также накладывать эмбарго, в случае если некоторые тестовые образцы предшествуют некоторым тренировочным образцам.

4. Выполнить подгонку классификаторов на $\binom{N}{N-k}$ тренировочных подмножествах и произвести прогнозы на соответствующих $\binom{N}{N-k}$ тестовых подмножествах.

5. Вычислить $\varphi[N,k]$ бэктестовых траекторий. Вы можете вычислить коэффициент Шарпа из каждой траектории, и из этого вывести для стратегии эмпирическое распределение коэффициента Шарпа (а не один коэффициент Шарпа, как в прямом методе или перекрестной проверке).

12.4.3. Примеры

Для $k = 1$ мы получим траекторию $\varphi[N, 1] = 1$, и в таком случае алгоритм CPCV сводится к перекрестной проверке. Следовательно, алгоритм CPCV можно понимать как обобщение перекрестной проверки для $k > 1$.

Для $k = 2$ мы получим траектории $\varphi[N, 2] = N - 1$. Это особо интересный случай, потому что во время тренировки классификатора на крупной порции данных, $\theta = 1 - 2/N$, мы можем генерировать почти столько же бэктестовых траекторий, сколько и групп, $N - 1$. Простое правило заключается в том, чтобы подразделять данные на $N = \varphi + 1$ групп, где φ — это число траекторий, на которое мы нацелены, а затем сформировать $\binom{N}{N-2}$ сочетаний. В пределе мы можем назначать одну группу в расчете на наблюдение, $N = T$, и генерировать $\varphi[T, 2] = T - 1$ траекторий во время тренировки классификатора на порции $\theta = 1 - 2/T$ данных в расчете на сочетание.

Если требуется еще больше траекторий, то мы можем увеличить $k \to N/2$, но, как объяснялось ранее, это будет стоить использования меньшей порции совокупности данных для тренировки. На практике для генерирования необходимых φ траекторий часто достаточно $k = 2$, если задать $N = \varphi + 1 \leq T$.

12.5. Как комбинаторная прочищенная перекрестная проверка справляется с бэктестовой переподгонкой

При заданной выборке одинаково распределенных взаимно независимых случайных величин $x_i \sim Z$, $i = 1, ..., I$, где Z — это стандартное нормальное распределение, ожидаемый максимум этой выборки можно аппроксимировать как

$$\mathrm{E}[\max\{x_i\}_{i=1,...,I}] \approx (1-\gamma)Z^{-1}\left[1-\frac{1}{I}\right]+\gamma Z^{-1}\left[1-\frac{1}{I}e^{-1}\right] \leq \sqrt{2\log[I]},$$

где $Z^{-1}[.]$ — это обратное кумулятивной функции распределения (CDF) от Z, $\gamma \approx 0.5772156649 \cdots$ — постоянная Эйлера—Маскерони, и $I \gg 1$ (см. Bailey и соавт. [2014], где демонстрируется доказательство). Теперь предположим, что исследователь бэктестирует I стратегий на финансовом инструменте, который ведет себя как мартингейл[1] с коэффициентами Шарпа $\{y_i\}_{i=1,\ldots,I}$, $E[y_i] = 0$, $\sigma^2[y_i] > 0$ и $\frac{y_i}{\sigma[y_i]} \sim Z$.

Несмотря на то что истинный коэффициент Шарпа равен нулю, мы ожидаем найти одну стратегию с коэффициентом Шарпа, равным:

$$E[\max\{y_i\}_{i=1,\ldots,I}] = E[\max]\{x_i\}_{i=1,\ldots,I}]\sigma[y_i].$$

Прямые (WF) бэктесты, то есть по прямому методу, демонстрируют высокую дисперсию, $\sigma[y_i] \gg 0$, как минимум по одной причине: крупная порция решений базируется на малой порции совокупности данных. Несколько наблюдений будут иметь большой вес на коэффициент Шарпа. Использование периода разогрева сократит длину бэктеста, что может внести свой вклад в увеличение дисперсии. Высокая дисперсия прямого метода приводит к ложным открытиям, потому что исследователи будут отбирать бэктест с максимальным *оценочным* коэффициентом Шарпа, даже если *истинный* коэффициент Шарпа равен нулю. Вот почему в контексте прямого бэктестирования непременно нужно контролировать число испытаний (I). Без этой информации невозможно определить групповую частоту ошибок с поправкой на эффект множественного тестирования (family-wise error rate, FWER), частоту ложных обнаружений (false discovery rate, FDR), вероятность бэктестовой переподгонки (PBO, см. главу 11) или аналогичный статистический показатель оценивания качества модели.

Перекрестно-проверочные бэктесты (раздел 12.3) устраняют этот источник дисперсии путем тренировки каждого классификатора на одинаковой и крупной порции совокупности данных. Хотя перекрестная проверка приводит к меньшему числу ложных открытий, чем прямой метод (WF), оба подхода по-прежнему оценивают коэффициент Шарпа из единственной траектории для стратегии i, y_i, и это оценивание может быть чрезвычайно волатильным. Напротив, комбинаторная прочищенная перекрестная проверка (CPCV) выводит распределение коэффициентов Шарпа из большого числа путей, $j = 1, \ldots, \varphi$, со средним $E[\{y_{i,j}\}_{j=1,\ldots,\varphi}] = \mu_i$ и дисперсией $\sigma^2[\{y_{i,j}\}_{j=1,\ldots,\varphi}] = \sigma_i^2$. Дисперсия выборочного среднего проверки CPCV равна

[1] Мартингейл (martingale) — тип инвестиционной стратегии, используемой трейдерами для капитализации убытков. По мере того как цены на акции снижаются, инвестор покупает больше инвестиций, чтобы расширить свой портфель по более низкой цене. Инвестор покупает больше акций по более низкой цене с *верой* в то, что цена в конечном итоге увеличится и принесет чистую прибыль. Например, инвестор может приобрести акции Google по цене 500 долларов за акцию, а затем развернуться и купить больше акций, если цена упадет до 450 долларов за акцию. Когда цена восстанавливается, инвестор получает больший финансовый возврат на инвестиции. — *Примеч. науч. ред.*

$$\sigma^2[\mu_i] = \varphi^{-2}(\varphi\sigma_i^2 + \varphi(\varphi-1)\sigma_i^2\bar{\rho}_i) = \varphi^{-1}\sigma_i^2(1+(\varphi-1)\bar{\rho}_i),$$

где σ_i^2 — это дисперсия коэффициентов Шарпа по всем траекториям для стратегии i, а $\bar{\rho}_i$ — средняя внедиагональная корреляция среди $\{y_{i,j}\}_{j=1,\ldots,\varphi}$. Проверка CPCV приводит к меньшему числу ложных открытий, чем стандартная перекрестная проверка и прямой метод WF, потому что из $\bar{\rho}_i < 1$ вытекает, что дисперсия выборочного среднего ниже дисперсии выборки,

$$\varphi^{-1}\sigma_i^2 \leq \sigma^2[\mu_i] < \sigma_i^2.$$

Чем больше некоррелированных траекторий, $\bar{\rho}_i \ll 1$, тем ниже будет дисперсия проверки CPCV, а в пределе проверка CPCV сообщит истинный коэффициент Шарпа $E[y_i]$ с нулевой дисперсией, $\lim_{\varphi\to\infty}\sigma^2[\mu_i] = 0$. Систематическое смещение при отборе будет отсутствовать, потому что из $i = 1, \ldots, I$ будет отбираться стратегия с самым высоким *истинным* коэффициентом Шарпа.

Разумеется, мы знаем, что нулевая дисперсия недостижима, так как φ имеет верхнюю границу, $\varphi \leq \varphi[T, \frac{T}{2}]$. И тем не менее для достаточно большого числа траекторий φ проверка (CPCV) может сделать дисперсию бэктеста настолько малой, что вероятность ложного обнаружения будет ничтожно мала.

В главе 11 мы утверждали, что бэктестовая переподгонка может быть самой важной открытой проблемой во всех математических финансах. Давайте посмотрим, как проверка CPCV помогает решать эту проблему на практике. Предположим, что исследователь отправляет стратегию в академический журнал, подкрепляя ее переподогнанным прямым бэктестом, отобранным из большого числа неуказанных испытаний. Журнал вполне может попросить исследователя повторить его эксперименты с использованием комбинаторной прочищенной перекрестной проверки для заданных N и k. Поскольку исследователь не знал заранее число и характеристики траекторий, которые будут бэктестироваться, его усилия по переподгонке будут легко побеждены. Статья будет отклонена или снята с рассмотрения. Надеюсь, что комбинаторная прочищенная перекрестная проверка (CPCV) будет использоваться для уменьшения числа ложных открытий, публикуемых в журналах.

Упражнения

12.1. Предположим, что вы разрабатываете импульсную стратегию для фьючерсного контракта, где прогноз основан на авторегрессивном процессе AR(1). Вы проводите бэктестирование этой стратегии при помощи метода опережения и получаете коэффициент Шарпа –1.5. Затем вы повторяете бэктестирование обратных рядов и получаете коэффициент Шарпа –1.5. Какие математиче-

ские обоснования пренебрежения вторым результатом вы можете предложить (если таковые имеются)?

12.2. Вы разрабатываете стратегию чередования для фьючерсного контракта. Ваш бэктест, проведенный методом опережения, дает коэффициент Шарпа 1.5. Вы увеличиваете длину подготовительного периода, и коэффициент Шарпа падает до 0.7. Вас это не останавливает, и вы представляете лишь результат с высоким коэффициентом Шарпа, утверждая, что стратегия с коротким подготовительным периодом более реалистична. Можно ли считать это ошибкой выборки?

12.3. Ваша стратегия достигла коэффициента Шарпа 1.5 по результатам бэктеста, проведенного методом опережения, и коэффициента Шарпа 0.7 по результатам бэктеста, проведенного методом кросс-валидации. Вас это не останавливает, и вы представляете лишь результат с высоким коэффициентом Шарпа, утверждая, что бэктест, проведенный методом опережения, точен в ретроспективном понимании, а бэктест, проведенный методом кросс-валидации, является симуляцией сценария или логическим упражнением. Можно ли считать это ошибкой выборки?

12.4. Ваша стратегия производит 100 000 прогнозов за определенный период. Вы хотите вывести комбинаторную очищенную кросс-валидацию распределений коэффициентов Шарпа за счет генерирования 1000 путей. Какие вероятные комбинации параметров (N и k) позволят вам достичь этого результата?

12.5. Вы нашли стратегию, которая дает коэффициент Шарпа 1.5 после бэктеста, проведенного методом опережения. Вы пишете научную статью, объясняющую теорию, которая может оправдать подобный результат, и отправляете ее в научный журнал. Редактор отвечает, что один из рецензентов просит вас повторить бэктест, используя метод комбинаторной очищенной кросс-валидации при $N = 100$ и $k = 2$ на основе вашего кода и полного набора данных. Вы следуете инструкциям и получаете коэффициент Шарпа –1 со стандартным отклонением в 0.5. На эмоциях вы прекращаете общение с рецензентом, снимаете свою заявку и подаете работу в другой журнал с более высоким импакт-фактором. Через полгода вашу работу принимают. Вы успокаиваете совесть, говоря себе, что виноват только журнал, который принял ложное открытие, не запросив тестирование через комбинаторную очищенную кросс-валидацию. Вы думаете: «В моем поступке нет ничего аморального, поскольку все так делают и это не запрещено». Какие аргументы вы можете привести в свое оправдание (научные или этические)?

13 Бэктестирование на синтетических данных

13.1. Актуальность

В этой главе мы рассмотрим альтернативный метод бэктестирования, в котором история используется для генерирования синтетической совокупности данных со статистическими характеристиками, оцененными на основе наблюдаемых данных. Это позволит нам пробэктестировать стратегию на большом числе ранее не встречавшихся синтетических тестовых подмножеств, тем самым снижая вероятность того, что эта стратегия будет переподогнана к отдельному подмножеству точек данных[1]. Это очень обширная тема, и для того чтобы достичь некоторой глубины, мы сосредоточимся на бэктестировании торговых правил.

13.2. Правила трейдинга

Инвестиционные стратегии можно определить как алгоритмы, которые постулируют существование рыночной неэффективности. Некоторые стратегии опираются на эконометрические модели предсказания цен с использованием макроэкономических величин, таких как ВВП или инфляция; другие стратегии используют фундаментальную и отчетную информацию для образования цен на ценные бумаги либо занимаются поиском арбитражно-подобных возможностей при ценообразовании производных продуктов и т. д. Например, предположим, что финансовые посредники склонны продавать облигации предыдущей эмиссии (off-the-run) государственных долговых обязательств США за два дня до открытых аукционов по их размещению с целью привлечения денежных средств, необходимых для покупки новой «бумаги». Можно было бы монетизировать эти знания, продавая облигации за три дня до аукциона. Но как? Каждая инвестиционная стратегия требует наличия тактики реализации, часто называемой «правилами биржевой торговли», или просто торговыми правилами.

---

[1] Хотел бы поблагодарить профессора Питера Карра (Peter Carr) из Нью-Йоркского университета за его вклад в эту главу.

Существуют десятки хедж-фондовых стилей, в каждом из которых выполняются десятки уникальных инвестиционных стратегий. Хотя стратегии по своему характеру могут быть весьма неоднородными, тактики относительно однородны. Торговые правила обеспечивают алгоритм, который необходимо соблюдать для входа и выхода из позиции. Например, в позицию входят, когда сигнал стратегии достигает определенного значения. Условия для выхода из позиции часто определяются посредством порогов для взятия прибыли и остановки убытка. Эти правила входа и выхода опираются на параметры, которые обычно калибруются посредством исторических симуляций. Такая практика приводит к проблеме *переподгонки бэктеста*, поскольку эти параметры нацелены на конкретные наблюдения внутривыборочно вплоть до того, что инвестиционная стратегия настолько привязана к прошлому, что она становится непригодной для будущего.

Важно уточнить, что нас интересуют условия коридора выхода, которые максимизируют результативность. Другими словами, позиция уже существует, и вопрос в том, как выйти из нее оптимально. С такой дилеммой часто сталкиваются исполнительные трейдеры, и ее не следует путать с определением порогов входа и выхода для инвестирования в ценную бумагу. Этот альтернативный вопрос исследуется, например, в публикации Bertram [2009].

В публикации Bailey и соавт. [2014, 2017] обсуждается проблема переподгонки бэктеста и предоставляются методы, позволяющие определить, в какой степени симулированная результативность может быть взвинчена из-за этой переподгонки. Хотя выяснение вероятности переподгонки бэктеста является полезным инструментом для отказа от излишних инвестиционных стратегий, было бы лучше избежать риска переподгонки, по крайней мере в контексте калибровки торгового правила. Теоретически это может быть достигнуто путем получения оптимальных параметров для торгового правила непосредственно из стохастического процесса, который генерирует данные, а не путем участия в исторических симуляциях. Именно такой подход мы применяем в этой главе. Используя всю историческую выборку целиком, мы охарактеризуем стохастический процесс, генерирующий наблюдаемый поток финансовых возвратов, и получим оптимальные значения параметров торгового правила без необходимости исторической симуляции.

13.3. Проблема

Предположим, что инвестиционная стратегия S инвестирует в $i = 1, ..., I$ возможностей или ставок. При каждом удобном случае i S занимает позицию из m_i единиц ценной бумаги X, где $m_i \in (-\infty, \infty)$. Сделка, которая вошла в такую возможность, была оценена по стоимости $m_i P_{i,0}$, где $P_{i,0}$ — это средняя цена за единицу, по которой m_i ценных бумаг участвовало в сделке. По мере того как другие участники рынка совершают сделки с ценной бумагой X, мы можем пересчитывать стоимость этой возможности i после t наблюдаемых сделок по текущим рыночным ценам (mark-to-market, MtM) как $m_i P_{i,t}$. Этот пересчет показывает стоимость возможности i,

если бы она была ликвидирована по цене, наблюдаемой на рынке после t сделок. Соответственно, мы можем вычислить прибыль/убыток возможности i по текущим рыночным ценам после t сделок как $\pi_{i,t} = m_i(P_{i,t} - P_{i,0})$.

Стандартное торговое правило обеспечивает логику выхода из возможности i при $t = T_i$. Это происходит, как только подтверждается одно из двух условий:

- $\pi_{i,T_i} \geq \overline{\pi}$, где $\overline{\pi} > 0$ является порогом взятия прибыли;
- $\pi_{i,T_i} \leq \underline{\pi}$, где $\underline{\pi} < 0$ является порогом остановки убытка.

Эти пороги эквивалентны горизонтальным барьерам, о которых мы говорили в контексте метаразметки (глава 3). Поскольку $\underline{\pi} < \overline{\pi}$, только одно из двух условий закрытия может спровоцировать закрытие на основе возможности i. При условии что возможность i может быть закрыта при T_i, ее конечная прибыль/убыток будет выражен как π_{i,T_i}. На стадии открытия каждой возможности главной целью является фиксирование ожидаемой прибыли $E_0[\pi_{i,T_i}] = m_i(E_0[\pi_{i,T_i}] - P_{i,0})$, где $E_0[P_{i,T_i}]$ — это спрогнозированная цена, а $P_{i,0}$ — это уровень открытия возможности i.

Определение 1: торговое правило. Торговое правило для стратегии S определяется множеством параметров $R := \{\underline{\pi}, \overline{\pi}\}$.

Один из способов калибровки торгового правила (методом полного перебора) состоит в том, чтобы:

1. Определить множество альтернативных значений R, $\Omega := \{R\}$.
2. Просимулировать исторически (пробэктестировать) результативность S согласно альтернативным значениям $R \in \Omega$.
3. Отобрать оптимальное R^*.

Более формально:

$$R^* = \arg\max_{R \in \Omega}\{SR_R\};$$
$$SR_R = \frac{E[\pi_{iT_i} \mid R]}{\sigma[\pi_{iT_i} \mid R]}, \tag{13.1}$$

где $E[.]$ и $\sigma[.]$ — это, соответственно, математическое ожидание и среднеквадратическое отклонение от π_{i,T_i}, обусловленные торговым правилом R, на $i = 1, ..., I$. Другими словами, уравнение (13.1) максимизирует коэффициент Шарпа S на I возможностях по всему пространству альтернативных торговых правил R (см. публикацию Bailey and Lopez de Prado [2012], где дается определение и анализ коэффициента Шарпа). Поскольку мы считаем с двумя переменными, максимизируя SR_R по выборке размера I, легко достичь переподгонки R. Тривиальная переподгонка происходит, когда пара $(\underline{\pi}, \overline{\pi})$ ориентируется на несколько выбросов. В публикации Bailey и соавт. [2017] предоставляется строгое определение бэкте-

стовой переподгонки, которое может быть применено к нашему исследованию торговых правил следующим образом.

Определение 2: переподогнанное торговое правило. R^* переподогнано, если $\mathrm{E}\left[\frac{\mathrm{E}[\pi_{j,T_j}|R^*]}{\sigma[\pi_{j,T_j}|R^*]}\right] < \mathrm{Me}_\Omega\left[\mathrm{E}\left[\frac{\mathrm{E}[\pi_{j,T_j}|R]}{\sigma[\pi_{j,T_j}|R]}\right]\right]$, где $j = I + 1, \ldots, J$, а $\mathrm{Me}_\Omega[.]$ — это медиана.

В интуитивном плане оптимальное внутривыборочное (IS, $i \in [1, I]$) торговое правило R^* является переподогнанным, когда от него ожидается отставание по результативности от медианы альтернативных вневыборочных торговых правил $R \in \Omega$ (OOS, $j \in [I + 1, J]$). Это по существу то же самое определение, которое мы использовали в главе 11 для получения вероятности PBO. В публикации Bailey и соавт. [2014] утверждается, что трудно не достичь переподгонки бэктеста, в особенности когда есть свободные переменные, способные ориентироваться на конкретные внутривыборочные наблюдения, либо число элементов в Ω велико. Торговое правило вводит такие свободные переменные, потому что R^* можно определить независимо из S. В итоге бэктест получает прибыль от случайного внутривыборочного шума, что делает R^* непригодным для вневыборочных возможностей. Те же авторы показывают, что переподгонка приводит к отрицательной вневыборочной результативности, когда $\Delta\pi_{i,t}$ проявляет внутрирядовую зависимость. В то время как вероятность PBO предоставляет полезный метод для оценивания того, в какой степени бэктест был переподогнан, было бы удобно избегать этой проблемы изначально[1]. Этой цели мы посвящаем следующий далее раздел.

13.4. Наш математический каркас

До сих пор мы не охарактеризовали стохастический процесс, из которого вынимаются наблюдения $\pi_{i,t}$. Мы заинтересованы в отыскании оптимального торгового правила (optimal trading rule, OTR) для тех сценариев, где переподгонка была бы наиболее разрушительной, например, когда $\pi_{i,t}$ проявляет внутрирядовую корреляцию. В частности, предположим дискретный процесс Орнштейна—Уленбека (O—U) на ценах

$$P_{i,t} = (1-\varphi)\mathrm{E}_0[P_{i,T_i}] + \varphi P_{i,t-1} + \sigma\varepsilon_{i,t} \tag{13.2}$$

такой, что случайные шоки одинаково распределены и взаимно независимы и $\varepsilon_{i,t} \sim N(0, 1)$. Начальное значение посева для этого процесса равно $P_{i,0}$, уровень, на который ориентируется возможность i, равен $\mathrm{E}_0[P_{i,T_i}]$, а φ определяет скорость,

[1] Данная стратегия вполне может по-прежнему быть результатом ретротестовой переподгонки, но, по крайней мере, торговое правило не внесло свой вклад в эту проблему.

с которой $P_{i,0}$ сходится к $\mathrm{E}_0[P_{i,T_i}]$. Поскольку $\pi_{i,t} = m_i(P_{i,t} - P_{i,0})$, из уравнения (13.2) вытекает, что результативность возможности i характеризуется процессом

$$\frac{1}{m_i}\pi_{i,t} = (1-\varphi)\mathrm{E}_0[P_{iT_i}] + P_{i,0} + \varphi P_{i,t-1} + \sigma\varepsilon_{i,t}. \tag{13.3}$$

Из доказательства утверждения 4 в публикации Bailey and Lopez de Prado [2013] можно показать, что распределение процесса, указанного в уравнении (13.2), является гауссовым с параметрами

$$\pi_{i,t} \sim N\left[m_i\left((1-\varphi)\mathrm{E}_0[P_{i,T_i}]\sum_{j=0}^{t-1}\varphi^i - P_{i,0}\right),\ m_i^2\sigma^2\sum_{j=0}^{t-1}\varphi^{2j}\right] \tag{13.4}$$

и необходимым и достаточным условием его стационарности является то, что $\varphi \in (-1, 1)$. С учетом множества входных параметров $\{\sigma, \varphi\}$ и начальных условий $\{P_{i,0}, \mathrm{E}_0[P_{i,T_i}]\}$, связанных с возможностью i, существует ли $R^* := (\underline{\pi}, \overline{\pi})$? Точно так же, в случае если стратегия S предсказывает цель $\overline{\pi}$ по прибыли, можем ли мы вычислить оптимальную остановку убытка $\underline{\pi}$ с учетом входных значений $\{\sigma, \varphi\}$? Если ответ на эти вопросы утвердительный, то бэктест для определения R^* не требуется, что позволит избежать проблемы переподгонки торгового правила. В следующем разделе мы покажем, как ответить на эти вопросы экспериментально.

13.5. Численное определение оптимальных торговых правил

В предыдущем разделе мы использовали спецификацию Орнштейна—Уленбека (O—U) для характеристики стохастического процесса, генерирующего финансовые возвраты стратегии S. В этом разделе мы представим процедуру численного вывода оптимального торгового правила (OTR) для любой спецификации в общем случае и для спецификации O—U в частности.

13.5.1. Алгоритм

Алгоритм состоит из пяти последовательных шагов.

Шаг 1. Мы оцениваем входные параметры $\{\sigma, \varphi\}$ путем линеаризации уравнения (13.2):

$$P_{i,t} = \mathrm{E}_0[P_{i,T_i}] + \varphi(P_{i,t-1} - \mathrm{E}_0[P_{i,T_i}] + \xi_t. \tag{13.5}$$

Затем мы можем сформировать векторы X и Y, упорядочив возможности:

$$X = \begin{bmatrix} P_{0,0} - \mathrm{E}_0[P_{0,T_0}] \\ P_{0,1} - \mathrm{E}_0[P_{0,T_0}] \\ \cdots \\ P_{0,T-1} - \mathrm{E}_0[P_{0,T_0}] \\ \cdots \\ P_{I,0} - \mathrm{E}_0[P_{I,T_I}] \\ \cdots \\ P_{I,T-1} - \mathrm{E}_0[P_{I,T_I}] \end{bmatrix}; \; Y = \begin{bmatrix} P_{0,1} \\ P_{0,2} \\ \cdots \\ P_{0,T} \\ \cdots \\ P_{I,1} \\ \cdots \\ P_{I,T} \end{bmatrix}; \; Z = \begin{bmatrix} \mathrm{E}_0[P_{0,T_0}] \\ \mathrm{E}_0[P_{0,T_0}] \\ \cdots \\ \mathrm{E}_0[P_{0,T_0}] \\ \cdots \\ \mathrm{E}_0[P_{I,T_I}] \\ \cdots \\ \mathrm{E}_0[P_{I,T_I}] \end{bmatrix} \tag{13.6}$$

Применяя обычные наименьшие квадраты (OLS) к уравнению (13.5), мы можем оценить исходные параметры спецификации O—U как

$$\begin{aligned} \hat{\varphi} &= \frac{\mathrm{cov}[Y,X]}{\mathrm{cov}[X,X]}; \\ \hat{\xi}_t &= Y - Z - \hat{\varphi}X; \\ \hat{\sigma} &= \sqrt{\mathrm{cov}[\hat{\xi}_t, \hat{\xi}_t]}, \end{aligned} \tag{13.7}$$

где cov [·, ·] — это ковариационный оператор.

Шаг 2. Мы строим решетку пар, состоящую из остановки убытка и взятия прибыли ($\underline{\pi}$, $\overline{\pi}$). Например, декартово произведение $\underline{\pi} = \{-1\sigma, -\sigma, ..., -10\sigma\}$ и $\overline{\pi} = \{\frac{1}{2}\sigma, \sigma, ..., 10\sigma\}$ дает нам 20 × 20 узлов, каждый из которых представляет собой альтернативное торговое правило $R \in \Omega$.

Шаг 3. Мы генерируем большое число траекторий (например, 100 000) для $\pi_{i,t}$, применяя наши оценки $\{\hat{\sigma}, \hat{\varphi}\}$. В качестве значений посева мы используем наблюдаемые начальные условия $\{P_{i,0}, \mathrm{E}_0[P_{i,T_i}]\}$, связанные с возможностью i. Поскольку невозможно владеть позицией в течение неограниченного периода времени, мы можем ввести максимальный период владения (например, 100 наблюдений), в который позиция закрывается, несмотря на то что $\underline{\pi} \le \pi_{i,100} \le \overline{\pi}$. Этот максимальный период владения эквивалентен вертикальному бару тройного барьерного метода (глава 3)[1].

Шаг 4. Мы применяем 100 000 траекторий, сгенерированных на шаге 3, на каждом узле решетки 20 × 20 ($\underline{\pi}$, $\overline{\pi}$), созданной на шаге 2. Для каждого узла мы применя-

[1] Торговое правило R можно охарактеризовать как функцию не от горизонтальных, а от трех барьеров. Это изменение не повлияет на процедуру. Оно просто добавит еще одну размерность в решетку (20 × 20 × 20). В данной главе мы не рассматриваем эту конфигурацию, потому что это сделало бы визуализацию метода менее интуитивно понятной.

ем логику остановки убытка и взятия прибыли, дающую нам 100 000 значений π_{iT_i}. Точно так же для каждого узла мы вычисляем коэффициент Шарпа, связанный с этим торговым правилом, как описано в уравнении (13.1). См. публикацию Bailey and Lopez de Prado [2012] относительно исследования интервала достоверности оценщика коэффициента Шарпа. Этот результат можно использовать тремя разными способами: шаги 5а, 5б и 5в.

Шаг 5а. Мы определяем пары $(\underline{\pi}, \overline{\pi})$ внутри решетки торговых правил, которые оптимальны с учетом входных параметров $\{\hat{\sigma}, \hat{\varphi}\}$ и наблюдаемых начальных условий $\{P_{i,0}, E_0[P_{i,T_i}]\}$.

Шаг 5б. Если стратегия S обеспечивает цель $\overline{\pi}_i$ по прибыли для конкретной возможности i, то мы можем использовать эту информацию в сочетании с результатами на шаге 4 с целью определения оптимальной остановки убытка $\underline{\pi}_i$.

Шаг 5в. Если трейдер имеет максимальную остановку убытка $\underline{\pi}_i$, предписанную руководством фонда для возможности i, то мы можем использовать эту информацию в сочетании с результатами шага 4 с целью определения оптимального взятия прибыли $\underline{\pi}_i$ в диапазоне остановок убытков $[0, \underline{\pi}_i]$.

В публикации Bailey and Lopez de Prado [2013] доказывается, что период полураспада[1] процесса в уравнении (13.2) равен $\tau = -\frac{\log[2]}{\log[\varphi]}$ при соблюдении требования, что $\varphi \in (0, 1)$. Из этого результата мы можем определить значение φ, связанное с определенным периодом полураспада τ как $\varphi = 2^{-1/\tau}$.

13.5.2. Реализация

Листинг 13.1 обеспечивает реализацию на Python экспериментов, проведенных в этой главе. Функция `main` порождает декартово произведение параметров $(E_0[P_{iT_i}], \tau)$, характеризующих стохастический процесс из уравнения (13.5). Без потери общности во всех симуляциях мы использовали $\sigma = 1$. Затем для каждой пары $(E_0[P_{iT_i}], \tau)$ функция `batch` вычисляет коэффициенты Шарпа, связанные с разными торговыми правилами.

Листинг 13.1. Исходный код Python для определения оптимальных торговых правил

```
import numpy as np
from random import gauss
from itertools import product
#------------------------------------
```

[1] В физике понятие полураспада служит для измерения скорости распада конкретного вещества. Полураспад — это время, затрачиваемое данным количеством вещества на распад до половины его массы. Применительно к данной теме полураспад показывает медленность процесса либо время достижения ожидаемого значения. — *Примеч. науч. ред.*

```
def main():
    rPT=rSLm=np.linspace(0,10,21)
    count=0
    for prod_ in product([10,5,0,-5,-10],[5,10,25,50,100]):
        count+=1
        coeffs={'forecast':prod_[0],'hl':prod_[1],'sigma':1}
        output=batch(coeffs,nIter=1e5,maxHP=100,rPT=rPT,rSLm=rSLm)
    return output
```

Листинг 13.2 вычисляет решетку 20 × 20 коэффициентов Шарпа, по одному для каждого торгового правила ($\underline{\pi}$, $\bar{\pi}$) с учетом пары параметров ($E_0[P_{iT_i}]$, τ). Вертикальный барьер есть, потому что максимальный период владения установлен равным 100 (`maxHP=100`). Мы задали $P_{i,0} = 0$, так как именно расстояние ($P_{i,t-1} - E_0[P_{iT_i}]$) в уравнении (13.5) управляет схождением, а не конкретные абсолютные ценовые уровни. При касании первого из трех барьеров цена выхода сохраняется, и начинается следующая итерация. После завершения всех итераций (1E5) коэффициент Шарпа может быть вычислен для этой пары ($\underline{\pi}$, $\bar{\pi}$), и алгоритм переходит к следующей паре. Когда все пары торговых правил обработаны, результаты возвращаются в функцию `main`. Этот алгоритм может быть параллелизован, подобно тому как мы делали для тройного барьерного метода в главе 3. Мы оставляем эту задачу в качестве упражнения.

Листинг 13.2. Программный код Python для определения оптимальных торговых правил

```
def batch(coeffs,nIter=1e5,maxHP=100,rPT=np.linspace(.5,10,20),
    rSLm=np.linspace(.5,10,20),seed=0):
    phi,output1=2**(-1./coeffs['hl']),[]
    for comb_ in product(rPT,rSLm):
        output2=[]
        for iter_ in range(int(nIter)):
            p,hp,count=seed,0,0
            while True:
                p=(1-phi)*coeffs['forecast']+phi*p+coeffs['sigma']*gauss(0,1)
                cP=p-seed;hp+=1
                if cP>comb_[0] or cP<-comb_[1] or hp>maxHP:
                    output2.append(cP)
                    break
        mean,std=np.mean(output2),np.std(output2)
        print comb_[0],comb_[1],mean,std,mean/std
        output1.append((comb_[0],comb_[1],mean,std,mean/std))
    return output1
```

13.6. Экспериментальные результаты

В табл. 13.1 перечислены сочетания, проанализированные в данном исследовании. Несмотря на то что разные значения этих входных параметров приведут к разным численным результатам, применяемые сочетания позволяют проанализировать

наиболее общие случаи. Столбец «Прогноз» относится к $E_0[P_{iT_i}]$, столбец «Полураспад» относится к τ; столбец «Сигма» относится к σ, столбец «maxHP» обозначает максимальный период владения.

На следующих далее рисунках мы построили графики не-среднегодовых коэффициентов Шарпа, которые являются результатом разных сочетаний условий выхода для взятия прибыли и остановки убытка. Для простоты мы опустили отрицательный знак на оси y (остановки убытков). Коэффициенты Шарпа представлены в оттенках серого (более светлый указывает на более высокую результативность; более темный указывает на более низкую результативность), в формате, известном как тепловая карта. Результативность (π_{iT_i}) вычисляется на единицу ($m_i = 1$), находящуюся во владении, так как другие значения m_i просто будут перешкалировать результативность, не оказывая влияния на коэффициент Шарпа. Транзакционные издержки могут быть легко добавлены, но для образовательных целей лучше строить графики результатов без них, с тем чтобы вы могли видеть симметрию функций.

Таблица 13.1. Используемые в симуляциях сочетания входных параметров

Рисунок	Прогноз	Полураспад	Сигма	maxHP (максимальный период владения)
13.1	0	5	1	100
13.2	0	10	1	100
13.3	0	25	1	100
13.4	0	50	1	100
13.5	0	100	1	100
13.6	5	5	1	100
13.7	5	10	1	100
13.8	5	25	1	100
13.9	5	50	1	100
13.10	5	100	1	100
13.11	10	5	1	100
13.12	10	10	1	100
13.13	10	25	1	100
13.14	10	50	1	100
13.15	10	100	1	100
13.16	–5	5	1	100
13.17	–5	10	1	100
13.18	–5	25	1	100

Рисунок	Прогноз	Полураспад	Сигма	maxHP (максимальный период владения)
13.19	−5	50	1	100
13.20	−5	100	1	100
13.21	−10	5	1	100
13.22	−10	10	1	100
13.23	−10	25	1	100
13.24	−10	50	1	100
13.25	−10	100	1	100

13.6.1. Случаи с нулевым долгосрочным равновесием

Случаи с нулевым долгосрочным равновесием согласуются с деятельностью маркетмейкеров[1], которые предоставляют ликвидность исходя из того, что ценовые отклонения от текущих уровней будут корректироваться самостоятельно с течением времени. Чем меньше τ, тем меньше авторегрессионный коэффициент ($\varphi = 2^{-1/\tau}$ $1/\tau$). Небольшой авторегрессионный коэффициент в сочетании с нулевой ожидаемой прибылью имеет эффект, который состоит в том, что большинство пар ($\underline{\pi}_i$, $\bar{\pi}_i$) выдают нулевую результативность.

Рисунок 13.1 показывает теплокарту для сочетания параметров $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{0, 5, 1\}$. Полураспад настолько мал, что результативность максимизируется в узком диапазоне сочетаний малых взятий прибылей с большими остановками убытков. Другими словами, оптимальное торговое правило состоит во владении ценностями достаточно долго до тех пор, пока не возникает небольшая прибыль, даже за счет того, что приходится сталкиваться с некими пятикратными или семикратными нереализованными убытками. Коэффициенты Шарпа высоки, достигая уровней около 3.2. Это на самом деле то, что многие маркетмейкеры делают на практике, и согласуется с «дилеммой асимметричных выплат», описанной в публикации Easley и соавт. [2011]. Наихудшее возможное торговое правило в этой конфигурации состояло бы в том, чтобы сочетать короткую остановку убытка с большим порогом взятия прибыли, чего маркетмейкеры избегают на практике. Результативность ближе всего к нейтральной по диагонали решетке, где взятия прибылей и остановки убытков симметричны. Этот результат следует иметь в виду во время маркировки наблюдений тройным барьерным методом (глава 3).

[1] Маркетмейкер (market maker) — брокер, дилер или инвестиционная компания, которая принимает на себя рыночный риск (системный риск), вступая во владение ценной бумагой и торгуя ею в качестве принципала. Маркетмейкеры обязаны постоянно выдавать котировки цен спроса и предложения, а также гарантировать полную продажу или поглощение ценной бумаги по определенной цене. — *Примеч. науч. ред.*

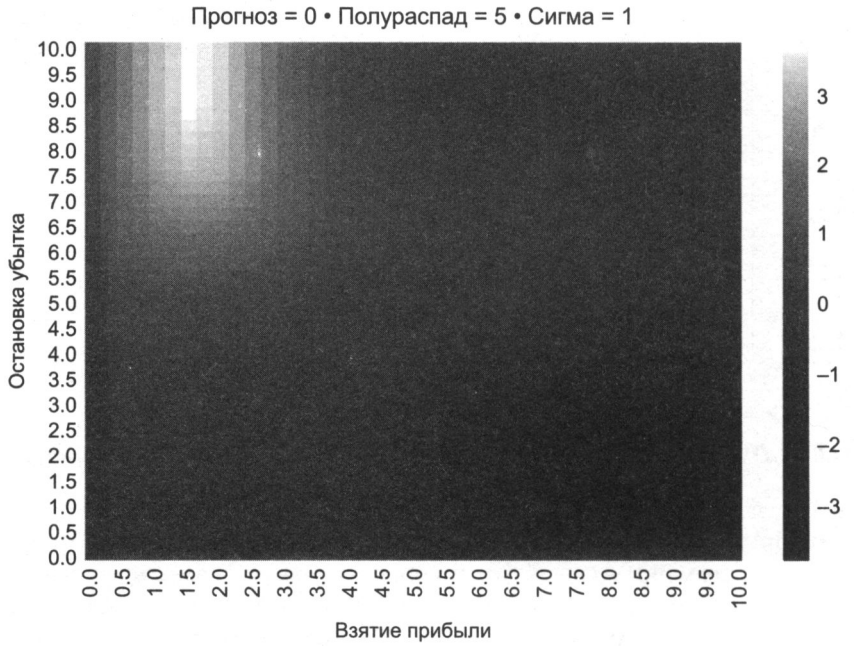


Рис. 13.1. Тепловая карта для $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{0, 5, 1\}$

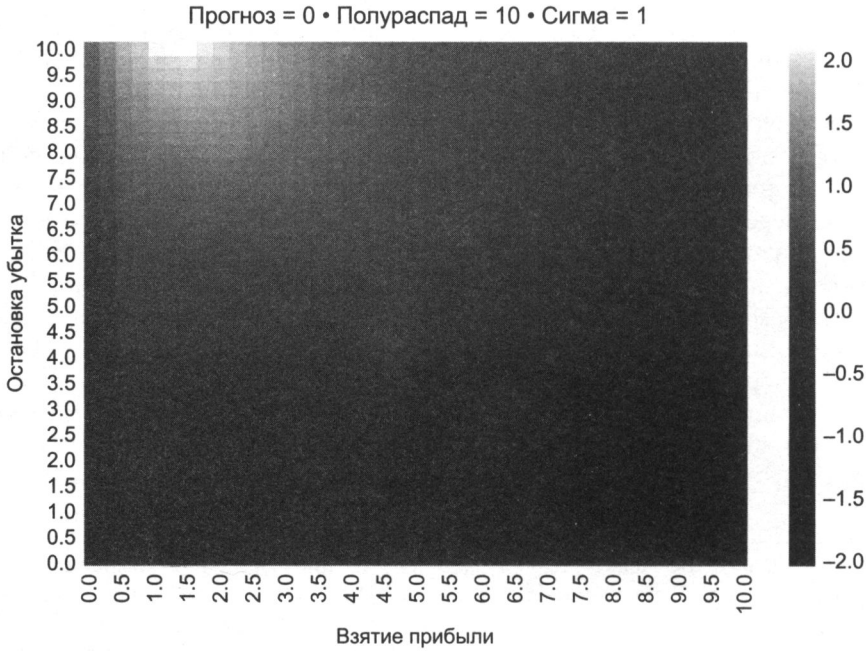


Рис. 13.2. Тепловая карта для $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{0, 10, 1\}$

Рисунок 13.2 показывает, что при увеличении τ с 5 до 10 участки наибольшей и наименьшей результативности рассеиваются по решетке пар ($\underline{\pi}_i$, $\overline{\pi}_i$), в то время как коэффициенты Шарпа уменьшаются. Причина в том, что по мере увеличения периода полураспада увеличивается и величина авторегрессионного коэффициента (напомним, что $\varphi = 2^{-1/\tau}$), а это приближает процесс к случайному блужданию.

На рис. 13.3 $\tau = 25$, что опять же рассеивает участки наибольшей и наименьшей результативности при одновременном снижении коэффициента Шарпа. Рисунок 13.4 ($\tau = 50$) и рис. 13.5 ($\tau = 100$) продолжают эту прогрессию. В конце концов, по мере того как $\varphi \to 1$, нет никаких распознаваемых участков, где результативность может быть максимизирована.

Калибровка торгового правила на случайном блуждании посредством исторических симуляций приведет к переподгонке бэктеста, потому что будет отобрано одно случайное сочетание взятия прибыли и остановки убытка, которое по случаю максимизировало коэффициент Шарпа. Вот почему тестирование синтетических данных так важно: чтобы избежать выбора стратегии из-за того, что некая счастливая статистическая случайность имела место в прошлом (одиночная случайная траектория). Наша процедура предотвращает переподгонку, распознавая, что результативность не проявляет устойчивой закономерности, указывая на отсутствие оптимального торгового правила.

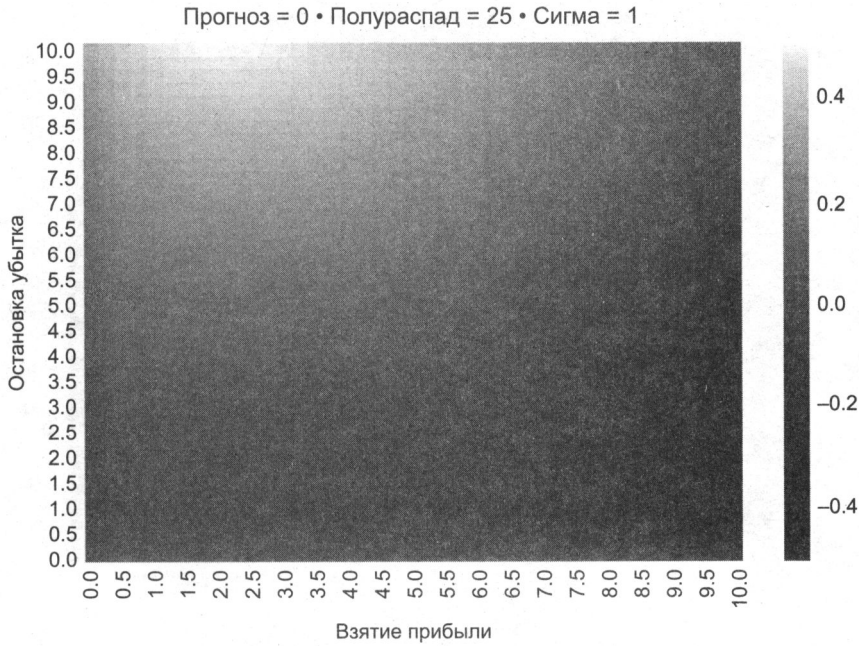


Рис. 13.3. Тепловая карта для $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{0, 25, 1\}$

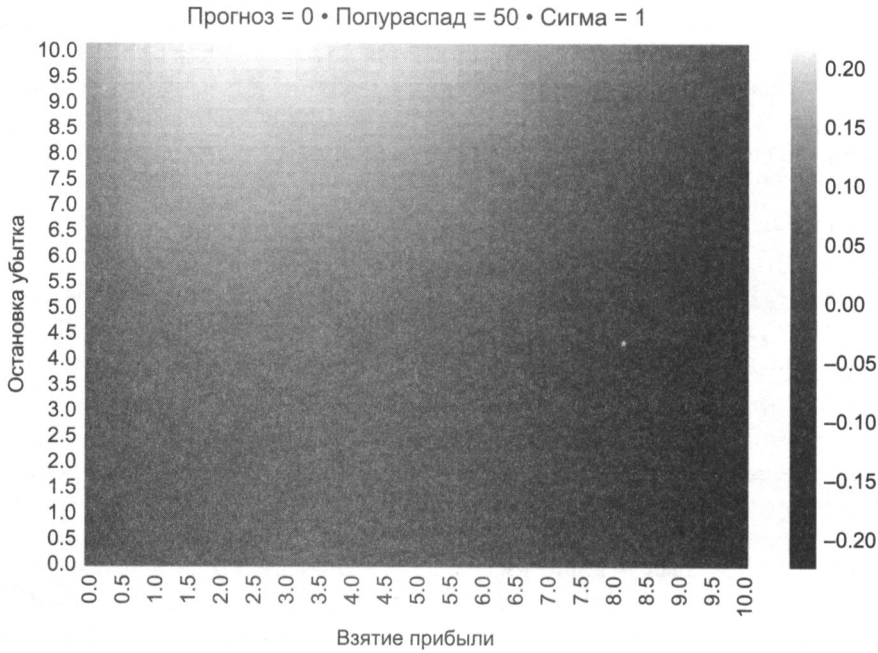


Рис. 13.4. Тепловая карта для $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{0, 50, 1\}$

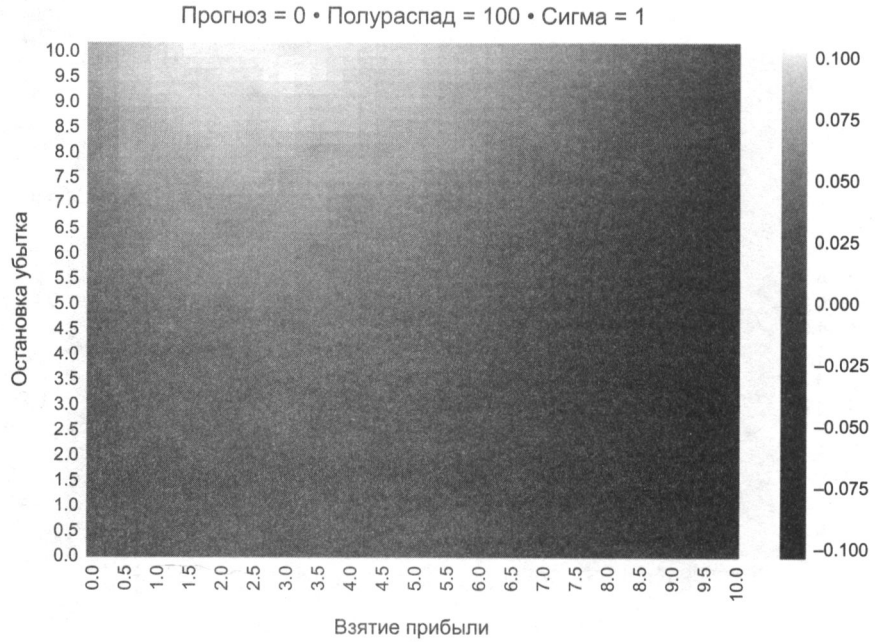


Рис. 13.5. Тепловая карта для $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{0, 100, 1\}$

13.6.2. Случаи с положительным долгосрочным равновесием

Случаи с положительным долгосрочным равновесием согласуются с деятельностью позиционеров[1], таких как хеджевый фонд или менеджер активов. На рис. 13.6 показаны результаты для параметрического сочетания $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{5, 5, 1\}$. Поскольку позиции, как правило, зарабатывают деньги, оптимальное взятие прибыли выше, чем в предыдущих случаях, центрированных вокруг 6, с остановками убытка, которые варьируются между 4 и 10. Участок оптимального торгового правила принимает характерную прямоугольную форму в результате сочетания широкого диапазона остановок убытка с более узким диапазоном взятия прибыли. Результативность является самой высокой во всех экспериментах, с коэффициентами Шарпа около 12.

На рис. 13.7 мы увеличили период полураспада с $\tau = 5$ до $\tau = 10$. Теперь оптимальная результативность достигается при взятии прибыли, центрированном вокруг 5, с остановками убытка в диапазоне между 7 и 10. Диапазон оптимального взятия прибыли шире, а диапазон оптимальных остановок убытка сужается, образуя бывший прямоугольный участок ближе к форме квадрата. Опять же, больший период полураспада приближает процесс к случайному блужданию, и поэтому результативность теперь относительно ниже, чем раньше, с коэффициентами Шарпа около 9.

На рис. 13.8 мы сделали $\tau = 25$. Оптимальное взятие прибыли теперь центрировано вокруг 3, в то время как оптимальные остановки убытка находятся в диапазоне между 9 и 10. Предыдущий квадратный участок оптимальной результативности уступил место полукругу малых взятий прибыли с большими порогами остановки убытка. Снова мы видим ухудшение результативности, с коэффициентами Шарпа 2.7.

На рис. 13.9 период полураспада увеличен до $\tau = 50$. В результате участок оптимальной результативности рассеивается, в то время как коэффициенты Шарпа продолжают падать до 0.8. Это тот же эффект, который мы наблюдали в случае нулевого долгосрочного равновесия (раздел 13.6.1), только с той разницей, что поскольку теперь $E_0[P_{i,T_i}] > 0$, нет симметричного участка худшей результативности.

---

[1] Позиционер (position-taker), или покупатель позиций, — это физическое или юридическое лицо, которое должно принимать преобладающие позиции на рынке, не располагая долей рынка для того, чтобы влиять на рыночную позицию самостоятельно. Аналогичным образом, покупатель цен (price-taker) — это физическое или юридическое лицо, которое должно принимать преобладающие цены на рынке, не располагая долей рынка, чтобы влиять на рыночную цену самостоятельно. В большинстве конкурентных рынков фирмы являются покупателями цен. Если фирмы устанавливают на свою продукцию более высокие цены, чем преобладающие рыночные цены, то потребители просто покупают ее у другого продавца по более низкой цене. На фондовом рынке индивидуальные инвесторы считаются покупателями цен, а маркетмейкерами — те, кто устанавливает цену на ценную бумагу и предлагает ее на рынке. — *Примеч. науч. ред.*

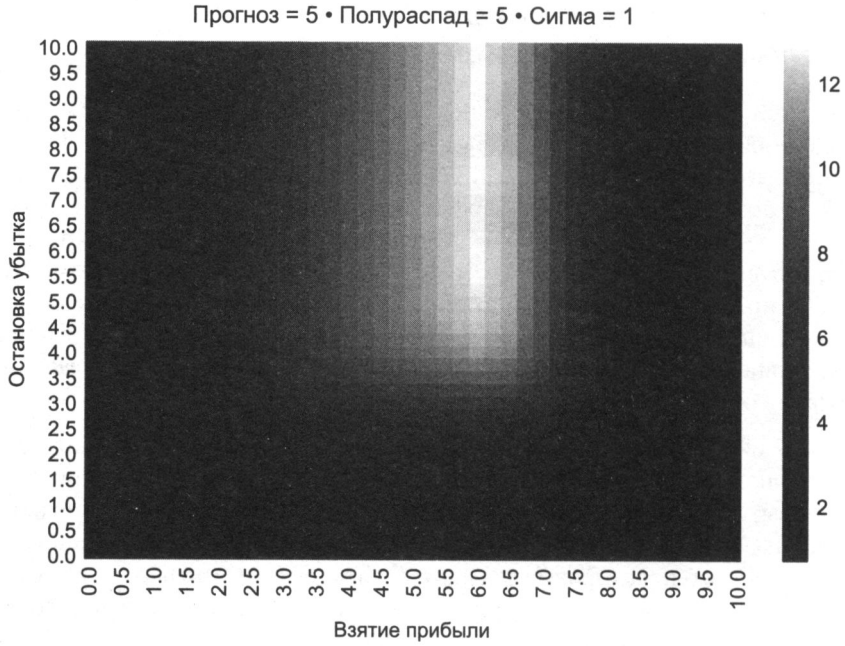


Рис. 13.6. Тепловая карта $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{5, 5, 1\}$

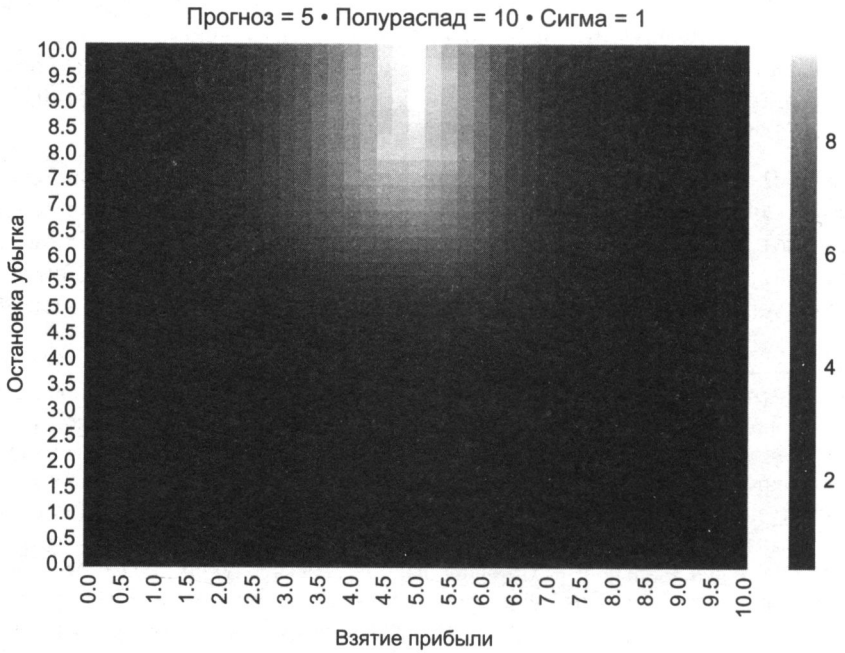


Рис. 13.7. Тепловая карта для $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{5, 10, 1\}$

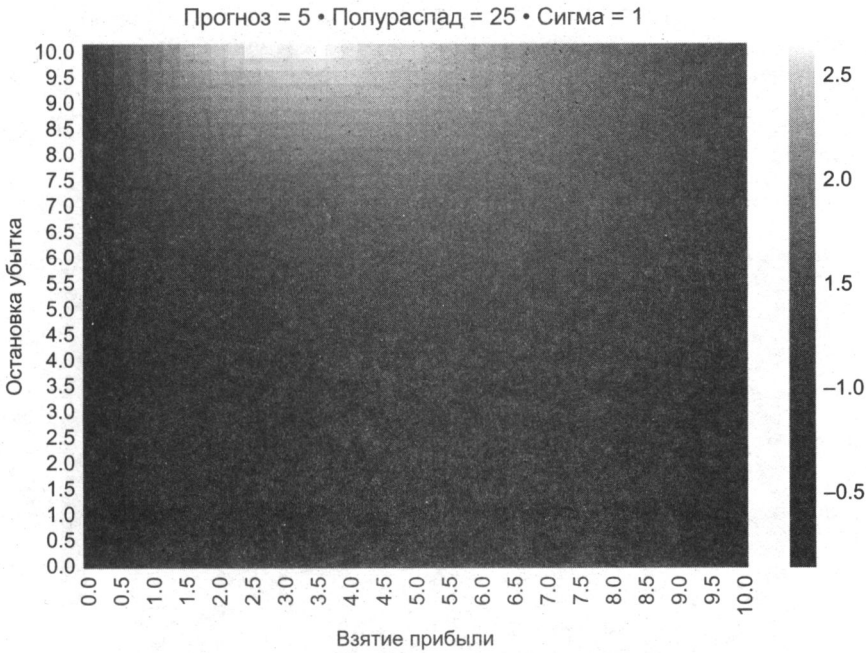


Рис. 13.8. Тепловая карта для $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{5, 25, 1\}$

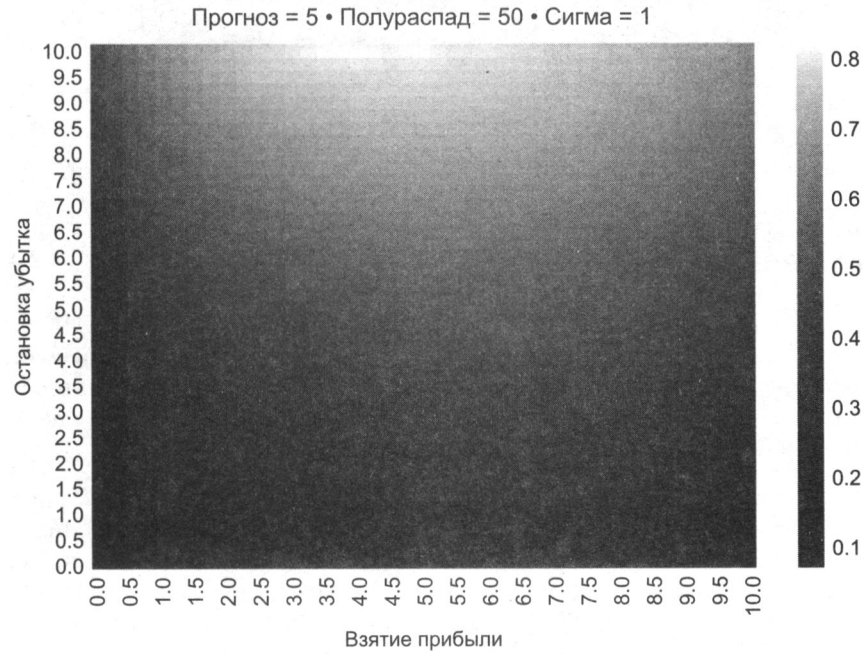


Рис. 13.9. Тепловая карта для $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{5, 50, 1\}$

На рис. 13.10 видно, что $\tau = 100$ приводит к естественному завершению описанного выше тренда. Процесс теперь настолько близок к случайному блужданию, что максимальный коэффициент Шарпа составляет всего 0.32.

Мы наблюдаем аналогичную закономерность на рис. 13.11–13.15, где $E_0[P_{i,T_i}] = 10$ и τ постепенно увеличиваются с 5 соответственно до 10, 25, 50 и 100.

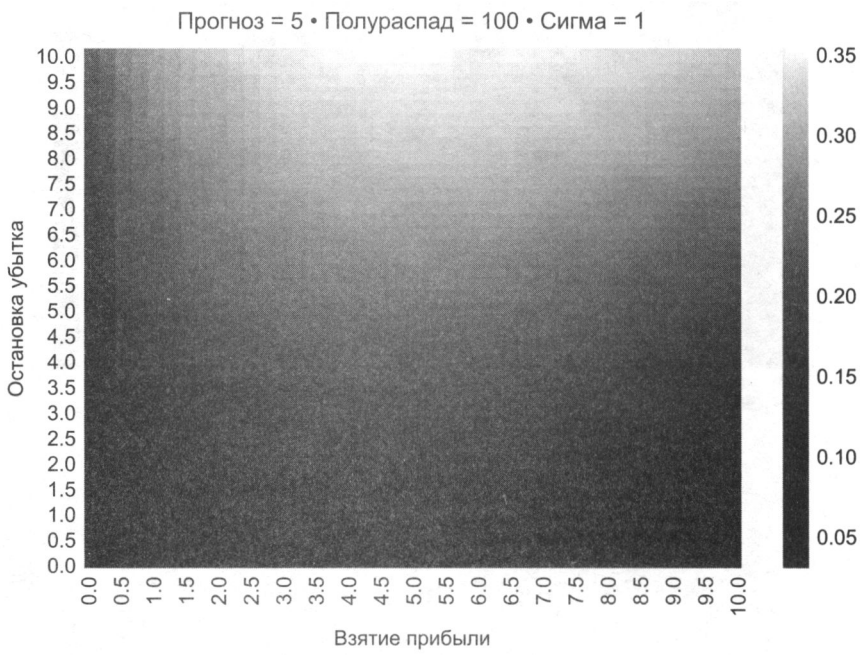


Рис. 13.10. Тепловая карта для $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{5, 100, 1\}$

13.6.3. Случаи с отрицательным долгосрочным равновесием

Рациональный участник рынка не будет инициировать позицию, исходя из того что убыток является ожидаемым исходом. Однако если трейдер признаёт, что убытки являются ожидаемым результатом ранее существовавшей позиции, то ему по-прежнему нужна стратегия, которая принудительно останавливает эту позицию, при этом минимизируя такие убытки.

Мы получили рис. 13.16 в результате применения параметров $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{-5, 5, 1\}$. Если мы сравним рис. 13.16 с рис. 13.6, то создается впечатление, будто один является повернутым дополнением другого. Рисунок 13.6 напоминает повернутый фотонегатив рис. 13.16. Причина в том, что прибыль на рис. 13.6 транслируется в убыток на рис. 13.16, а убыток на рис. 13.6 транслируются в прибыль на рис. 13.16. Один случай является обратной картиной другого, так же как проигрыш азартного игрока — это выигрыш для заведения.

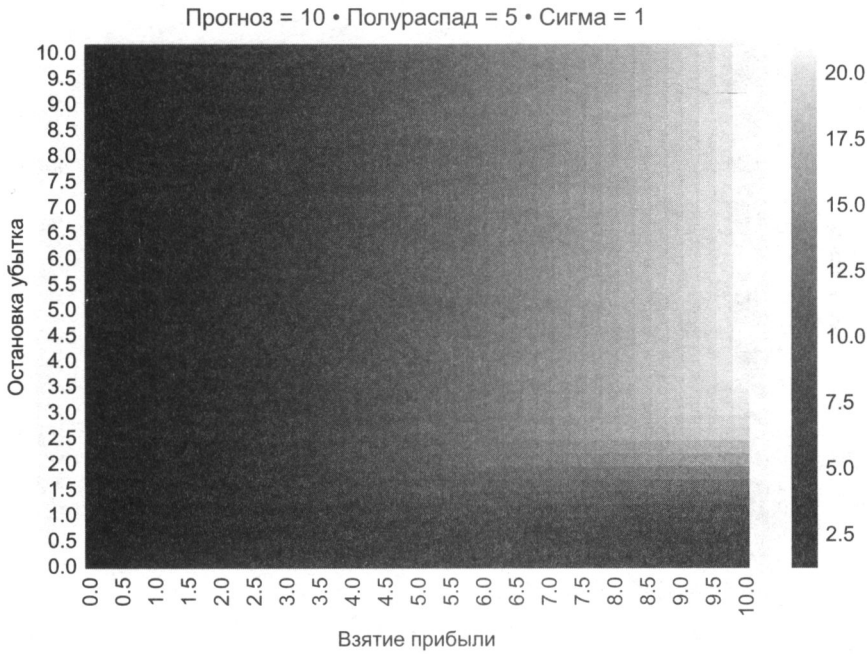


Рис. 13.11. Тепловая карта для $\{E_0[P_{t,T_i}], \tau, \sigma\} = \{10, 5, 1\}$

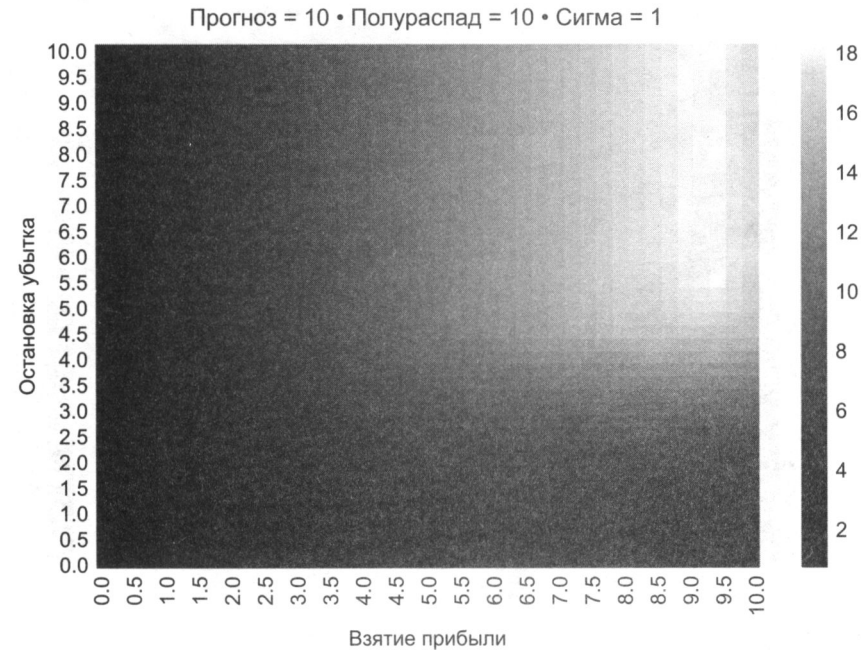


Рис. 13.12. Тепловая карта для $\{E_0[P_{t,T_i}], \tau, \sigma\} = \{10, 10, 1\}$

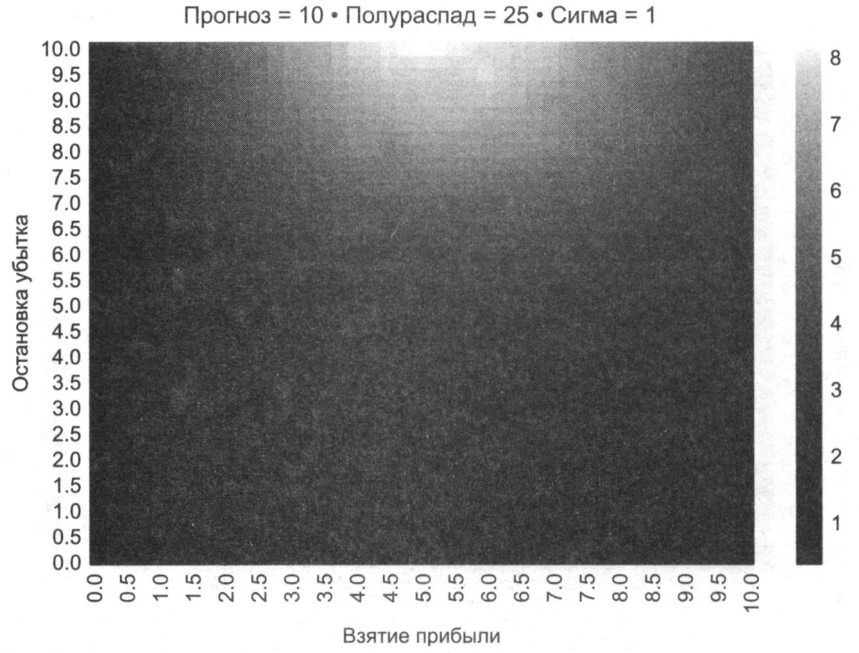


Рис. 13.13. Тепловая карта для $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{10, 25, 1\}$

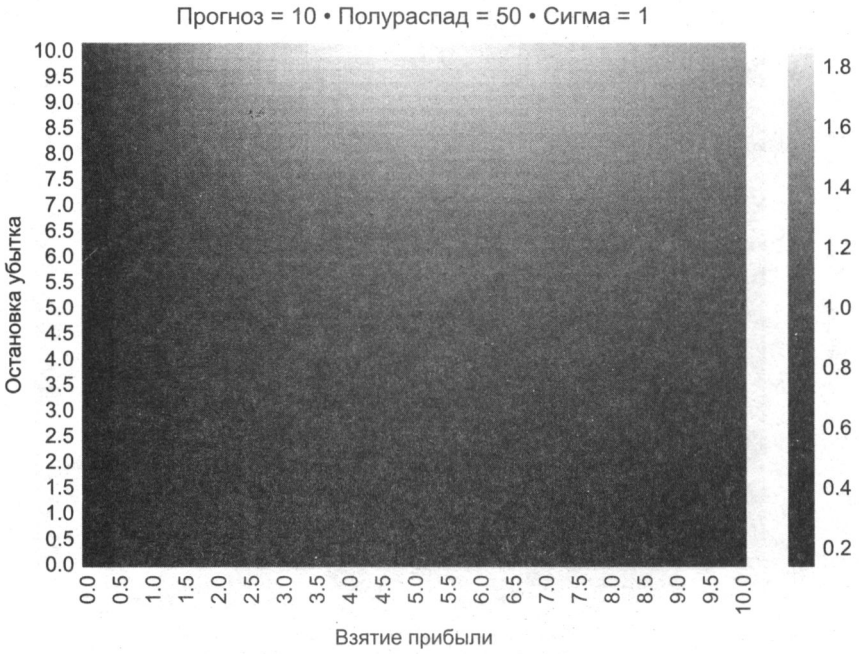


Рис. 13.14. Тепловая карта для $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{10, 50, 1\}$

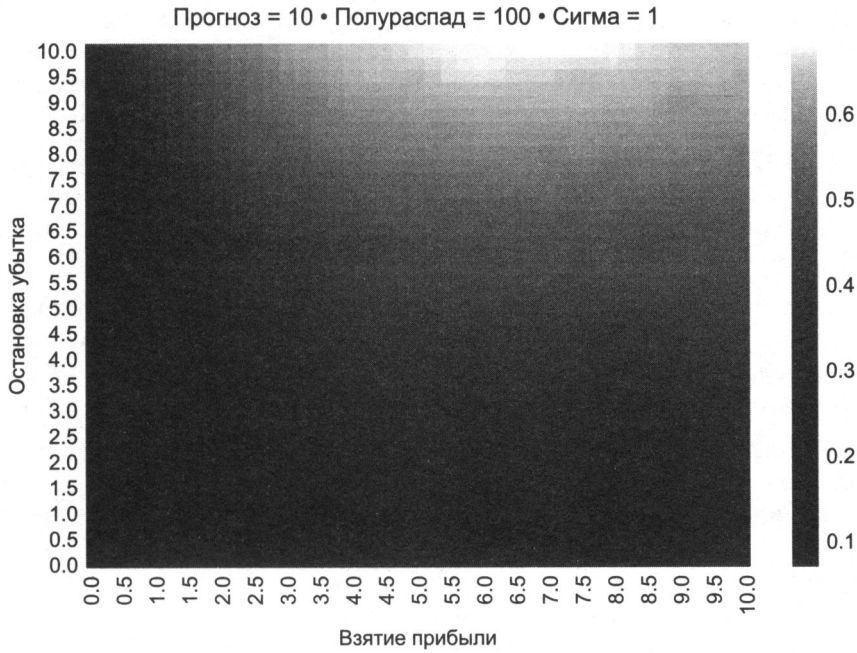


Рис. 13.15. Тепловая карта для $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{10, 100, 1\}$

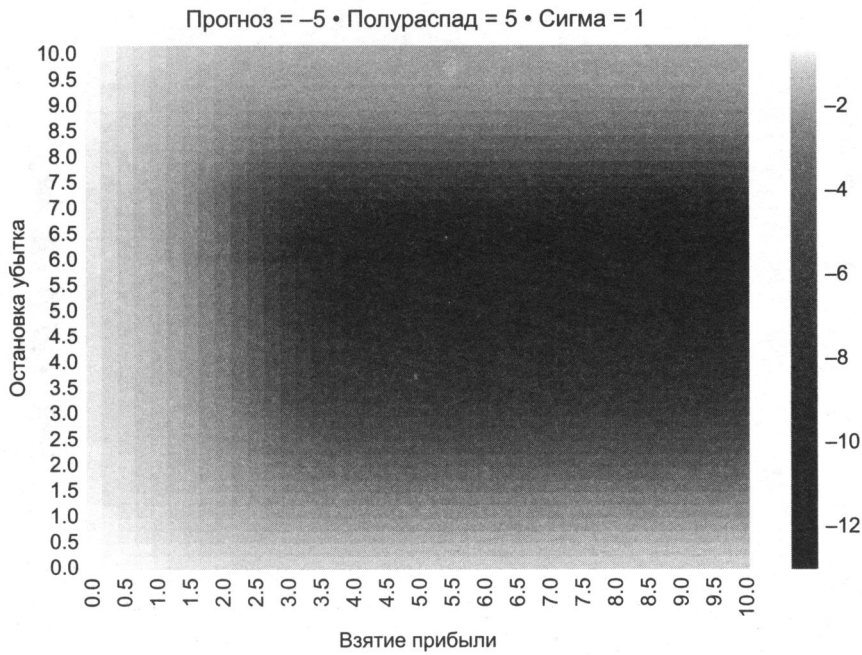


Рис. 13.16. Тепловая карта для $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{-5, 5, 1\}$

Как и ожидалось, коэффициенты Шарпа — отрицательные, с участком худшей результативности вокруг остановки убытка, равного 6, а порогами взятия прибыли в диапазоне между 4 и 10. Теперь прямоугольная форма соответствует участку не лучшей результативности, а худшей, с коэффициентами Шарпа около –12.

На рис. 13.17 $\tau = 10$, и теперь близость к случайному блужданию играет в нашу пользу. Участок худшей результативности рассеивается, и прямоугольный участок становится квадратом. Результативность становится менее отрицательной, с коэффициентами Шарпа около –9.

Эту знакомую прогрессию можно заметить на рис. 13.18, 13.19 и 13.20, поскольку τ поднят до 25, 50 и 100. Опять же, по мере приближения процесса к случайному блужданию результативность выравнивается, и оптимизация торгового правила становится упражнением по переподгонке бэктеста.

Рисунки 13.21–13.25 повторяют тот же процесс для $E_0[P_{i,T_i}] = -10$ и τ, который поступательно увеличивается с 5 до 10, 25, 50 и 100. Возникает та же закономерность, комплементарно повернутая к случаю с положительным долгосрочным равновесием.

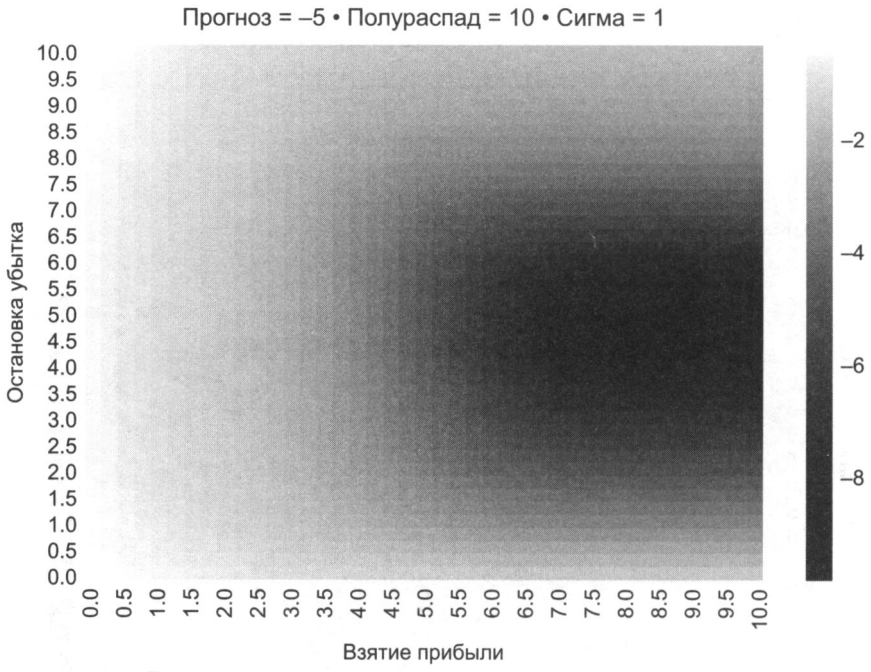


Рис. 13.17. Тепловая карта для $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{-5, 10, 1\}$

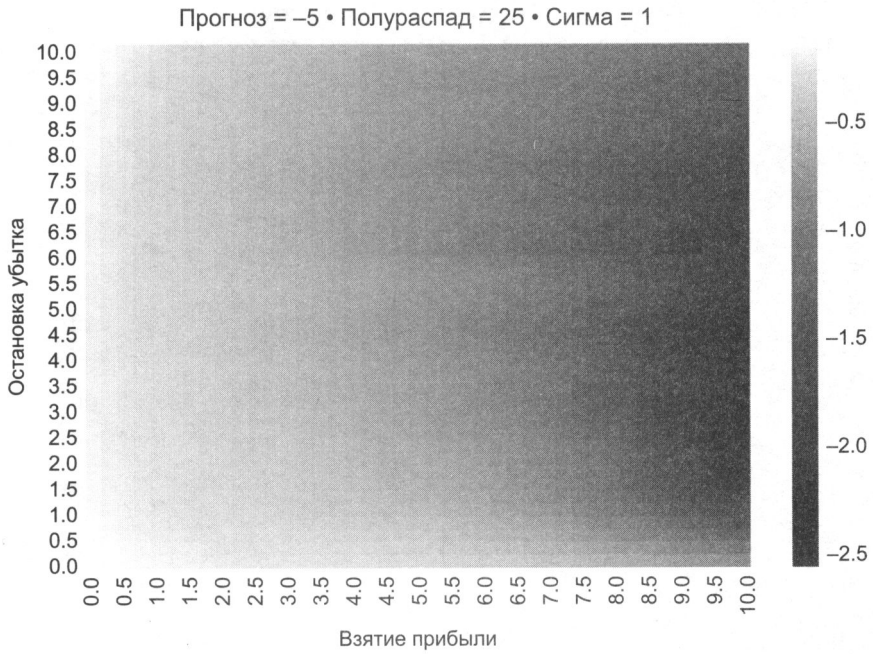


Рис. 13.18. Тепловая карта для $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{-5, 25, 1\}$

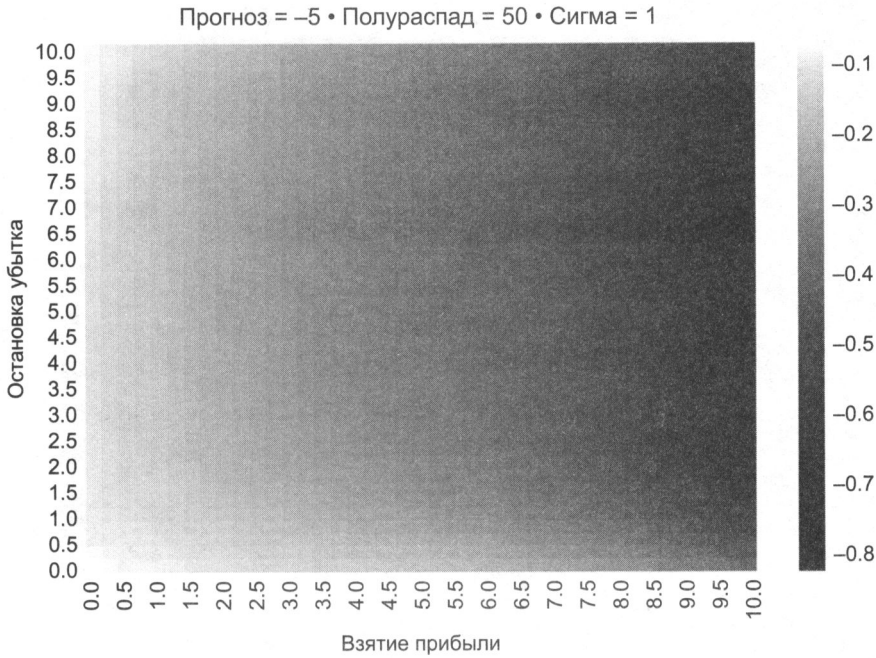


Рис. 13.19. Тепловая карта для $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{-5, 50, 1\}$

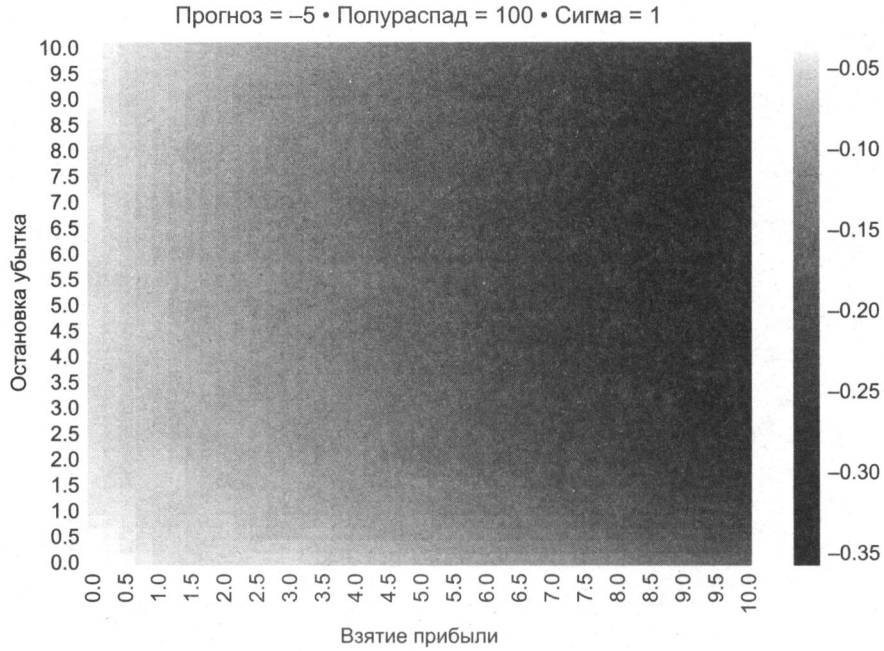


Рис. 13.20. Тепловая карта для $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{-5, 100, 1\}$

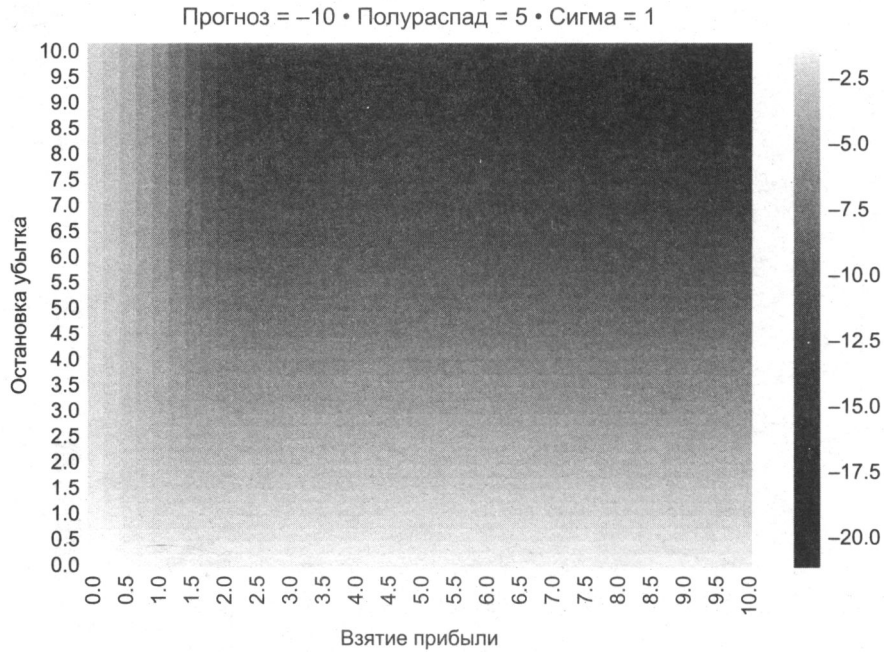


Рис. 13.21. Тепловая карта для $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{-10, 5, 1\}$

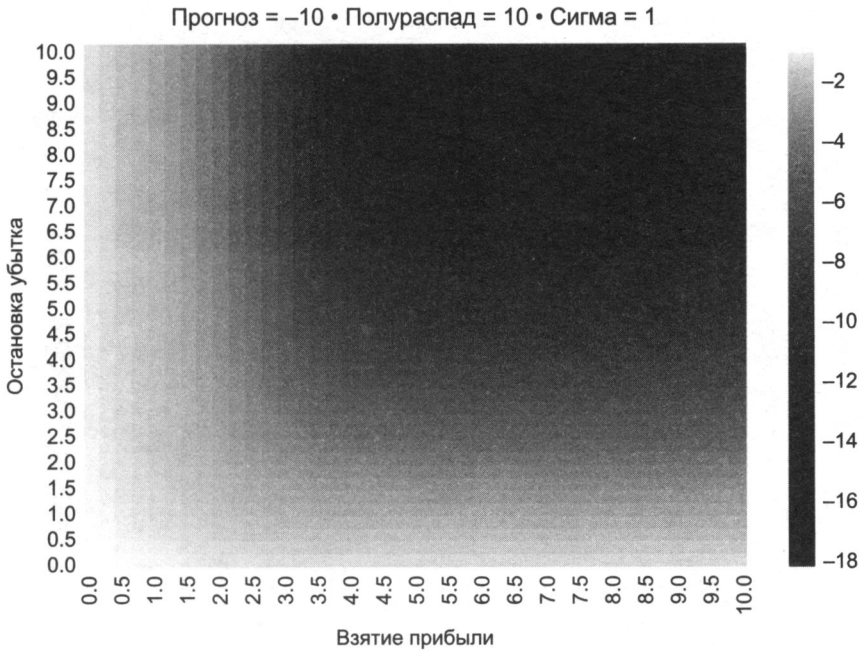


Рис. 13.22. Тепловая карта для $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{-10, 10, 1\}$

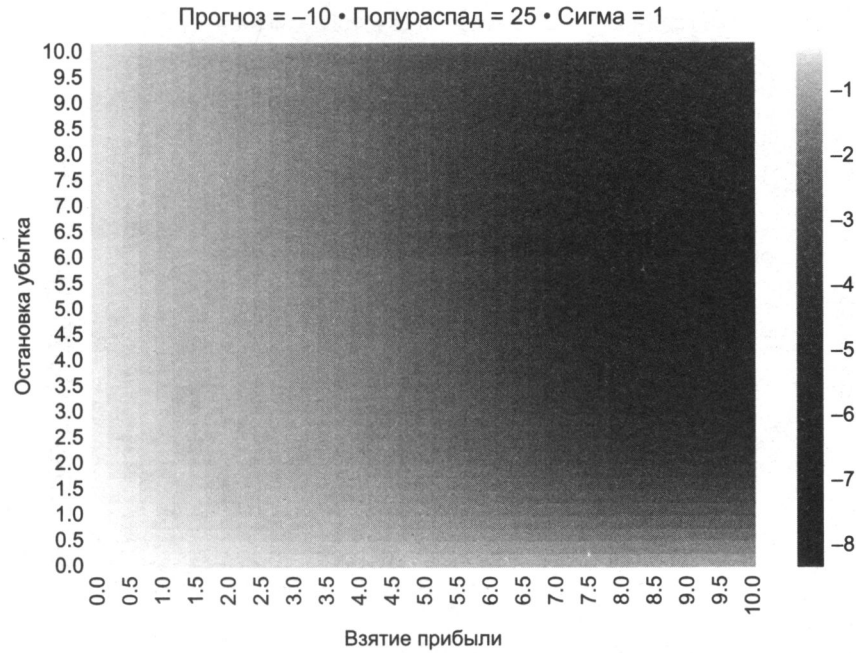


Рис. 13.23. Тепловая карта для $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{-10, 25, 1\}$

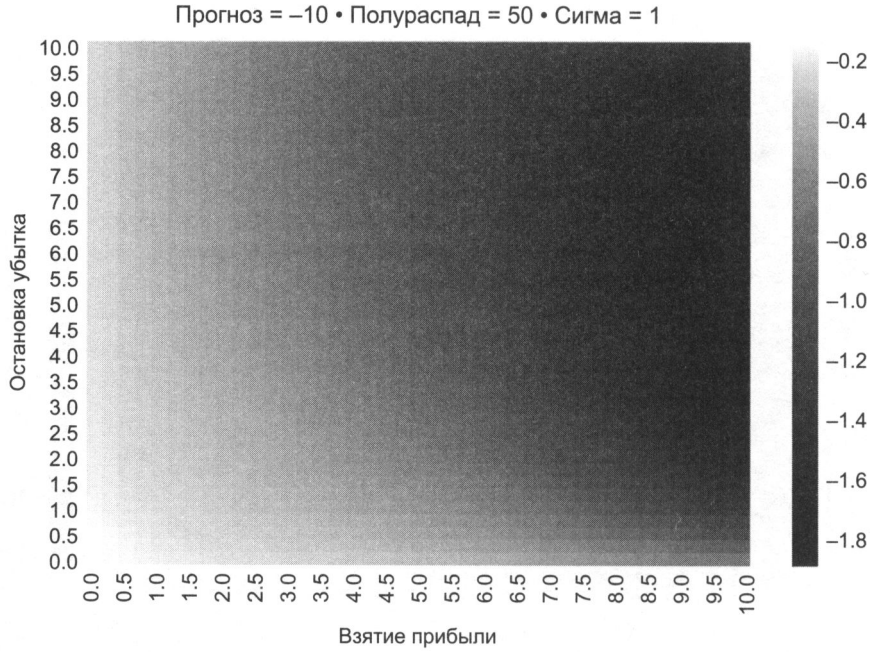


Рис. 13.24. Тепловая карта для $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{-10, 50, 1\}$

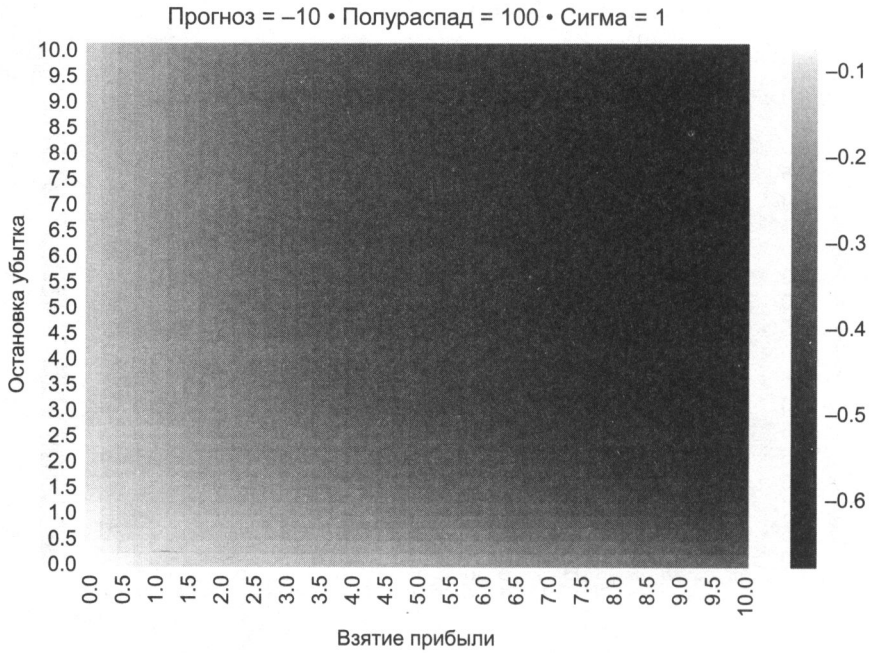


Рис. 13.25. Тепловая карта для $\{E_0[P_{i,T_i}], \tau, \sigma\} = \{-10, 100, 1\}$

13.7. Выводы

В этой главе мы показали, как экспериментально определять оптимальную торговую стратегию, связанную с ценами, подчиняющимися дискретному процессу Орнштейна—Уленбека (O—U). Поскольку выведение такой торговой стратегии не является результатом исторической симуляции, наша процедура позволяет избежать рисков, связанных с переподгонкой бэктеста к одной траектории. Вместо этого оптимальное торговое правило (OTR) выводится из характеристик базового стохастического процесса, который управляет ценами. Такой же подход может быть применен к процессам, отличным от процесса O—U, и мы сосредоточились на этом конкретном процессе только в образовательных целях.

Хотя в этой главе мы не выводим решение проблемы оптимальных торговых стратегий в закрытой форме, наши экспериментальные результаты, похоже, поддерживают следующее предположение об оптимальном торговом правиле OTR:

Предположение: с учетом того что цены финансового инструмента характеризуются дискретным процессом O—U, существует уникальное оптимальное торговое правило с точки зрения сочетания взятия прибыли и остановки убытка, которое максимизирует коэффициент Шарпа данного торгового правила.

С учетом того что эти оптимальные торговые правила могут быть выведены численно в течение нескольких секунд, практически нет стимула для получения решения в закрытой форме. По мере того как он становится все более распространенным в математических исследованиях, экспериментальный анализ данного предположения может помочь нам достичь цели даже при отсутствии доказательства. Для того чтобы доказать эту гипотезу, могут потребоваться годы, если не десятилетия, и все же все проведенные до сих пор эксперименты подтверждают ее эмпирически. Давайте я скажу так: вероятность того, что это предположение ложно, ничтожна по сравнению с вероятностью того, что вы добьетесь переподгонки вашего торгового правила, проигнорировав гипотезу. Следовательно, рациональный курс действий состоит в том, чтобы допустить, что гипотеза верна, и определить оптимальное торговое правило с помощью синтетических данных. В худшем случае торговое правило будет неоптимальным, но все же оно почти наверняка превзойдет переподогнанное торговое правило.

Упражнения

13.1. Предположим, что вы являетесь исполнительным трейдером. Клиент звонит вам с заявкой покрыть короткую позицию, в которую он вошел по цене 100. Он дает вам два условия выхода: взятие прибыли на 90 и остановка убытка на 105.

(a) Если допустить, что, по мнению клиента, цена подчиняется процессу Орнштейна—Уленбека (O—U), разумны ли эти уровни? Для каких параметров?

(б) Можете ли вы придумать альтернативный стохастический процесс, при котором эти уровни имеют смысл?

13.2. Выполните подгонку временного ряда долларовых баров фьючерсного контракта E-mini S&P 500 к процессу Орнштейна–Уленбека (O–U). С учетом этих параметров:

(а) Постройте теплокарту коэффициентов Шарпа для разных уровней взятия прибыли и остановки убытка.

(б) Каким будет оптимальное торговое правило?

13.3. Повторите упражнение 13.2, на этот раз на временном ряде долларовых баров:

(а) 10-летнего фьючерсного контракта на билеты казначейства США.

(б) Фьючерсного контракта на товарную нефть WTI.

(в) Будут ли результаты существенно отличаться? Оправдывает ли это то, что исполнительные трейдеры специализируются на конкретных продуктах?

13.4. Повторите упражнение 13.2 после подразделения временного ряда на две части:

(а) Первый временной ряд заканчивается 15 марта 2009 года.

(б) Второй временной ряд заканчивается 16 марта 2009 года.

(в) Значительно ли отличаются оптимальные торговые правила?

13.5. Сколько времени, по вашим оценкам, потребуется для выведения оптимальных торговых стратегий на 100 наиболее ликвидных фьючерсных контрактах в мире? С учетом результатов упражнения 13.4 как часто, по вашему мнению, вам может потребоваться рекалибровка оптимальных торговых стратегий? Имеет ли смысл вычислять эти данные предварительно?

13.6. Параллелизуйте листинги 13.1 и 13.2 с помощью модуля `mpEngine`, описанного в главе 20.

14 Статистические показатели бэктеста

14.1. Актуальность

В предыдущих главах мы изучили три парадигмы бэктестирования: во-первых, исторические симуляции (прямой метод WF, главы 11 и 12). Во-вторых, сценарные симуляции (метод перекрестной проверки CV и метод комбинаторной прочищенной перекрестной проверки CPCV, глава 12). В-третьих, симуляции на синтетических данных (глава 13). Независимо от выбранной вами парадигмы бэктестирования, вам необходимо сообщать результаты в соответствии с рядом статистических данных, которые инвесторы будут использовать для сравнения и вынесения суждения о вашей стратегии относительно конкурентов. В этой главе мы обсудим некоторые из наиболее часто используемых статистических показателей оценивания результативности. Некоторые из этих статистических показателей включены в глобальные стандарты инвестиционной результативности (global investment performance standards, GIPS)[1], однако для всестороннего анализа результативности требуются показатели, специфичные для рассматриваемых стратегий МО.

14.2. Виды статистических показателей бэктеста

Статистические показатели бэктеста включают в себя метрические показатели, которые используются инвесторами для оценки значимости и сравнения различных инвестиционных стратегий. Эти показатели должны помочь нам выявить потенциально проблемные аспекты стратегии, такие как существенные асимметричные риски или малую емкость. В целом, они могут быть разделены на следующие категории: общие характеристики, показатели результативности, показатели интервалов/просадок, показатели дефицита реализации, показатели эффективности финансового возврата/риска, классификационные балльные оценки и показатели атрибутирования.

[1] Для получения дополнительной информации посетите https://www.gipsstandards.org.

14.3. Основные характеристики

Общие характеристики бэктеста проявляются в следующих статистических показателях:

- **Временной диапазон** (time range) задает начальную и конечную даты. Период, используемый для тестирования стратегии, должен быть достаточно продолжительным, чтобы включать в себя всеобъемлющее число режимов (Bailey and Lopez de Prado [2012]).

- **Средние активы в управлении** (average AUM) — это средняя долларовая стоимость активов под управлением. В целях вычисления этого среднего долларовое значение длинных и коротких позиций считается положительным вещественным числом.

- **Емкость** (capacity) — емкость стратегии можно измерить как наивысшую стоимость AUM, которая обеспечивает целевую результативность с учетом риска. Минимальная стоимость AUM необходима для обеспечения надлежащего размера ставок (глава 10) и диверсификации рисков (глава 16). Помимо этой минимальной стоимости AUM, результативность будет снижаться по мере увеличения стоимости AUM из-за более высоких транзакционных издержек и более низкого оборота.

- **Кредитное плечо** (leverage) измеряет объем заимствований, необходимых для достижения заявленных результатов. Если имеет место кредитное плечо, ему должны быть назначены издержки. Одним из способов измерения кредитного плеча является отношение среднего размера долларовой позиции к средней стоимости AUM.

- **Максимальный размер долларовой позиции** (maximum dollar position size) информирует нас о том, занимала ли стратегия временами долларовые позиции, которые значительно превышали среднюю стоимость AUM. В целом мы будем отдавать предпочтение стратегиям, которые занимают максимальные долларовые позиции, близкие к средней стоимости AUM, указывая на то, что они не опираются на возникновение экстремальных событий (возможно, выбросов).

- **Соотношение длинных позиций** (ratio of longs) — соотношение длинных позиций (лонгов) показывает, какая доля ставок связана с длинными позициями. В длинно-коротких рыночных нейтральных стратегиях это значение в идеале близко к 0.5. Если нет, то стратегия может иметь позиционное смещение, либо бэктестированный период может быть слишком коротким и нерепрезентативным для будущих рыночных условий.

- **Частота ставок** (frequency of bets) — это число ставок в год в бэктесте. Последовательность позиций на одной стороне считается частью одной ставки.

Ставка заканчивается, когда позиция выравнивается или переворачивается на противоположную сторону. Число ставок всегда меньше, чем число сделок (трейдов). Число сделок дает завышенную оценку числа независимых возможностей, открываемых стратегией.

- **Средний период владения** (average holding period) — это среднее число дней, в течение которых ставка держится во владении. Высокочастотные стратегии могут владеть позицией в течение доли секунд, в то время как низкочастотные стратегии могут владеть позицией в течение нескольких месяцев или даже лет. Короткие периоды владения могут ограничить возможности стратегии. Период владения связан с частотой ставок, но от нее отличается. Например, стратегия может делать ставки на ежемесячной основе, рядом с публикацией данных занятости в несельскохозяйственном секторе, где каждая ставка держится во владении всего несколько минут.
- **Среднегодовой оборот** (annualized turnover) измеряет отношение среднего числа долларов, торгуемых в год, к среднегодовой стоимости AUM. Высокий оборот может произойти даже при небольшом числе ставок, так как стратегия может потребовать постоянной регулировки позиции. Высокий оборот может также иметь место при небольшом числе сделок, если каждая сделка предусматривает переворот позиции между максимальной длинной и максимальной короткой.
- **Корреляция с базовым универсумом** (correlation to underlying) — это корреляция между финансовыми возвратами стратегии и возвратами базового инвестиционного универсума. Когда корреляция значительно положительна или отрицательна, стратегия по существу владеет или шортит инвестиционный универсум без особой дополнительной стоимости.

В листинге 14.1 приведен алгоритм, который производит временные штампы боковых (flat) или переворотных (flip) сделок из временно́го ряда библиотеки pandas с целевыми позициями (`tPos`). Он дает нам число сделанных ставок.

Листинг 14.1. Получение времен ставок из временного ряда целевых позиций

```
# ставка происходит между боковыми позициями или позиционными переворотами
df0=tPos[tPos==0].index
df1=tPos.shift(1);df1=df1[df1!=0].index
bets=df0.intersection(df1) # боковое движение
df0=tPos.iloc[1:]*tPos.iloc[:-1].values
bets=bets.union(df0[df0<0].index).sort_values() # tPos переворачивается
if tPos.index[-1] not in bets:bets=bets.append(tPos.index[-1:]) # последняя
                                                                # ставка
```

Листинг 14.2 иллюстрирует реализацию алгоритма, который оценивает средний период владения стратегии с учетом временного ряда целевых позиций (`tPos`) библиотеки pandas.

Листинг 14.2. Реализация оценщика периода владения

```
def getHoldingPeriod(tPos):
    # Получить средний период владения (в днях), используя
    # алгоритм сопряжения среднего времени входа
    hp,tEntry=pd.DataFrame(columns=['dT','w']),0.
    pDiff,tDiff=tPos.diff(),(tPos.index-tPos.index[0])/np.timedelta64(1,'D')
    for i in xrange(1,tPos.shape[0]):
        if pDiff.iloc[i]*tPos.iloc[i-1]>=0: # увеличился или не изменился
            if tPos.iloc[i]!=0:
                tEntry=(tEntry*tPos.iloc[i-1]+tDiff[i]*pDiff.iloc[i])/
                        tPos.iloc[i]
        else: # уменьшился
            if tPos.iloc[i]*tPos.iloc[i-1]<0: # перевернулся
                hp.loc[tPos.index[i],['dT','w']]=(tDiff[i]-tEntry,abs
                                                  (tPos.iloc[i-1]))
                tEntry=tDiff[i] # сбросить время входа
            else:
                hp.loc[tPos.index[i],['dT','w']]=(tDiff[i]-tEntry,abs
                                                  (pDiff.iloc[i]))
    if hp['w'].sum()>0: hp=(hp['dT']*hp['w']).sum()/hp['w'].sum()
    else: hp=np.nan
    return hp
```

14.4. Результативность

Статистические показатели результативности — это цифры в долларах и финансовых возвратах без поправки на риск. Несколько полезных мер результативности включают:

- **Прибыль и убыток** (profit and loss, PnL) — общая сумма долларов (или эквивалент в валюте номинала), сгенерированная за весь бэктест, включая ликвидационные издержки из конечной позиции.
- **Прибыль и убыток от длинных позиций** (PnL from long positions) — часть суммы PnL в долларах, которая была сгенерирована исключительно длинными позициями. Это интересное значение для оценки смещения длинно-коротких рыночных нейтральных стратегий.
- **Среднегодовая возвратность** (annualized rate of return) — средневзвешенный по времени годовой уровень совокупного финансового возврата, включая дивиденды, купоны, затраты и т. д.
- **Соотношение попаданий** (hit ratio) — доля ставок, которые привели к положительной сумме PnL.
- **Средний возврат от попаданий** (average return from hits) — средний финансовый возврат от ставок, которые генерировали прибыль.
- **Средний возврат от промахов** (average return from misses) — средний финансовый возврат от ставок, которые генерировали убыток.

14.4.1. Взвешенная по времени возвратность

Совокупный финансовый возврат — это возвратность реализованных и нереализованных прибылей и убытков, включая начисленные проценты, выплаченные купоны и дивиденды за период измерения. Правила GIPS рассчитывают взвешенную по времени возвратность (time-weighted rate of returns, TWRR), скорректированную на внешние денежные потоки (CFA Institute [2010]). Периодические и подпериодические финансовые возвраты геометрически связаны. Для периодов, начинающихся 1 января 2005 года или после этой даты, правила GIPS предписывают рассчитывать портфельные финансовые возвраты с учетом среднесуточных внешних денежных потоков.

Мы можем вычислить возвратность TWRR, определив стоимость портфеля на момент каждого внешнего денежного потока[1]. Возвратность TWRR для портфеля i между подпериодами $[t-1, t]$ обозначается $r_{i,t}$ с уравнениями

$$r_{i,t} = \frac{\pi_{i,t}}{K_{i,t}};$$

$$\pi_{i,t} = \sum_{j=1}^{J}[(\Delta P_{j,t} + A_{j,t})\theta_{i,j,t-1} + \Delta\theta_{i,j,t}(P_{j,t} - \bar{P}_{j,t-1})];$$

$$K_{i,t} = \sum_{j=1}^{J}\tilde{P}_{j,t-1}\theta_{i,j,t-1} + \max\left\{0, \sum_{j=1}^{J}\bar{\bar{P}}_{j,t}\Delta\theta_{i,j,t}\right\},$$

где

- $\pi_{i,t}$ — пересчитанная по текущим рыночным ценам (mark-to-market, MtM) прибыль или убыток для портфеля i в момент времени t;
- $K_{i,t}$ — рыночная стоимость активов, управляемых портфелем i в подпериоде t. Включение члена max{.} предназначено для финансирования дополнительных закупок (наращивания);
- $A_{j,t}$ — начисленные проценты или дивиденды, выплаченные одной единицей инструмента j в момент времени t;
- $P_{j,t}$ — чистая цена ценной бумаги j в момент времени t;
- $\theta_{i,j,t}$ — владения (то есть содержимое) портфеля i по ценной бумаге j в момент времени;
- $\tilde{P}_{j,t}$ — это грязная стоимость ценной бумаги j в момент t;
- $\bar{P}_{j,t}$ — среднетранзакционная чистая цена портфеля i по ценной бумаге j за подпериод t;

[1] Внешние денежные потоки — это активы (денежные средства или инвестиции), которые входят в портфель или выходят из него. Например, выплаты по доходам в виде дивидендов и процентов не считаются внешними денежными потоками.

- $\bar{\bar{P}}_{j,t}$ — среднетранзакционная грязная цена портфеля i по ценной бумаге j за подпериод t.

Считается, что приток денежных средств происходит в начале дня, а отток денежных средств — в конце дня. Эти подпериодические финансовые возвраты связаны геометрически:

$$\varphi_{i,T} = \prod_{t=1}^{T}(1 + r_{i,t}).$$

Переменную $\varphi_{i,T}$ можно понимать как результативность одного доллара, вложенного в портфель i за весь его срок существования, $t = 1, ..., T$. Наконец, среднегодовая возвратность портфеля i равна

$$R_i = (\varphi_{i,T})^{-y_i} - 1,$$

где y_i — это количество лет, прошедших между $r_{i,1}$ и $r_{i,T}$.

14.5. Интервалы

Инвестиционные стратегии редко приносят финансовые возвраты из одинаково распределенного взаимно независимого случайного процесса. При отсутствии этого свойства ряды стратегии с финансовыми возвратами демонстрируют частые интервалы, или отрезки. Интервалы (runs) — это непрерывные последовательности финансовых возвратов с одинаковым знаком. Следовательно, интервалы увеличивают риск понесения убытков, который необходимо оценивать с помощью надлежащих метрических показателей.

14.5.1. Концентрация финансовых возвратов

С учетом временного ряда финансовых возвратов из ставок $\{r_t\}_{t=1,...,T}$ мы можем высчитать два весовых ряда w^- и w^+:

$$r^+ = \{r_t \,|\, r_t \geq 0\}_{t=1,...,T};$$
$$r^- = \{r_t \,|\, r_t < 0\}_{t=1,...,T};$$

$$w^+ = \left\{ r_t^+ \left(\sum_t r_t^+ \right)^{-1} \right\}_{t=1,...,T};$$

$$w^- = \left\{ r_t^- \left(\sum_t r_t^- \right)^{-1} \right\}_{t=1,...,T}.$$

Руководствуясь индексом Херфиндаля–Хиршмана (HHI) для $\|w^+\| > 1$, где $\|.\|$ – это размер вектора, мы определим концентрацию положительных финансовых возвратов как

$$h^+ = \frac{\sum_t t(w_t^+)^2 - \|w^+\|^{-1}}{1 - \|w^+\|^{-1}} = \left(\frac{\mathrm{E}[(r_t^+)^2]}{\mathrm{E}[r_t^+)^2]} - 1\right)(\|r^+\| - 1)^{-1}$$

и эквивалент для концентрации отрицательных финансовых возвратов, для $\|w^-\| > 1$ как

$$h^- = \frac{\sum_t t(w_t^-)^2 - \|w^-\|^{-1}}{1 - \|w^-\|^{-1}} = \left(\frac{\mathrm{E}[(r_t^-)^2]}{\mathrm{E}[r_t^-)^2]} - 1\right)(\|r^-\| - 1)^{-1}.$$

Из неравенства Йенсена мы знаем, что $\mathrm{E}[r_1^+]^2 \le \mathrm{E}[(r_1^+)^2]$. И поскольку $\frac{\mathrm{E}[r_1^+]^2}{\mathrm{E}[(r_1^+)^2]} \le$ $\le \|r^+\|$, мы делаем вывод, что $\mathrm{E}[r_t^+]^2 \le \mathrm{E}[(r_t^+)]^2 \le \mathrm{E}[r_t^+]^2 \|r^+\|$ с эквивалентной границей на отрицательных финансовых возвратах из ставок. Эти определения имеют несколько интересных свойств:

1) $0 \le h^+ \le 1$;

2) $h^+ = 0 \Leftrightarrow w_t^+ = \|w^+\|^{-1}, \forall t$ (равномерные финансовые возвраты);

3) $h^+ = 1 \Leftrightarrow \exists i | w_t^+ = \sum_t w_t^+$ (только один ненулевой финансовый возврат).

Легко получить аналогичное выражение для концентрации ставок по месяцам, $h[t]$. Листинг 14.3 реализует эти понятия. В идеале нас интересуют стратегии, в которых финансовые возвраты *ставок* демонстрируют:

- высокий коэффициент Шарпа;
- большое число ставок в год, $\|r^+\| + \|r^-\| = T$;
- высокое соотношение попаданий (относительно низкий $\|r^-\|$);
- низкий h^+ (без правого толстого хвоста);
- низкий h^- (без левого толстого хвоста);
- низкий $h[t]$ (ставки не концентрированы во времени).

Листинг 14.3. Алгоритм получения концентрации HHI

```
rHHIPos=getHHI(ret[ret>=0]) # концентрация положительных возвратов на ставку
rHHINeg=getHHI(ret[ret<0]) # концентрация отрицательных возвратов на ставку
tHHI=getHHI(ret.groupby(pd.TimeGrouper(freq='M')).count()) # концентрация
                                                           # ставки/месяц
#-------------------------------------
```

```
def getHHI(betRet):
    if betRet.shape[0]<=2: return np.nan
    wght=betRet/betRet.sum()
    hhi=(wght**2).sum()
    hhi=(hhi-betRet.shape[0]**-1)/(1.-betRet.shape[0]**-1)
    return hhi
```

14.5.2. Просадка и время нахождения ниже уровня воды

В интуитивном плане просадка (drawdown, DD) — это максимальный убыток, понесенный инвестициями между двумя отметками линии уровня воды (high-watermark, HWM) подряд. Время нахождения ниже уровня воды (time under water, TuW) — это время, прошедшее между отметкой HWM и моментом, когда стоимость PnL превышает предыдущую максимальную стоимость PnL. В этих понятиях лучше всего разобраться, прочитав листинг 14.4. Этот исходный код производит ряды DD и TuW либо 1) из ряда финансовых возвратов (`dollars = False`), либо 2) из ряда долларовой результативности (`dollar = True`). На рис. 14.1 приведен пример рядов DD и TuW.

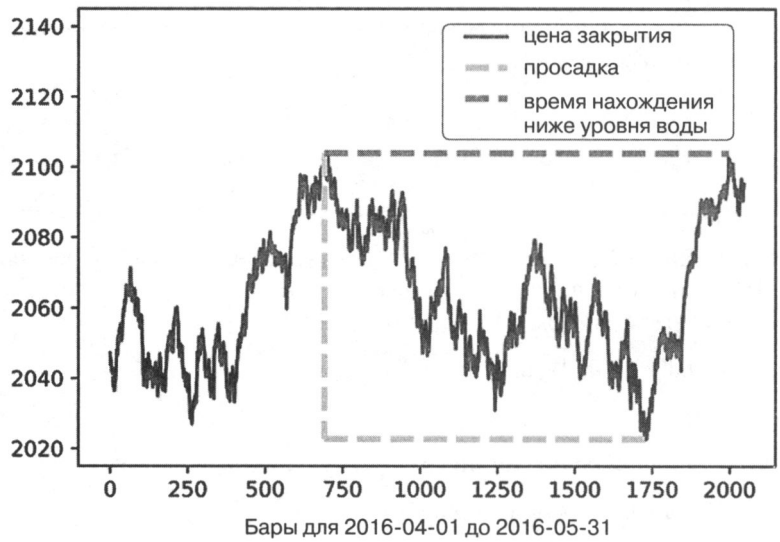


Рис. 14.1. Примеры просадок и времени нахождения ниже уровня воды (TuW)

Листинг 14.4. Выведение последовательности DD и TuW

```
def computeDD_TuW(series,dollars=False):
    # вычислить ряды просадок и связанного с ними времени ниже уровня воды
    df0=series.to_frame('pnl')
    df0['hwm']=series.expanding().max()
    df1=df0.groupby('hwm').min().reset_index()
```

```
df1.columns=['hwm','min']
df1.index=df0['hwm'].drop_duplicates(keep='first').index # время hwm
df1=df1[df1['hwm']>df1['min']] # за hwm следует просадка
if dollars: dd=df1['hwm']-df1['min']
else: dd=1-df1['min']/df1['hwm']
tuw=((df1.index[1:]-df1.index[:-1])/np.timedelta64(1,'Y')).values # в годах
tuw=pd.Series(tuw,index=df1.index[:-1])
return dd,tuw
```

14.5.3. Статистические показатели интервалов для оценивания результативности

Несколько полезных мер интервальных статистических показателей включают в себя:

- **Индекс HHI на положительных финансовых возвратах** (HHI index on positive returns): в листинге 14.3 это `getHHI(ret[ret>=0])`.
- **Индекс HHI на отрицательных финансовых возвратах** (HHI index on negative returns): в листинге 14.3 это `getHHI(ret[ret<0])`.
- **Индекс HHI на времени между ставками** (HHI index on time between bets): в листинге 14.3 это `getHHI(ret.groupby(pd.TimeGrouper(freq='M')).count())`.
- **95-процентильная DD** (95-percentile DD) — это 95-й процентиль ряда DD, полученный в листинге 14.4.
- **95-процентильное TuW** (95-percentile TuW) — это 95-й процентиль ряда TuW, полученный в листинге 14.4.

14.6. Дефицит реализации

Инвестиционные стратегии часто оказываются безуспешными из-за неправильных допущений относительно издержек исполнения. Некоторые важные связанные с ними меры включают в себя:

- **Брокерская комиссия в расчете на финансовый оборот** (broker fees per turnover) — это комиссионные, выплачиваемые брокеру за оборачиваемость портфеля, включая биржевые сборы.
- **Среднее соскальзывание в расчете на финансовый оборот** (average slippage per turnover) — это издержки исполнения, за исключением брокерских сборов, участвующих в обороте одного портфеля. Например, сюда входит убыток, вызванный покупкой ценной бумаги по цене исполнения выше средней цены на момент отправки ордера исполняющему брокеру.
- **Долларовая результативность в расчете на финансовый оборот** (dollar performance per turnover) — это соотношение между долларовой результативностью

(включая брокерские сборы и издержки соскальзывания) и общим оборотом портфеля. Она означает, насколько дороже может стать исполнение до того, как стратегия станет безубыточной.

- **Финансовый возврат на издержки исполнения** (return on execution costs) — это соотношение между долларовой результативностью (включая брокерские сборы и издержки соскальзывания) и общими издержками исполнения. Для того чтобы гарантировать, что стратегия выживет при худшем сценарии, чем ожидалось, он должен быть большим числом.

14.7. Эффективность

До сих пор все статистические показатели результативности учитывали прибыли, убытки и издержки. В этом разделе мы учитываем риски, сопряженные с достижением этих результатов.

14.7.1. Коэффициент Шарпа

Предположим, что избыточные финансовые возвраты стратегии (свыше безрисковой ставки), $\{r_t\}_{t=1,\ldots,T}$, являются одинаково распределенными взаимно независимыми гауссовыми случайными величинами со средним μ и дисперсией σ^2. Коэффициент Шарпа (Sharpe ratio, SR) определяется как

$$SR = \frac{\mu}{\sigma}.$$

Коэффициент Шарпа предназначен для оценивания навыков конкретной стратегии или инвестора. Поскольку μ, σ обычно неизвестны, истинное значение коэффициента Шарпа не может быть известно наверняка. Неизбежным следствием этого является то, что расчеты коэффициента Шарпа могут быть предметом существенных ошибок оценивания.

14.7.2. Вероятностный коэффициент Шарпа

Вероятностный коэффициент Шарпа (probabilistic SR, PSR) дает скорректированную оценку коэффициента Шарпа, устраняя инфляционный эффект, вызванный короткими рядами с асимметричными финансовыми возвратами и/или финансовыми возвратами с толстыми хвостами. При определяемом пользователем эталонном[1] коэффициенте Шарпа (SR*) и наблюдаемом коэффициенте Шарпа

[1] Эталон может быть установлен по умолчанию равным нулевому значению (то есть сравнение с отсутствием инвестиционного навыка).

($\widehat{SR}$) вероятностный коэффициент Шарпа оценивает вероятность того, что наблюдаемый $\widehat{SR}$ больше, чем гипотетический SR^*. Согласно Bailey and Lopez de Prado [2012], вероятностный коэффициент Шарпа (PSR) можно оценить как

$$\widehat{PSR}[SR^*] = Z\left[\frac{(\widehat{SR} - SR^*)\sqrt{T-1}}{\sqrt{1-\hat{\gamma}_3\widehat{SR} + \frac{\hat{\gamma}_4 - 1}{4}\widehat{SR}^2}}\right],$$

где $Z[.]$ — это кумулятивная функция распределения (CDF) стандартного нормального распределения, T — число наблюдаемых финансовых возвратов, $\hat{\gamma}_3$ — асимметрия финансовых возвратов и $\hat{\gamma}_4$ — эксцесс финансовых возвратов ($\hat{\gamma}_4 = 3$ для гауссовых финансовых возвратов). Для заданного коэффициента SR^* вероятностный коэффициент $\widehat{PSR}$ повышается вместе с бóльшим коэффициентом Шарпа $\widehat{SR}$ (в исходной частоте отбора образцов, то есть не в годовом выражении), либо с более продолжительной предысторией (T), либо положительно асимметричными финансовыми возвратами ($\hat{\gamma}_3$), но он уменьшается вместе с более толстыми хвостами ($\hat{\gamma}_4$). На рис. 14.2 построен график вероятностного коэффициента Шарпа $\widehat{PSR}$ для $\hat{\gamma}_4 = 3$, $\widehat{SR} = 1.5$ и $SR^* = 1.0$ как функции от $\hat{\gamma}_3$ и T.

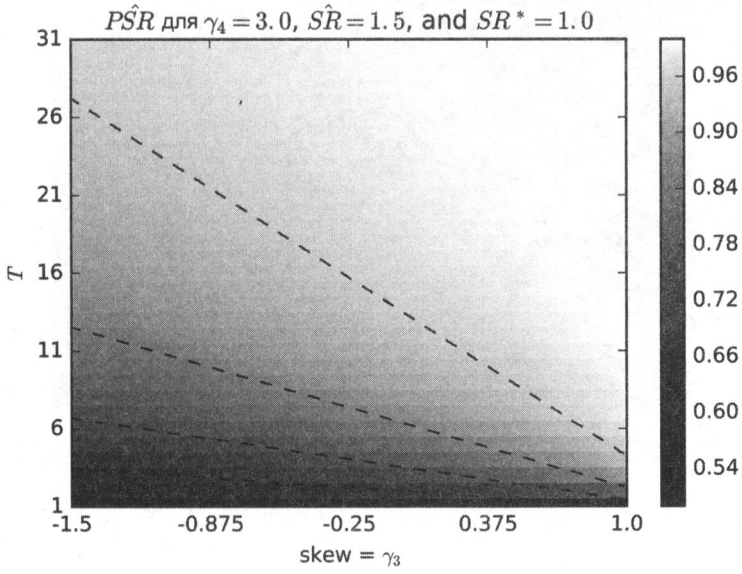


Рис. 14.2. Вероятностный коэффициент Шарпа как функция от асимметрии и длины выборки

14.7.3. Дефлированный коэффициент Шарпа

Дефлированный коэффициент Шарпа (deflated SR, DSR) — это вероятностный коэффициент Шарпа (PSR), где порог отклонения откорректирован для того, чтобы отражать множественность испытаний. Согласно Bailey and Lopez de Prado [2014], дефлированный коэффициент Шарпа можно оценить как $\widehat{PSR}$ [SR^*], где эталонный коэффициент Шарпа, SR^*, больше не определяется пользователем. Вместо этого SR^* оценивается как

$$SR^* = \sqrt{V[\{\widehat{SR}_n\}]}\left((1-\gamma)Z^{-1}\left[1-\frac{1}{N}\right]+\gamma Z^{-1}\left[1-\frac{1}{N}e^{-1}\right]\right),$$

где $V[\{\widehat{SR}_n\}]$ — это дисперсия по всем оцененными коэффициентом Шарпа испытаниям, N — число независимых испытаний, $Z[.]$ — кумулятивная функция распределения (CDF) стандартного нормального распределения, γ — постоянная Эйлера—Маскерони, а $n = 1, ..., N$. На рис. 14.3 построен график SR^* как функции от $V[\{SR_n\}]$ и N.

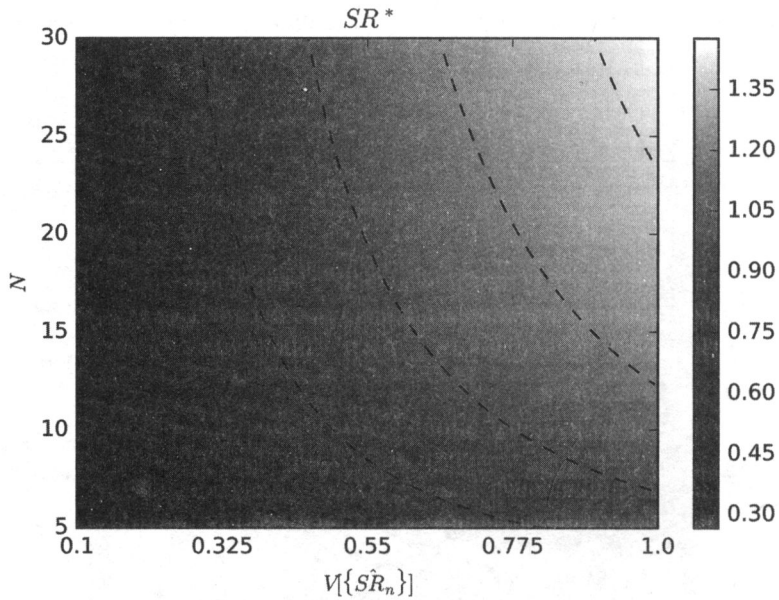


Рис. 14.3. SR^* как функция от $V[\{\widehat{SR}_n\}]$ и N

Мотивировка пересчитанного коэффициента Шарпа заключается в следующем: при наличии набора оценок коэффициента Шарпа $\{\widehat{SR}_n\}$ его ожидаемое максимальное значение выше нуля, даже если верный коэффициент Шарпа равен нулю. Исходя из нулевой гипотезы о том, что действующий коэффициент Шарпа равен

нулю, $H_0 : SR = 0$, мы знаем, что ожидаемое максимальное значение наблюдаемого коэффициента ($\widehat{SR}$) может быть оценено как исходный коэффициент (SR^*). Действительно, исходный коэффициент Шарпа растет с большой скоростью по мере проведения независимых испытаний N или вследствие того, что испытания обладают более высокой дисперсией ($V[\{\widehat{SR}_n\}]$). Исходя из этого мы можем вывести третий закон бэктестирования.

ЛИСТИНГ 14.5. ТРЕТИЙ ЗАКОН БЭКТЕСТИРОВАНИЯ МАРКОСА. БОЛЬШИНСТВО ОТКРЫТИЙ В ФИНАНСАХ ЯВЛЯЮТСЯ ЛОЖНЫМИ ИЗ-ЗА ЕГО НАРУШЕНИЯ

«О каждом результате бэктеста следует сообщать в комплексе со всеми испытаниями, участвующими в его производстве. Без этой информации невозможно оценить вероятность "ложного обнаружения", полученного в результате бэктеста».

— Маркос Лопез де Прадо
Машинное обучение: алгоритмы для бизнеса (2018)

14.7.4. Статистические показатели эффективности

Полезные статистические показатели эффективности включают:

- **Среднегодовой коэффициент Шарпа** (annualized Sharpe ratio) — это значение коэффициента Шарпа, в пересчете в годовом исчислении с коэффициентом $\sqrt{a}$, где a — это среднее число финансовых возвратов, наблюдавшихся за год. Данный распространенный метод пересчета в годовое исчисление основан на допущении, что финансовые возвраты одинаково распределены и взаимно независимы.
- **Информационное соотношение** (information ratio): это эквивалент портфельного коэффициента Шарпа, который измеряет результативность портфеля относительно эталонного показателя. Он представляет собой среднегодовой коэффициент между средним избыточным финансовым возвратом и ошибкой отслеживания. Избыточный финансовый возврат измеряется как портфельный финансовый возврат, превышающий эталонный финансовый возврат. Ошибка отслеживания оценивается как среднеквадратическое отклонение избыточных финансовых возвратов.
- **Вероятностный коэффициент Шарпа** (probabilistic Sharpe ratio) — исправляет инфляционные эффекты коэффициента Шарпа, вызванные ненормальными финансовыми возвратами или длиной предыстории. Для стандартного уровня значимости 5 % он должен превышать 0.95. Он может быть вычислен на абсолютных или относительных финансовых возвратах.
- **Дефлированный коэффициент Шарпа** (deflated Sharpe ratio) — исправляет инфляционные эффекты коэффициента Шарпа, вызванные ненормальными

финансовыми возвратами, длиной предыстории и множественным тестированием/систематическим смещением при отборе. Для стандартного уровня значимости 5 % он должен превышать 0.95. Он может быть вычислен на абсолютных или относительных финансовых возвратах.

14.8. Классификационные балльные оценки

В контексте метамаркировочных стратегий (глава 3, раздел 3.6) полезно понимать результативность оверлейного алгоритма изолированно. Напомним, что первичный алгоритм идентифицирует возможности, а вторичный (оверлейный, наложенный) алгоритм решает, использовать их или пропустить. Несколько полезных статистических показателей включают:

- **Правильность** (accuracy) — это доля возможностей, правильно промаркированных оверлейным алгоритмом.

$$\text{правильность} = \frac{TP + TN}{TP + TN + FP + FN},$$

где TP — число истинных утверждений, TN — число истинных отрицаний, а FP — число ложных утверждений и FN — число ложных отрицаний.

- **Точность** (precision) — это доля истинных утверждений среди предсказанных утверждений,

$$\text{точность} = \frac{TP}{TP + FP}.$$

- **Полнота** (recall) — это доля истинных утверждений среди утверждений,

$$\text{полнота} = \frac{TP}{TP + FN}.$$

- **Оценка F1** (F1-score) — для метамаркировочных приложений точность может не быть адекватной классификационной балльной оценкой. Предположим, что после применения метамаркировки существует гораздо больше отрицательных случаев (метка '0'), чем утвердительных (метка '1'). В рамках этого сценария классификатор, который предсказывает, что каждый случай будет отрицательным, достигнет высокой точности, даже если полнота = 0 и точность не определена. Оценка F1 исправляет этот недостаток, оценивая классификатор с точки зрения (одинаково взвешенного) гармонического среднего точности и отзыва,

$$F_1 = 2\frac{\text{точность} \times \text{полнота}}{\text{точность} + \text{полнота}}.$$

В качестве ремарки рассмотрим необычный сценарий, где после применения метамаркировки имеется намного больше утвердительных случаев, чем отрицательных. Классификатор, который предсказывает, что все случаи будут утвердительными, достигнет TN = 0 и FN = 0, следовательно, правильность = точность и полнота = 1. Правильность будет высокой, и оценка F1 не будет меньше правильности, даже если классификатор не способен проводить различие между наблюдаемыми образцами. Одним из решений было бы поменять местами определения утвердительных и отрицательных случаев, с тем чтобы преобладали отрицательные случаи, а затем оценить с помощью меры F1.

- **Отрицательная логарифмическая потеря** (negative log-loss) была представлена в главе 9, раздел 9.4, в контексте регулировки гиперпараметров. Пожалуйста, обратитесь к этому разделу для получения дополнительной информации. Ключевое концептуальное различие между точностью и отрицательной логарифмической потерей заключается в том, что отрицательная логарифмическая потеря учитывает не только правильность наших прогнозов, но и вероятность этих прогнозов.

Смотрите визуальное представление точности, отзыва и точности в главе 3, раздел 3.7. Таблица 14.1 характеризует четыре вырожденных случая бинарной классификации. Как вы можете видеть, оценка F1 не определена в двух из этих случаев. По этой причине, когда библиотеке scikit-learn поручается вычислить F1 на выборке без наблюдаемых единиц или без предсказанных единиц, она напечатает предупреждение (`UndefinedMetricWarning`) и установит значение F1 равным 0.

Таблица 14.1. Четыре вырожденных случая бинарной классификации

Условие	Коллапс	Правильность	Точность	Полнота	F1-мера
Наблюдаются все 1	TN=FP=0	= полнота	1	[0,1]	[0,1]
Наблюдаются все 0	TP=FN=0	[0,1]	0	NaN	NaN
Прогнозируются все 1	TN=FN=0	= точность	[0,1]	1	[0,1]
Прогнозируются все 0	TP=FP=0	[0,1]	NaN	0	NaN

Когда все наблюдаемые значения утвердительны (метка '1'), нет истинных отрицаний или ложных утверждений, поэтому точность равна 1, отзыв является положительным вещественным числом между 0 и 1 (включительно), правильность равна отзыву. Тогда $F_1 = 2\frac{\text{полнота}}{1+\text{полнота}} \geq \text{полнота}$.

Когда все предсказанные значения утвердительны (метка '1'), нет истинных отрицаний или ложных отрицаний, поэтому точность является положительным

вещественным числом от 0 до 1 (включительно), отзыв равен 1, а правильность равна точности. Тогда $F_1 = 2\frac{\text{точность}}{1+\text{точность}} \geq \text{точность}$.

14.9. Атрибутирование

Атрибутирование результативности предназначено для декомпозиции стоимости PnL с точки зрения классов риска. Например, менеджер портфеля корпоративных облигаций, как правило, хочет понять, на сколько его результативность зависит от его подверженности следующим классам рисков: риску срочности, кредитному риску, риску ликвидности, риску экономического сектора, валютному риску, суверенному риску, эмитентному риску и т. д. Окупились ли его ставки на срочность? В каких кредитных сегментах он преуспевает? Или он должен сосредоточиться на своих навыках выбора эмитента?

Эти риски не ортогональны, поэтому между ними всегда есть наложение. Например, высоколиквидные облигации, как правило, имеют короткую срочность и высокий кредитный рейтинг и, как правило, выпускаются крупными компаниями с большими суммами задолженности в долларах США. В результате сумма атрибутированных (приписываемых) стоимостей PnL не будет соответствовать общему PnL, но по крайней мере мы сможем вычислить коэффициент Шарпа (или информационное соотношение) на класс риска. Пожалуй, самым популярным примером такого подхода является многофакторный метод Барра. См. публикации Barra [1998, 2013] и Zhang and Rachev [2004] для получения более подробной информации.

Равный интерес представляет атрибутирование стоимости PnL по категориям в каждом классе. Например, класс срочности может быть подразделен между краткосрочным (менее 5 лет), среднесрочным (от 5 до 10 лет) и долгосрочным (свыше 10 лет). Эта атрибуция стоимости PnL может быть выполнена следующим образом: во-первых, чтобы избежать проблемы наложения, о которой мы говорили ранее, нам нужно убедиться, что каждый член инвестиционного универсума принадлежит к одной и только одной категории каждого класса риска в любой момент времени. Другими словами, для каждого класса риска мы разделяем весь инвестиционный универсум на ненакладывающиеся подразделы. Во-вторых, для каждого класса риска мы формируем один индекс на категорию риска. Например, мы вычислим результативность индекса краткосрочных облигаций, еще один индекс среднесрочных облигаций и еще один индекс долгосрочных облигаций. Весовые коэффициенты для каждого индекса представляют собой решкалированные веса нашего инвестиционного портфеля, так что вес каждого индекса в сумме составляет единицу. В-третьих, мы повторяем второй шаг, но на этот раз мы формируем эти индексы категории риска, используя веса из инвестиционного универсума (например, Markit iBoxx Investment Grade), снова решкалированные так, чтобы веса каждого индекса в сумме составляли единицу. В-четвертых, мы

вычисляем показатели результативности, которые мы обсуждали ранее в главе, по каждому из этих индексов и избыточных финансовых возвратов. Для ясности, в данном контексте избыточный финансовый возврат краткосрочного индекса — это возврат с использованием (решкалированных) портфельных перевесовок (шаг 2) за вычетом возврата с использованием (решкалированных) перевесовок универсума (шаг 3).

Упражнения

14.1. Стратегия демонстрирует высокий оборот, высокое кредитное плечо и большое число ставок, с коротким периодом владения, низким финансовым возвратом на издержки исполнения и высоким коэффициентом Шарпа. Вероятно ли, что она имеет большую емкость? Какого рода стратегией, по вашему мнению, она является?

14.2. На совокупности данных с долларовыми барами для фьючерсного контракта E-mini S&P 500 вычислите:

(а) Индекс HHI на положительных финансовых возвратах.

(б) Индекс HHI на отрицательных финансовых возвратах.

(в) Индекс HHI на времени между барами.

(г) 95-процентильную просадку (DD).

(д) 95-процентильное время нахождения ниже уровня воды (TuW).

(е) Среднегодовой финансовый возврат.

(ж) Средние финансовые возвраты от попаданий (положительные финансовые возвраты).

(з) Средние финансовые возвраты от промахов (отрицательные финансовые возвраты).

(и) Среднегодовой коэффициент Шарпа.

(к) Информационное соотношение, где эталоном является безрисковая ставка.

(л) Вероятностный коэффициент Шарпа.

(м) Дефлированный коэффициент Шарпа, где мы исходим из того, что было 100 испытаний и дисперсия коэффициента Шарпа в испытаниях равнялась 0.5.

14.3. Рассмотрите стратегию, которая является продолжительным фьючерсным контрактом на четные годы и коротким фьючерсным контрактом на нечетные годы.

(а) Повторите расчеты из упражнения 14.2.

(б) Какова корреляция с базовым универсумом?

14.4. Результаты двухлетнего бэктеста показывают, что месячные финансовые возвраты имеют среднее значение 3.6 % и среднеквадратическое отклонение 0.079 %.

(а) Каков коэффициент Шарпа?

(б) Какой среднегодовой коэффициент Шарпа?

14.5. В продолжение упражнения 14.1:

(а) Финансовые возвраты имеют асимметрию, равную 0, и эксцесс, равный 3. Каков вероятностный коэффициент Шарпа?

(б) Финансовые возвраты имеют асимметрию, равную −2.448, и эксцесс, равный 10.164. Каков вероятностный коэффициент Шарпа?

14.6. Каким будет вероятностный коэффициент Шарпа из 14.2.6 при условии, что бэктест продолжался 3 года?

14.7. Пятилетний бэктест имеет среднегодовой коэффициент Шарпа, равный 2.5, вычисленный на среднесуточных финансовых возвратах. Асимметрия составляет −3 и эксцесс 10.

(а) Каков вероятностный коэффициент Шарпа?

(б) С целью нахождения лучшего результата было проведено 100 испытаний. Дисперсия коэффициентов Шарпа на этих испытаниях составляет 0.5. Каков дефлированный коэффициент Шарпа?

15 Понимание риска стратегии

15.1. Актуальность

Как мы видели в главах 3 и 13, инвестиционные стратегии часто реализуются с точки зрения позиций, которыми владеют до тех пор, пока не будет выполнено одно из двух условий: 1) условие выхода из позиции с прибылями (взятие прибыли) или 2) условие выхода из позиции с убытками (остановка убытка). Даже когда стратегия явно не объявляет остановку убытка, всегда существует неявный предел остановки убытка, при котором инвестор больше не может финансировать свою позицию (маржин колл) или несет ущерб, вызванный увеличением нереализованного убытка. Поскольку большинство стратегий имеют (явно или неявно) эти два условия выхода, имеет смысл моделировать распределение исходов посредством биномиального процесса. Это, в свою очередь, поможет нам понять, какие сочетания частоты ставок, рисков и выплат являются неэкономичными. Цель этой главы — помочь вам оценить, когда стратегия уязвима к небольшим изменениям в любой из этих величин.

15.2. Симметричные выплаты

Рассмотрим стратегию, которая производит n одинаково распределенных взаимно независимых ставок в год, где исход X_i ставки $i \in [1, n]$ представляет собой прибыль $\pi > 0$ с вероятностью $P[X_i = \pi] = p$ и убыток $-\pi$ с вероятностью $P[X_i = -\pi] = 1 - p$. Вы можете представить p как точность бинарного классификатора, в котором утвердительный исход означает заключение ставки на возможность, а отрицательный исход означает пропуск возможности: истинные утверждения вознаграждаются, ложные утверждения наказываются, и отрицательные исходы (будь то истинные или ложные) выплат не имеют. Поскольку исходы ставок $\{X_i\}_{i=1,\ldots,n}$ независимы, мы будем вычислять ожидаемые моменты в расчете на ставку. Ожидаемая прибыль от одной ставки составляет $E[X_i] = \pi p + (-\pi)(1 - p) = \pi(2p - 1)$. Дисперсия составляет $V[X_i] = E[X_i^2] - E[X_i^2]$, где $E[X_i^2] = \pi^2 p + (-\pi)^2(1 - p) = \pi^2$, следовательно, $V[X_i] = \pi^2 - \pi^2 (2p - 1)^2 = \pi^2[1-(2p - 1)^2] = 4\pi^2 p(1 - p)$. Для n одинаково распределенных взаимно независимых ставок в год среднегодовой коэффициент Шарпа (θ) равен

$$\theta[p,n]=\frac{n\mathrm{E}[X_i]}{\sqrt{nV[X_i]}}=\underbrace{\frac{2p-1}{2\sqrt{p(1-p)}}}_{\substack{t\text{-значение } p \\ \text{относительно } H_0:\ p=\frac{1}{2}}}\sqrt{n}.$$

Обратите внимание, как π уравновешивает приведенное выше уравнение, потому что выплаты симметричны. Так же как и в гауссовом случае, $\theta[p,n]$ можно понимать как решкалированное t-значение[1]. Этим иллюстрируется тот факт, что даже для малого $p>\frac{1}{2}$ коэффициент Шарпа может быть сделан высоким для достаточно большого n.

Это служит экономической основой для высокочастотного трейдинга, где p может быть чуть выше .5, а залогом успешной биржевой деятельности является увеличение n. Коэффициент Шарпа является функцией от точности, а не от правильности, потому что пропуск возможности (отрицательное утверждение) не вознаграждается или наказывается напрямую (хотя слишком много отрицательных утверждений может привести к малому n, что будет сводить коэффициент Шарпа к нулю).

Например, для $p=.55$, $\frac{2p-1}{2\sqrt{p(1-p)}}=0.1005$ и для достижения среднегодового коэффициента Шарпа, равного 2, требуется 396 ставок в год. Листинг 15.1 проверяет этот результат экспериментально. Рисунок 15.1 показывает коэффициент Шарпа как функцию от точности для разных частот ставок.

Листинг 15.1. Коэффициент Шарпа как функция от числа ставок

```
out,p=[],.55
for i in xrange(1000000):
    rnd=np.random.binomial(n=1,p=p)
    x=(1 if rnd==1 else -1)
    out.append(x)
print np.mean(out),np.std(out),np.mean(out)/np.std(out)
```

Решив для $0\le p\le 1$, мы получим $-4p^2+4p-\frac{n}{\theta^2+n}=0$ с решением

$$p=\frac{1}{2}\left(1+\sqrt{1-\frac{n}{\theta^2+n}}\right).$$

Это уравнение совершенно ясно выражает компромисс между точностью (p) и частотой (n) для заданного коэффициента Шарпа (θ). Например, для того чтобы давать среднегодовой коэффициент Шарпа, равный 2, стратегии, которая производит только еженедельные ставки ($n=52$), потребуется довольно высокая точность $p=0.6336$.

---

[1] t-значение (t-value) — стандартизованная версия проверочной статистики, используемой в качестве критерия при проверке статистической гипотезы. — *Примеч. науч. ред.*

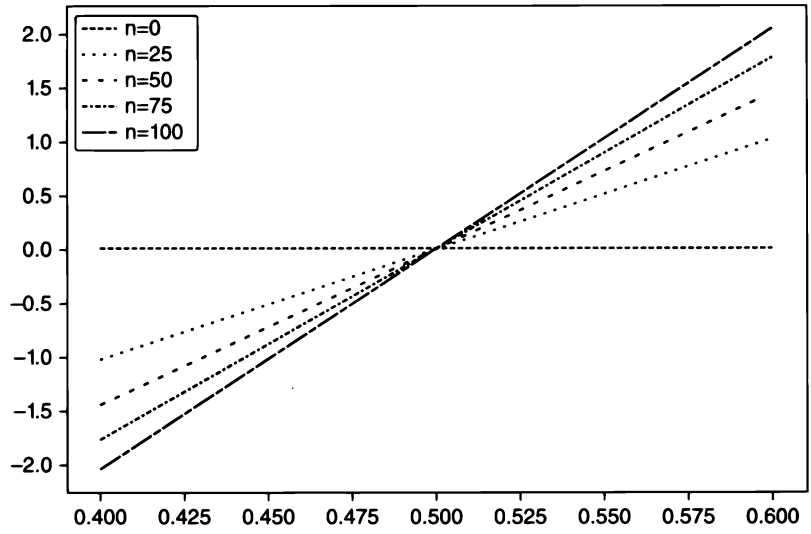


Рис. 15.1. Связь между точностью (ось x) и коэффициентом Шарпа (ось y) для разных частот заключения ставок (n)

15.3. Асимметричные выплаты

Рассмотрим стратегию, которая производит n одинаково распределенных взаимно независимых ставок в год, где исход X_i ставки $i \in [1, n]$ равен π_+ с вероятностью $P[X_i = \pi_+] = p$, а исход π_- ($\pi_- < \pi_+$) случается с вероятностью $P[X_i = \pi\_] = 1 - p$. Ожидаемая прибыль от одной ставки составляет $E[X_i] = p\pi_+ + (1-p)\pi_- = (\pi^+ - \pi_-)p + \pi_-$. Дисперсия составляет $V[X_i] = E[X_i^2] - E[X_i]^2$, где

$$E[X_i^2] = p\pi_+^2 + (1-p)\pi_-^2 = (\pi_+^2 - \pi_-^2)p + \pi_-^2,$$

следовательно, $V[X_i] = (\pi_+ - \pi_-)^2 p(1-p)$. Для n одинаково распределенных взаимно независимых ставок в год годовой коэффициент Шарпа (θ) равен

$$\theta[p, n, \pi_-, \pi_+] = \frac{nE[X_i]}{\sqrt{nV[X_i]}} = \frac{(\pi_+ - \pi_-)p + \pi_-}{(\pi_+ - \pi_-)\sqrt{p(1-p)}}\sqrt{n}.$$

И для $\pi_- = -\pi_+$ мы видим, что это уравнение сводится к симметричному случаю: $\theta[p, n, -\pi_+, \pi_+] = \frac{2\pi_+ p + \pi_+}{2\pi_+\sqrt{p(1-p)}}\sqrt{n} = \frac{2p-1}{2\sqrt{p(1-p)}}\sqrt{n} = \theta[p, n]$. Например, для $n = 260$, $\pi_- = -.01$, $\pi_+ = .005$, $p = .7$, мы получим $\theta = 1.173$.

Наконец, мы можем решить предыдущее уравнение для $0 \le p \le 1$ и получить

$$p = \frac{-b + \sqrt{b^2 - 4ac}}{2a}.$$

где:

- $a = (n + \theta^2)(\pi_+ - \pi_-)^2$;
- $b = [2n\pi - \theta^2(\pi_+ - \pi_-)](\pi_+ - \pi_-)$;
- $c = n\pi_-^2$.

Примечание: листинг 15.2 проверяет эти символические операции с помощью Python-овской оболочки SymPy Live, работающей на облачной службе Google App Engine: http://live.sympy.org/.

Листинг 15.2. Использование библиотеки SymPy для символических операций

```
>>> from sympy import *
>>> init_printing(use_unicode=False,wrap_line=False,no_global=True)
>>> p,u,d=symbols('p u d')
>>> m2=p*u**2+(1-p)*d**2
>>> m1=p*u+(1-p)*d
>>> v=m2-m1**2
>>> factor(v)
```

Приведенное выше уравнение отвечает на следующий вопрос: при заданном торговом правиле, характеризуемом параметрами $\{\pi_-, \pi_+, n\}$, какова степень точности p, необходимая для достижения коэффициента Шарпа, равного θ^*? Например, для того чтобы получить $\theta = 2$ для $n = 260$, $\pi_- = -.01$, $\pi_+ = .005$, нам потребуется $p = .72$. Благодаря большому числу ставок очень малое изменение в p (с $p = .7$ до $p = .72$) продвинуло коэффициент Шарпа с $\theta = 1.173$ до $\theta = 2$. С другой стороны, это также говорит нам о том, что данная стратегия уязвима для малых изменений в p. Листинг 15.3 реализует выведение предполагаемой точности. На рис. 15.2 показана предполагаемая точность как функция от n и π_-, где $\pi_+ = 0.1$, а $\theta^* = 1.5$. По мере того как для заданного n порог π_- становится отрицательнее, требуется более высокая степень p, необходимая для достижения θ^* для заданного порога π_+. По мере того как для заданного порога π_- число n становится меньше, требуется более высокая степень p, необходимая для достижения θ^* для заданного π_+.

Листинг 15.3. Вычисление предполагаемой точности

```
def binHR(sl,pt,freq,tSR):
    '''
    При заданном торговом правиле, характеризующемся параметрами {sl,pt,freq},
    какова минимальная точность, требуемая для достижения
    коэффициента Шарпа, равного tSR?
    1) Входы
    sl: порог остановки убытка
    pt: порог взятия прибыли
```

```
    freq: число ставок в год
    tSR: целевой среднегодовой коэффициент Шарпа
    2) Выход
    p: минимальная степень точности p, требуемая для достижения tSR
    '''
    a=(freq+tSR**2)*(pt-sl)**2
    b=(2*freq*sl-tSR**2*(pt-sl))*(pt-sl)
    c=freq*sl**2
    p=(-b+(b**2-4*a*c)**.5)/(2.*a)
    return p
```

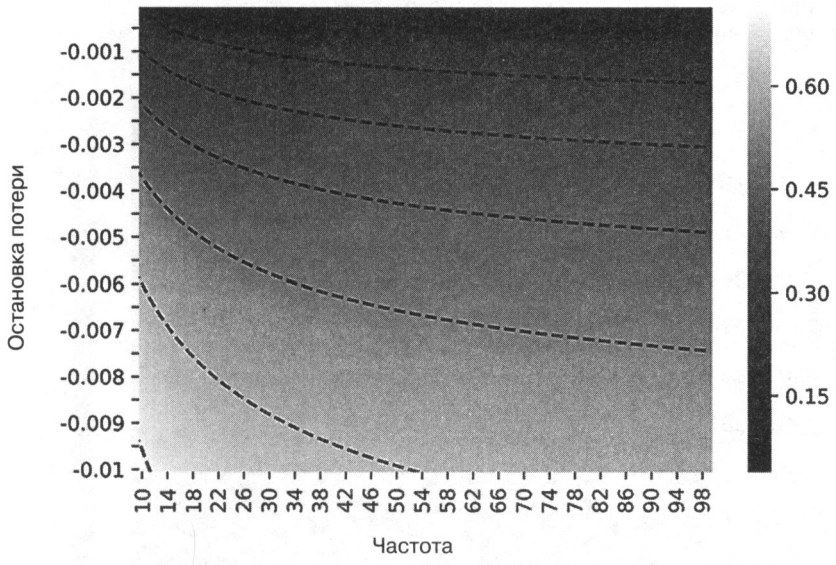


Рис. 15.2. Тепловая карта предполагаемой точности от функции n и π_-, где $\pi_+ = 0.1$, а $\theta^* = 1.5$

Листинг 15.4 решает $\theta[p, n, \pi_-, \pi_+]$ для предполагаемой частоты заключения ставок n. На рис. 15.3 показана предполагаемая частота в зависимости от p и π_-, где $\pi_+ = 0.1$, а $\theta^* = 1.5$. По мере того как для заданной степени p порог π_- становится отрицательнее, требуется более высокое число n, необходимое для достижения θ^* для данного порога π_+. По мере того как для заданного порога π_- степень p становится меньше, требуется более высокое число n, необходимое для достижения θ^* для заданного порога π_+.

Листинг 15.4. Вычисление предполагаемой частоты заключения ставок

```
def binFreq(sl,pt,p,tSR):
    '''
    При заданном торговом правиле, характеризующемся параметрами {sl, pt,
    freq}, какое число ставок в год необходимо для достижения коэффициента
    Шарпа tSR со степенью точности p?
```

```
Примечание: уравнение с радикалами, проверьте наличие постороннего
решения.
1) Входы
sl: порог остановки убытка
pt: порог взятия прибыли
p: степень точности p
tSR: целевой среднегодовой коэффициент Шарпа
2) Выход
freq: число необходимых ставок в год
'''
freq=(tSR*(pt-sl))**2*p*(1-p)/((pt-sl)*p+sl)**2 # возможно постороннее
if not np.isclose(binSR(sl,pt,freq,p),tSR): return
return freq
```

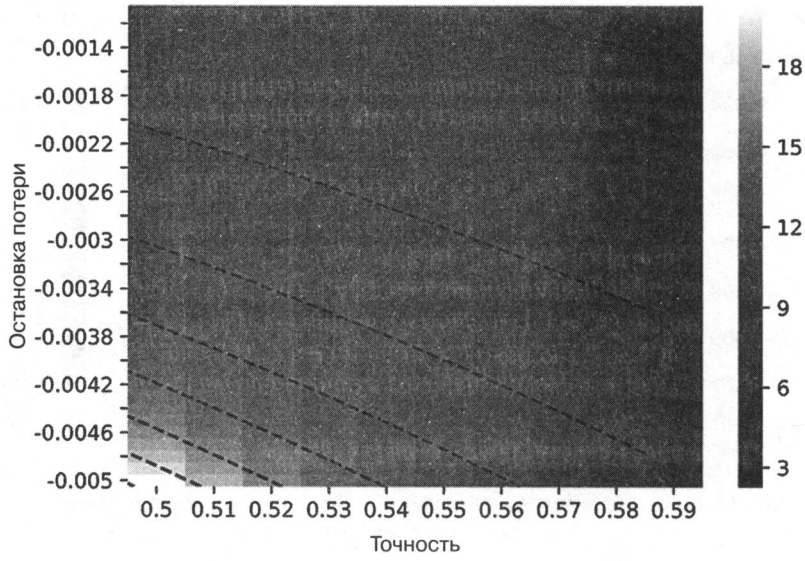


Рис. 15.3. Предполагаемая частота как функция от p и $c = 0.1$ и $= 1.5$

15.4. Вероятность неуспешности стратегии

В приведенном выше примере параметры $\pi_- = -.01$, $\pi_+ = .005$ устанавливаются портфельным менеджером и передаются трейдерам с исполнительными ордерами. Параметр $n = 260$ также устанавливается портфельным менеджером, так как он решает, на какую возможность стоит делать ставки. Два параметра, которые не находятся под контролем портфельного менеджера, — это степень p (определяется рынком) и коэффициент θ^* (цель, поставленная инвестором). Поскольку p неизвестна, мы можем смоделировать ее как случайную величину с математическим ожиданием $E[p]$. Давайте определим через θ^* значение p, ниже которого стратегия будет отставать по результативности от целевого коэффициента Шар-

па θ^*, то есть $p_{\theta^*} = \max\{p|\theta \leq \theta^*\}$. Мы можем использовать приведенные выше уравнения (или функцию `binHR`), для того чтобы сделать вывод, что для $p_{\theta^*} = 0 = \frac{2}{3}$, $p < p_{\theta^*} = 0 \Rightarrow \theta \leq 0$. Этим подчеркиваются риски, связанные с данной стратегией, потому что относительно малое падение в p (с $p = .7$ до $p = .67$) смоет всю прибыль. Данная стратегия по своей природе рискованна, даже если находящиеся во владении портфельные активы такими не являются. Это именно то критическое различие, которое мы хотим зафиксировать в этой главе: *риск стратегии* не следует путать с *портфельным риском*.

Большинство фирм и инвесторов вычисляют, отслеживают и сообщают о портфельном риске, не понимая, что он ничего не говорит нам о риске самой стратегии. Риск стратегии — это не риск лежащего в основе базового портфеля, рассчитанный главным управляющим по рискам. Риск стратегии — это риск того, что инвестиционная стратегия с течением времени будет безуспешной, и этот вопрос в гораздо большей степени относится к компетенции главного управляющего по инвестициям. Ответ на вопрос «какова вероятность того, что эта стратегия окажется безуспешной?» эквивалентен вычислению $P[p < p_{\theta^*}]$. Следующий далее алгоритм поможет нам вычислить риск стратегии.

15.4.1. Алгоритм

В этом разделе мы опишем процедуру вычисления $P[p < p_{\theta^*}]$. При заданном временном ряде исходов ставок $\{\pi_t\}_{t=1,\ldots,T}$ во-первых, мы оцениваем $\pi_- = E[\{\pi_t|\pi_t \leq 0\}_{t=1,\ldots,T}]$ и $\pi_+ = E[\{\pi_t|\pi_t > 0\}_{t=1,\ldots,T}]$. В качестве альтернативы $\{\pi_-, \pi_+\}$ можно получить из подгонки смеси двух гауссиан с помощью алгоритма EF3M[1] (Lopez de Prado and Foreman [2014]). Во-вторых, годовая частота n задается отношением $n = \frac{T}{y}$, где y — это число лет, прошедших между $t = 1$ и $t = T$. В-третьих, мы бутстрапируем распределение p следующим образом:

1. Для итераций $i = 1, \ldots, I$:
   - **(а)** Взять $\lfloor nk \rfloor$ образцов из $\{\pi_t\}_{t=1,\ldots,T}$ с возвратом, где k — это число лет, используемых инвесторами для оценки качества стратегии (например, 2 года). Мы обозначаем множество этих взятых образцов как $\{\pi_j^{(i)}\}_{j=1,\ldots,\lfloor nk \rfloor}$.
   - **(б)** Вывести наблюдаемую точность из итерации i как $p_i = \frac{1}{\lfloor nk \rfloor} \|\{\pi_j^{(i)}|\pi_j^{(i)} > 0\}_{j=1,\ldots,\lfloor nk \rfloor}\|$.

[1] Алгоритм точной подгонки первых трех моментов (exact fit of the firsrt three moments, EF3M) основан на биологической эволюции и разработан для ранней идентификации деэволюции стратегии. — *Примеч. науч. ред.*

2. Выполнить подгонку функции плотности вероятности (PDF) p, обозначаемую, как $f[p]$, применив оценщика ядерной плотности на $\{p_i\}_{i=1,\ldots,I}$.

Для достаточно больших k мы можем аппроксимировать этот третий шаг как $f[p] \sim N[\bar{p}, \bar{p}(1-\bar{p})]$, где $\bar{p} = \mathrm{E}[p] = \frac{1}{T}\|\{\pi_t^{(i)} \| \pi_t^{(i)}| > 0\}_{t=1,\ldots,T}\|$. В-четвертых, при заданном пороге θ^* (коэффициенте Шарпа, отделяющем неуспех от успеха) выводим p_{θ^*} (см. раздел 15.4). В-пятых, риск стратегии вычисляется как $\mathrm{P}[p < p_{\theta^*}] = \int_{-\infty}^{p_{\theta^*}} f[p]dp$.

15.4.2. Реализация

Листинг 15.5 приводит одну возможную реализацию этого алгоритма. Обычно мы игнорируем стратегии, где $P[p < p_{\theta^*}] > .05$, как слишком рискованные, даже если они инвестируют в низковолатильные инструменты. Причина в том, что даже если они не потеряют много денег, вероятность того, что они не смогут достичь своей цели, слишком высока. Для развертывания стратегии ее разработчик должен найти способ сократить p_{θ^*}.

Листинг 15.5. Расчет рисков стратегии на практике

```
import numpy as np,scipy.stats as ss
#---------------------------------------
def mixGaussians(mu1,mu2,sigma1,sigma2,prob1,nObs):
    # Случайные выемки образцов из смеси гауссиан
    ret1=np.random.normal(mu1,sigma1,size=int(nObs*prob1))
    ret2=np.random.normal(mu2,sigma2,size=int(nObs)-ret1.shape[0])
    ret=np.append(ret1,ret2,axis=0)
    np.random.shuffle(ret)
    return ret
#---------------------------------------
def probFailure(ret,freq,tSR):
    # Вывести вероятность того, что стратегия может быть безуспешной
    rPos,rNeg=ret[ret>0].mean(),ret[ret<=0].mean()
    p=ret[ret>0].shape[0]/float(ret.shape[0])
    thresP=binHR(rNeg,rPos,freq,tSR)
    risk=ss.norm.cdf(thresP,p,p*(1-p)) # аппроксимация бутстраповского отбора
    return risk
#---------------------------------------
def main():
    #1) Параметры
    mu1,mu2,sigma1,sigma2,prob1,nObs=.05,-.1,.05,.1,.75,2600
    tSR,freq=2.,260
    #2) Сгенерировать выборку из смеси
    ret=mixGaussians(mu1,mu2,sigma1,sigma2,prob1,nObs)
    #3) Вычислить вероятность неуспеха
    probF=probFailure(ret,freq,tSR)
    print 'Вероятность, что стратегия будет безуспешна',probF
    return
#---------------------------------------
if __name__=='__main__':main()
```

Данный подход имеет некоторое сходство с вероятностным коэффициентом Шарпа (PSR) (см. главу 14, а также публикации Bailey and Lopez de Prado [2012, 2014]). Вероятностный коэффициент Шарпа выводит вероятность того, что истинный коэффициент Шарпа превышает заданный порог в рамках негауссовых финансовых возвратов. Аналогичным образом, представленный в этой главе метод выводит вероятность безуспешности стратегии, основываясь на асимметричных бинарных исходах. Ключевое различие состоит в том, что, в отличие от вероятностного коэффициента Шарпа, который не проводит различия между параметрами, находящимися под контролем или вне контроля портфельного менеджера, обсуждаемый здесь метод позволяет портфельному менеджеру изучить жизнеспособность стратегии с учетом параметров, находящихся под его контролем: $\{\pi_-, \pi_+, n\}$. Это полезно при разработке либо оценке жизнеспособности торговой стратегии.

Упражнения

15.1. Портфельный менеджер намерен запустить стратегию, которая нацелена на среднегодовой коэффициент Шарпа, равный 2. Ставки имеют степень точности, равную 60 %, с недельной частотой. Условия выхода равны 2 % для взятия прибыли и −2 % для остановки убытка.

(а) Жизнеспособна ли эта стратегия?

(б) При прочих равных условиях какова требуемая степень точности, которая сделает стратегию прибыльной?

(в) Для какой частоты ставок цель достижима?

(г) Для какого порога взятия прибыли цель достижима?

(д) Какой будет альтернативная остановка убытка?

15.2. В продолжение стратегии из упражнения 15.1:

(а) Какова чувствительность коэффициента Шарпа к изменению каждого параметра на 1 %?

(б) С учетом этих чувствительностей и приняв, что все параметры одинаково трудно улучшить, какой из них является самым доступным?

(в) Влияет ли изменение какого-либо из параметров в упражнении 15.1 на другие? Например, модифицирует ли изменение частоты ставок степень точности и т. д.?

15.3. Предположим, что у вас есть стратегия, которая генерирует ежемесячные ставки в течение двух лет с финансовыми возвратами, подчиняющимися смеси двух гауссовых распределений. Первое распределение имеет среднее значение −0.1 и среднеквадратическое отклонение 0.12. Второе распределение имеет среднее значение 0.06 и среднеквадратическое отклонение 0.03.

Вероятность того, что взятый образец приходит из первого распределения, равна 0.15.

(а) В соответствии с публикациями Lopez de Prado and Peijan [2004] и Lopez de Prado and Foreman [2014] выведите первые четыре момента для финансовых возвратов смеси.

(б) Рассчитайте среднегодовой коэффициент Шарпа.

(в) Используя эти моменты, вычислите вероятностный коэффициент Шарпа PSR[1] (см. главу 14). Вы бы отказались от этой стратегии на 95 %-ном уровне достоверности?

15.4. Используя листинг 15.5, вычислите $P[p < p_{\theta^*=1}]$ для стратегии, описанной в упражнении 15.3. Вы бы отказались от этой стратегии на уровне достоверности 0.05? Согласуется ли этот результат с $PSR[\theta^*]$?

15.5. Насколько оба эти метода дополняют друг друга? Какой результат, по вашим ожиданиям, будет точнее в общем случае: $PSR[\theta^*]$ или $P[p < p_{\theta^*=1}]$? Как эти два метода дополняют друг друга?

15.6. Перепроверьте результаты из главы 13, используя новые знания, полученные в этой главе.

(а) Имеет ли смысл асимметрия между порогами взятия прибыли и остановки убытка в оптимальных торговых правилах?

(б) Каков диапазон p, предполагаемый на рис. 13.1, для суточной частоты ставок?

(в) Каков диапазон p, предполагаемый на рис. 13.5, для недельной частоты ставок?

16 Распределение финансовых активов

16.1. Актуальность

В этой главе представлен подход на основе иерархического паритета рисков (hierarchical risk parity, HRP)[1]. Портфели на основе иерархического паритета рисков решают три главные проблемы квадратических оптимизаторов в целом и алгоритма критической линии Марковица (critical line algorithm, CLA) в частности: нестабильность, концентрацию и пониженную результативность. В подходе на основе иерархического паритета рисков применяется современная математика (теория графов и методы машинного обучения) для построения диверсифицированного портфеля на основе информации, содержащейся в ковариационной матрице. Однако в отличие от квадратических оптимизаторов, данный подход не требует обратимости ковариационной матрицы. Фактически, подход на основе HRP может вычислять портфель на (плохо) вырожденной или даже сингулярной ковариационной матрице, что невозможно для квадратических оптимизаторов. Эксперименты Монте-Карло показывают, что данный метод обеспечивает более низкую вневыборочную дисперсию, чем алгоритм CLA, хотя целевой задачей оптимизации в этом алгоритме является минимальная дисперсность. Подход на основе HRP производит менее рискованные портфели вневыборочно по сравнению с традиционными методами паритета рисков. Исторические аналитические исследования также показали, что данный подход мог бы достигать более высокой результативности, чем стандартные подходы (Kolanovic и соавт. [2017], Raffinot [2017]). Практическое применение подхода HRP заключается в определении размещений между многочисленными стратегиями МО.

16.2. Проблема выпуклой портфельной оптимизации

Конструирование портфеля является, пожалуй, самой распространенной финансовой задачей. На ежедневной основе инвестиционные менеджеры должны

[1] Краткая версия этой главы вышла в журнале Portfolio Management (Портфельный менеджмент), т. 42, № 4, с. 59–69, лето 2016 г.

формировать портфели, учитывающие их мнения и прогнозы в отношении рисков и финансовых возвратов. Это изначальный вопрос, на который 24-летний Гарри Марковиц попытался ответить более шести десятилетий назад. Его монументальная проницательность заключалась в признании того, что различные уровни риска связаны с разными оптимальными портфелями с точки зрения скорректированных на риск финансовых возвратов, отсюда и понятие «эффективной границы» (Markowitz [1952]). Одно из последствий заключается в том, что размещение всех активов в инвестиции с наибольшими ожидаемыми финансовыми возвратами редко бывает оптимальным. Вместо этого мы должны учитывать корреляции между альтернативными инвестициями, тем самым конструируя диверсифицированный портфель.

Прежде чем получить докторскую степень в 1954 году, Марковиц оставил академию, чтобы начать работать на Rand Corporation, где он разработал алгоритм критической линии. Алгоритм критической линии (CLA) — это процедура квадратической оптимизации, специально спроектированная для задач портфельной оптимизации с ограничениями в виде неравенств. Этот алгоритм примечателен тем, что гарантирует, что точное решение будет найдено после известного числа итераций и что он изобретательно обходит условия Каруша—Куна—Таккера (Kuhn and Tucker [1951]). Описание и реализацию этого алгоритма с открытым исходным кодом можно найти в публикации Bailey and Lopez de Prado [2013]. Удивительно, но большинство финансовых практиков до сих пор, по всей видимости, не знают об алгоритме CLA, так как они часто опираются на универсальные методы квадратического программирования, которые не гарантируют правильного решения или времени остановки.

Несмотря на гениальность теории Марковица, ряд практических задач делают решения алгоритма CLA несколько ненадежными. Главное предостережение заключается в том, что небольшие отклонения в прогнозируемых финансовых возвратах приводят к тому, что этот алгоритм производит очень разные портфели (Michaud [1998]). С учетом того что финансовые возвраты редко могут быть спрогнозированы с достаточной точностью, многие авторы предпочли полностью от них отказаться и сосредоточиться на ковариационной матрице. Это привело к риск-ориентированным подходам к размещению активов, ярким примером которых является «паритет рисков» (Jurczenko [2015]). Отбрасывание прогнозов финансовых возвратов улучшает, но не предотвращает проблемы нестабильности. Причина в том, что методы квадратического программирования требуют инверсии положительно определенной ковариационной матрицы (все собственные значения должны быть положительными). Эта инверсия склонна к большим ошибкам, когда ковариационная матрица численно плохо обусловлена, то есть когда она имеет высокое число обусловленности (Bailey and Lopez de Prado [2012]).

16.3. Проклятие Марковица

Число обусловленности ковариационной, корреляционной (или нормальной, следовательно, диагонализируемой) матрицы является абсолютным значением

соотношения между ее максимальными и минимальными (по модулям) собственными значениями. На рис. 16.1 показаны отсортированные собственные значения нескольких корреляционных матриц, где число обусловленности — это соотношение между первым и последним значениями каждой строки. Это число является наименьшим для диагональной корреляционной матрицы, которая является обратной себе самой. Когда мы добавляем коррелируемые (мультиколлинеарные) инвестиции, число обусловленности растет. В какой-то момент число обусловленности настолько велико, что числовые ошибки делают обратную матрицу слишком неустойчивой: малое изменение на любой записи приведет к совершенно другому обратному. Проклятие Марковица в следующем: чем больше инвестиции коррелируются, тем больше потребность в диверсификации и тем больше вероятность того, что мы получим нестабильные решения. Преимущества диверсификации зачастую с избытком перекрываются ошибками оценивания.

Увеличение размера ковариационной матрицы только ухудшит ситуацию, так как каждый коэффициент ковариации оценивается с меньшими степенями свободы.

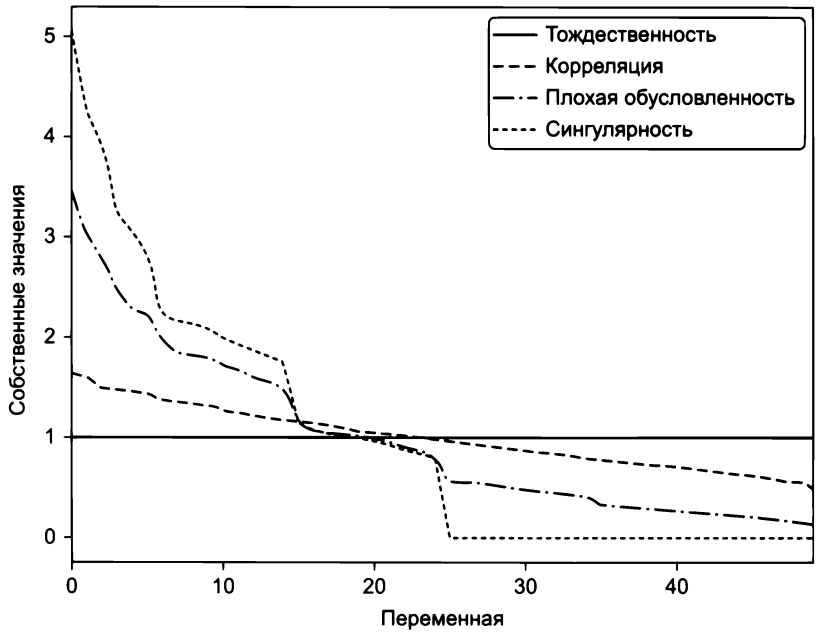


Рис. 16.1. Визуальное представление проклятия Марковица.

Диагональная корреляционная матрица имеет наименьшее число обусловленности. По мере добавления коррелированных инвестиций максимальное собственное значение становится больше, а минимальное собственное значение становится меньше. Число обусловленности быстро растет, что приводит к неустойчивым инверсно-корреляционным матрицам. В какой-то момент преимущества диверсификации с избытком перекрываются ошибками оценивания

В общем случае, нам нужно по крайней мере $1/2N(N+1)$ одинаково распределенных взаимно независимых наблюдений, для того чтобы оценить ковариационную матрицу размера N, которая не является сингулярной. Например, оценивание обратимой ковариационной матрицы размера 50 требует, по крайней мере, 5 лет суточных одинаково распределенных взаимно независимых данных. Как известно большинству инвесторов, корреляционные структуры не остаются неизменными в течение таких длительных периодов при любом разумном уровне достоверности. О серьезности этих проблем свидетельствует тот факт, что, как было доказано, даже наивные (равновзвешенные) портфели одерживают верх над среднедисперсной риск-ориентированной оптимизацией вневыборочно (De Miguel и соавт. [2009]).

16.4. От геометрических связей к иерархическим

В последние годы эти проблемы нестабильности получили значительное внимание, как тщательно описано в публикации Kolm и соавт. [2014]. Большинство альтернатив пытаются достичь устойчивости путем встраивания дополнительных ограничений (Clarke и соавт. [2002]), введения байесовых априорностей (Black and Litterman [1992]) или улучшения численной устойчивости обратной ковариационной матрицы (Ledoit and Wolf [2003]).

Все обсуждавшиеся до сих пор методы, хотя и опубликованы в последние годы, получены из (очень) классических областей математики: геометрии, линейной алгебры и дифференциального и интегрального исчисления. Корреляционная матрица — это линейно-алгебраический объект, который измеряет косинусы углов между любыми двумя векторами в векторном пространстве, образованном последовательностью финансовых возвратов (см. Calkin and Lopez de Prado [2014a, 2015b]). Одной из причин неустойчивости квадратических оптимизаторов является то, что векторное пространство моделируется как полный (полносвязный) граф, где каждый узел является потенциальным кандидатом на замену другого. В алгоритмических терминах инвертирование матрицы означает оценивание частных корреляций по всему полному графу. На рис. 16.2, *а* визуализируются связи, вытекающие из ковариационной матрицы 50×50, то есть 50 вершин и 1225 ребер. Эта сложная структура увеличивает малые ошибки оценивания, что приводит к неправильным решениям. Интуитивно хотелось бы отбросить лишние ребра.

Рассмотрим на минуту практические последствия такой топологической структуры. Предположим, что инвестор желает построить диверсифицированный портфель из ценных бумаг, в том числе сотни акций, облигаций, хедж-фонды, недвижимость, частные размещения и т. д. Некоторые инвестиции кажутся более близкими заменителями друг друга, а другие — взаимодополняющими. Например, акции могут быть сгруппированы по ликвидности, размеру, отрасли и региону, где акции в рамках данной группы конкурируют за размещения. При принятии

решения о размещении в публично торгуемые финансовые акции такого крупного банка США, как J. P. Morgan, мы будем рассматривать вопрос о добавлении или сокращении размещения в другой крупный публично торгуемый банк США, такой как Goldman Sachs, а не банк небольшой общины в Швейцарии или холдинг недвижимости в Карибском бассейне. Однако для корреляционной матрицы все инвестиции являются потенциальными заменителями друг друга. Другими словами, корреляционные матрицы не имеют понятия *иерархии*. Это отсутствие иерархической структуры позволяет весам свободно варьироваться непредусмотренными способами, что является первопричиной нестабильности портфеля CLA. На рис. 16.2, *б* показана иерархическая структура, называемая деревом. Древовидная структура вводит два желательных признака: 1) она имеет только $N - 1$ ребер, соединяющих N вершин, поэтому веса балансируют только между сверстниками на разных иерархических уровнях; и 2) веса распределяются сверху вниз, в соответствии с тем, сколько менеджеров активов строят свои портфели (например, от класса активов к секторам и далее к конкретным ценным бумагам). По этим причинам иерархические структуры лучше предназначены для того, чтобы давать не только стабильные, но и интуитивные результаты.

В этой главе мы изучим новый метод конструирования инвестиционного портфеля, который решает проблемы алгоритма CLA, используя современную математику: теорию графов и машинное обучение. В данном методе иерархического паритета рисков (HRP) используется информация, содержащаяся в ковариационной матрице, не требующая ее инверсии или положительной определенности. Метод HRP может даже вычислять портфель, основанный на сингулярной ковариационной матрице. Данный алгоритм работает в три этапа: древовидная кластеризация, квазидиагонализации и рекурсивное дробление на две части.

16.4.1. Деревовидная кластеризация

Рассмотрим $T \times N$-матрицу наблюдений X, такую как ряд финансовых возвратов из N переменных за T периодов. Мы хотели бы объединить эти N векторов-столбцов в иерархическую структуру из кластеров, чтобы размещения могли течь вниз по течению через древовидный граф.

Во-первых, мы вычисляем корреляционную $N \times N$-матрицу с записями $\rho = \{\rho_{ij}\}_{i,j=1,\ldots,N}$, где $\rho_{ij} = \rho[X_i, X_j]$. Мы определяем метрический показатель расстояния $d: (X_i, X_j) \subset$ $\subset B \to \mathbb{R} \in [0, 1]$, $d_{ij} = d[X_i, X_j] = \sqrt{\frac{1}{2}(1 - \rho_{i,j})}$, где B — это декартово произведение элементов в $\{1, \ldots, i, \ldots, N\}$. Он позволяет нам вычислить $N \times N$ — матрицу расстояний $D = \{d_{ij}\}_{i,j=1,\ldots,N}$. Матрица D является собственным метрическим пространством (см. дополнение 16.A.1 относительно доказательства), в том смысле что $d[x, y] \geq 0$ (неотрицательность), $d[x, y] = 0 \Leftrightarrow X = Y$ (коинцидентность), $d[x, y] = d[Y, X]$ (симметрия) и $d[X, Z] \leq d[x, y] + d[Y, Z]$ (субаддитивность). См. пример 16.1.

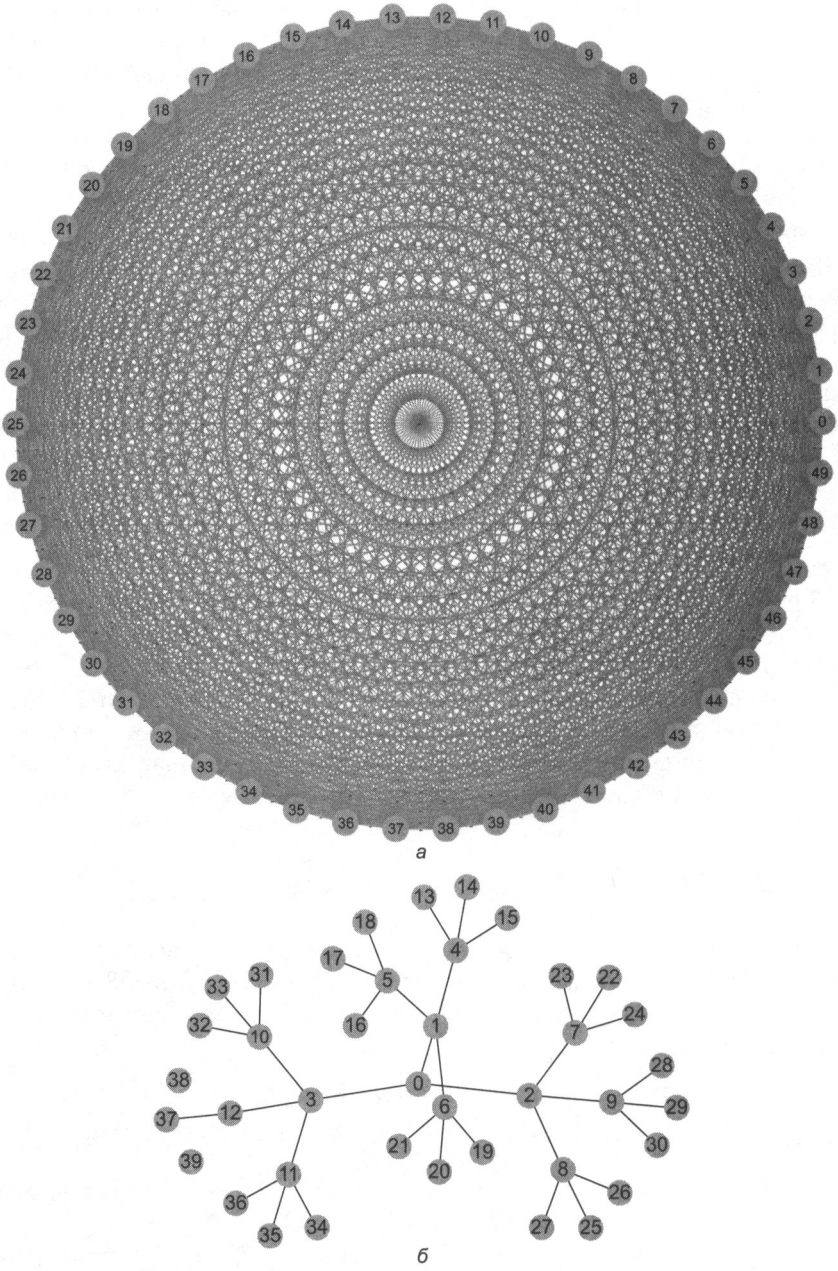


Рис. 16.2. Полный граф (сверху) и древовидный граф (снизу).

Корреляционные матрицы можно представить в виде полных графов, в которых отсутствует понятие иерархии: каждая инвестиция взаимозаменяема другой. Напротив, древовидные структуры встраивают иерархические связи

$$\{\rho_{i,j}\} = \begin{bmatrix} 1 & .7 & .2 \\ .7 & 1 & -.2 \\ .2 & -.2 & 1 \end{bmatrix} \rightarrow \{d_{i,j}\} = \begin{bmatrix} 0 & .3873 & .6325 \\ .3873 & 0 & .7746 \\ .6325 & .7746 & 0 \end{bmatrix}.$$

Пример 16.1. Кодирование корреляционной матрицы p как матрицы расстояний D

Во-вторых, мы вычисляем евклидово расстояние между любыми двумя векторами-столбцами D, $\tilde{d} : (D_i, D_j) \subset B \rightarrow \mathbb{R} \in [0,\sqrt{N}]$, $\tilde{d}_{i,j} = \tilde{d}\ [D_i,D_j] = \sqrt{\sum_{n=1}^{N}(d_{n,i} - d_{n,j})^2}$. Обратите внимание на разницу между метрическими показателями расстояния $d_{i,j}$ и $\tilde{d}_{i,j}$. В отличие от $d_{i,j}$, которое определяется на векторах-столбцах X, $\tilde{d}_{i,j}$ определяется по векторах-столбцах D (расстояние расстояний). Следовательно, $\tilde{d}$ — это расстояние, определенное на всем метрическом пространстве D, так как каждое $\tilde{d}_{i,j}$ является функцией от всей корреляционной матрицы (а не конкретной перекрестно-корреляционной пары). См. пример 16.2.

$$\{d_{i,j}\} = \begin{bmatrix} 0 & .3873 & .6325 \\ .3873 & 0 & .7746 \\ .6325 & .7746 & 0 \end{bmatrix} \rightarrow \{\tilde{d}_{i,j}\}_{i,j=\{1,2,3\}} = \begin{bmatrix} 0 & .5659 & .9747 \\ .5659 & 0 & 1.1225 \\ .9747 & 1.1225 & 0 \end{bmatrix}.$$

Пример 16.2. Евклидово расстояние корреляционных расстояний

В-третьих, мы кластеризуем пару столбцов (i^*,j^*) такую, что $(i^*,j^*) = \underset{i \neq j}{\text{argmin}}(i,j)\{\tilde{d}_{i,j}\}$, и обозначаем этот кластер через $u[1]$. См. пример 16.3.

$$\{\tilde{d}_{i,j}\}_{i,j=\{1,2,3\}} = \begin{bmatrix} 0 & .5659 & .9747 \\ .5659 & 0 & 1.1225 \\ .9747 & 1.1225 & 0 \end{bmatrix} \rightarrow u[1] = (1,2).$$

Пример 16.3. Кластеризация элементов

В-четвертых, нам нужно определить расстояние между вновь сформированным кластером $u[1]$ и одиночными (некластеризованными) элементами, чтобы можно было обновить $\{\tilde{d}_{ij}\}$. В иерархическом кластерном анализе это называется «критерием связи». Например, мы можем определить расстояние между элементом i из $\tilde{d}$ и новым кластером u [1] как $\dot{d}_{i,u[1]} = \min[\{\tilde{d}_{i,j}\}_{j \in u[1]}]$ (алгоритм ближайшей точки). См. пример 16.4.

$$u[1] = (1,2) \rightarrow \{\dot{d}_{i,u[1]}\} = \begin{bmatrix} \min\,[0, .5659] \\ \min\,[.5659, 0] \\ \min\,[.9747, 1.1225] \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ .9747 \end{bmatrix}.$$

Пример 16.4. Обновление матрицы $\{\tilde{d}_{ij}\}$ новым кластером u

В-пятых, матрица $\{\tilde{d}_{i,j}\}$ обновляется путем добавления $\dot{d}_{i,u[1]}$ и отбрасывания кластеризованных столбцов и строк $j \in u[1]$. См. пример 16.5.

$$\{\tilde{d}_{i,j}\}_{i,j=\{1,2,3,4\}} = \begin{bmatrix} 0 & .5659 & .9747 & 0 \\ .5659 & 0 & 1.1225 & 0 \\ .9747 & 1.1225 & 0 & .9747 \\ 0 & 0 & .9747 & 0 \end{bmatrix}$$

$$\{\tilde{d}_{i,j}\}_{i,j=\{3,4\}} = \begin{bmatrix} 0 & .9747 \\ .9747 & 0 \end{bmatrix}$$

Пример 16.5. Обновление матрицы $\{\tilde{d}_{i,j}\}$ новым кластером u

В-шестых, применяемые рекурсивно, шаги 3, 4 и 5 позволяют добавить $N-1$ таких кластеров в матрицу D, после чего конечный кластер будет содержать все исходные элементы и алгоритм кластеризации останавливается. См. пример 16.6.

$$\{\tilde{d}_{i,j}\}_{i,j=\{3,4\}} = \begin{bmatrix} 0 & .9747 \\ .9747 & 0 \end{bmatrix} \rightarrow u[2] = (3, 4) \rightarrow \text{Стоп.}$$

Пример 16.6. Рекурсия в поисках оставшихся кластеров

На рис. 16.3 показаны кластеры, формируемые на каждой итерации для данного примера, а также расстояния $\tilde{d}_{i^*,j^*}$, инициировавшие каждый кластер (третий шаг). Данная процедура может быть применена к широкому спектру метрических показателей расстояния $d_{i,j}$, $\tilde{d}_{i,j}$ и $\dot{d}_{i,u}$, помимо тех, которые проиллюстрированы в этой главе. См. публикацию Rokach and Maimon [2005] относительно альтернативных метрических показателей, обсуждение вектора Фидлера и метода спектральной кластеризации Стюарта в публикации Brualdi [2010], а также алгоритмы в библиотеке scipy[1]. Листинг 16.1 содержит пример древовидной кластеризации с использованием функциональности библиотеки scipy.

Листинг 16.1. Древовидная кластеризация с использованием функциональности библиотеки scipy

```
import scipy.cluster.hierarchy as sch
import numpy as np
import pandas as pd
cov,corr=x.cov(),x.corr()
dist=((1-corr)/2.)**.5 # матрица расстояний
link=sch.linkage(dist,'single') # матрица связей
```

[1] Дополнительные метрические показатели см. на http://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.distance.pdist.html, http://docs.scipy.org/doc/scipy-0.16.0/reference/generated/scipy.cluster.hierarchy.linkage.html.

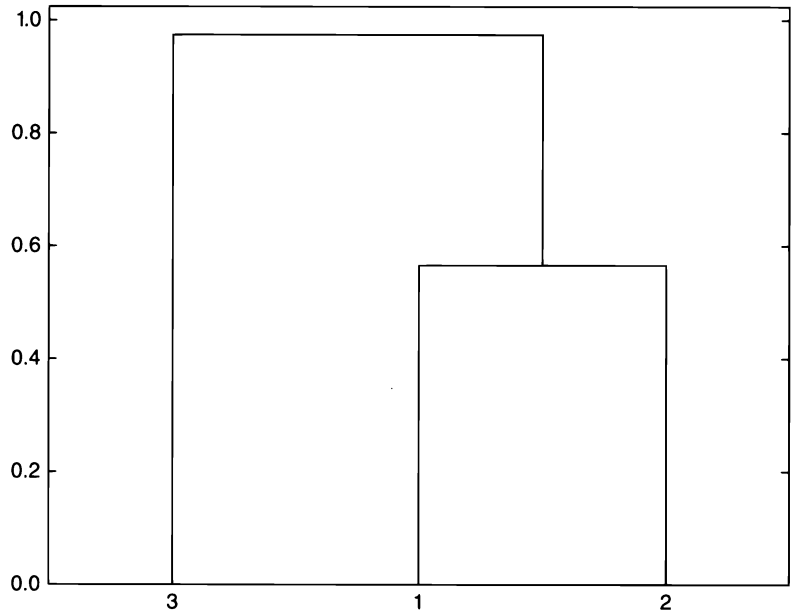


Рис. 16.3. Последовательность формирования кластеров.

Древовидная структура, выведенная из нашего численного примера, здесь показана в виде дендрограммы. Ось ординат измеряет расстояние между двумя объединяемыми листьями

Этот этап позволяет определить матрицу связей как $(N-1) \times 4$ матрицу со структурой $Y = \{(y_{m,1}, y_{m,2}, y_{m,3}, y_{m,4})\}_{m=1,\ldots,N-1}$ (то есть с одним 4-элементным кортежем на кластер). Элементы $(y_{m,1}, y_{m,2})$ сообщают о составляющих. Элемент $y_{m,3}$ сообщает расстояние между $y_{m,1}$ и $y_{m,2}$, то есть $y_{m,3} = \tilde{d}_{y_{m,1}, y_{m,2}}$. Элемент $y_{m,4} \leq N$ сообщает число исходных элементов, включенных в кластер m.

16.4.2. Квазидиагонализация

Этот этап реорганизует строки и столбцы ковариационной матрицы таким образом, что наибольшие значения лежат вдоль диагонали. Данная квазидиагонализация ковариационной матрицы (не требующая замены базиса) проявляет полезное свойство: схожие инвестиции помещаются рядом, и разнородные инвестиции помещаются далеко друг от друга (см. рис. 16.5 и 16.6 с примерами). Алгоритм работает следующим образом: известно, что каждая строка матрицы связей объединяет две ветви в одну. Мы рекурсивно заменяем кластеры в $(y_{N-1,1}, y_{N-1,2})$ их компонентами до тех пор, пока больше не останется кластеров. Эти замены сохраняют порядок кластеризации. На выходе получается отсортирован-

ный список исходных (некластеризованных) элементов. Эта логика реализована в листинге 16.2.

Листинг 16.2. Квазидиагонализация

```
def getQuasiDiag(link):
    # Отсортировать кластеризованные элементы по расстоянию
    link=link.astype(int)
    sortIx=pd.Series([link[-1,0],link[-1,1]])
    numItems=link[-1,3] # число исходных элементов
    while sortIx.max()>=numItems:
        sortIx.index=range(0,sortIx.shape[0]*2,2) # создать пространство
        df0=sortIx[sortIx>=numItems] # отыскать кластеры
        i=df0.index;j=df0.values-numItems
        sortIx[i]=link[j,0] # элемент 1
        df0=pd.Series(link[j,1],index=i+1)
        sortIx=sortIx.append(df0) # элемент 2
        sortIx=sortIx.sort_index() # пересортировать
        sortIx.index=range(sortIx.shape[0]) # реиндексировать
    return sortIx.tolist()
```

16.4.3. Рекурсивное дробление пополам

Этап 2 произвел квазидиагональную матрицу. Инверсно-дисперсное размещение оптимально для диагональной ковариационной матрицы (см. дополнение 16.A.2 с доказательством). Мы можем воспользоваться этими фактами двумя разными способами: 1) снизу вверх, чтобы определить дисперсию сплошного подмножества как дисперсию инверсно-дисперсного размещения; либо 2) сверху вниз, чтобы раздробить размещения между смежными подмножествами в обратной пропорции в их агрегированные дисперсии. Следующий ниже алгоритм формализует эту идею:

1. Алгоритм инициализируется путем:

   а) задания списка элементов: $L = \{L_0\}$, при $L_0 = \{n\}_{n=1,...,N}$;

   б) назначения единичного веса всем элементам: $w_n = 1, \forall n = 1, ..., N$.

2. Если $|L_i| = 1, \forall L_i \in L$, то алгоритм останавливается.

3. Для каждого $L_i \in L$ такого, что $|L_i| > 1$:

   а) рассечь L_i на две подгруппы, $L_i^{(1)} \cup L_i^{(2)} = L_i$, где $|L_i^{(1)}| = \text{int}[\frac{1}{2}|L_i|]$ с сохранением упорядоченности;

   б) определить дисперсию $L_i^{(j)}, j = 1, 2$ в качестве квадратичной формы $\tilde{V}_i^{(j)} \equiv \equiv \tilde{w}_i^{(j)'} V_i^{(j)} \tilde{w}_i^{(j)}$, где $V_i^{(j)}$ — это ковариационная матрица между составляю-

щими дробления надвое $L_i^{(j)}$, а $\tilde{w}_i^{(j)} = \text{diag}[V_i^{(j)}]^{-1} \frac{1}{\text{tr}[\text{diag}[V_i^{(j)}]^{-1}]}$, где diag[.] и tr[.] — это диагональный оператор и оператор следа;

в) вычислить фактор дробления $\alpha_i = 1 - \frac{\tilde{V}_i^{(1)}}{\tilde{V}_i^{(1)} + \tilde{V}_i^{(2)}}$ такой, что $0 \le \alpha_i \le 1$;

г) перешкалировать размещения w_n с коэффициентом α_i, $\forall n \in L_i^{(1)}$;

д) перешкалировать размещения w_n с коэффициентом $(1 - \alpha_i)$, $\forall n \in L_i^{(2)}$.

4. Вернуться к шагу 2.

Шаг 3б использует квазидиагонализацию снизу вверх, потому что он определяет дисперсию подразделения $L_i^{(j)}$ с использованием обратных дисперсионных перевесовок $\tilde{w}_i^{(j)}$). Шаг 3в использует квазидиагонализацию сверху вниз, потому что он подразделяет вес в обратной пропорции относительно дисперсии кластера. Этот алгоритм гарантирует, что $0 \le w_i \le 1$, $\forall i = 1, ..., N$ и $\sum_{i=1}^{N} w_i = 1$, потому что на каждой итерации мы подразделяем веса, полученные от более высоких иерархических уровней. На этом этапе можно легко ввести ограничения путем замены уравнений в шагах 3в, 3г и 3д в соответствии с предпочтениями пользователя. Этап 3 реализован в листинге 16.3.

Листинг 16.3. Рекурсивное дробление на две части (бисекция)

```
def getRecBipart(cov,sortIx):
    # Вычислить выделение капитала портфелем HRP
    w=pd.Series(1,index=sortIx)
    cItems=[sortIx] # инициализировать все элементы в одном кластере
    while len(cItems)>0:
        cItems=[i[j:k] for i in cItems for j,k in ((0,len(i)/2),\
            (len(i)/2,len(i))) if len(i)>1] # бисекция
        for i in xrange(0,len(cItems),2): # выполнить разбор в парах
            cItems0=cItems[i] # кластер 1
            cItems1=cItems[i+1] # кластер 2
            cVar0=getClusterVar(cov,cItems0)
            cVar1=getClusterVar(cov,cItems1)
            alpha=1-cVar0/(cVar0+cVar1)
            w[cItems0]*=alpha # вес 1
            w[cItems1]*=1-alpha # вес 2
    return w
```

На этом мы закончим первое описание алгоритма HRP, который решает проблему размещения в лучшем случае за детерминированное логарифмическое время, $T(n) = O(\log_2[n])$, и в худшем случае за детерминированное линейное время, $T(n) = O(n)$. Далее мы применим на практике то, что узнали, и оценим точность метода вневыборочно.

16.5. Численный пример

Мы начнем с симулирования матрицы наблюдений X, порядка (10 000 × 10). Корреляционная матрица визуализирована на рис. 16.4 как тепловая карта. Рисунок 16.5 показывает дендрограмму результирующих кластеров (этап 1). На рис. 16.6 показана та же корреляционная матрица, реорганизованная в блоки в соответствии с выявленными кластерами (этап 2). Приложение 16.A.3 предоставляет исходный код, используемый для генерирования данного числового примера.

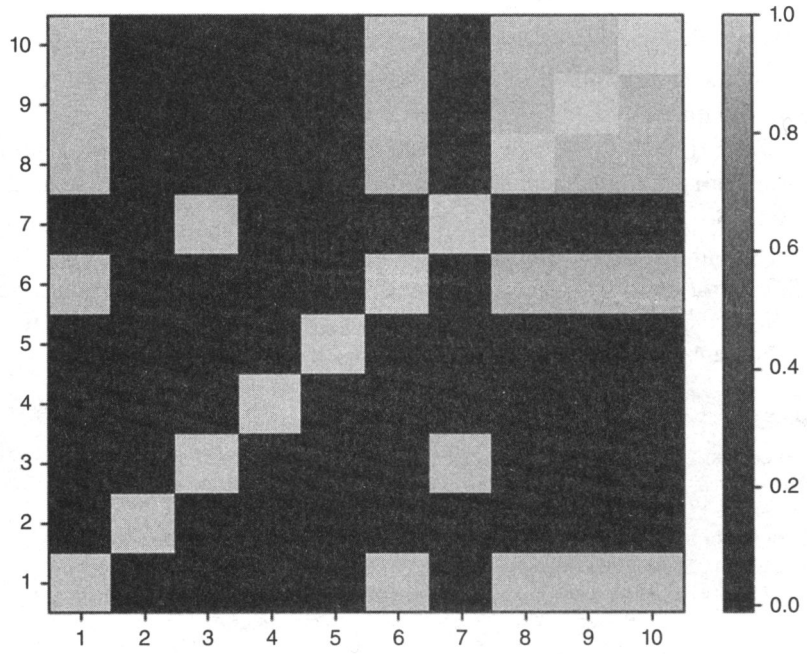


Рис. 16.4. Тепловая карта оригинальной ковариационной матрицы.

Эта корреляционная матрица была вычислена с помощью функции generateData из листинга 16.4 (см. раздел 16.A.3). Последние пять столбцов частично коррелированы с несколькими из первых пяти рядов

На этих случайных данных мы вычисляем размещения портфеля HRP (этап 3) и сравниваем их с размещениями из двух конкурирующих методологий: 1) квадратическая оптимизация, представленная минимально-дисперсным портфелем CLA (единственный портфель эффективной границы, который не зависит от средних значений финансовых возвратов), и 2) традиционный паритет рисков, примером которого является инверсно-дисперсный портфель (inverse-variance portfolio, IVP). См. публикацию Bailey and Lopez de Prado [2013] с описанием комплексной реализа-

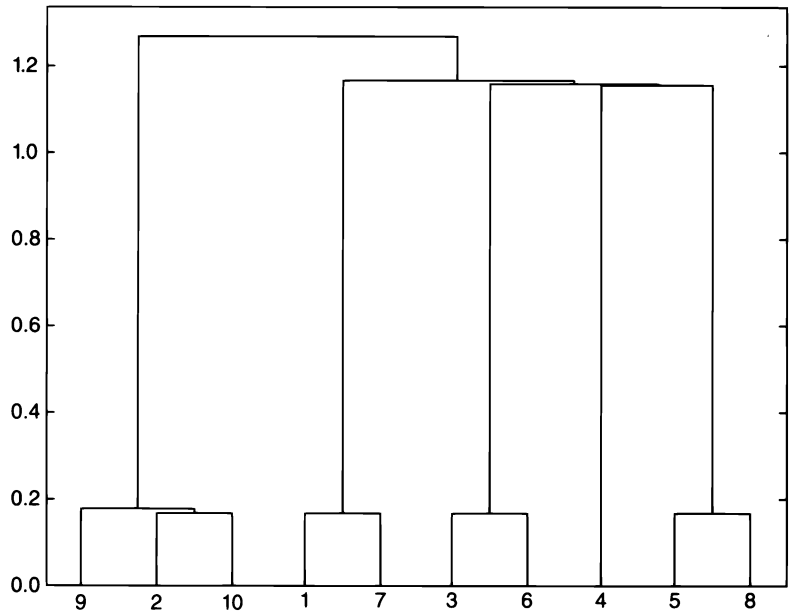


Рис. 16.5. Древовидная диаграмма формирования кластеров.

Кластеризующая процедура правильно определила, что ряды 9 и 10 являются перемещениями ряда 2, следовательно (9, 2, 10) сгруппированы вместе. Аналогичным образом, 7 является перемещением 1, 6 является перемещением 3 и 8 является перемещением 5. Единственный исходный элемент, который не был перемещен, — это 4, и это тот элемент, для которого кластеризующий алгоритм не нашел сходства

ции портфеля CLA и приложение 16.A.2 с выведением формулы портфеля IVP. Мы применяем стандартные ограничения, что $0 \leq w_i \leq 1$ (неотрицательность), $\forall i = 1, ..., N$ и $\sum_{i=1}^{N} w_i = 1$ (полная инвестиция). Попутно заметим, что число обусловленности для ковариационной матрицы в данном примере составляет всего 150.9324, не особо высокое и поэтому не является неблагоприятным для портфеля CLA.

Из размещений активов в табл. 16.1 мы можем оценить несколько стилизованных признаков. Во-первых, портфель алгоритма критической линии (CLA) концентрирует 92.66 % размещений на лучших пяти находящихся во владении портфельных активах, в то время как портфель иерархического паритета рисков (HRP) концентрирует только 62.57 %. Во-вторых, портфель CLA присваивает нулевой вес трем инвестициям (без ограничения $0 \leq w_i$; размещение было бы отрицательным). В-третьих, портфель HRP, похоже, находит компромисс между концентрированным решением портфеля CLA и инверсно-дисперсным портфельным (IVP) размещением традиционного паритета рисков. Читатель может использовать исходный код в приложении 16.A.3 и перепроверить, что эти результаты обычно справедливы для альтернативных случайных ковариационных матриц.

Таблица 16.1. Сравнение трех распределений

Вес	CLA	HPR	IVP
1	14.44 %	7.00 %	10.36 %
2	19.93 %	7.59 %	10.28 %
3	19.73 %	10.84 %	10.36 %
4	19.87 %	19.03 %	10.25 %
5	18.68 %	9.72 %	10.31 %
6	0.00 %	10.19 %	9.74 %
7	5.86 %	6.62 %	9.80 %
8	1.49 %	9.10 %	9.65 %
9	0.00 %	7.12 %	9.64 %
10	0.00 %	12.79 %	9.61 %

Характерный исход трех изученных портфельных методов: метод CLA концентрирует веса на малом числе инвестиций, следовательно, становится предрасположенным идиосинкразическим шокам. Метод IVP равномерно рассеивает веса по всем инвестициям, игнорируя корреляционную структуру. Это делает его уязвимым для системных шоков. Метод HRP находит компромисс между диверсификацией по всем инвестициям и диверсификацией по кластеру, что делает его более устойчивым к обоим типам шоков

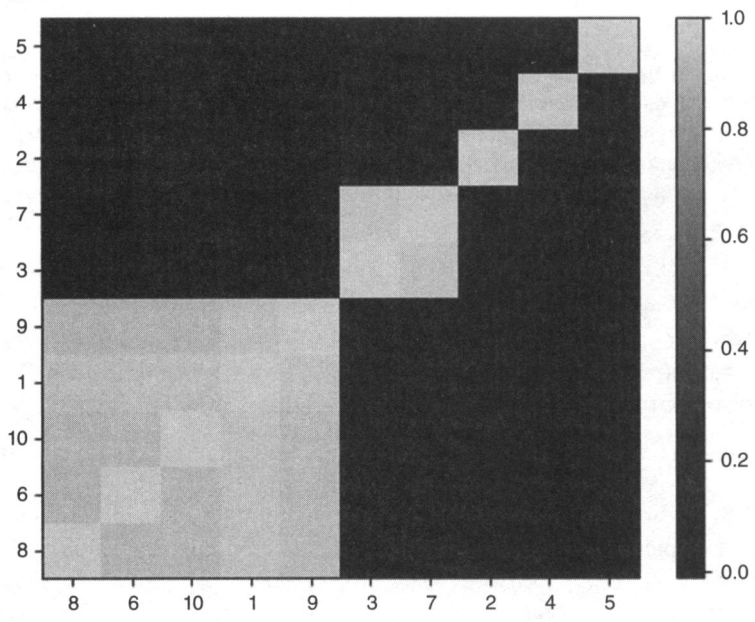


Рис. 16.6. Кластеризация ковариационной матрицы.

Этап 2 квазидиагонализует корреляционную матрицу в том смысле, что наибольшие значения лежат вдоль диагонали. Однако в отличие от анализа главных компонент (PCA) или аналогичных процедур, портфель HRP не требует изменения базиса. Портфель HRP решает проблему размещения робастно, работая с исходными инвестициями

Экстремальной концентрацией в портфеле CLA движет его цель минимизировать риск инвестиционного портфеля. И все же оба портфеля имеют очень похожее среднеквадратическое отклонение ($\sigma_{HRP} = 0.4640$, $\sigma_{CLA} = 0.4486$). Поэтому портфель CLA отбросил половину инвестиционного универсума в пользу незначительного снижения риска. Реальность, конечно, заключается в том, что портфель CLA обманчиво диверсифицирован, потому что любая бедственная ситуация, влияющая на лучшие пять размещений, будет иметь гораздо большее негативное влияние на портфель CLA, чем на портфель HRP.

16.6. Вневыборочные симуляции Монте-Карло

В нашем численном примере портфель CLA имеет более низкий риск, чем портфель HRP внутривыборочно. Однако портфель с минимальной дисперсией внутри выборки не обязательно является портфелем с минимальной дисперсией вне выборки. Для нас было бы слишком легко подобрать конкретную историческую совокупность данных, где портфель HRP превосходит портфель CLA и портфель IVP (см. Bailey and Lopez de Prado [2014] и вспомните наше обсуждение вопроса систематического смещения при отборе в главе 11). Вместо этого в данном разделе мы следуем парадигме бэктестирования, описанной в главе 13, и посредством симуляций Монте-Карло вневыборочно оцениваем результативность портфеля HRP, сравнивая его с размещениями минимально-дисперсного портфеля CLA и инверсно-дисперсного портфеля IVP традиционного паритета рисков. Это также поможет нам понять, какие признаки делают метод предпочтительнее остальных, независимо от эпизодических контрпримеров.

Во-первых, мы генерируем 10 рядов случайных гауссовых финансовых возвратов (520 наблюдений, эквивалентных двум годам суточной истории) с 0 средним и произвольным стандартным отклонением, равным 10 %. Реальные цены демонстрируют частые скачки (Merton [1976]), и финансовые возвраты не являются межсекторально независимыми, поэтому в наши сгенерированные данные мы должны добавить случайные шоки и случайную корреляционную структуру. Во-вторых, мы вычисляем портфели HRP, CLA и IVP, оглядываясь на 260 наблюдений (год суточной истории). Эти портфели переоцениваются и ребалансируются каждые 22 наблюдения (что эквивалентно месячной периодичности). В-третьих, мы вычисляем вневыборочные финансовые возвраты, связанные с этими тремя портфелями. Эта процедура повторяется 10 000 раз.

Все средние портфельные финансовые возвраты вне выборки по существу равны 0, как и ожидалось. Критическое различие происходит от дисперсии вневыборочных портфельных финансовых возвратов: $\sigma^2_{CLA} = 0.1157$, $\sigma^2_{IVP} = 0.0928$ и $\sigma^2_{HRP} = 0.0671$. Хотя цель портфеля CLA состоит в том, чтобы предоставить самую низкую дисперсию (то есть цель его оптимизационной программы), его резуль-

тативность показывает самую высокую дисперсию вне выборки и на 72.47 % более высокую дисперсию, чем в портфеле HRP. Этот экспериментальный вывод согласуется с историческими свидетельствами в публикации De Miguel и соавт. [2009]. Другими словами, портфель HRP улучшит вневыборочный коэффициент Шарпа стратегии портфеля CLA примерно на 31.3 %, что является довольно значительным стимулом. Допущение, что ковариационная матрица диагональна, привносит некоторую устойчивость в портфель IVP; однако его дисперсия по-прежнему на 38.24 % больше, чем в портфеле HRP. Это снижение дисперсии вне выборки критически важно для инвесторов с паритетом рисков, если учесть использование ими значительного кредитного плеча. См. публикацию Bailey и соавт. [2014], где вопрос внутривыборочной и вневыборочной результативности обсуждается шире.

Математическое доказательство превосходства портфеля HRP по результативности над портфелем на основе алгоритма CLA Марковица и портфелем IVP традиционного паритета рисков несколько запутанно и выходит за рамки этой главы. Интуитивно мы можем понять вышеуказанные эмпирические результаты следующим образом. Шоки, влияющие на конкретные инвестиции, наказывают концентрацию в портфеле CLA. Шоки, связанные с несколькими коррелированными инвестициями, наказывают незнание портфелем IVP корреляционной

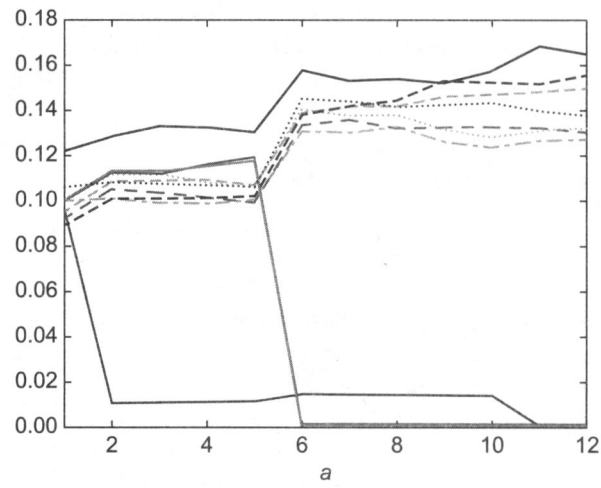


Рис. 16.7. а) временные ряды размещений для портфеля IVP.

Между первым и вторым ребалансированием одна инвестиция получает своеобразный шок, который увеличивает ее дисперсию. Ответ портфеля IVP заключается в сокращении размещений в эти инвестиции и рассеивании прежнего риска на все другие инвестиции. Между пятым и шестым ребалансированием две инвестиции находятся под воздействием общего шока. Реакция портфеля IVP такая же. По этой причине размещения между семью незатронутыми инвестициями с течением времени растут, независимо от их соотношения

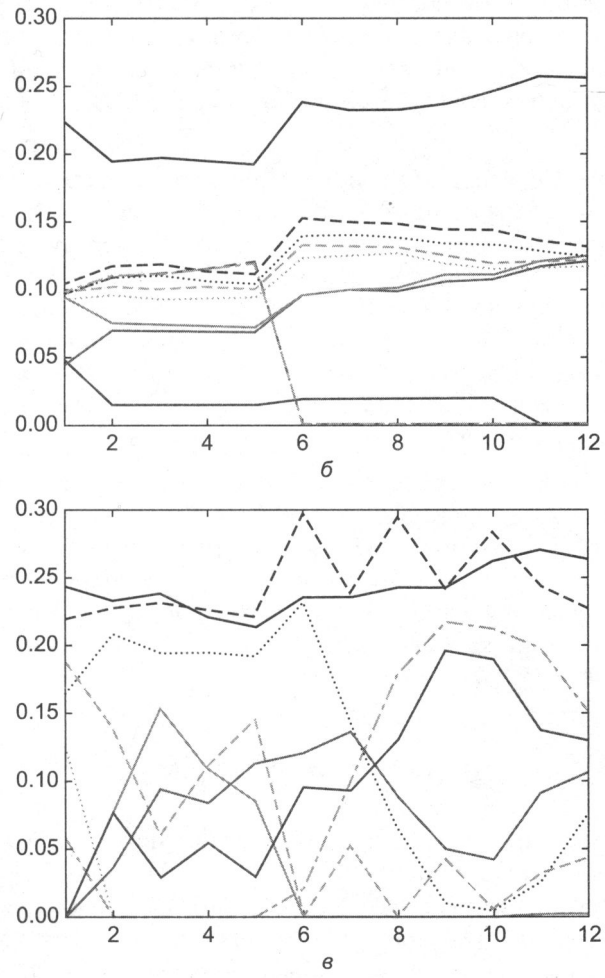


Рис. 16.7 (*продолжение*)

б) временные ряды размещений для портфеля HRP.

Ответ портфеля HRP на идиосинкразический шок заключается в сокращении размещений в затронутые инвестиции и использовании этой сокращенной суммы для увеличения размещений в коррелированные инвестиции, которые не были затронуты. В ответ на общий шок портфель HRP сокращает размещения в затронутые инвестиции и увеличивает размещения в некоррелированные инвестиции (с меньшей дисперсией)

в) временные ряды размещений для портфеля CLA.

Размещения портфеля CLA реагируют беспорядочно на идиосинкразические и общие шоки. Если бы мы учли издержки на ребалансирование, то результативность портфеля CLA была бы очень отрицательной

структуры. Портфель HRP обеспечивает более эффективную защиту как от общих, так и от специфических шоков, находя компромисс между диверсификацией всех инвестиций и диверсификацией по кластерам инвестиций на нескольких иерархических уровнях. На рис. 16.7 показаны временные ряды размещений для первого из 10 000 прогонов.

Приложение 16.A.4 предоставляет исходный код на языке Python, который реализует вышеупомянутое исследование. Читатель может поэкспериментировать с разными конфигурациями параметров и сделать аналогичные выводы. В частности, вневыборочное превосходство портфеля HRP становится еще более существенным для более крупных инвестиционных универсумов, или когда добавляется больше шоков, или рассматривается более сильная корреляционная структура, или учитываются издержки на ребалансировку. Каждая из этих ребалансировок портфеля CLA порождает транзакционные издержки, которые с течением времени могут накапливаться до непомерно высоких убытков.

В частности, эффективность метода иерархического паритета рисков за пределами выборки становится еще более существенной для более крупных инвестиционных областей, при добавлении новых шоков, при использовании более сильной корреляционной структуры или с учетом повторной балансировки расходов. Каждая подобная повторная балансировка методом критических линий сопряжена с расходами на транзакции, которые могут аккумулировать недопустимые потери с течением времени.

16.7. Дальнейшие исследования

Представленная в этой главе методология является гибкой, масштабируемой и допускает несколько вариаций одних и тех же идей. Используя предоставленный исходный код, читатели могут исследовать и оценивать то, какие конфигурации метода HRP лучше всего подходят для их конкретной задачи. Например, на этапе 1 они могут применить альтернативные определения $d_{i,j}$, $\tilde{d}_{i,j}$ и $\dot{d}_{i,u}$ или различные алгоритмы кластеризации, такие как бикластеризация; на этапе 3 они могут использовать различные функции для $\tilde{w}_m$ и α или альтернативные ограничения на размещения. Вместо рекурсивного дробления пополам (бисекции) этап 3 может также дробить размещения сверху вниз, используя кластеры из этапа 1.

Относительно просто встроить прогнозируемые финансовые возвраты, сжатие по Ледуа—Вульфа и взгляды в стиле Блэка—Литтермана на иерархический подход. На самом деле, любознательный читатель, вполне возможно, понял, что по своей сути метод HRP представляет собой робастную процедуру, позволяющую избегать матричных инверсий, и те же идеи, лежащие в основе метода HRP, могут быть использованы для замены многих эконометрических регрессионных методов, известных своими нестабильными результатами (например, VAR или VECM). На рис. 16.8 показана: а) большая корреляционная матрица ценных бумаг с фиксиро-

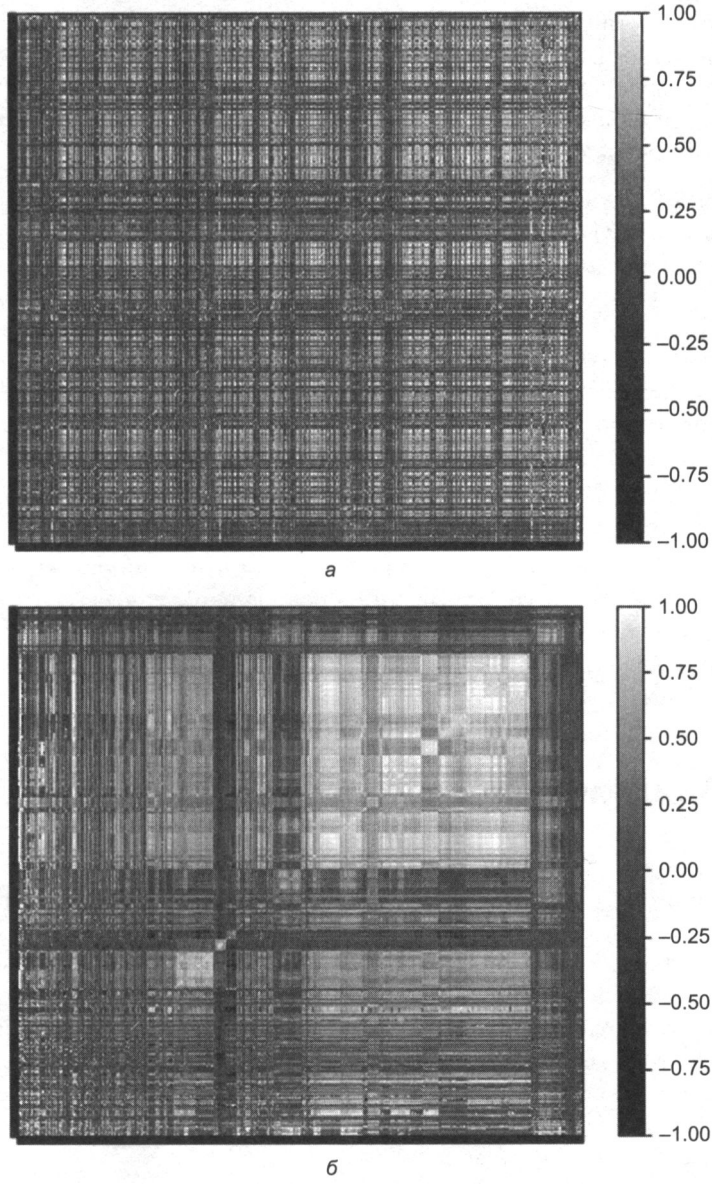


Рис. 16.8. Корреляционная матрица до и после кластеризации.

Методология, описанная в этой главе, может быть применена к задачам, выходящим за рамки оптимизации. Например, анализ PCA большого универсума с фиксированным доходом страдает теми же недостатками, которые мы описали для CLA. Методы с использованием малых данных, разработанные десятилетия и столетия назад (факторные модели, регрессионный анализ, эконометрия), не способны распознавать иерархический характер финансовых больших данных

ванной процентной ставкой до кластеризации и б) после кластеризации с более чем 2,1 млн записей. Традиционные методы оптимизации или эконометрические методы не справляются с распознаванием иерархической структуры финансовых больших данных, где числовые нестабильности подрывают преимущества анализа, что приводит к ненадежным и вредным исходам.

В публикации Kolanovic и соавт. [2017] авторы провели продолжительное исследование метода HRP и пришли к выводу, что «портфель HRP обеспечивает превосходные скорректированные на риск финансовые возвраты. В то время как оба портфеля, и HRP, и MV (минимально дисперсный), обеспечивают самую высокую доходность, портфели HRP соответствуют целям волатильности намного лучше, чем портфели MV. Мы также провели симуляционные исследования, подтверждающие робастность наших результатов, в которых портфель HRP последовательно обеспечивает превосходную результативность по сравнению с портфелем MV и другими риск-ориентированными стратегиями <...> Портфели HRP действительно диверсифицированы с более высоким числом некоррелированных внешних воздействий и менее экстремальными весами и рисковыми размещениями».

В публикации Raffinot [2017] делается вывод о том, что «эмпирические результаты свидетельствуют о том, что портфели на основе иерархической кластеризации являются робастными, по-настоящему диверсифицированными и достигают статистически более высоких скорректированных на риск результативностей, чем широко используемые методы портфельной оптимизации».

16.8. Заключение

Точные аналитические решения могут показывать гораздо худшую результативность, чем приближенные решения на основе машинного обучения. Хотя математически правильные квадратические оптимизаторы в целом и алгоритм критической линии (CLA) Марковица в частности, как известно, в общем случае обеспечивают ненадежные решения из-за их нестабильности, концентрации и пониженной результативности. Ключевая причина этих проблем заключается в том, что квадратические оптимизаторы требуют инверсии ковариационной матрицы. Проклятие Марковица заключается в том, что чем больше инвестиции коррелированны, тем больше потребность в диверсифицированном портфеле и тем больше ошибки оценивания этого портфеля.

В этой главе мы выявили основной источник нестабильности квадратических оптимизаторов: матрица размера N связана с полным графом с $\frac{1}{2}N(N-1)$ ребрами.

При таком числе ребер, соединяющих узлы графа, весам разрешено балансировать с полной свободой. Это отсутствие иерархической структуры означает, что малые ошибки оценивания приведут к совершенно другим решениям. Метод HRP за-

меняет ковариационную структуру древовидной структурой, достигая трех целей: 1) в отличие от традиционных методов паритета рисков, он полностью использует информацию, содержащуюся в ковариационной матрице, 2) восстанавливается стабильность весов и 3) решение интуитивно по конструкции. Алгоритм сходится за детерминированное логарифмическое (лучший случай) или линейное (худший случай) время.

Метод HRP является робастным, визуальным и гибким, позволяя пользователю вводить ограничения или манипулировать древовидной структурой без ущерба для алгоритмического поиска. Эти свойства вытекают из того, что метод HRP не требует ковариационной обратимости. Действительно, метод HRP может вычислять портфель на (плохо) вырожденной или даже сингулярной ковариационной матрице.

В центре внимания этой главы находится конструирование портфеля; однако читатель найдет другие практические применения для принятия решений в условиях неопределенности, в особенности в присутствии почти сингулярной ковариационной матрицы: размещение портфельных менеджеров, размещение по всем алгоритмическим стратегиям, бэггирование и бустирование сигналов машинно-обучающихся алгоритмов, прогнозы из случайных лесов, замена неустойчивых эконометрических моделей (модель векторной авторегрессии VAR, векторная модель коррекции ошибок VECM) и т. д.

Конечно, квадратические оптимизаторы, такие как алгоритм критической линии (CLA), производят минимально дисперсный портфель внутривыборочно (то есть его целевую функцию). Эксперименты Монте-Карло показывают, что метод HRP обеспечивает более низкую вневыборочную дисперсию, чем алгоритм CLA или традиционные методы паритета рисков (инверсно-дисперсный портфель IVP). С тех пор как компания Bridgewater впервые ввела паритет рисков в 1990-х годах, некоторые крупнейшие менеджеры активов запустили фонды, которые следуют этому подходу, с общими активами свыше 500 миллиардов долларов. Учитывая широкое использование ими кредитного плеча, этим фондам следует извлечь выгоду из принятия более стабильного метода размещения на основе паритета рисков, который позволит им достичь более высоких скорректированных на риск финансовых возвратов и более низких издержек на ребалансировку.

Дополнение

16.A.1. Корреляционный метрический показатель

Рассмотрим два вещественно-значных вектора X и Y размером T и корреляционную переменную $\rho[x, y]$ с единственным условием, что $\sigma[x, y] = \rho[x, y]\sigma[X]\sigma[Y]$, где

$\sigma[x, y]$ — это ковариация (совместная дисперсия) между двумя векторами, и $\sigma[.]$ — среднеквадратическое отклонение. Обратите внимание, что корреляция Пирсона не единственная корреляция, удовлетворяющая этим требованиям.

Докажем, что $d[x, y] = \sqrt{\frac{1}{2}(1-\rho[x,y])}$ является истинным метрическим показателем. Во-первых, евклидово расстояние между двумя векторами равно $d[x, y] = \sqrt{\sum_{t=1}^{T}(X_t - Y_t)^2}$. Во-вторых, мы z-стандартизируем эти векторы как $x = \frac{X - \bar{X}}{\sigma[X]}$, $y = \frac{Y - \bar{Y}}{\sigma[Y]}$. Следовательно, $0 \le \rho[x, y] = \rho[x, y]$. В-третьих, мы получаем евклидово расстояние $d[x, y]$ как

$$d[x,y] = \sqrt{\sum_{t=1}^{T}(x_t - y_t)^2} = \sqrt{\sum_{t=1}^{T}x_t^2 + \sum_{t=1}^{T}y_t^2 - 2\sum_{t=1}^{T}x_t y_t} =$$

$$= \sqrt{T + T - 2T\sigma[x,y]} = \sqrt{2T\left(1 - \underbrace{\rho[x,y]}_{=\rho[x,y]}\right)} = \sqrt{4T}d[x,y].$$

Другими словами, расстояние $d[x, y]$ является линейным кратным евклидову расстоянию между векторами $\{X, Y\}$ после z-стандартизации, следовательно, оно наследует истинные метрические свойства евклидова расстояния.

Точно так же мы можем доказать, что $d[x,y] = \sqrt{1-|\rho[x,y]|}$ снижается до истинного метрического показателя на частном $\mathbb{Z}/2\mathbb{Z}$. Для этого мы переопределяем $y = \frac{Y - \bar{Y}}{\sigma[Y]}\text{sgn}[\rho[x, y]]$, где $\text{sgn}[.]$ — это знаковый оператор, такой что $0 \le \rho[x, y] = |\rho[x, y]|$. Тогда

$$d[x,y] = \sqrt{2T\left(1 - \underbrace{\rho[x,y]}_{=|\rho[x,y]|}\right)} = \sqrt{2T}d[x,y].$$

16.A.2. Инверсно-дисперсное размещение

Этап 3 (см. раздел 16.4.3) дробит вес в обратной пропорции к дисперсии подмножества. Теперь мы докажем, что такое размещение оптимально, когда ковариационная матрица диагональна. Рассмотрим стандартную квадратическую оптимизационную задачу размера N,

$$\min_{\omega} \omega' V \omega$$

$$\text{так что } \omega' a = 1_I$$

с решением $\omega = \frac{V^{-1}a}{a'V^{-1}a}$. Для характеристического вектора $a = 1_N$ решением является минимально дисперсный портфель. Если V является диагональной, то $\omega_n = \frac{V_{n,n}^{-1}}{\sum_{i=1}^{N} V_{i,i}^{-1}}$. В частном случае $N = 2$ $\omega_1 = \frac{\frac{1}{V_{1,1}}}{\frac{1}{V_{1,1}} + \frac{1}{V_{2,2}}} = 1 - \frac{\frac{1}{V_{1,1}}}{\frac{1}{V_{1,1}} + \frac{1}{V_{2,2}}}$, то есть то, как этап 3 дробит вес между двумя половинами подмножества.

16.A.3. Воспроизведение численного примера

Листинг 16.4 можно использовать для воспроизведения наших результатов и симулирования дополнительных численных примеров. Функция `generateData` производит матрицу временных рядов, где число `size0` векторов не коррелировано, а число `size1` векторов коррелировано. Читатель может изменить посев `np.random.seed` в функции `generateData` для запуска альтернативных примеров и получения представления о том, как работает метод HRP. Функция `linkage` библиотеки scipy может использоваться для выполнения этапа 1 (раздел 16.4.1), функция `getQuasiDiag` выполняет этап 2 (раздел 16.4.2) и функция `getRecBipart` выполняет этап 3 (раздел 16.4.3).

Листинг 16.4. Полная реализация алгоритма иерархического паритета рисков (HRP)

```
import matplotlib.pyplot as mpl
import scipy.cluster.hierarchy as sch,random,numpy as np,pandas as pd
#------------------------------------
def getIVP(cov,**kargs):
    # Вычислить инверсно-дисперсный портфель
    ivp=1./np.diag(cov)
    ivp/=ivp.sum()
    return ivp
#------------------------------------
def getClusterVar(cov,cItems):
    # Вычислить дисперсию в расчете на кластер
    cov_=cov.loc[cItems,cItems] # matrix slice
    w_=getIVP(cov_).reshape(-1,1)
    cVar=np.dot(np.dot(w_.T,cov_),w_)[0,0]
    return cVar
#------------------------------------
def getQuasiDiag(link):
```

```
    # Отсортировать кластеризованные элементы по расстоянию
    link=link.astype(int)
    sortIx=pd.Series([link[-1,0],link[-1,1]])
    numItems=link[-1,3] # число исходных элементов
    while sortIx.max()>=numItems:
        sortIx.index=range(0,sortIx.shape[0]*2,2) # создать пространство
        df0=sortIx[sortIx>=numItems] # отыскать кластеры
        i=df0.index;j=df0.values-numItems
        sortIx[i]=link[j,0] # элемент 1
        df0=pd.Series(link[j,1],index=i+1)
        sortIx=sortIx.append(df0) # элемент 2
        sortIx=sortIx.sort_index() # пересортировать
        sortIx.index=range(sortIx.shape[0]) # реиндексировать
    return sortIx.tolist()
#-------------------------------------
def getRecBipart(cov,sortIx):
    # Вычислить выделение капитала портфелем HRP
    w=pd.Series(1,index=sortIx)
    cItems=[sortIx] # инициализировать все элементы в одном кластере
    while len(cItems)>0:
        cItems=[i[j:k] for i in cItems for j,k in ((0,len(i)/2), \
            (len(i)/2,len(i))) if len(i)>1] # дробление на две части
        for i in xrange(0,len(cItems),2): # выполнить разбор в парах
            cItems0=cItems[i] # кластер 1
            cItems1=cItems[i+1] # кластер 2
            cVar0=getClusterVar(cov,cItems0)
            cVar1=getClusterVar(cov,cItems1)
            alpha=1-cVar0/(cVar0+cVar1)
            w[cItems0]*=alpha # вес 1
            w[cItems1]*=1-alpha # вес 2
    return w
#-------------------------------------
def correlDist(corr):
    # Матрица расстояний на основе корреляции, где 0<=d[i,j]<=1
    # Это собственный метрический показатель расстояния
    dist=((1-corr)/2.)**.5 # матрица расстояний
    return dist
#-------------------------------------
def plotCorrMatrix(path,corr,labels=None):
    # Теплокарта корреляционной матрицы
    if labels is None: labels=[]
    mpl.pcolor(corr)
    mpl.colorbar()
    mpl.yticks(np.arange(.5,corr.shape[0]+.5),labels)
    mpl.xticks(np.arange(.5,corr.shape[0]+.5),labels)
    mpl.savefig(path)
    mpl.clf();mpl.close() # сбросить объект pylab
    return
#-------------------------------------
def generateData(nObs,size0,size1,sigma1):
    # Временной ряд коррелированных переменных
    #1) генерирование некоррелированных данных
```

```
    np.random.seed(seed=12345);random.seed(12345)
    x=np.random.normal(0,1,size=(nObs,size0)) # каждая строка - это переменная
    #2) создание корреляции между переменными
    cols=[random.randint(0,size0-1) for i in xrange(size1)]
    y=x[:,cols]+np.random.normal(0,sigma1,size=(nObs,len(cols)))
    x=np.append(x,y,axis=1)
    x=pd.DataFrame(x,columns=range(1,x.shape[1]+1))
    return x,cols
#-----------------------------------
def main():
    #1) сгенерировать коррелированные данные
    nObs,size0,size1,sigma1=10000,5,5,.25
    x,cols=generateData(nObs,size0,size1,sigma1)
    print [(j+1,size0+i) for i,j in enumerate(cols,1)]
    cov,corr=x.cov(),x.corr()
    #2) вычислить и построить график корреляционной матрицы
    plotCorrMatrix('HRP3_corr0.png',corr,labels=corr.columns)
    #3) кластеризовать
    dist=correlDist(corr)
    link=sch.linkage(dist,'single')
    sortIx=getQuasiDiag(link)
    sortIx=corr.index[sortIx].tolist() # восстановить метки
    df0=corr.loc[sortIx,sortIx] # переупорядочить
    plotCorrMatrix('HRP3_corr1.png',df0,labels=df0.columns)
    #4) размещение капитала
    hrp=getRecBipart(cov,sortIx)
    print hrp
    return
#-----------------------------------
if __name__=='__main__':main()
```

16.A.4. Воспроизведение эксперимента Монте-Карло

Листинг 16.5 реализует эксперименты Монте-Карло на трех методах размещения: HRP, CLA и IVP. Все библиотеки являются стандартными, за исключением HRP, которая приводится в приложении 16.A.3, и CLA, которую можно найти в публикации Bailey and Lopez de Prado [2013]. Функция `generateData` симулирует коррелированные данные с двумя типами случайных шоков: общими для разных инвестиций и специфичными для одной инвестиции. Есть два шока каждого типа: один положительный и один отрицательный. Переменные для экспериментов задаются в качестве аргументов `hrpMC`. Они были выбраны произвольно, и пользователь может поэкспериментировать с альтернативными сочетаниями.

Листинг 16.5. Эксперимент Монте-Карло с вневыборочной результативностью метода иерархического паритета рисков HRP

```
import scipy.cluster.hierarchy as sch,random,numpy as np,pandas as pd,CLA
from HRP import correlDist,getIVP,getQuasiDiag,getRecBipart
#-----------------------------------
```

```
def generateData(nObs,sLength,size0,size1,mu0,sigma0,sigma1F):
    # Временной ряд коррелированных переменных
    #1) сгенерировать случайные некоррелированные данные
    x=np.random.normal(mu0,sigma0,size=(nObs,size0))
    #2) создать корреляцию между переменными
    cols=[random.randint(0,size0-1) for i in xrange(size1)]
    y=x[:,cols]+np.random.normal(0,sigma0*sigma1F,size=(nObs,len(cols)))
    x=np.append(x,y,axis=1)
    #3) добавить общий случайный шок
    point=np.random.randint(sLength,nObs-1,size=2)
    x[np.ix_(point,[cols[0],size0])]=np.array([[-.5,-.5],[2,2]])
    #4) добавить специфичный случайный шок
    point=np.random.randint(sLength,nObs-1,size=2)
    x[point,cols[-1]]=np.array([-.5,2])
    return x,cols
#-------------------------------------
def getHRP(cov,corr):
    # Сконструировать иерархический портфель
    corr,cov=pd.DataFrame(corr),pd.DataFrame(cov)
    dist=correlDist(corr)
    link=sch.linkage(dist,'single')
    sortIx=getQuasiDiag(link)
    sortIx=corr.index[sortIx].tolist() # восстановить метки
    hrp=getRecBipart(cov,sortIx)
    return hrp.sort_index()
#-------------------------------------
def getCLA(cov,**kargs):
    # Вычислить минимально дисперсионный портфель CLA
    mean=np.arange(cov.shape[0]).reshape(-1,1) # Не используется портфелем CLA
    lB=np.zeros(mean.shape)
    uB=np.ones(mean.shape)
    cla=CLA.CLA(mean,cov,lB,uB)
    cla.solve()
    return cla.w[-1].flatten()
#-------------------------------------
def hrpMC(numIters=1e4,nObs=520,size0=5,size1=5,mu0=0,sigma0=1e-2, \
    sigma1F=.25,sLength=260,rebal=22):
    # Эксперимент Monte Carlo на HRP
    methods=[getIVP,getHRP,getCLA]
    stats,numIter={i.__name__:pd.Series() for i in methods},0
    pointers=range(sLength,nObs,rebal)
    while numIter<numIters:
        print numIter
        #1) подготовить данные для одного эксперимента
        x,cols=generateData(nObs,sLength,size0,size1,mu0,sigma0,sigma1F)
        r={i.__name__:pd.Series() for i in methods}
        #2) вычислить портфели внутривыборочно
        for pointer in pointers:
            x_=x[pointer-sLength:pointer]
            cov_,corr_=np.cov(x_,rowvar=0),np.corrcoef(x_,rowvar=0)
            #3) вычислить результативность вневыборочно
            x_=x[pointer:pointer+rebal]
```

```
            for func in methods:
                w_=func(cov=cov_,corr=corr_) # обратный вызов
                r_=pd.Series(np.dot(x_,w_))
                r[func.__name__]=r[func.__name__].append(r_)
        #4) оценить и сохранить результаты
        for func in methods:
            r_=r[func.__name__].reset_index(drop=True)
            p_=(1+r_).cumprod()
            stats[func.__name__].loc[numIter]=p_.iloc[-1]-1
        numIter+=1
    #5) сообщить результаты
    stats=pd.DataFrame.from_dict(stats,orient='columns')
    stats.to_csv('stats.csv')
    df0,df1=stats.std(),stats.var()
    print pd.concat([df0,df1,df1/df1['getHRP']-1],axis=1)
    return
#---------------------------------------
if __name__=='__main__':hrpMC()
```

Упражнения

16.1. При заданном ряде прибылей и убытков PnL на N инвестиционных стратегиях:

(а) Выровняйте их по средней частоте их ставок (например, еженедельные наблюдения для стратегий, которые торгуют на еженедельной основе). Подсказка: такое выравнивание данных иногда называется «отбором с пониженной частотой».

(б) Вычислите ковариацию (совместную дисперсию) их финансовых возвратов, V.

(в) Идентифицируйте иерархические кластеры среди N стратегий.

(г) Постройте график кластеризированной корреляционной матрицы из N стратегий.

16.2. Используя кластеризированную корреляционную матрицу V из упражнения 16.1:

(а) Вычислите размещения по методу HRP.

(б) Вычислите размещения по методу CLA.

(в) Вычислите размещения по методу IVP.

16.3. Используя ковариационную матрицу V из упражнения 16.1:

(а) Выполните спектральное разложение $VW = W\Lambda$.

(б) Сформируйте массив ε, вынув N случайных чисел из распределения $U[0,1]$.

(в) Сформируйте $N \times N$ матрицу $\tilde{\Lambda}$, где $\tilde{\Lambda}_{n,n} = N\varepsilon_n \Lambda_{n,n} \left(\sum_{n=1}^{N} \varepsilon_n\right)^{-1}$, $n = 1, ..., N$.

(г) Вычислите $\tilde{V} = W\tilde{\Lambda}W^{-1} - 1$.

(д) Повторите упражнение 16.2, используя $\tilde{V}$ в качестве ковариационной матрицы. На какой метод размещения больше всего повлияло перешкалирование спектральных дисперсий?

16.4. Как бы вы изменили метод HRP для того, чтобы произвести размещения, которые в сумме составляют 0, где $|w_n| \leq 1$, $\forall n = 1, ..., N$?

16.5. Можете ли вы придумать простой способ встраивания ожидаемых финансовых возвратов в размещения по методу HRP?

Часть 4
ПОЛЕЗНЫЕ ФИНАНСОВЫЕ ПРИЗНАКИ

17 Структурные сдвиги

17.1. Актуальность

При разработке инвестиционной стратегии на основе машинного обучения мы, как правило, хотим делать ставки, когда происходит слияние факторов, предсказанный исход которых обеспечивает благоприятный скорректированный на риск финансовый возврат. Структурные сдвиги, как и переход от одного рыночного режима к другому, являются одним из примеров такого слияния, и он представляет особый интерес. Например, закономерность возвращения к среднему значению может смениться импульсной закономерностью. Когда этот переход происходит, большинство участников рынка оказываются застигнутыми врасплох и будут делать дорогостоящие ошибки. Такого рода ошибки являются основой для многих прибыльных стратегий, потому что игроки на проигравшей стороне, как правило, осознают свою ошибку, когда уже слишком поздно. Прежде чем они акцептуют свои убытки, они будут действовать иррационально, пытаясь владеть позицией и надеясь на возвращение. Иногда в отчаянии они даже увеличивают проигрышную позицию. В конце концов, они будут вынуждены остановить убыток или принудительно выйти из игры. Структурные сдвиги предлагают одно из самых лучших соотношений риск/вознаграждение[1]. В этой главе мы рассмотрим несколько методов, которые измеряют вероятность структурных сдвигов с целью построения на их основе информативных признаков.

17.2. Типы проверок на структурные сдвиги

Мы можем отнести проверки на структурные сдвиги к двум общим категориям:

- **Проверки на основе фильтра CUSUM** (CUSUM test): они проверяют, значительно ли отклоняются кумулятивные ошибки прогнозирования от белого шума.

[1] Структурный сдвиг (structural break), или изменение, или разрыв, представляет собой неожиданный сдвиг во временно́м ряде, который может привести к огромным ошибкам предсказания и ненадежности модели в целом. — *Примеч. науч. ред.*

- **Проверки взрываемости** (explosiveness test): помимо отклонения от белого шума, эти проверки оценивают, показывает ли процесс экспоненциальный рост или коллапс, так как это несовместимо со случайным блужданием или стационарным процессом и не может сохраняться в таком состоянии длительное время.
- **Правохвостные единично-корневые проверки** (right-tail unit-root test): эти проверки оценивают наличие экспоненциального роста или коллапса, исходя при этом из авторегрессионной спецификации.
- **Суб- и супермартингейловые проверки** (sub/super-martingale test): эти проверки оценивают наличие экспоненциального роста или коллапса в рамках разнообразных функциональных форм.

17.3. Проверки на основе фильтра CUSUM

В главе 2 мы представили фильтр CUSUM, который мы применили в контексте событийно-управляемого отбора баров. Идея заключалась в том, чтобы отбирать бар всякий раз, когда какая-то величина, например кумулятивные предсказательные ошибки, превысила предопределенный порог. Эта идея может быть расширена на проверки наличия структурных сдвигов.

17.3.1. Проверка CUSUM Брауна—Дарбина—Эванса на рекурсивных остатках

Эта проверка была предложена в публикации Brown, Durbin and Evans [1975]. Будем считать, что в каждом наблюдении $t = 1, ..., T$ мы ведем подсчет с помощью массива признаков x_t, предсказывающих значение y_t. Матрица X_t составлена из временного ряда признаков $t \leq T$, $\{x_i\}_{i=1,...,t}$. Упомянутые авторы предлагают вычислять оценки β рекурсивных наименьших квадратов (recursive least squares, RLS), основываясь на спецификации

$$y_t = \beta_t' x_t + \varepsilon_t,$$

которая подгоняется на подвыборках ($[1, k + 1]$, $[1, k + 2]$, ... , $[1, T]$), давая $T - k$ оценок по методу наименьших квадратов $(\hat{\beta}_{k+1}, ..., \hat{\beta}_T)$. Мы можем вычислить стандартизированные рекурсивные остатки с забеганием на 1 шаг вперед как:

$$\hat{\omega}_t = \frac{y_t - \hat{\beta}_{t-1}' x_t}{\sqrt{f_t}};$$
$$f_t = \hat{\sigma}_\varepsilon^2 \left[1 + x_t' (X_t' X_t)^{-1} x_t\right].$$

Статистический показатель CUSUM определяется как

$$S_t = \sum_{j=k+1}^{t} \frac{\hat{\omega}_j}{\hat{\sigma}_\omega}$$
$$\hat{\sigma}_\omega^2 = \frac{1}{T-k}\sum_{t=k}^{T}(\hat{\omega}_t - \mathrm{E}[\hat{\omega}_t])^2.$$

Согласно нулевой гипотезе, если β — это некая постоянная величина, $H_0 : \beta_t = \beta$, то $S_t \sim N[0, t-k-1]$. Один нюанс этой процедуры заключается в том, что отправная точка выбирается совершенно произвольно, и вследствие этого результаты могут быть противоречивы.

17.3.2. Проверка CUSUM Чу—Стинчкомба—Уайта на уровнях

Эта проверка вытекает из публикации Homm and Breitung [2012]. Она упрощает предыдущий метод, отбрасывая $\{x_t\}_{t=1,\ldots,T}$ и исходя из того, что $H_0 : \beta_t = 0$, то есть мы не прогнозируем никаких изменений ($\mathrm{E}_{t-1}[\Delta y_t] = 0$). Это позволит нам работать непосредственно с y_t уровнями, тем самым снижая вычислительную нагрузку. Мы вычисляем стандартизированный отход логарифмической цены y_t относительно логарифмической цены y_n, $t > n$ как

$$S_{n,t} = (y_t - y_n)(\hat{\sigma}_t\sqrt{t-n})^{-1};$$
$$\hat{\sigma}_t^2 = (t-1)^{-1}\sum_{i=2}^{t}(\Delta y_i)^2.$$

Согласно нулевой гипотезе $H_0 : \beta_t = 0$, тогда $S_{n,t} \sim N[0, 1]$. Зависимое от времени критическое значение для *односторонней проверки* равно

$$c_a[n,t] = \sqrt{b_a + \log[t-n]}.$$

Упомянутые авторы посредством метода Монте-Карло вывели, что $b_{0,05} = 4{,}6$. Одним из недостатков этого метода является то, что опорный уровень y_n задается несколько произвольно. Для того чтобы преодолеть эту ловушку, мы могли бы оценивать $S_{n,t}$ на ряде из обратно сдвигающихся окон $n \in [1, t]$ и подбирать $S_t = \sup_{n\in[1,t]}[S_{n,t}]$.

17.4. Проверки взрываемости

Проверки взрываемости могут быть в целом разделены на проверки, которые проверяют наличие одного пузыря, и проверки, которые проверяют наличие нескольких пузырей. В этом контексте пузыри не ограничиваются ценовыми ралли, но

и включают активные распродажи. Проверки, которые допускают многочисленные пузыри, являются более робастными, в том смысле что цикл пузырь — схлопывание — пузырь приводит к тому, что ряд выглядит стационарным для проверки с единственным пузырем. В публикациях Maddala and Kim [1998] и Breitung [2014] предлагаются хорошие обзоры литературы по этой теме.

17.4.1. Проверка Дики—Фуллера по типу Чоу

Семейство проверок взрываемости было обусловлено работой Грегори Чоу начиная с публикации Chow [1960]. Рассмотрим авторегрессионный процесс первого порядка

$$y_t = \rho y_{t-1} + \varepsilon_t,$$

где ε_t — это белый шум. Нулевая гипотеза заключается в том, что y_t подчиняется случайному блужданию, H_0: $\rho = 1$, и альтернативная гипотеза заключается в том, что y_t начинается как случайное блуждание, но изменяется в момент времени $\tau^* T$, где $\tau^* \in (0, 1)$, во взрывной процесс:

$$H_1 : y_t = \begin{cases} y_{t-1} + \varepsilon_t \text{ для } t = 1, ..., \tau^* T \\ \rho y_{t-1} + \varepsilon_t \text{ для } t = \tau^* T + 1, ..., T \text{ при } \rho > 1. \end{cases}$$

В момент времени T мы можем провести тест на переход (от случайного блуждания к процессу резкого подъема), который случается в момент времени τT (время разрыва). Чтобы проверить эту гипотезу, нам необходимо применить следующее правило:

$$\Delta y_t = \delta y_{t-1} D_t[\tau^*] + \varepsilon_t,$$

где $D_t[\tau^*]$ — это фиктивная переменная, которая принимает нулевое значение, если $t < \tau^* T$, и принимает единичное значение, если $t \geq \tau^* T$. Затем нулевая гипотеза H_0: $\delta = 0$ проверяется относительно (односторонней) альтернативы $H_1 : \delta > 1$:

$$DFC_{\tau^*} = \frac{\hat{\delta}}{\hat{\sigma}_\delta}.$$

Главным недостатком этого метода является то, что τ^* неизвестно. Для решения этой проблемы в публикации Andrews [1993] была предложена новая проверка, в которой все возможные τ^* опробываются внутри некоторого интервала $\tau^* \in \in [\tau_0, 1 - \tau_0]$. Как объясняется в публикации Breitung [2014], мы должны опустить некоторые из возможных τ^* в начале и в конце выборки, чтобы гарантировать, что любой из режимов был подогнан достаточным числом наблюдений (в $D_t[\tau^*]$

должно быть достаточно нулей и достаточно единиц). Проверочный статистический показатель для неизвестного τ^* является максимумом всех $T(1 - 2\tau_0)$ значений $DFC_{\tau\bullet}$.

$$SDFC = \sup_{\tau^* \in [\tau_0, 1-\tau_0]} \{DFC_{\tau\bullet}\}.$$

Еще одним недостатком подхода Чоу является то, что данный подход исходит из того, что существует только одна дата сдвига $\tau^* T$ и что пузырь продолжается до конца выборки (обратное переключение на случайное блуждание отсутствует). Для ситуаций, когда существует три или более режимов (случайное блуждание → пузырь → случайное блуждание...), нам нужно обсудить проверку SADF (то есть дополненную супремумом проверку Дики—Фуллера).

17.4.2. Супремально расширенный тест Дики—Фуллера

По словам авторов публикации Phillips, Wu and Yu [2011], «стандартные единичнокорневые и коинтеграционные проверки являются неподходящими инструментами для обнаружения поведения пузырей, поскольку они не могут эффективно различать стационарный процесс и периодически коллапсирующую модель пузырей. Закономерности периодически коллапсирующих пузырей в данных больше похожи на данные, сгенерированные из единичного корня или стационарной авторегрессии, чем из потенциально взрывного процесса». Для того чтобы устранить этот недостаток, упомянутые авторы предлагают выполнять подгонку регрессионной спецификации

$$\Delta y_t = \alpha + \beta y_{t-1} + \sum_{l=1}^{L} \gamma_l \Delta y_{t-l} + \varepsilon_t,$$

где мы выполняем проверку гипотез $H_0 : \beta \le 0$, $H_1 : \beta > 0$. Руководствуясь работой Andrews [1993], авторы публикаций Phillips and Yu [2011] и Phillips, Wu and Yu [2011] предложили дополненную супремумом проверку Дики—Фуллера (supremum augmented Dickey-Fuler, SADF). Проверка SADF выполняет подгонку приведенной выше регрессии в каждой конечной точке t с расширяющимися назад начальными точками, а затем вычисляет

$$SADF_t = \sup_{t_0 \in [1, t-\tau]} \{ADF_{t_0, t}\} = \sup_{t_0 \in [1, t-\tau]} \left\{ \frac{\hat{\beta}_{t_0, t}}{\hat{\sigma}_{\hat{\beta}_{t_0, t}}} \right\},$$

где $\hat{\beta}_{t_0, t}$ оценивается на выборке, которая начинается в t_0 и заканчивается в t, τ — это минимальная длина выборки, используемая в анализе, t_0 — левая граница расширяющегося назад окна и $t = \tau, ..., T$. Для оценивания SADF правая часть окна фиксируется в t. Стандартная проверка ADF является частным случаем SADF, где $\tau = t - 1$.

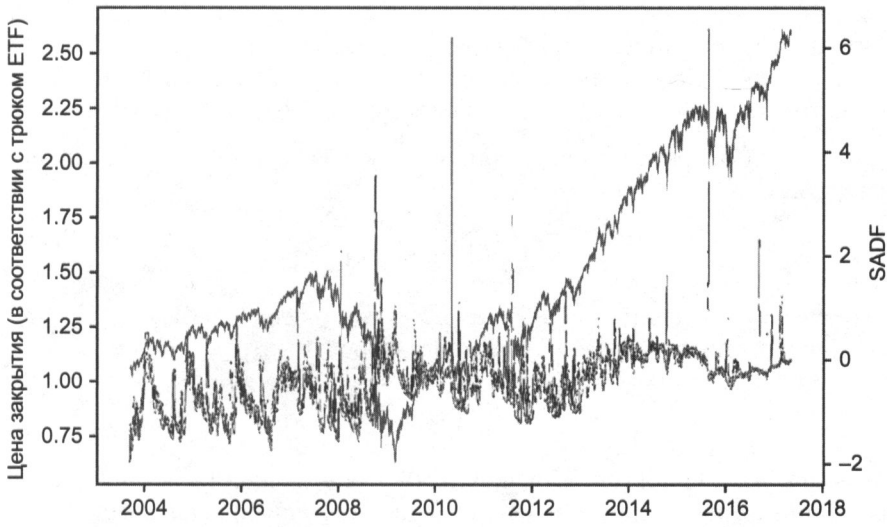


Рис. 17.1. Цены (левая ось y) и проверка SADF во временной динамике

Между методами $SADF_t$ и SADF существует два важных различия: во-первых, метод $SADF_t$ вычисляется на каждом $t \in [\tau, T]$, тогда как метод SDFC вычисляется только на T. Во-вторых, вместо введения фиктивной переменной метод SADF рекурсивно расширяет начало выборки ($t_0 \in [1, t - \tau]$). Пробуя все сочетания вложенного двойного цикла на (t_0, t), метод SADF не исходит из известного числа переключений режимов или дат сдвига. На рис. 17.1 показан ценовой ряд фьючерса E-mini S&P 500 после применения трюка ETF (глава 2, раздел 2.4.1), а также результаты проверки SADF, полученные из этого ценового ряда. Линия SADF подскакивает вверх, когда цены демонстрируют поведение, похожее на пузырь, и возвращается к низким уровням, когда пузырь схлопывается. В следующих разделах мы обсудим некоторые усовершенствования оригинального метода SADF Филлипса.

17.4.2.1. Абсолютные цены против логарифмических цен

В литературе часто можно встретить исследования, которые проводят проверки наличия структурного сдвига на сырых ценах. В этом разделе мы рассмотрим причину, почему следует отдавать предпочтение логарифмическим ценам, в особенности во время работы с длинными временны́ми рядами, включающими пузыри и их схлопывание.

Для сырых цен $\{y_t\}$, если нулевая гипотеза проверки ADF отвергается, то это означает, что цены являются стационарными, с конечной дисперсией. Из этого вытекает, что прибыль $\frac{y_t}{y_{t-1}} - 1$ не является инвариантной ко времени, поскольку волатильность финансовых возвратов должна уменьшаться по мере роста цен и увеличиваться по мере падения цен, с тем чтобы поддерживать ценовую дисперсию

постоянной. Когда мы выполняем проверку ADF на сырых ценах, мы исходим из того, что дисперсия финансовых возвратов не инвариантна к ценовым уровням. Если дисперсия финансовых возвратов оказывается инвариантной к ценовым уровням, то модель будет конструктивно гетероскедастичной.

Напротив, если мы работаем с логарифмическими ценами, то в спецификации проверки ADF будет указано, что

$$\Delta \log[y_t] \propto \log[y_{t-1}].$$

Давайте внесем изменение в переменную $x_t = ky_t$. Теперь $\log[x_t] = \log[k] + \log[y_t]$, и спецификация проверки ADF будет констатировать, что

$$\Delta \log[x_t] \propto \log[y_{t-1}] \propto \log[y_{t-1}].$$

Согласно этой альтернативной спецификации, основанной на логарифмических ценах, ценовые уровни обусловливают среднее значение финансовых возвратов, а не волатильность финансовых возвратов. На практике данная разница может не иметь значения для малых выборок, где $k \approx 1$, но проверка SADF выполняет регрессии на протяжении десятилетий, и пузыри производят уровни, которые существенно различаются между режимами ($k \neq 1$).

17.4.2.2. Вычислительная сложность

Алгоритм выполняется за $O(n^2)$, так как число проверок ADF, которое требуется методом SADF, для общей длины выборки T, равняется

$$\sum_{t=\tau}^{T} t - \tau + 1 = \frac{1}{2}(T - \tau + 2)(T - \tau + 1) = \binom{T - \tau + 2}{2}.$$

Рассмотрим матричное представление спецификации ADF, где $X \in \mathbb{R}^{T \times N}$ и $y \in \mathbb{R}^{T \times 1}$. Решение одной регрессии проверки ADF связано с операциями с плавающей запятой (так называемыми флопами, floating point operations, FlOPs), которые перечислены в табл. 17.1.

В общей сложности это дает $f(N, T) = N^3 + N^2(2T + 3) + N(4T - 1) + 2T + 2$ флопов на одну оценку проверки ADF. Одно обновление SADF требует $g(N, T, \tau) = \sum_{t=\tau}^{T} f(N, t) + T - \tau$ флопов ($T - \tau$ операций для того, чтобы найти максимальный статистический показатель проверки ADF), и оценивание полной серии SADF требует $\sum_{t=\tau}^{T} g(N, T, \tau)$.

Рассмотрим долларовый барный ряд на фьючерсном контракте E-mini S&P 500. Для $(T, N) = (356631.3)$ оценка ADF требует 11 412 245 флопов и обновление SADF требует 2 034 979 648 799 операций (примерно 2.035 терафлопов). Полный времен-

ной ряд SADF требует 241 910 974 617 448 672 операции (примерно 242 петафлопа). Это число будет быстро расти, по мере того как будет продолжать расти T. И эта оценка не включает заведомо дорогостоящие операции, такие как выравнивание, предобработка данных, входные-выходные задания и т. д. Излишне говорить, что двойной цикл этого алгоритма требует большого числа операций. Для оценки рядов проверки SADF в разумные сроки может потребоваться высокопроизводительный вычислительный кластер, эффективно выполняющий параллелизованную реализацию данного алгоритма. В главе 20 будут представлены несколько полезных в таких ситуациях стратегий параллелизации.

Таблица 17.1. Флопы в расчете на оценку ADF

Матричная операция	Флопы
$o_1 = X'y$	$(2T-1)N$
$o_2 = X'X$	$(2T-1)N^2$
$o_3 = o_2^{-1}$	$N^3 + N^2 + N$
$o_4 = o_3 o_1$	$2N^2 - N$
$o_5 = y - Xo_4$	$T + (2N-1)T$
$o_6 = o_5' o_5$	$2T - 1$
$o_7 = o_3 o_6 \frac{1}{T-N}$	$2 + N^2$
$o_8 = \frac{o_4[0,0]}{\sqrt{o_7[0,0]}}$	1

17.4.2.3. Условия экспоненциального поведения

Рассмотрим спецификацию с нулевым лагом для логарифмических цен, $\Delta\log[y_t] = \alpha + \beta\log[y_{t-1}] + \varepsilon_t$. Она может быть переписана как $\log[\tilde{y}_t] = (1+\beta)\log[\tilde{y}_{t-1}] + \varepsilon_t$, где $\log[\tilde{y}_t] = \log[y_t] + \frac{\alpha}{\beta}$. Откатив назад на t дискретных шагов, мы получим $\mathrm{E}[\log[\tilde{y}_t]] = (1+\beta)^t\log[\tilde{y}_0]$ либо $\mathrm{E}[\log[y_t]] = -\frac{\alpha}{\beta} + (1+\beta)^t(\log[y_0] + \frac{\alpha}{\beta})$. Индекс t может быть сброшен в заданный момент времени, для того чтобы спроецировать будущую траекторию $y_0 \to y_t$ после следующих t шагов. Это позволяет выявлять условия, характеризующие три состояния для данной динамической системы:

- Устойчивое: $\beta < 0 \Rightarrow \lim_{t\to\infty}\mathrm{E}[\log[y_t]] = -\frac{\alpha}{\beta}$.
  - Неравновесность равняется $\log[y_t] - (-\frac{\alpha}{\beta}) = \log[\tilde{y}_t]$.

- Тогда $\frac{\mathrm{E}[\log[\tilde{y}_t]]}{\log[\tilde{y}_0]} = (1+\beta)^t = \frac{1}{2}$ при $t = -\frac{\log[2]}{\log[1+\beta]}$ (полураспад).

- Единично-корневое: $\beta = 0$, где система нестационарна и ведет себя как мартингейл.

- Взрывное: $\beta > 0$, где $\lim_{t\to\infty}\mathrm{E}[\log[y_t]] = \begin{cases} -\infty, \text{ если } \log[y_0] < \frac{\alpha}{\beta} \\ +\infty, \text{ если } \log[y_0] > \frac{\alpha}{\beta} \end{cases}$.

17.4.2.4. Квантильная проверка ADF

Проверка SADF берет супремум ряда на t-значениях, $SADF_t = \sup_{t_0 \in [1, t-\tau]}\{ADF_{t_0,t}\}$. Взятие экстремального значения создает некоторые проблемы робастности, при которых оценки SADF могут существенно варьироваться в зависимости от частоты отбора и конкретных временны́х штампов выборок. Более робастный оценщик экстремумов ADF будет следующим. Во-первых, пусть $s_t = \{ADF_{t_0,t}\}_{t_0 \in [0, t_1 - \tau]}$. Во-вторых, мы определяем $Q_{t,q} = Q[s_t, q]$ q-й квантиль s_t как меру центральности высоких значений ADF, где $q \in [0, 1]$. В-третьих, мы определяем $\dot{Q}_{t,q,\nu} = Q_{t,q+\nu} - Q_{t,q-\nu}$ при $0 < \nu \leq \min\{q, 1-q\}$ как меру разброса высоких значений ADF. Например, мы могли бы установить $q = 0.95$ и $\nu = 0.025$. Отметим, что проверка SADF является лишь частным случаем квантильной проверки QADF, где $SADF_t = Q_{t,1}$, а $\dot{Q}_{t,q,\nu}$ не определен, так как $q = 1$.

17.4.2.5. Условная проверка ADF

В качестве альтернативы мы можем решить проблемы устойчивости проверки SADF, вычисляя условные моменты. Пусть $f[x]$ равно функции распределения вероятностей от $s_t = \{ADF_{t_0,t}\}_{t_0 \in [0, t_1 - \tau]}$, с $x \in s_t$. Тогда мы определяем $C_{t,q} = K^{-1}\int_{Q_{t,q}}^{\infty} xf[x]dx$ как меру центральности высоких значений ADF и $\dot{C}_{t,q} = \sqrt{K^{-1}\int_{Q_{t,q}}^{\infty}(x - C_{t,q})^2 f[x]dx}$ как меру разброса высоких значений ADF с регуляризационной постоянной $K = \int_{Q_{t,q}}^{\infty} f[x]dx$. Например, мы могли бы использовать $q = 0.95$.

По конструкции, $C_{t,q} \leq SADF_t$. Диаграмма рассеяния $SADF_t$ относительно $C_{t,q}$ показывает эту нижнюю границу как восходящую линию с приближенно единичным градиентом (см. рис. 17.2). Когда SADF вырастает за пределы −1.5, мы можем оценить некоторые горизонтальные траектории, совместимые с внезапным расширением правого толстого хвоста в s_t. Другими словами, $(SADF_t - C_{t,q})/\dot{C}_{t,q}$ может достигать значительно крупных значений, даже если мера $C_{t,q}$ относительно мала, потому что $SADF_t$ чувствительна к выбросам.

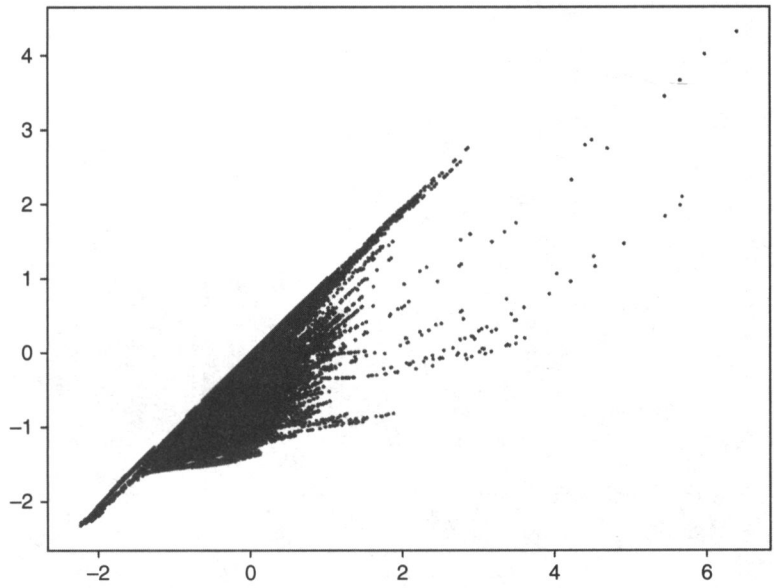


Рис. 17.2. Проверка SADF (ось x) против проверки CADF (ось y)

На рис. 17.3, *а* построен график $(SADF_t - C_{t,q})/\dot{C}_{t,q}$ для цен фьючерса E-mini S&P 500 во временной динамике. На рис. 17.3, *б* показана диаграмма рассеяния $(SADF_t - C_{t,q})/\dot{C}_{t,q}$ относительно $SADF_t$, вычисленная на ценах фьючерса E-mini S&P 500. Данная диаграмма служит доказательством того, что выбросы в s_t смещают $SADF_t$ вверх.

17.4.2.6. Реализация алгоритма SADF

В этом разделе представлена реализация алгоритма SADF. Этот исходный код предназначен не для быстрого вычисления проверки SADF, а для разъяснения шагов его вычисления. В листинге 17.1 приведен внутренний цикл алгоритма SADF: та часть, которая оценивает $SADF_t = \sup_{t_0 \in [1, t-\tau]} \left\{ \frac{\hat{\beta}_{t_0,t}}{\hat{\sigma}_{\hat{\beta}_{t_0,t}}} \right\}$, то есть сдвигающий назад компонент алгоритма. Внешний цикл (здесь не показанный) повторяет это вычисление для продвижения t, $\{SADF_t\}_{t=1,\ldots,T}$. Аргументы алгоритма следующие:

- `logP`: ряд библиотеки pandas, содержащий логарифмические цены.
- `minSL`: минимальная длина выборки (τ), используемая конечной регрессией.
- `constant`: временна́я трендовая компонента регрессии.
  - `'nc'`: временной тренд отсутствует, только константа.
  - `'ct'`: константа плюс линейный временной тренд.

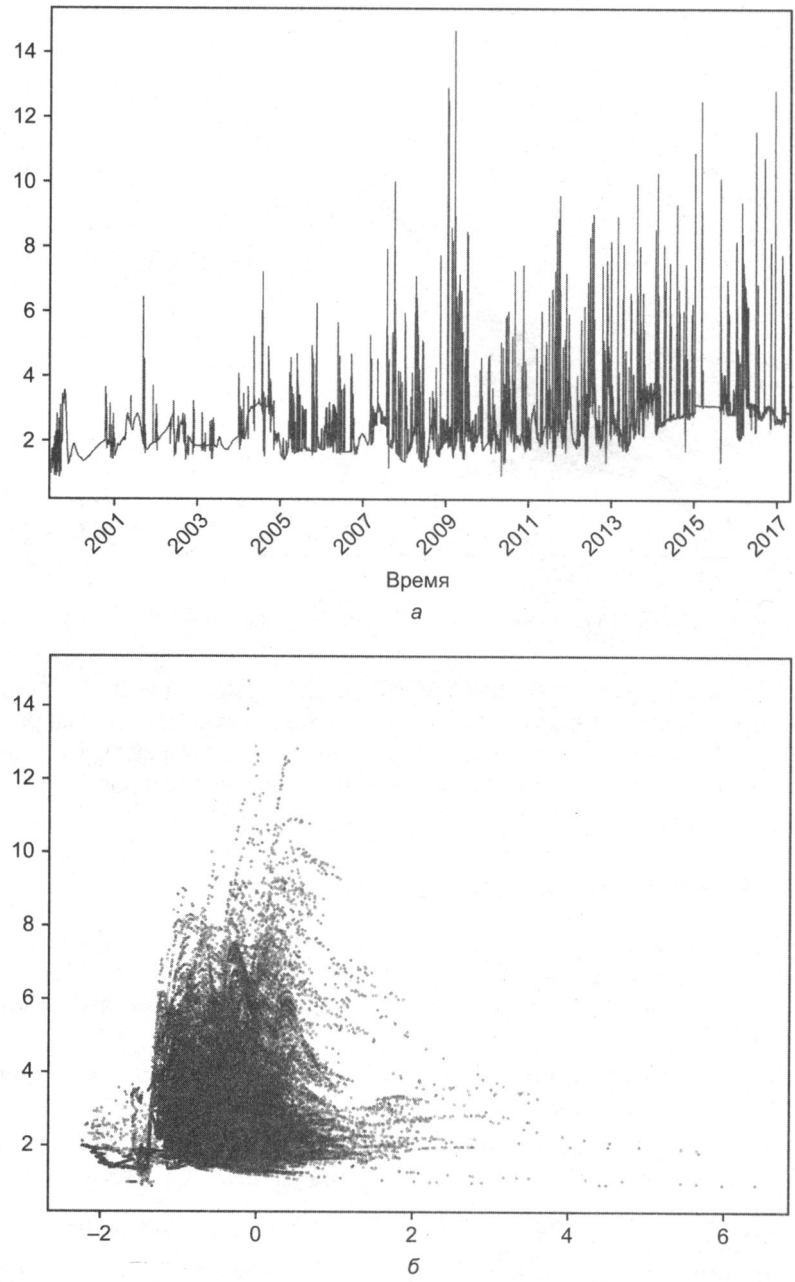


Рис. 17.3. а) $(SADF_t - C_{t,q})/\dot{C}_{t,q}$ во временной динамике; б) $(SADF_t - C_{t,q})/\dot{C}_{t,q}$ (ось y) как функция $SADF_t$ (ось x)

- `'ctt'`: константа плюс квадратичный (2-й степени) полиномиальный временной тренд.

❍ `lags`: число лагов, используемых в спецификации ADF.

Листинг 17.1. Внутренний цикл алгоритма SADF

```
def get_bsadf(logP,minSL,constant,lags):
    y,x=getYX(logP,constant=constant,lags=lags)
    startPoints,bsadf,allADF=range(0,y.shape[0]+lags-minSL+1),None,[]
    for start in startPoints:
        y_,x_=y[start:],x[start:]
        bMean_,bStd_=getBetas(y_,x_)
        bMean_,bStd_=bMean_[0,0],bStd_[0,0]**.5
        allADF.append(bMean_/bStd_)
        if allADF[-1]>bsadf: bsadf=allADF[-1]
    out={'Time':logP.index[-1],'gsadf':bsadf}
    return out
```

В листинге 17.2 приводится функция `getXY`, которая подготавливает объекты numpy, необходимые для проведения рекурсивных проверок.

Листинг 17.2. Подготовка совокупностей данных

```
def getYX(series,constant,lags):
    series_=series.diff().dropna()
    x=lagDF(series_,lags).dropna()
    x.iloc[:,0]=series.values[-x.shape[0]-1:-1,0] # лаговый уровень
    y=series_.iloc[-x.shape[0]:].values
    if constant!='nc':
        x=np.append(x,np.ones((x.shape[0],1)),axis=1)
        if constant[:2]=='ct':
            trend=np.arange(x.shape[0]).reshape(-1,1)
            x=np.append(x,trend,axis=1)
        if constant=='ctt':
            x=np.append(x,trend**2,axis=1)
    return y,x
```

В листинге 17.3 приводится функция `lagDF`, которая применяет к кадру данных лаги (запаздывания), указанные в ее аргументе `lags`.

Листинг 17.3. Применение аргумента lags к кадру данных

```
def lagDF(df0,lags):
    df1=pd.DataFrame()
    if isinstance(lags,int): lags=range(lags+1)
    else: lags=[int(lag) for lag in lags]
    for lag in lags:
        df_=df0.shift(lag).copy(deep=True)
        df_.columns=[str(i)+'_'+str(lag) for i in df_.columns]
        df1=df1.join(df_,how='outer')
    return df1
```

Наконец, в листинге 17.4 приводится функция `getBetas`, которая выполняет фактические регрессии.

Листинг 17.4. Подгонка спецификации проверки ADF

```
def getBetas(y,x):
    xy=np.dot(x.T,y)
    xx=np.dot(x.T,x)
    xxinv=np.linalg.inv(xx)
    bMean=np.dot(xxinv,xy)
    err=y-np.dot(x,bMean)
    bVar=np.dot(err.T,err)/(x.shape[0]-x.shape[1])*xxinv
    return bMean,bVar
```

17.4.3. Суб- и супермартингейловые проверки

В этом разделе мы представим проверки взрываемости, которые не зависят от стандартной спецификации проверки ADF. Рассмотрим процесс, который является суб- либо супермартингейловым. При заданных неких наблюдениях $\{y_t\}$ мы хотели бы проверить существование взрывного временного тренда, $H_0 : \beta = 0$, $H_1 : \beta \neq 0$, согласно альтернативным спецификациям:

- Полиномиальный тренд (SM-Poly1):

$$y_t = \alpha + \gamma t + \beta t^2 + \varepsilon_t.$$

- Полиномиальный тренд (SM-Poly2):

$$\log[y_t] = \alpha + \gamma t + \beta t^2 + \varepsilon_t.$$

- Экспоненциальный тренд (SM-Exp):

$$y_t = \alpha e^{\beta t} + \varepsilon_t \Rightarrow \log[y_t] = \log[a] + \beta t + \xi_t.$$

- Степенной тренд (SM-Power):

$$y_t = \alpha t^{\beta} + \varepsilon_t \Rightarrow \log[y_t] = \log[a] + \beta \log[t] + \xi_t.$$

Подобно проверке SADF, мы подгоняем любую из этих спецификаций к каждой конечной точке $t = \tau,..., T$, с расширяющимися назад начальными точками, а затем вычисляем

$$SMT_t = \sup_{t_0 \in [1, t-\tau]} \left\{ \frac{\left| \hat{\beta}_{t_0,t} \right|}{\hat{\sigma}_{\hat{\beta}_{t_0,t}}} \right\}.$$

Причина абсолютного значения в том, что мы одинаково заинтересованы во взрывном росте и коллапсе. В случае простой регрессии (Greene [2008], с. 48), дисперсия β равна $\hat{\sigma}^2_{\beta} = \frac{\hat{\sigma}^2_{\varepsilon}}{\hat{\sigma}^2_{xx}(t-t_0)}$, следовательно, $\lim_{t\to\infty} \hat{\sigma}_{\beta_{t_0,t}} = 0$. Тот же результат обобщается на случай многомерной линейной регрессии (Greene [2008], с. 51–52). $\hat{\sigma}^2_{\beta}$ слабого долгосрочного пузыря может быть меньше, чем $\hat{\sigma}^2_{\beta}$ сильного краткосрочного пузыря, следовательно, смещая метод в сторону долгосрочных пузырей. Для того чтобы исправить это смещение, мы можем штрафовать большие длины выборки, определив коэффициент $\varphi \in [0, 1]$, который дает лучшие сигналы взрываемости.

$$SMT_t = \sup_{t_0 \in [1, t-\tau]} \left\{ \frac{\left|\hat{\beta}_{t_0,t}\right|}{\hat{\sigma}_{\hat{\beta}_{t_0,t}} (t-t_0)^{\varphi}} \right\}.$$

Например, при $\varphi = 0.5$ мы компенсируем меньшее $\hat{\sigma}_{\beta_{t_0,t}}$, связанное с большей длиной выборки, в случае простой регрессии. Для $\varphi \to 0$ проверка SMT_t будет демонстрировать более длинные тренды, так как эта компенсация ослабевает и долгосрочные пузыри маскируют краткосрочные пузыри. Для $\varphi \to 1$ SMT_t становится шумнее, потому что более короткие пузыри отбираются за счет долгосрочных пузырей. Следовательно, это является естественным способом корректировки сигнала взрываемости, с тем чтобы он фильтровал возможности, нацеленные на определенный период владения. Признаки, используемые машинно-обучающимся алгоритмом, могут включать SMT_t, оцененную из широкого диапазона значений φ.

Упражнения

17.1. На долларовом барном ряде на фьючерсном контракте E-mini S&P 500:

(а) Примените метод Брауна–Дарбина–Эванса. Проверьте, сможет ли он предсказать пузырь доткомов.

(б) Примените метод Чу–Стинчкомба–Уайта. Проверьте, сможет ли он обнаружить пузырь 2007–2008 годов.

17.2. На долларовом барном ряде на фьючерсном контракте E-mini S&P 500:

(а) Вычислите проверку взрываемости SDFC (по типу Чоу). Какая дата сдвига отбирается данным методом? Это то, чего вы ожидали?

(б) Вычислите и постройте график значений проверки SADF для этого ряда. Наблюдаете ли вы экстремальные всплески вокруг пузыря доткомов и перед Великой рецессией? Вызвали ли схлопывания также и всплески?

17.3. На основе упражнения 17.2:

(а) Определите периоды, в которых ряд демонстрировал:

i) спокойное состояние;

ii) состояние единичного корня;

iii) состояние резкого подъема.

(б) Рассчитайте квантильный асимптотически непараметрический критерий.

(в) Рассчитайте условный асимптотически непараметрический критерий.

17.4. На долларовом барном ряде на фьючерсном контракте E-mini S&P 500:

(а) Вычислите проверку SMT для SM-Poly1 и SM-Poly 2, где $\varphi = 1$. Какова их корреляция?

(б) Вычислите проверку SMT для SM-Exp, где $\varphi = 1$ и $\varphi = 0.5$. Какова их корреляция?

(в) Вычислите проверку SMT для SM-Power, где $\varphi = 1$ и $\varphi = 0.5$. Какова их корреляция?

17.5. Если вычислить обратную величину каждой цены, то ряд $\{y_t^{-1}\}$ превратит пузыри в схлопывания и схлопывания в пузыри.

(а) Нужна ли эта трансформация для выявления схлопываний?

(б) Какие методы в этой главе могут идентифицировать схлопывания без необходимости такого преобразования?

18 Энтропийные признаки

18.1. Актуальность

Ценовой ряд передает информацию о силах спроса и предложения. На идеальных рынках цены непредсказуемы, потому что каждое наблюдение транслирует все, что известно о продукте или услуге. При несовершенстве рынков цены формируются с частичной информацией, и поскольку некоторые агенты знают больше других, то могут эксплуатировать эту информационную асимметрию. Было бы полезно оценивать информационное содержимое ценового ряда и формировать признаки, на которых алгоритмы МО способны заучивать вероятные исходы. Например, алгоритм МО может обнаружить, что импульсные ставки выгоднее, когда цены несут мало информации, и что ставки на возврате к среднему значению выгоднее, когда цены несут много информации. В этой главе мы рассмотрим способы определения количества информации, содержащейся в ценовом ряде.

18.2. Энтропия Шеннона

В этом разделе мы рассмотрим несколько концепций из теории информации, которые будут полезны в оставшейся части главы. Читатель может найти полное изложение в публикации MacKay [2003]. Основоположник теории информации Клод Шеннон определил энтропию как среднее количество информации (на длинных сообщениях), производимой стационарным источником данных. Это количество представлено наименьшим числом бит на символ, необходимым для описания сообщения уникальным декодируемым способом. Математически Шеннон в работе Shannon [1948] определил энтропию дискретной случайной величины X с возможными значениями $x \in A$ как

$$H[X] = -\sum_{x \in A} p[x] \log_2 p[x]$$

с $0 \leq H[X] \leq \log_2 [\|A\|]$, где $p[x]$ — это вероятность x, $H[X] = 0 \Leftrightarrow \exists x | p[x] = 1$, $H[X] = \log_2[\|A\|] \Leftrightarrow p[x] = \frac{1}{\|A\|}$ для всех x и $\|A\|$ — размер множества A. Это можно интер-

претировать как взвешенное по вероятности среднее значение информационного содержимого в X, где биты информации измеряются как $\log_2 \frac{1}{p[x]}$. Обоснование для измерения информации как $\log_2 \frac{1}{p[x]}$ вытекает из наблюдения, что маловероятные исходы показывают больше информации, чем высоковероятные исходы. Другими словами, мы учимся, когда происходит что-то неожиданное. Аналогичным образом, избыточность определяется как

$$R[X] \equiv 1 - \frac{H[X]}{\log_2[\| A \|]}.$$

с $0 \leq R[X] \leq 1$. Колмогоров [1965] формализовал связь избыточности и сложности марковского информационного источника. Взаимная информация между двумя переменными определяется как отклонение Кульбака–Лейблера от совместной плотности вероятностей к произведению частных плотностей вероятностей[1].

$$MI[X,Y] = E_{f[x,y]}\left[\log \frac{f[x,y]}{f[x]f[y]}\right] = H[X] + H[Y] - H[X,Y].$$

Взаимная информация (mutual information, MI) всегда неотрицательна, симметрична и равна нулю тогда и только тогда, когда X и Y независимы. Для нормально распределенных величин взаимная информация тесно связана с известной корреляцией Пирсона ρ:

$$MI[X,Y] = -\frac{1}{2}\log[1-\rho^2].$$

Таким образом, взаимная информация является естественной мерой связи между величинами, независимо от того, являются ли они линейными или нелинейными по своей природе (Hausser and Strimmer [2009]). Нормализованная вариация информации — это метрика, выводимая из взаимной информации. Для ознакомления с несколькими оценщиками энтропии обратитесь:

- На языке R: http://cran.r-project.org/web/packages/entropy/entropy.pdf
- На языке Python: https://code.google.com/archive/p/pyentropy/

[1] Частное, или маржинальное (маргинальное), распределение подмножества совокупности случайных величин — это распределение вероятностей величин, содержащихся в *подмножестве*. Например, при заданных двух случайных величинах X и Y, совместное распределение которых известно, частным распределением X будет простое распределение вероятности X, усредненное по информации о Y. — *Примеч. науч. ред.*

18.3. Подстановочный (или максимально правдоподобный) оценщик

В этом разделе мы будем следовать изложению максимально правдоподобного оценщика энтропии в публикации Gao и соавт. [2008]. В начале номенклатура может показаться немного своеобразной (никаких каламбуров), но как только вы с ней познакомитесь, вам будет удобно. При заданной последовательности данных x_1^n, содержащей цепочку значений, начинающуюся в позиции 1 и заканчивающуюся в позиции n, можно сформировать словарь всех слов длины $w < n$ в этой последовательности A^w. Рассмотрим произвольное слово $y_1^w \in A^w$ длины w. Мы обозначаем через $\hat{p}_w[y_1^w]$ эмпирическую вероятность слова y_1^w в x_1^n. Из этого следует, что $\hat{p}_w[y_1^w]$ — это частота, с которой y_1^w появляется в x_1^n. Исходя из того, что данные генерируются стационарным и эргодическим процессом, закон больших чисел гарантирует, что для фиксированного w и большого n эмпирическое распределение p_w будет близко к истинному распределению p_w. В этих условиях естественным оценщиком скорости энтропии (то есть средней энтропии на бит) является

$$\hat{H}_{n,w} = -\frac{1}{w}\sum_{y_1^w \in A^w} \hat{p}_w[y_1^w]\log_2 \hat{p}_w[y_1^w].$$

Поскольку эмпирическое распределение также является максимально правдоподобной оценкой истинного распределения, его также часто называют максимально правдоподобным оценщиком энтропии. Значение w должно быть достаточно большим для $\hat{H}_{n,w}$, чтобы находиться достаточно близко к истинной энтропии H. Значение n должно быть намного больше w, с тем чтобы эмпирическое распределение порядка w находилось близко к истинному распределению. Листинг 18.1 реализует подстановочный оценщик энтропии[1].

Листинг 18.1. Подстановочный оценщик энтропии

```
import time,numpy as np
#---------------------------------------
def plugIn(msg,w):
    # Вычислить (MO) скорость подстановочной энтропии
    pmf=pmf1(msg,w)
    out=-sum([pmf[i]*np.log2(pmf[i]) for i in pmf])/w
    return out,pmf
#---------------------------------------
def pmf1(msg,w):
```

[1] Подстановочный оценщик энтропии (plug-in entropy estimator) — это энтропия распределения, где вероятности символов или блоков были заменены их относительными частотами в выборке. — *Примеч. науч. ред.*

```
    # Вычислить функцию вероятности (функцию распределения масс)
    # для одномерной дискретной случайной величины
    # len(msg)-w возникновений
    lib={}
    if not isinstance(msg,str): msg=''.join(map(str,msg))
    for i in xrange(w,len(msg)):
        msg_=msg[i-w:i]
        if msg_ not in lib: lib[msg_]=[i-w]
        else: lib[msg_]=lib[msg_]+[i-w]
    pmf=float(len(msg)-w)
    pmf={i:len(lib[i])/pmf for i in lib}
    return pmf
```

18.4. Оценщики на основе алгоритма LZ

Энтропию можно интерпретировать как меру сложности. Сложная последовательность содержит больше информации, чем обычная (предсказуемая) последовательность. Алгоритм Лемпеля–Зива (LZ) эффективно разлагает сообщение на избыточные подстроки (Ziv and Lempel [1978]). Мы можем оценить степень сжатия сообщения как функцию от числа элементов в словаре Лемпеля–Зива относительно длины сообщения. Интуитивная идея здесь заключается в том, что сложные сообщения имеют высокую энтропию, которая потребует больших словарей относительно длины передаваемой строки. Листинг 18.2 показывает реализацию алгоритма сжатия LZ.

Листинг 18.2. Библиотека, создаваемая с помощью алгоритма LZ

```
def lempelZiv_lib(msg):
    i,lib=1,[msg[0]]
    while i<len(msg):
        for j in xrange(i,len(msg)):
            msg_=msg[i:j+1]
            if msg_ not in lib:
                lib.append(msg_)
                break
        i=j+1
    return lib
```

В публикации Kontoyiannis [1998] предпринимается попытка эффективнее воспользоваться информацией, имеющейся в сообщении. Ниже мы воспроизведем точное резюме изложения публикации Gao и соавт. [2008]. Мы будем воспроизводить шаги из этой статьи, дополняя их фрагментами кода, которые реализуют их идеи. Определим L_i^n как 1 плюс длина самого длинного совпадения, найденного в n битах перед i:

$$L_i^n = 1 + \max\{l | x_i^{i+1} = x_j^{j+1} = \text{для некоторых } i-n \le j \le i-1, l \in [0,n]\}.$$

Листинг 18.3 реализует алгоритм, который определяет длину самого длинного совпадения. Вот несколько примечаний, достойных упоминания:

- Значение n постоянно для скользящего окна и $n = i$ для расширяющегося окна.
- Для вычисления L_i^n требуются данные x_{i-n}^{j+n-1}. Другими словами, индекс i должен находиться в центре окна. Это важно для того, чтобы гарантировать, что обе совпадающие строки имеют одинаковую длину. Если их длина не одинакова, l будет иметь ограниченный диапазон и ее максимум будет недооценен.
- Допускается некоторое наложение между двумя подстроками, хотя совершенно очевидно, что обе не могут начинаться с i.

Листинг 18.3. Функция, которая вычисляет длину самого длинного совпадения

```
def matchLength(msg,i,n):
    # Максимальная совпадающая длина+1, с наложением.
    # i>=n & len(msg)>=i+n
    subS=''
    for l in xrange(n):
        msg1=msg[i:i+l+1]
        for j in xrange(i-n,i):
            msg0=msg[j:j+l+1]
            if msg1==msg0:
                subS=msg1
                break # поиск более высокой l.
    return len(subS)+1,subS # совпавшая длина + 1
```

В работе Ornstein and Weiss [1993] было официально установлено, что

$$\lim_{n\to\infty} = \frac{L_i^n}{\log_2[n]} = \frac{1}{H}.$$

Контоянис использует этот результат для оценки скорости энтропии Шеннона. Он оценивает среднее $\frac{L_i^n}{\log_2[n]}$ и использует обратную величину этого среднего для оценки H. Общая интуитивная идея заключается в том, что по мере увеличения имеющейся в наличии истории мы ожидаем, что сообщения с высокой энтропией будут производить относительно более короткие неизбыточные подстроки. Напротив, сообщения с низкой энтропией будут создавать относительно более длинные неизбыточные подстроки по мере того, как мы выполняем разбор сообщения. При заданной реализации $x_{-\infty}^{\infty}$ данных, окна длиной $n \geq 1$ и числа совпадений $k \geq 1$ скользящеоконный LZ-оценщик $\hat{H}_{n,k} = \hat{H}_{n,k}[x_{-n+1}^{n+k-1}]$ определяется по формуле

$$\hat{H}_{n,k} = \left[\frac{1}{k}\sum_{i=1}^{k}\frac{L_i^n}{\log_2[n]}\right]^{-1}.$$

Аналогичным образом, возрастающеоконный LZ-оценщик $\hat{H}_n = \hat{H}_n[x_0^{2n-1}]$ определяется по формуле

$$\hat{H}_{n,k} = \left[\frac{1}{n}\sum_{i=2}^{n}\frac{L_i^i}{\log_2[i]}\right]^{-1}.$$

Размер окна n является постоянным при вычислении $\hat{H}_{n,k}$, следовательно L_i^n. Однако при вычислении $\hat{H}_n$ размер окна увеличивается вместе с i, следовательно L_i^i с $n = \frac{N}{2}$. В данном случае с расширяющимся окном длина сообщения N должна быть четным числом, чтобы гарантировать, что все биты были разобраны (напомним, что x_i находится в центре, поэтому последний бит сообщения с нечетной длиной не будет прочитан).

Эти выражения были выведены в рамках допущений: стационарность, эргодичность, что процесс принимает конечное множество значений и что процесс удовлетворяет условию Дёблина. В интуитивном плане это условие требует, чтобы после конечного числа шагов r, независимо от того, что произошло раньше, все что угодно может произойти с положительной вероятностью. Получается, что этого условия Дёблина можно избежать, если мы рассмотрим модифицированную версию приведенных выше оценщиков:

$$\tilde{H}_{n,k} = \frac{1}{k}\sum_{i=1}^{k}\frac{\log_2[n]}{L_i^n};$$

$$\tilde{H}_n = \frac{1}{n}\sum_{i=2}^{n}\frac{\log_2[i]}{L_i^i}.$$

Один практический вопрос при оценивании $\tilde{H}_{n,k}$ состоит в том, как определять размер окна n. В публикации Gao и соавт. [2008] утверждается, что $k + n = N$ должно быть приближенно равно длине сообщения. Учитывая, что смещение L_i^n имеет порядок $O\left[1/\log_2[n]\right]$ и дисперсия L_i^n имеет порядок $O[1/k]$, компромисс между смещением и дисперсией сбалансирован вокруг $k \approx O[(\log_2[n])^2]$. То есть можно выбрать n такое, что $N \approx n + (\log_2[n])^2$. Например, при $N = 2^8$ сбалансированный по смещению/дисперсии размер окна будет равен $n \approx 198$, и в этом случае $k \approx 58$.

Контояннис [1998] доказал, что $\hat{H}[X]$ сходится к скорости энтропии Шеннона с вероятностью 1 по мере приближения n к бесконечности. Листинг 18.4 реализует идеи, обсуждаемые в публикации Gao и соавт. [2008], которые являются развитием идей, высказанных в публикации Kontoyiannis [1997], путем поиска максимальной избыточности между двумя подстроками одинакового размера.

Листинг 18.4. Реализация алгоритмов, описанных в публикации Gao и соавт. [2008]

```
def konto(msg,window=None):
    '''
    * LZ оценка энтропии Контоянниса, версия 2013 г. (центрированное окно).
    * Инверсия средней длины кратчайшей неизбыточной подстроки.
    * Если неизбыточные подстроки короткие, то текст сильно энтропийный.
    * window== None для расширяющегося окна, в этом случае len(msg)%2==0
    * Если конец сообщения более релевантен, то попробовать konto(msg [::-1])
    '''
    out={'num':0,'sum':0,'subS':[]}
    if not isinstance(msg,str): msg=''.join(map(str,msg))
    if window is None:
        points=xrange(1,len(msg)/2+1)
        ENCODING SCHEMES 269
    else:
        window=min(window,len(msg)/2)
        points=xrange(window,len(msg)-window+1)
    for i in points:
        if window is None:
            l,msg_=matchLength(msg,i,i)
            out['sum']+=np.log2(i+1)/l # чтобы избежать условия Дёблина
        else:
            l,msg_=matchLength(msg,i,window)
            out['sum']+=np.log2(window+1)/l # чтобы избежать условия Дёблина
        out['subS'].append(msg_)
        out['num']+=1
    out['h']=out['sum']/out['num']
    out['r']=1-out['h']/np.log2(len(msg)) # избыточность, 0<=r<=1
    return out
#---------------------------------------
if __name__=='__main__':
    msg='101010'
    print konto(msg*2)
    print konto(msg+msg[::-1])
```

Одним из предостережений этого метода является то, что скорость энтропии определяется в пределе. По словам Контоянниса, «мы фиксируем крупное число N как размер нашей базы данных». Теоремы, используемые в работе Контоянниса, доказывают асимптотическую сходимость; однако нигде не заявляется о свойстве монотонности. Если сообщение короткое, то решением может быть повторение одного и того же сообщения несколько раз.

Второе предостережение состоит в том, что, поскольку окно для сопоставления должно быть симметричным (той же длины для словаря, что и для сопоставляемой подстроки), последний бит рассматривается для сопоставления, только если длина сообщения соответствует четному числу. Одним из решений является удаление первого бита сообщения нечетной длины.

Третье предостережение заключается в том, что некоторые конечные биты будут отклонены, когда им предшествуют нерегулярные последовательности. Это также

является следствием симметричного окна сопоставления. Например, скорость энтропии «10000111» равна скорости энтропии «10000110», а это означает, что последний бит нерелевантен вследствие несопоставимой цепочки «11» в шестом и седьмом бите. Когда конец сообщения особо релевантен, хорошим решением может быть анализ энтропии обратного сообщения. Этим не только гарантируется, что используются конечные биты (то есть начальные после реверсирования), но на самом деле они будут использоваться для потенциального сопоставления каждого бита. Продолжая предыдущий пример, скорость энтропии «11100001» составляет 0.96, а скорость энтропии «01100001» — 0.84.

18.5. Схемы кодирования

Оценивание энтропии требует кодирования сообщения. В этом разделе мы рассмотрим несколько используемых в литературе схем кодирования, которые основаны на финансовых возвратах. Хотя это не обсуждается ниже, рекомендуется кодировать информацию из дробно (а не целочисленно) дифференцированных рядов (глава 4), поскольку они все еще содержат некоторую память.

18.5.1. Двоичное кодирование

Оценивание скорости энтропии требует дискретизации непрерывной величины, такой, чтобы каждому значению можно было присвоить код из конечного алфавита. Например, поток финансовых возвратов r_t может быть закодирован в соответствии со знаком, 1 для $r_t > 0$, 0 для $r_t < 0$, удаляя случаи, когда $r_t = 0$. Двоичное кодирование возникает естественным образом в случае рядов с финансовыми возвратами, отобранных из ценовых баров (то есть баров, содержащих цены, колеблющиеся между двумя симметричными горизонтальными барьерами, центрированными вокруг начальной цены), потому что $|r_t|$ приближенно константно.

Когда $|r_t|$ может принимать широкий диапазон результатов, двоичное кодирование отбрасывает потенциально полезную информацию. Это особенно актуально при работе с внутридневными барами, на которые влияет гетероскедастичность, обусловленная неоднородностью тиковых данных. Одним из способов частично решить эту гетероскедастичность является отбор цен в соответствии с подчиненным стохастическим процессом. Примерами этого являются сделочные бары и объемные бары, которые содержат фиксированное число сделок или сделок на фиксированную величину объема (см. главу 2). Работая по этим нехронологическим, приводимым в действие рынком часам, мы делаем отбор образцов чаще в периоды высокой активности и реже в периоды меньшей активности, тем самым регулируя распределение $|r_t|$ и уменьшая потребность в большом алфавите.

18.5.2. Квантильное кодирование

Если не используются ценовые бары, то вполне вероятно, что потребуется более двух кодов. Один подход состоит в назначении кода каждому r_t согласно квантилю,

которому он принадлежит. Границы квантилей определяются с использованием внутривыборочного периода (тренировочного подмножества). Каждой букве будет назначено одинаковое число наблюдений для совокупной внутренней выборки и близкое к одинаковому числу наблюдений в расчете на букву вне выборки. При использовании этого метода некоторые коды охватывают бóльшую долю диапазона r_t, чем другие. Это равномерное (внутривыборочное) или близкое к равномерному (вневыборочное) распределение кодов имеет тенденцию к увеличению скорости энтропии в среднем.

18.5.3. Сигма-кодирование

В качестве альтернативного подхода, вместо того чтобы задавать число кодов, мы могли бы позволить ценовому потоку определять фактический словарь. Предположим, мы задаем шаг дискретизации, σ. Затем мы назначаем 0 для $r_t \in [\min\{r\}, \min\{r\}+\sigma)$, 1 для $r_t \in [\min\{r\} + \sigma, \min\{r\} + 2\sigma)$ и так далее до тех пор, пока все наблюдения не будут закодированы с общим числом ceil $\left[\frac{\max[r]-\min[r]}{\sigma}\right]$ кодов, где ceil[.] — это функция округления вверх до ближайшего целого. В отличие от квантильного кодирования, теперь каждый код охватывает ту же долю диапазона r_t. Поскольку коды распределены неравномерно, скорости энтропии будут, как правило, меньше, чем в среднем в квантильном кодировании; однако появление «редкого» кода вызовет всплески скорости энтропии.

18.6. Энтропия гауссова процесса

Энтропия нормального одинаково распределенного взаимно независимого случайного процесса (см. Norwich [2003]) может быть выведена как

$$H = \frac{1}{2}\log[2\pi e\sigma^2].$$

Для стандартного нормального процесса $H \approx 1.42$. Есть по крайней мере два использования этого результата. Во-первых, это позволяет нам оценивать результативность оценщика энтропии. Мы можем вынимать образцы из стандартного нормального распределения и находить то, какое сочетание оценщика, длины сообщения и кодирования дает нам оценку $\hat{H}$ энтропии, достаточно близкую к теоретически полученному значению H. Например, на рис. 18.1 построен график бутстрапированных распределений оценок энтропии в рамках 10-, 7-, 5- и 2-буквенных кодировок на сообщениях длиной 100, используя метод Контоянниса. Для алфавитов не менее 10 букв алгоритм из листинга 18.4 предоставляет правильный ответ. Когда алфавиты слишком малы, информация отбрасывается, и энтропия недооценивается.

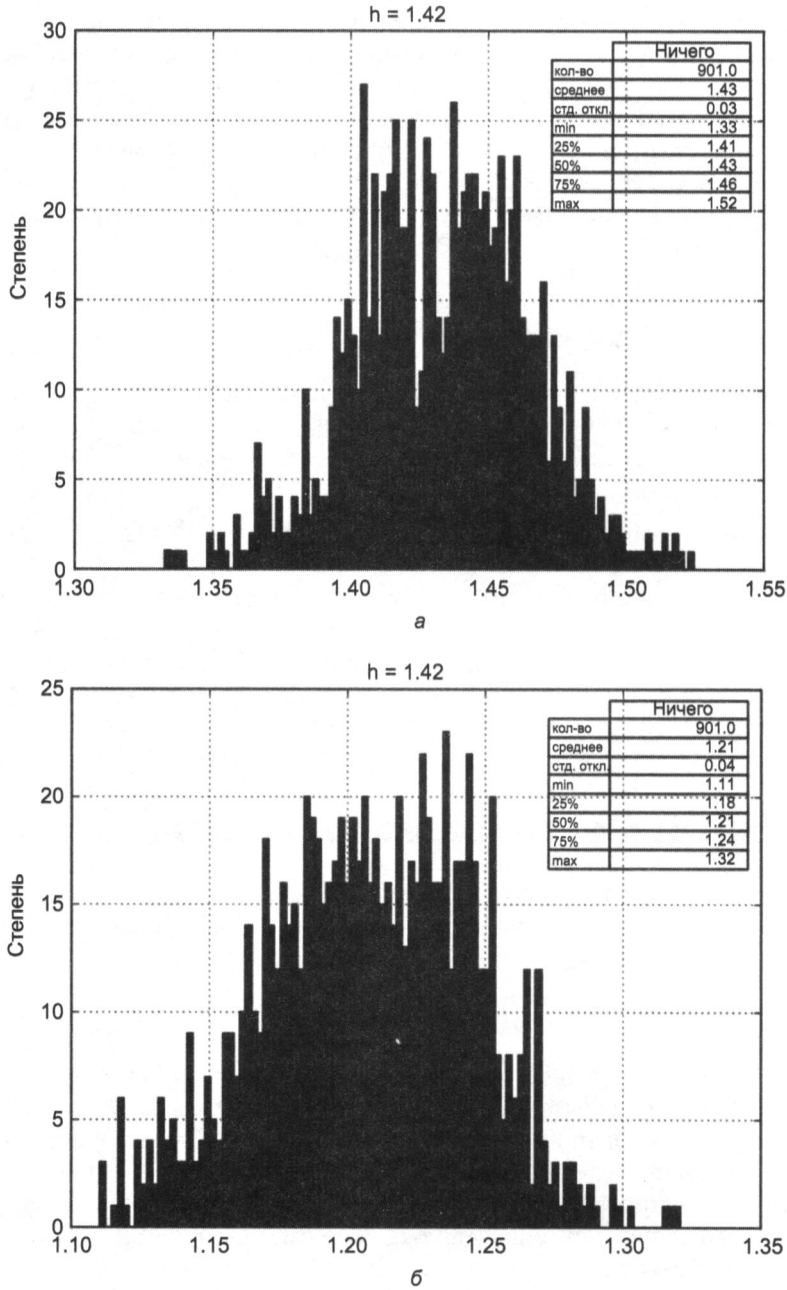


Рис. 18.1. Распределение оценок энтропии в условиях 10-буквенных (вверху) и 7-буквенных (внизу) кодировок на сообщениях длиной 100

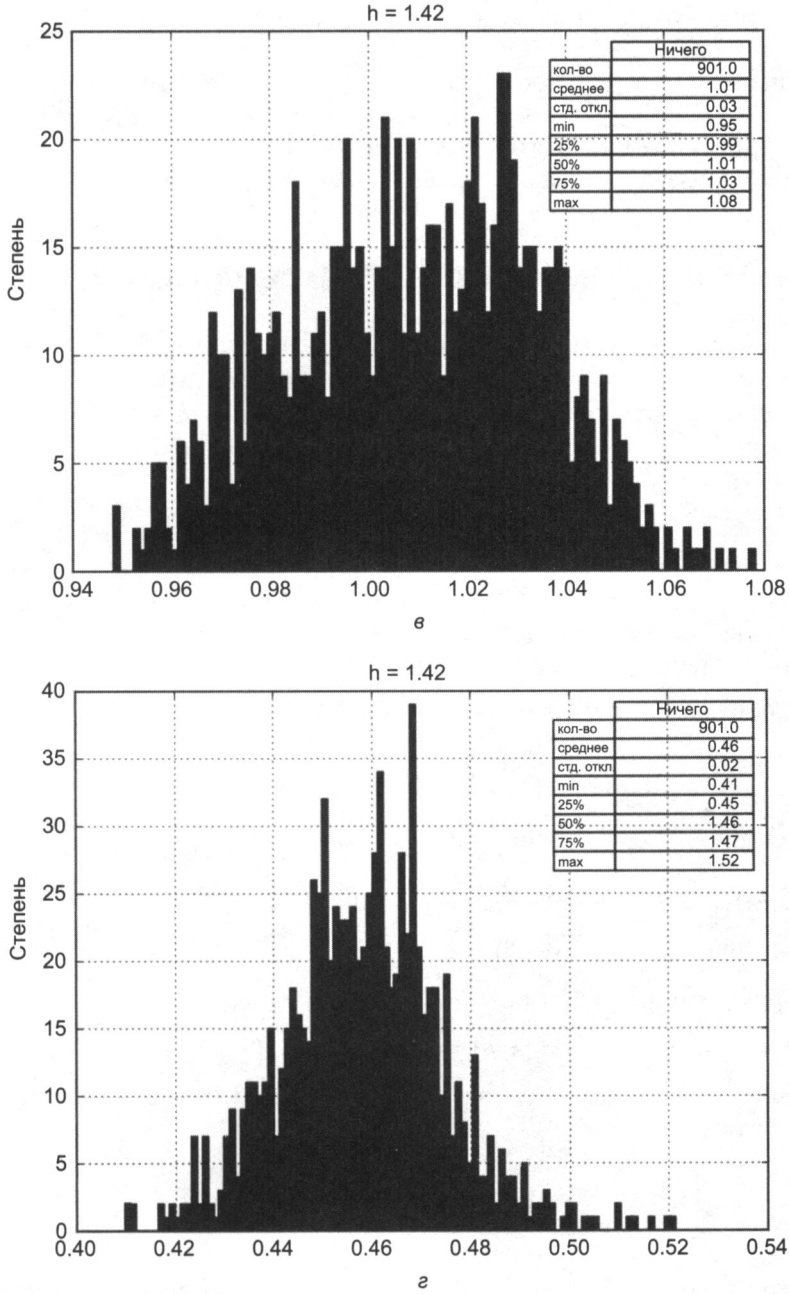


Рис. 18.1. (*Окончание*) Распределение оценок энтропии в условиях 5-буквенных (вверху) и 2-буквенных (внизу) кодировок на сообщениях длиной 100

Во-вторых, мы можем использовать приведенное выше уравнение для соединения энтропии с волатильностью, отметив, что $\sigma_H = \frac{e^{H-1/2}}{\sqrt{2\pi}}$. Это дает нам оценку энтропийно-предполагаемой волатильности, при условии что финансовые возвраты действительно извлекаются из нормального распределения.

18.7. Энтропия и обобщенное среднее

Ниже приводится интересный способ думать об энтропии. Рассмотрим множество вещественных чисел $x = \{x_i\}_{i=1,\ldots,n}$ и весов $p = \{p_i\}_{i=1,\ldots,n}$ таких, что $0 \le p_i \le 1, \forall i$ и $\sum_{i=1}^{n} p_i = 1$. Обобщенное взвешенное среднее x с весами p на степени $q \neq 0$ определяется как

$$M_q[x,p] = \left(\sum_{i=1}^{n} p_i x_i^q \right)^{1/q}.$$

Для $q < 0$ нам необходимо, чтобы $x_i > 0, \forall i$. Причина, по которой это среднее является обобщенным, заключается в том, что другие средние могут быть получены как частные случаи:

- Минимальное: $\lim_{q \to -\infty} M_q[x,p] = \min_i\{x_i\}$.
- Гармоническое среднее: $M_{-1}[x,p] = \left(\sum_{i=1}^{n} p_i x_i^{-1} \right)^{-1}$.
- Геометрическое среднее: $\lim_{q \to 0} M_q[x,p] = e^{\sum_{i=1}^{n} p_i \log[x_i]} = \prod_{i=1}^{n} x_i^{p_i}$.
- Арифметическое среднее: $M_1[x, \{n^{-1}\}_{i=1,\ldots,n}] = n^{-1} \sum_{i=1}^{n} x_i$.
- Взвешенное среднее: $M_1[x,p] = \sum_{i=1}^{n} p_i x_i$.
- Квадратичное среднее: $M_2[x,p] = (\sum_{i=1}^{n} p_i x_i^2)^{1/2}$.
- Максимальное: $\lim_{q \to +\infty} M_q[x,p] = \max_i\{x_i\}$.

В контексте теории информации интересным частным случаем является $x = \{p_i\}_{i=1,\ldots,n}$, следовательно:

$$M_q[p,p] = \left(\sum_{i=1}^{n} p_i p_i^q \right)^{1/q}.$$

Определим величину $N_q[p] = \frac{1}{M_{q-1}[p,p]}$ для $q \neq 1$. Опять же, для $q < 1$ в $N_q[p]$ мы должны иметь $p_i > 0, \forall i$. Если $p_i = \frac{1}{k}$ для $k \in [1,n]$ разных индексов и $p_i = 0$ в другом

месте, то вес распределяется равномерно по k разным элементам и $N_q[p] = k$ для $q > 1$. Другими словами, $N_q[p]$ дает нам *эффективное число* или *многообразие* элементов в p, согласно некоторой схеме взвешивания, задаваемой q.

Используя неравенство Йенсена, мы можем доказать, что $\frac{\partial M_q[p,p]}{\partial q} \geq 0$, следовательно, $\frac{\partial N_q[p]}{\partial q} \leq 0$. Меньшие значения q назначают более равномерный вес элементам раздела, давая относительно больший вес менее общим элементам, и $\lim_{q\to 0} N_p[p]$ — это просто общее число ненулевых p_i.

Энтропия Шеннона равняется $H[p] = \sum_{i=1}^{n} - p_i \log[p_i] = -\log[\lim_{q\to 0} M_p[p] = \log[\lim_{q\to 1} N_q[p]]$. Этим показывается, что энтропия может быть интерпретирована как логарифм *эффективного числа* элементов в списке p, где $q \to 1$. Рисунок 18.2 иллюстрирует, как эффективные логарифмические числа для семейства случайно сгенерированных p массивов сходятся к энтропии Шеннона по мере приближения q к 1. Обратите внимание также на то, как их поведение стабилизируется по мере роста q.

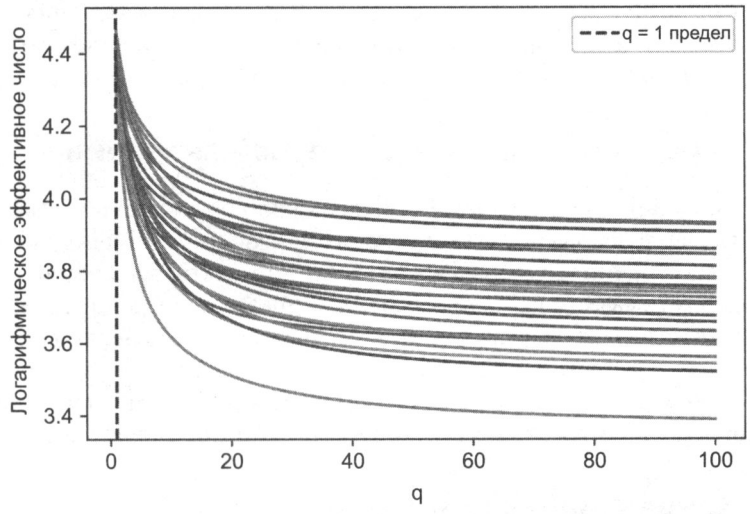


Рис. 18.2. Логарифмические эффективные числа для семейства случайно сгенерированных p массивов

В интуитивном плане, энтропия измеряет информацию как уровень *многообразия*, содержащегося в случайной величине. Эта интуитивная идея формализуется через понятие обобщенного среднего. Из этого следует, что энтропия Шеннона является частным случаем меры многообразия (отсюда и ее связь с волатильностью). Теперь мы можем определять и вычислять альтернативные меры многообразия, помимо энтропии, где $q \neq 1$.

18.8. Несколько финансовых приложений энтропии

В этом разделе мы представим несколько приложений энтропии для моделирования финансовых рынков.

18.8.1. Рыночная эффективность

Когда арбитражные механизмы эксплуатируют полное множество возможностей, цены мгновенно отражают весь объем доступной информации, становясь непредсказуемыми (то есть мартингейловыми), без каких-либо различимых закономерностей. И наоборот, когда арбитраж не совершенен, цены содержат неполные объемы информации, что приводит к предсказуемым закономерностям. Закономерности возникают, когда цепочка содержит избыточную информацию, что активирует ее сжатие. Энтропийная скорость цепочки определяет ее оптимальную скорость сжатия. Чем выше энтропия, тем меньше избыточность, тем больше информативное содержание. Следовательно, энтропия ценовой цепочки говорит нам о степени эффективности рынка в данный момент времени. «Разжатый» рынок является эффективным рынком, поскольку ценовая информация не избыточна. «Сжатый» рынок является неэффективным рынком, поскольку ценовая информация избыточна. Пузыри образуются на сжатых (низкоэнтропийных) рынках.

18.8.2. Генерирование максимальной энтропии

В ряде публикаций Fiedor [2014a, 2014b, 2014c] для оценки количества энтропии, присутствующей в ценовых рядах, предлагается использовать методику из работы Kontoyiannis [1997]. Автор утверждает, что из возможных будущих исходов наиболее выгодным может оказаться тот, который максимизирует энтропию, поскольку она наименее предсказуема фриквентистскими статистическими моделями. Это сценарий «черного лебедя», который, скорее всего, вызовет остановки убытков (стоп-лоссы), тем самым генерируя механизм обратной связи, который усилит и усугубит движение, что приведет к интервалам в знаках временны́х рядов финансовых возвратов.

18.8.3. Концентрация портфеля

Рассмотрим ковариационную $N \times N$-матрицу V, вычисленную на финансовых возвратах. Во-первых, мы вычисляем разложение матрицы по собственным значениям, $VW = W\Lambda$. Во-вторых, мы получаем вектор факторных нагрузок как $f_\omega = W'\omega$, где ω — это вектор размещений, $\sum_{n=1}^{n} \omega_n = 1$.[1] В-третьих, мы получаем долю риска, вносимую каждой главной компонентой (Bailey and Lopez de Prado [2012]), как

[1] В качестве альтернативы мы могли бы работать с вектором портфельных активов во владении, если бы ковариационная матрица была вычислена на ценовых изменениях.

$$\theta_i = \frac{[f_\omega]_i^2 \Lambda_{i,i}}{\sum_{n=1}^{N} [f_\omega]_n^2 \Lambda_{n,n}},$$

где $\sum_{i=1}^{N} \theta_i = 1$, а $\theta_i \in [0, 1]$, $\forall i = 1, ..., N$. В-четвертых, в публикации Meucci [2009] предложено следующее энтропийно-мотивированное определение концентрации портфеля:

$$H = 1 - \frac{1}{N} e^{-\sum_{n=1}^{N} \theta_i \log[\theta_i]}.$$

На первый взгляд, такое определение концентрации портфеля может показаться поразительным, поскольку θ_i не является вероятностью. Связь между этим понятием концентрации и энтропией обусловлена обобщенным средним, которое мы обсуждали в разделе 18.7.

18.8.4. Микроструктура рынка

В публикации Easley и соавт. [1996, 1997] показано, что когда шансы хороших новостей/плохих новостей равны, вероятность информированной биржевой торговли (probability of informed trading, PIN) может быть получена как

$$PIN = \frac{\alpha\mu}{\alpha\mu + 2\varepsilon},$$

где μ — это скорость прибытия информированных трейдеров, ε — скорость прибытия неинформированных трейдеров и α — вероятность информационного события. Вероятность PIN можно интерпретировать как долю ордеров, которые возникают от информированных трейдеров относительно совокупного потока ордеров.

Внутри объемного бара размера V мы можем классифицировать тики как покупку или продажу в соответствии с неким алгоритмом, таким как тиковое правило или алгоритм Ли—Рэди выявления направления сделки[1]. Пусть V_τ^B равен сумме объемов от покупных тиков, входящих в объемный бар τ, и V_τ^S равен сумме объемов от продажных тиков внутри объемного бара τ. В публикациях Easley и соавт. [2012a, 2012b] отмечается, что $E[|V_\tau^B - V_\tau^S|] \approx \alpha\mu$ и что ожидаемый общий объем равен $E[V_\tau^B + V_\tau^S] = \alpha\mu + 2\varepsilon$. С помощью объемных часов (Easley и соавт. [2012c]) мы можем установить значение $E[V_\tau^B + V_\tau^S] = \alpha\mu + 2\varepsilon = V$ экзогенно. Это означает, что в условиях объемных часов вероятность PIN сводится к

$$VPIN = \frac{\alpha\mu}{\alpha\mu + 2\varepsilon} = \frac{\alpha\mu}{V} \approx \frac{1}{V} E[|2V_\tau^B - V|] = E[|2v_\tau^B - 1|],$$

[1] См. https://wrds-www.wharton.upenn.edu/pages/support/applications/microstructure-research/lee-and-ready-1991-algorithm/. — *Примеч. науч. ред.*

где $\nu_\tau^B = \frac{V_\tau^B}{V}$. Обратите внимание, что $2\nu_\tau^B - 1$ представляет несбалансированность потока ордеров, OI_τ, то есть ограниченную вещественно-значную величину, где $OI_\tau \in [-1,1]$. Таким образом, теория VPIN обеспечивает формальную связь между вероятностью информированной биржевой торговли (PIN) и устойчивостью несбалансированностей потока ордеров в условиях объемных часов. См. главу 19 для получения дополнительной информации об этой микроструктурной теории.

Устойчивая несбалансированность потока ордеров является необходимым, но недостаточным условием для неблагоприятного выбора. Для того чтобы маркетмейкеры предоставляли ликвидность информированным трейдерам, данная несбалансированность потока ордеров $|OI_\tau|$ также должна была быть относительно непредсказуемой. Другими словами, маркетмейкеры не выбираются неблагоприятно, когда их прогноз несбалансированности потока ордеров точен, даже если $|OI_\tau| \gg 0$. Для того чтобы определить вероятность неблагоприятного выбора, мы должны определить величину непредсказуемости несбалансированности потока ордеров. Мы можем определить это, применив теорию информации.

Рассмотрим длинную последовательность символов. Когда эта последовательность содержит несколько избыточных закономерностей, она охватывает уровень сложности, который затрудняет описание и предсказание. В публикации Колмогорова [1965] сформулирована эта связь между избыточностью и сложностью. В теории информации сжатие без потерь — это задача идеального описания последовательности минимальным числом бит. Чем больше избыточностей последовательность содержит, тем выше степень сжатия. Энтропия характеризует избыточность источника, отсюда и его колмогоровская сложность и предсказуемость. Мы можем использовать эту связь между избыточностью последовательности и ее непредсказуемостью (маркетмейкерами) для получения вероятности неблагоприятного выбора.

Далее мы обсудим одну конкретную процедуру, которая выводит вероятность неблагоприятного выбора в зависимости от сложности, коренящейся в несбалансированности потока ордеров. Во-первых, при заданной последовательности объемных баров, индексированных по $\tau = 1, ..., N$, причем каждый бар размера V, определим часть объема, классифицированную как покупка, $\nu_\tau^B \in [0, 1]$. Во-вторых, мы вычисляем q-квантили на $\{\nu_\tau^B\}$, которые определяют множество K из q непересекающихся подмножеств, $K = \{K_1, ..., K_q\}$. В-третьих, мы производим отображение из каждого ν_τ^B в одно из непересекающихся подмножеств, $f: \nu_\tau^B \to \{1, ..., q\}$, где $f[\nu_\tau^B] = i \Leftrightarrow \nu_\tau^B \in K_i, \forall_i \in [1, q]$. В-четвертых, мы квантуем $\{\nu_\tau^B\}$, назначая каждому значению ν_τ^B индекс подмножества K, которому оно принадлежит, $f[\nu_\tau^B]$. В результате получаем трансляцию множества несбалансированностей ордеров $\{\nu_\tau^B\}$ в квантованное сообщение $X = [f[\nu_1^B], f[\nu_2^B], ..., f[\nu_N^B]]$. В-пятых, мы оцениваем энтропию $H[X]$, используя алгоритм LZ Контоянниса. В-шестых, мы получаем кумулятивную функцию распределения, $F[H[X]]$, и используем временной ряд $\{F[H[X_\tau]]\}_{\tau=1,...,N}$ как признак, который предсказывает неблагоприятный выбор.

Упражнения

18.1. Сформируйте долларовые бары на фьючерсе E-mini S&P 500:

(а) Проквантуйте ряд финансовых возвратов с помощью двоичного метода.

(б) Проквантуйте ряд финансовых возвратов с помощью квантильного кодирования, используя 10 букв.

(в) Проквантуйте ряд финансовых возвратов с помощью сигма-кодирования, где σ — это среднеквадратическое отклонение финансовых возвратов всех баров.

(г) Вычислите энтропию трех кодированных рядов, используя подстановочный метод.

(д) Вычислите энтропию трех кодированных рядов, используя метод Контоянниса с размером окна 100.

18.2. Используя бары из упражнения 18.1:

(а) Вычислите ряд финансовых возвратов $\{r_t\}$.

(б) Закодируйте ряд следующим образом: 0, если $r_t r_{t-1} < 0$, и 1, если $r_t r_{t-1} \geq 0$.

(в) Подразделите ряд на 1000 ненакладывающихся подмножеств равного размера (возможно, вам придется отбросить некоторые наблюдения в начале).

(г) Вычислите энтропию каждого из 1000 кодированных подмножеств, используя подстановочный метод.

(д) Вычислите энтропию каждого кодированного подмножества с помощью метода Контоянниса с размером окна 100.

(е) Вычислите корреляцию между результатами 18.2.г и 18.2.д.

18.3. Извлеките 1000 наблюдений из стандартного нормального распределения:

(а) Какова истинная энтропия этого процесса?

(б) Промаркируйте наблюдения в соответствии с восемью квантилями.

(в) Оцените энтропию с помощью подстановочного метода.

(г) Оцените энтропию с помощью метода Контоянниса:

i) используйте размер окна 10;

ii) используйте размер окна 100.

18.4. Используя наблюдения из упражнения 18.3, $\{x_t\}_{t=1,\ldots,1000}$:

(а) Вычислите $y_t = \rho_{y_{t-1}} + x_t$, где $\rho = .5$, $y_0 = 0$.

(б) Промаркируйте y_t наблюдений по восьми квантилям.

(в) Оцените энтропию с помощью подстановочного метода.

(г) Оцените энтропию с помощью метода Контоянниса:

i) используйте размер окна 10;

ii) используйте размер окна 100.

18.5. Предположим, что у вас есть портфель с 10 портфельными активами во владении с равными долларовыми размещениями.

(а) Доля общего риска, вносимая i-й главной компонентой, составляет $\frac{1}{10}$, $i = 1, ..., 10$. Рассчитайте энтропию портфеля.

(б) Доля общего риска, вносимая i-й главной компонентой, составляет $1-\frac{1}{55}$, $i = 1, ..., 10$. Рассчитайте энтропию портфеля.

(в) Доля общего риска, вносимая i-й главной компонентой, составляет $\alpha\frac{1}{10}+(1-\alpha)(1-\frac{i}{55})$, $i = 1, ..., 10$, $\alpha \in [0, 1]$. Постройте энтропию портфеля в качестве функции a.

19 Микроструктурные признаки

19.1. Актуальность

Рыночная микроструктура изучает «процесс и результаты обмена активами по четким правилам биржевой торговли» (O'Hara [1995]). Микроструктурные совокупности данных включают первичную информацию о процессе аукциона, такую как отмены заказов, книга двойного аукциона, очереди, частичные исполнения, сторона агрессора[1], исправления, замены и т. д. Основным источником являются сообщения по протоколу FIX (Financial Information eXchange, биржа финансовой информации), которые можно приобрести на биржах. Уровень детализации, содержащийся в сообщениях FIX, дает исследователям возможность понять, как участники рынка скрывают и раскрывают свои намерения. Это делает микроструктурные данные одним из наиболее важных компонентов для построения предсказательных признаков, используемых в машинном обучении.

19.2. Обзор литературы

Глубина и сложность теорий рыночной микроструктуры со временем эволюционировали как функция от величины и разнообразия имеющихся данных. Первое поколение моделей использовало исключительно ценовую информацию. Двумя основополагающими результатами этого раннего этапа являются модели классификации сделок (такие, как тиковое правило) и модель Ролла (Roll [1984]). Второе поколение моделей появилось после того, как стали доступны совокупности объемных данных и исследователи переключили свое внимание на изучение влияния объема на цены. Два примера этого поколения моделей — Kyle [1985] и Amihud [2002].

---

[1] Агрессор (agressor) — это трейдер, который забирает ликвидность с рынка. Вместо того чтобы делать ставку на акции, агрессор покупает на рынке по текущей цене предложения. Он также продает по текущим рыночным ценам и не указывает цену продажи. Покупая доступные акции или контракты по текущим рыночным ценам, агрессор размещает ордера, которые имеют немедленное исполнение. — *Примеч. науч. ред.*

Третье поколение моделей появилось после 1996 года, когда Морин О'Хара, Дэвид Исли и другие опубликовали свою теорию «вероятности информированной биржевой торговли» (probability of informed trading, PIN) (Easley и соавт. [1996]). Это стало большим прорывом, потому что вероятность PIN объяснила спред между ценой спроса и ценой предложения как следствие последовательного стратегического решения между поставщиками ликвидности (маркетмейкерами) и позиционерами (информированными трейдерами). В сущности, она проиллюстрировала, что маркетмейкеры были продавцами опциона, который будет выбран информированными трейдерами в неблагоприятных условиях, а спред между ценой спроса и ценой предложения — это премия, которую они взимают за этот опцион. В публикации Easley и соавт. [2012a, 2012b] объясняется, как оценивать VPIN, высокочастотную оценку вероятности PIN в рамках отбора на основе объема.

Это основные теоретические основы, используемые микроструктурной литературой. В публикациях O'Hara [1995] и Hasbrouck [2007] предлагается хороший сборник низкочастотных микроструктурных моделей. В публикации Easley и соавт. [2013] представлена современная трактовка высокочастотных микроструктурных моделей.

19.3. Первое поколение: ценовые последовательности

Первое поколение микроструктурных моделей было посвящено оценке спреда между ценой спроса и ценой предложения как косвенных индикаторов для неликвидности. Они сделали это с ограниченными данными и без навязывания стратегической или последовательной структуры процессу торговли.

19.3.1. Тиковое правило

В книге двойного аукциона котировки размещаются для продажи ценной бумаги на различных ценовых уровнях (ценах предложения) или для покупки ценной бумаги на различных ценовых уровнях (ценах спроса). Цены предложения всегда превышают цены спроса, потому что в противном случае было бы мгновенное совпадение. Сделка происходит, когда покупателю подходит цена предложения либо продавцу подходит цена спроса. Каждая сделка имеет покупателя и продавца, но только одна сторона инициирует сделку.

Тиковое правило — это алгоритм, используемый для определения в сделке стороны агрессора. Сделка, инициированная покупкой, маркируется меткой «1», и сделка, инициируемая продажей, маркируется меткой «–1», согласно следующей логике:

$$b_t = \begin{cases} 1 & \text{если } \Delta p_t > 0 \\ -1 & \text{если } \Delta p_t < 0 \\ b_{t-1} & \text{если } \Delta p_t = 0 \end{cases},$$

где p_t — это цена сделки, индексированная по $t = 1, ..., T$, а b_0 произвольно устанавливается равным 1. В ряде исследований было установлено, что тиковое правило достигает высокой точности классификации, несмотря на его относительную простоту (Aitken and Frino [1996]). Конкурирующие методы классификации включают Lee and Ready [1991] и Easley и соавт. [2016].

Трансформации ряда $\{b_t\}$ могут приводить к информативным признакам. Такие трансформации включают: 1) фильтры Калмана на его будущее математическое ожидание, $E_t[b_{t+1}]$; 2) структурные сдвиги на таких предсказаниях (глава 17); 3) энтропия $\{b_t\}$ последовательности (глава 18); 4) t-значения из проверок Вальда—Вольфовица отрезков на $\{b_t\}$; 5) дробное дифференцирование кумулятивного $\{b_t\}$ ряда, $\sum_{i=1}^{t} b_i$ (глава 5) и т. д.

19.3.2. Модель Ролла

В публикации Roll [1984] была описана одна из первых моделей, предложивших объяснение эффективного спреда между ценой спроса и ценой предложения, в котором ценная бумага торгуется. Оно полезно в том, что спреды между ценой спроса и ценой предложения являются функцией ликвидности, поэтому модель Ролла можно рассматривать как раннюю попытку измерить ликвидность ценной бумаги. Рассмотрим ряд срединных цен[1] $\{m_t\}$, где цены подчиняются случайному блужданию без дрейфа,

$$m_t = m_{t-1} + u_t,$$

следовательно, изменения цен $\Delta m_t = m_t - m_t - 1$ независимо и идентично изымаются из нормального распределения

$$\Delta m_t \sim N[0, \sigma_u^2].$$

Эти допущения, конечно, противоречат всем эмпирическим наблюдениям, которые позволяют предположить, что финансовые временные ряды имеют дрейф, они гетероскедастичны, проявляют внутрирядовую зависимость и их распределение финансовых возвратов ненормально. Но при надлежащей процедуре отбора, как

[1] Срединная цена (mid-price) — это цена между лучшей ценой продавцов актива, или ценой предложения (ask), и лучшей ценой покупателей актива, или ценой спроса (bid). — *Примеч. науч. ред.*

мы видели в главе 2, эти допущения не могут быть слишком нереалистичны. Наблюдаемые цены, $\{p_t\}$, являются результатом последовательной торговли против спреда между ценой спроса и ценой предложения:

$$p_t = m_t + b_t c,$$

где c — это половина спреда между ценой спроса и ценой предложения и $b_t \in \{-1, 1\}$ — сторона агрессора. Модель Ролла принимает допущение, что покупки и продажи равновероятны, $P[b_t = 1] = P[b_t = -1] = \frac{1}{2}$, внутрирядово независимы, $E[b_t b_{t-1}] = 0$, и независимы от шума, $E[b_t u_t] = 0$. Исходя из этих допущений, Ролл выводит значения c и σ_u^2 следующим образом:

$$\sigma^2[\Delta p_t] = E[(\Delta p_t)^2] - (E[(\Delta p_t)])^2 = 2c^2 + \sigma_u^2;$$
$$\sigma[\Delta p_t, \Delta p_{t-1}] = -c^2,$$

в результате чего $c = \sqrt{\max\{0, -\sigma[\Delta p_t, \Delta p_{t-1}]\}}$ и $\sigma_u^2 = \sigma^2[\Delta p_t] + 2\sigma[\Delta p_t, \Delta p_{t-1}]$. В заключение, спред между ценой спроса и ценой предложения является функцией от внутрирядовой ковариации ценовых изменений, а истинно (ненаблюдаемый) ценовой шум, за исключением микроструктурного шума, является функцией от наблюдаемого шума и внутрирядовой ковариации ценовых изменений.

Читатель может усомниться в необходимости модели Ролла в настоящее время, когда совокупности данных включают цены спроса и предложения на нескольких уровнях торговой книги. Одной из причин, почему модель Ролла по-прежнему используется, несмотря на ее ограниченность, является то, что она предлагает относительно прямой способ определения *эффективного* спреда между ценой спроса и ценой предложения на ценные бумаги, которые либо торгуются редко, или когда публикуемые котировки не репрезентативны относительно уровней, на которых маркетмейкеры готовы предоставлять ликвидность (например, корпоративные, муниципальные и агентские облигации). Используя оценки Ролла, мы можем получить информативные признаки относительно состояния ликвидности рынка.

19.3.3. Оценщик волатильности максимум-минимум

В публикации Beckers [1983] показано, что оценщики волатильности, основанные на максимальных (high) и минимальных (low) ценах, являются более точными, чем стандартные оценщики волатильности, основанные на ценах закрытия. В работе Parkinson [1980] выводится, что для непрерывно наблюдаемых цен, подчиняющихся геометрическому броуновскому движению,

$$\mathrm{E}\left[\frac{1}{T}\sum_{t=1}^{T}\left(\log\left[\frac{H_t}{L_t}\right]\right)^2\right] = k_1\sigma_{HL}^2$$
$$\mathrm{E}\left[\frac{1}{T}\sum_{t=1}^{T}\left(\log\left[\frac{H_t}{L_t}\right]\right)\right] = k_2\sigma_{HL}$$

где $k_1 = 4\log[2]$, $k_2 = \sqrt{\frac{8}{\pi}}$, H_t — максимальная цена для бара t, а L_t — минимальная цена для бара t. Тогда волатильностный признак σ_{HL} может быть робастно оценен, основываясь на наблюдаемых максимальных-минимальных (high-low) ценах.

19.3.4. Корвин и Шульц

Основываясь на работе Beckers [1983], Корвин и Шульц (Corwin and Schultz [2012]) вводят оценщика спреда между ценой спроса и ценой предложения на основе максимальных и минимальных (high-low) цен. Оценщик базируется на двух принципах: во-первых, максимальные цены почти всегда сопоставляются с ценой предложения, а минимальные цены почти всегда сопоставляются с ценой спроса. Соотношение максимальных цен к минимальным ценам отражает фундаментальную волатильность, а также спред между ценой спроса и ценой предложения. Во-вторых, компонента соотношения максимальных цен к минимальным ценам, которая обусловлена волатильностью, возрастает пропорционально времени, прошедшему между двумя наблюдениями.

Корвин и Шульц показывают, что этот спред, в процентах от цены, может быть оценен как

$$S_t = \frac{2(e^{a_t} - 1)}{1 + e^{a_t}},$$

где

$$a_t = \frac{\sqrt{2\beta_t} - \sqrt{\beta_t}}{3 - 2\sqrt{2}} - \sqrt{\frac{\gamma_t}{3 - 2\sqrt{2}}};$$
$$\beta_t = E\left[\sum_{j=0}^{1}\left[\log\left(\frac{H_{t-j}}{L_{t-j}}\right)\right]^2\right];$$
$$\gamma_t = \left[\log\left(\frac{H_{t-1,t}}{L_{t-1,t}}\right)\right]^2,$$

а $H_{t-1,t}$ — это максимальная цена на двух барах ($t-1$ и t), тогда как $L_{t-1,t}$ — минимальная цена на двух барах ($t-1$ и t). Поскольку $\alpha_t < 0 \Rightarrow S_t < 0$, авторы рекомендуют устанавливать отрицательные альфа равными 0 (см. Corwin and Schultz [2012], с. 727). Листинг 19.1 реализует данный алгоритм. Функция `corwinSchultz` получает два аргумента: кадр данных, содержащий ряды с двумя столбцами (`High`, `Low`), и целое значение `sl`, определяющее длину выборки, используемой для оценки β_t.

Листинг 19.1. Реализация алгоритма Корвина—Шульца

```
def getBeta(series,sl):
    hl=series[['High','Low']].values
    hl=np.log(hl[:,0]/hl[:,1])**2
    hl=pd.Series(hl,index=series.index)
    beta=pd.stats.moments.rolling_sum(hl,window=2)
    beta=pd.stats.moments.rolling_mean(beta,window=sl)
    return beta.dropna()
#---------------------------------------
def getGamma(series):
    h2=pd.stats.moments.rolling_max(series['High'],window=2)
    l2=pd.stats.moments.rolling_min(series['Low'],window=2)
    gamma=np.log(h2.values/l2.values)**2
    gamma=pd.Series(gamma,index=h2.index)
    return gamma.dropna()
#---------------------------------------
def getAlpha(beta,gamma):
    den=3-2*2**.5
    alpha=(2**.5-1)*(beta**.5)/den
    alpha-=(gamma/den)**.5
    alpha[alpha<0]=0 # установить отрицательные альфа равными 0
                     # (см. с. 727 работы)
    return alpha.dropna()
#---------------------------------------
def corwinSchultz(series,sl=1):
    # Примечание: S<0, только если alpha<0
    beta=getBeta(series,sl)
    gamma=getGamma(series)
    alpha=getAlpha(beta,gamma)
    spread=2*(np.exp(alpha)-1)/(1+np.exp(alpha))
    startTime=pd.Series(series.index[0:spread.shape[0]],index=spread.index)
    spread=pd.concat([spread,startTime],axis=1)
    spread.columns=['Spread','Start_Time'] # 1st loc used to compute beta
    return spread
```

Обратите внимание, что волатильность не появляется в окончательных уравнениях Корвина—Шульца. Причина в том, что волатильность была заменена оценщиком волатильности на основе максимумов-минимумов. В качестве побочного продукта этой модели мы можем получить волатильность Беккера—Паркинсона, как показано в листинге 19.2.

Листинг 19.2. Оценивание волатильности для цен максимум-минимум (high-low)

```
def getSigma(beta,gamma):
    k2=(8/np.pi)**.5
    den=3-2*2**.5
    sigma=(2**-.5-1)*beta**.5/(k2*den)
    sigma+=(gamma/(k2**2*den))**.5
    sigma[sigma<0]=0
    return sigma
```

Эта процедура особенно полезна на рынке корпоративных облигаций, где нет централизованной книги заказов, и торги происходят через заявки на конкурентной основе (BWIC[1]). Результирующий признак, спред между ценой спроса и ценой предложения S, может быть рекурсивно оценен в скользящем окне, а значения могут быть сглажены с помощью фильтра Калмана.

19.4. Второе поколение: стратегические модели сделок

Микроструктурные модели второго поколения сосредоточены на понимании и измерении неликвидности. Неликвидность является важным информативным признаком в финансовых моделях МО, поскольку это риск, который имеет связанную с ним премию. Эти модели имеют более сильную теоретическую основу, чем модели первого поколения, поскольку они объясняют торговлю как стратегическое взаимодействие между информированными и неинформированными трейдерами. При этом они обращают внимание на ориентированный (по знаку) объем и несбалансированность потока ордеров.

Большинство этих признаков оцениваются с помощью регрессий. На практике я заметил, что t-значения, связанные с этими микроструктурными оценкáми, более информативны, чем сами (средние) оценки. Хотя в литературе об этом наблюдении не упоминается, есть хороший аргумент в пользу предпочтения признаков, основанных на t-значениях, по сравнению с признаками, основанными на средних значениях: t-значения перешкалируются стандартным отклонением ошибки оценивания, тем самым встраивая еще одну размерность информации, отсутствующей в средних оценках.

[1] BWIC (bid wanted in competition, требуется заявка на конкурентной основе) — ситуация, когда институциональный инвестор представляет свой список заявителей на облигации различным дилерам ценных бумаг, дилеры делают свои заявки на указанные бумаги, и с теми, кто предлагает самые высокие заявки, заключаются контракты. — *Примеч. науч. ред.*

19.4.1. Лямбда Кайла

В публикации Kyle [1985] представлена следующая стратегическая модель сделки. Рассмотрим рисковый актив с терминальным значением $v \sim N[p_0, \Sigma_0]$, а также двух трейдеров:

- Шумный трейдер, который торгует величиной $u = N[0, \sigma_u^2]$, независимо от v.
- Информированный трейдер, который знает v и который проявляет спрос на величину x через рыночный ордер.

Маркетмейкер наблюдает за общим потоком ордеров $y = x + u$ и соответствующим образом устанавливает цену p. В этой модели маркетмейкеры не способны отличать ордера, исходящие от шумных трейдеров и от информированных трейдеров. Они регулируют цены как функцию от несбалансированности потока, так как она может указывать на наличие информированного трейдера. Следовательно, существует положительная связь между изменением цены и несбалансированностью потока ордеров, которая называется влиянием на рынок.

Информированный трейдер догадывается, что маркетмейкер имеет линейную функцию корректировки цены, $p = \lambda y + \mu$, где λ — это обратная мера ликвидности. Прибыли информированного трейдера равняются $\pi = (v - p)x$, они максимизируются в $x = \frac{v - \mu}{2\lambda}$ с условием второго порядка $\lambda > 0$.

С другой стороны, маркетмейкер догадывается, что спрос информированного трейдера является линейной функцией от v: $x = \alpha + \beta v$, из которой вытекает $\alpha = -\frac{\mu}{2\lambda}$ и $\beta = \frac{1}{2\lambda}$. Обратите внимание, что более низкая ликвидность означает более высокую λ, что означает более низкий спрос со стороны информированного трейдера.

Кайл утверждает, что маркетмейкер должен найти равновесие между максимизацией прибыли и эффективностью рынка и что в рамках указанных выше линейных функций единственное возможное решение возникает, когда

$$\mu = p_0;$$

$$\alpha = p_0 \sqrt{\frac{\sigma_u^2}{\sum_0}};$$

$$\lambda = \frac{1}{2} \sqrt{\frac{\sum_0}{\sigma_u^2}};$$

$$\beta = \sqrt{\frac{\sigma_u^2}{\sum_0}}.$$

Наконец, математическое ожидание прибыли информированного трейдера может быть переписано как

$$E[\pi] = \frac{(v - p_0)^2}{2}\sqrt{\frac{\sigma_u^2}{\sum_0}} = \frac{1}{4\lambda}(v - p_0)^2.$$

Из этого следует, что информированный трейдер имеет три источника прибыли:

- Неверное ценообразование ценной бумаги.
- Дисперсия чистого потока ордеров шумного трейдера. Чем выше шум, тем легче информированный трейдер может скрывать свои намерения.
- Обратная величина дисперсии терминальной ценной бумаги. Чем ниже волатильность, тем легче монетизировать неверное ценообразование.

В модели Кайла переменная λ захватывает (объясняет) влияние на цену. Неликвидность увеличивается вместе с неопределенностью относительно v и уменьшается вместе с величиной шума. Как признак, она может быть оценена путем подгонки регрессии

$$\Delta p_t = \lambda(b_t V_t) + \varepsilon_t,$$

где $\{p_t\}$ — это временной ряд цен, $\{b_t\}$ — это временной ряд флагов агрессора, $\{V_t\}$ — временной ряд торгуемых объемов и, следовательно, $\{b_t V_t\}$ — временной ряд ориентированного (по знаку) объема или чистого потока ордеров. На рис. 19.1 построен график лямбд Кайла, оцененных на ряде фьючерса E-mini S&P 500.

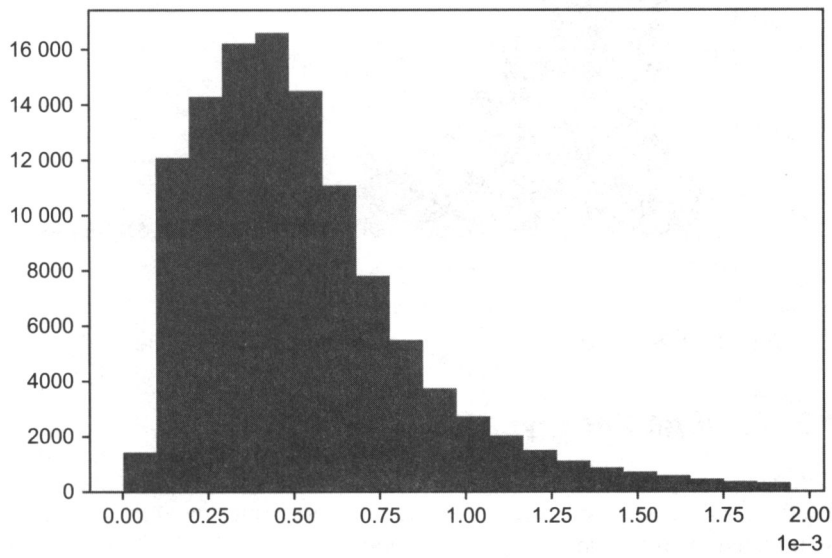


Рис. 19.1. Лямбды Кайла, вычисленные на фьючерсе E-mini S&P 500

19.4.2. Лямбда Амихуда

Амихуд в своей работе (Amihud [2002]) изучает положительную связь между абсолютной доходностью и неликвидностью. В частности, он вычисляет суточный ценовой отклик, связанный с однодолларовым объемом торговли, и утверждает, что его значение — это косвенный индикатор влияния на цену. Одна из возможных реализаций этой идеи имеет вид:

$$|\Delta \log[\tilde{p}_\tau]| = \lambda \sum_{t \in B_\tau} (p_t V_t) + \varepsilon_\tau,$$

где B_τ — это множество сделок, включенных в бар τ, $\tilde{p}_\tau$ — цена закрытия в баре τ, а $p_t V_t$ — долларовый объем, участвующий в сделке $t \in B_\tau$. Несмотря на кажущуюся простоту, в публикации Hasbrouck [2009] было обнаружено, что суточные оценки лямбды Амихуда демонстрируют высокую ранговую корреляцию с внутридневными оценками эффективного спреда. На рис. 19.2 представлена гистограмма лямбд Амихуда, оцененных на ряде фьючерса E-mini S&P 500.

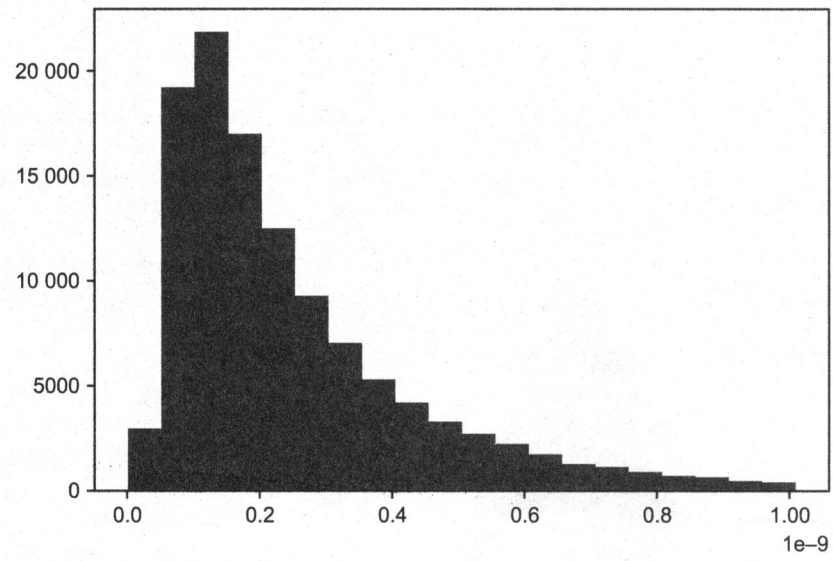


Рис. 19.2. Лямбды Амихуда, оцененные на фьючерсе E-mini S&P 500

19.4.3. Лямбда Хасбрука

Хасбрук в своей публикации (Hasbrouck [2009]) следует идеям Кайла и Амихуда и применяет их для оценивания коэффициента влияния на цену, основываясь на внутридневных транзакционных данных trade-and-quote (TAQ). Для получения байесовой оценки спецификации регрессии он использует генератор выборок по Гиббсу:

$$\log[\tilde{p}_{i,\tau}] - \log[\tilde{p}_{i,\tau-1}] = \lambda_i \sum_{t \in B_{i,\tau}} (b_{i,t} \sqrt{p_{i,t} V_{i,t}}) + \varepsilon_{i,\tau},$$

где $B_{i,\tau}$ — это множество сделок, включенных в бар τ для ценной бумаги i с $i = 1, \ldots, I$, $\tilde{p}_{i,\tau}$ — цена закрытия в баре τ для ценной бумаги i, $b_{i,t} \in \{-1, 1\}$ определяет, была ли сделка $t \in B_{i,\tau}$ инициирована покупкой либо продажей, и $p_{i,t} V_{i,t}$ — долларовый объем, участвующий в сделке $t \in B_{i,\tau}$. Тогда мы можем оценить лямбду λ_i для каждой ценной бумаги и использовать ее в качестве признака, который аппроксимирует эффективную стоимость торговли (влияние на рынок).

В соответствии с большей частью литературы, для отбора тиков Хасбрук рекомендует пятиминутные временные бары. Однако по причинам, рассмотренным в главе 2, более качественные результаты могут быть достигнуты с помощью методов стохастического отбора, синхронизированных с рыночной активностью. На рис. 19.3 построена гистограмма лямбд Хасбрука, оцененных на ряду фьючерса E-mini S&P 500.

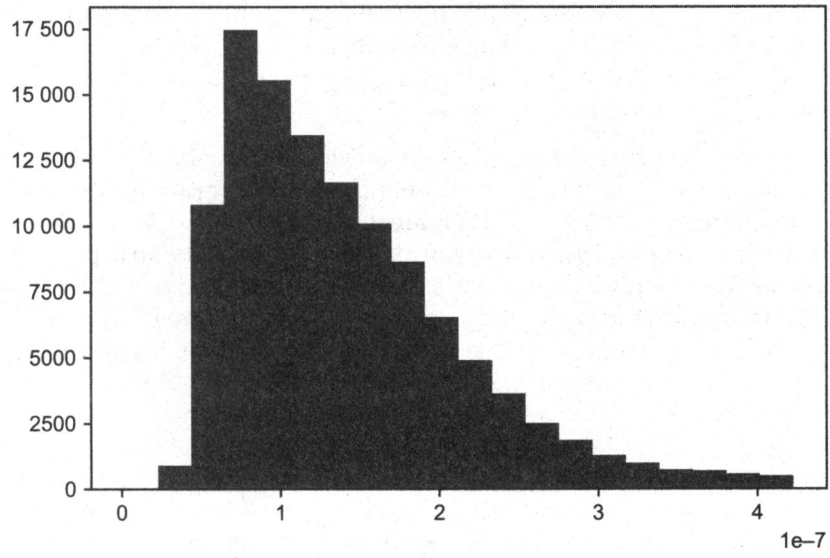


Рис. 19.3. Лямбды Хасбрука, оцененные на фьючерсе E-mini S&P 500

19.5. Третье поколение: модели последовательных сделок

Как мы видели в предыдущем разделе, стратегические модели сделок характеризуются наличием одного информированного трейдера, который может заключать сделки многократно. В этом разделе мы обсудим альтернативную модель,

в которой случайно отобранные трейдеры приходят на рынок последовательно и независимо.

С момента своего появления, модели последовательных сделок стали очень популярны среди маркетмейкеров. Одна из причин заключается в том, что они встраивают источники неопределенности, с которыми сталкиваются поставщики ликвидности, а именно вероятность того, что произошло информационное событие, вероятность того, что такое событие является отрицательным, скорость прибытия шумных трейдеров и скорость прибытия информированных трейдеров. С помощью этих переменных маркетмейкеры должны динамически обновлять котировки и управлять своими запасами.

19.5.1. Вероятность информационно обусловленной торговли

В публикации Easley и соавт. [1996] используются данные сделок для определения вероятности информационно обусловленной торговли (PIN) индивидуальных ценных бумаг. Эта микроструктурная модель рассматривает торговлю как игру между маркетмейкерами и позиционерами, которая повторяется на протяжении многочисленных торговых периодов.

Обозначим цену ценной бумаги через S с текущей стоимостью S_0. Однако после того, как определенная величина новой информации будет встроена в цену, S будет либо S_B (плохой новостью), либо S_G (хорошей новостью). Существует вероятность α, что новая информация поступит в пределах временно́го отрезка анализа, вероятность δ, что новая информация будет плохой, и вероятность $(1 - \delta)$, что новая информация будет хорошей. Эти авторы доказывают, что математическое ожидание цены ценной бумаги может быть вычислено в момент времени t как

$$\mathrm{E}[S_t] = (1-\alpha_t)S_0 + \alpha_t[\delta_t S_B + (1-\delta_t)S_G].$$

Подчиняясь распределению Пуассона, информированные трейдеры прибывают со скоростью μ, и неинформированные трейдеры — со скоростью ε. Тогда для того, чтобы избежать убытков от информированных трейдеров, маркетмейкеры выходят в безубыток на уровне B_t цены спроса

$$\mathrm{E}[B_t] = E[S_t] - \frac{\mu\alpha_t\delta_t}{\varepsilon + \mu\alpha_t\delta_t}(\mathrm{E}[S_t] - S_B).$$

И безубыточный уровень A_t цены предложения в момент времени t должен быть

$$\mathrm{E}[A_t] = \mathrm{E}[S_t] + \frac{\mu\alpha_t(1-\delta_t)}{\varepsilon + \mu\alpha_t(1-\delta_t)}(S_G - \mathrm{E}[S_t]).$$

Отсюда следует, что безубыточный спред между ценой спроса и ценой предложения определяется как

$$\mathrm{E}[A_t - B_t] = \frac{\mu\alpha_t(1-\delta_t)}{\varepsilon + \mu\alpha_t(1-\delta_t)}(S_G - \mathrm{E}[S_t]) + \frac{\mu\alpha_t\delta_t}{\varepsilon + \mu\alpha_t\delta_t}(\mathrm{E}[S_t] - S_B).$$

В стандартном случае при $\delta_t = \frac{1}{2}$ мы получаем

$$\delta_t = \frac{1}{2} \Rightarrow \mathrm{E}[A_t - B_t] = \frac{\alpha_t\mu}{\alpha_t\mu + 2\varepsilon}(S_G - S_B).$$

Это уравнение говорит нам о том, что критический фактор, определяющий ценовой диапазон, в котором маркетмейкеры предоставляют ликвидность, равен

$$\mathrm{PIN}_t = \frac{\alpha_t\mu}{\alpha_t\mu + 2\varepsilon}.$$

Нижний индекс t указывает на то, что вероятности α и δ оцениваются в этот момент времени. Авторы применяют байесов процесс обновления для встраивания информации после того, как каждая сделка прибывает на рынок.

Для того чтобы определить значение PIN_t, мы должны оценить четыре ненаблюдаемых параметра, а именно $\{\alpha, \delta, \mu, \varepsilon\}$. Подход с максимально правдоподобным оцениванием должен выполнить подгонку смеси из трех распределений Пуассона,

$$\begin{aligned}\mathrm{P}[V^B, V^S] = (1-\alpha)\mathrm{P}[V^B, \varepsilon]\mathrm{P}[V^S, \varepsilon] + \alpha(\delta\mathrm{P}[V^B, \varepsilon]\mathrm{P}[V^S, \mu+\varepsilon] + \\ + (1-\delta)\mathrm{P}[V^B, \mu+\varepsilon]\mathrm{P}[V^S, \varepsilon]),\end{aligned}$$

где V^B — это объем, торгуемый против цены предложения (сделок, инициированных покупкой), и V^S — объем, торгуемый против цены спроса (сделок, инициированных продажей).

19.5.2. Объемно-синхронизированная вероятность информированной торговли

В работе Easley и соавт. [2008] было доказано, что

$$E[V^B - V^S] = (1-\alpha)(\varepsilon - \varepsilon) + \alpha(1-\delta)(\varepsilon - (\mu+\varepsilon)) + \alpha\delta(\mu + \varepsilon - \varepsilon) = \alpha\mu(1-2\delta)$$

и в частности, что для достаточно большого значения μ

$$E[|V^B - V^S|] \approx \alpha\mu.$$

В публикации Easley и соавт. [2011] предложена высокочастотная оценка вероятности PIN, которую авторы назвали объемно-синхронизированной вероятностью информированной торговли (volume-synchronized probability of informed trading, VPIN). Данная процедура использует *объемные часы*, которые синхронизируют отбор данных с рыночной активностью, в соответствии с тем, как зафиксировано объемом (см. главу 2). Тогда мы можем оценить

$$\frac{1}{n}\sum_{\tau=1}^{n}\left|V_\tau^B - V_\tau^S\right| \approx \alpha\mu,$$

где V_τ^B — это сумма объемов от сделок, инициированных покупками в пределах объемного бара τ, V_τ^S — это сумма объемов, инициированных продажами в пределах объемного бара τ, и n — число баров, использованных для порождения этой оценки. Поскольку все объемные бары имеют одинаковый размер V, мы знаем, что по конструкции

$$\frac{1}{n}\sum_{\tau=1}^{n}\left(V_\tau^B - V_\tau^S\right) = V = \alpha\mu + 2\varepsilon.$$

Следовательно, вероятность PIN можно оценить высокочастотно как

$$VPIN_\tau = \frac{\sum_{\tau=q}^{n}\left|V_\tau^B - V_\tau^S\right|}{\sum_{\tau=q}^{n}\left(V_\tau^B - V_\tau^S\right)} = \frac{\sum_{\tau=q}^{n}\left|V_\tau^B - V_\tau^S\right|}{nV}.$$

Дополнительные сведения и тематические исследования вероятности VPIN см. в публикации Easley и соавт. [2013]. Используя линейные регрессии, в публикации Andersen and Bondarenko [2013] авторы пришли к выводу, что вероятность VPIN не является хорошим предиктором волатильности. Однако ряд исследований показали, что VPIN действительно обладает предсказательной силой. Вот лишь несколько публикаций: Abad and Yague [2012], Bethel и соавт. [2012], Cheung и соавт. [2015], Kim и соавт. [2014], Song и соавт. [2014], Van Ness и соавт. [2017] и Wei и соавт. [2013]. В любом случае, линейная регрессия — это метод, который уже был известен математикам в XVIII веке (Stigler [1981]), и экономисты не должны удивляться, когда он не распознает сложные нелинейные закономерности на финансовых рынках XXI века.

19.6. Дополнительные признаки из микроструктурных совокупностей данных

Признаки, которые мы изучали в разделах 19.3–19.5, были предложены теорией рыночной микроструктуры. Вдобавок мы должны рассмотреть альтернативные признаки, которые, хотя и не предлагаются теорией, по нашим подозрениям, несут

важную информацию о том, как участники рынка работают, и об их намерениях в будущем. При этом мы будем использовать мощь машинно-обучающихся алгоритмов, которые могут научиться использовать эти признаки, не будучи специально направляемыми теорией.

19.6.1. Распределение объемов ордеров

В публикации Easley и соавт. [2016] изучается частота сделок в расчете на размер сделки и обнаруживается, что сделки с округленными размерами — аномально часты. Например, частотности быстро затухают как функция от размера сделки, за исключением округленных размеров {5, 10, 20, 25, 50, 100, 200, ...}. Эти авторы приписывают этот феномен так называемым «мышиным» или «GUI»-трейдерам, то есть трейдерам-людям, которые отправляют ордера, нажимая кнопки на мониторе с графическим пользовательским интерфейсом (graphical user interface, GUI). В случае с E-mini S&P 500, например, размер 10 встречается в 2,9 раза чаще, чем размер 9, размер 50 в 10,9 раза более вероятен, чем размер 49, размер 100 встречается в 16,8 раза чаще, чем размер 99, размер 200 в 27,2 раза более вероятен, чем размер 199, размер 250 встречается в 32,5 раза чаще, чем размер 249, размер 500 встречается в 57,1 раза чаще, чем размер 499. Такие закономерности не типичны для «кремниевых трейдеров», которые обычно запрограммированы на рандомизацию сделок, для того чтобы скрывать свой след на рынках.

Полезным признаком может быть определение нормальной частоты сделок с округленными размерами и отслеживание отклонений от их математического ожидания. Машинно-обучающийся алгоритм может, например, определять, связана ли более крупная, чем обычно, доля сделок округленного размера с трендами, поскольку трейдеры-люди склонны делать ставки с фундаментальным ви́дением, мнением или убеждением. И наоборот, более низкая, чем обычно, доля сделок округленного размера может увеличивать вероятность того, что цены будут двигаться вбок, поскольку кремниевые трейдеры обычно не имеют долгосрочного ви́дения.

19.6.2. Скорости отмены, лимитные и рыночные ордера

В публикации Eisler и соавт. [2012] изучается влияние рыночных ордеров, лимитных ордеров и отмен котировок. Эти авторы считают, что акции с малым тиком реагируют на эти события иначе, чем акции с большим тиком. Они приходят к выводу, что определение размера этих величин имеет прямое отношение к моделированию динамики спреда между ценой спроса и ценой предложения.

В исследовании Easley и соавт. [2012] также утверждается, что большие скорости отмены котировок могут свидетельствовать о низкой ликвидности, так как участники публикуют котировки, которые не намерены исполняться. Они рассматривают четыре категории хищнических алгоритмов:

- **Наводнители котировками** (quote stuffers): они занимаются «латентным арбитражем». Их стратегия предусматривает наводнение бирж сообщениями с единственной целью замедлить конкурирующие алгоритмы, вынуждая их выполнять разбор этих сообщений, как проигнорировать которые знают только инициаторы.
- **Соблазнители котировками** (quote danglers): эта стратегия отправляет котировки, которые заставляют выжатого трейдера преследовать цену против его интересов. В публикации O'Hara [2011] представлены доказательства их подрывной деятельности.
- **Выжиматели ликвидности** (liquidity squeezers): когда огорченный крупный инвестор вынужден откатывать свою позицию назад, хищнические алгоритмы торгуют в том же направлении, сливая как можно больше ликвидности. В результате цены проскакивают мимо, и выжиматели ликвидности получают прибыль (Carlin и соавт. [2007]).
- **Стайные охотники** (pack hunters): хищники, охотящиеся независимо друг от друга, узнают о деятельности друг друга и образуют стаю для того, чтобы максимизировать шансы вызвать каскадный эффект (Donefer [2010], Fabozzi и соавт. [2011], Jarrow and Protter [2011]). В публикации NANEX [2011] показано то, что, по всей видимости, является стайными охотниками, которые форсируют стоп-лосс. Хотя их индивидуальные действия слишком малы, чтобы вызвать подозрения у регулятора, их коллективные действия могут быть рыночно-манипулятивными. Когда это так, очень трудно доказать их сговор, поскольку они координируют свои действия децентрализованно и спонтанно.

Эти хищнические алгоритмы задействуют отмены котировок и различные типы ордеров с целью неблагоприятного выбора маркетмейкеров. Они оставляют разнящиеся сигнатуры в торговой истории, и измерение скоростей отмены котировок, лимитных и рыночных ордеров может стать основой для полезных признаков, информирующих об их намерениях.

19.6.3. Исполнительные алгоритмы TWAP

В публикации Easley и соавт. [2012] продемонстрировано, как распознать наличие исполнительных алгоритмов, нацеленных на конкретную средневзвешенную по времени цену (time-weighted average price, TWAP). Алгоритм TWAP – это алгоритм, разрезающий большой ордер на малые ордера, которые отправляются через регулярные промежутки времени в попытке достичь заранее определенной средневзвешенной по времени цены. Эти авторы берут выборку фьючерсных сделок E-mini S&P 500 в период с 7 ноября 2010 года по 7 ноября 2011 года. Они делят сутки на 24 часа, и за каждый час они добавляют объем, торгуемый в каждую секунду, независимо от минуты. Затем они строят график этих агрегированных объемов как поверхность, где ось x назначается объему в секунду, ось y

назначается часу суток, а ось *z* назначается агрегированному объему. Такой анализ позволяет нам увидеть распределение объема внутри каждой минуты в течение суток и искать низкочастотных трейдеров, исполняющих свои массивные ордера в хронологическом времени-пространстве. Наибольшая концентрация объема внутри минуты, как правило, происходит в течение первых нескольких секунд, почти каждый час в течение суток. Это особенно верно в 00:00–01:00 GMT (вокруг открытия азиатских рынков), 05:00–09:00 GMT (вокруг открытия британских и европейских фондовых бирж), 13:00–15:00 GMT (вокруг открытия фондовых бирж США) и в 20:00–21:00 GMT (ближе к закрытию фондовых бирж США).

Полезным признаком МО может быть оценивание несбалансированности ордеров в начале каждой минуты и определение наличия устойчивой компоненты. Затем это может быть использовано для опережения крупных институциональных инвесторов, в то время как более крупная порция их ордера TWAP еще будет находиться на рассмотрении.

19.6.4. Опционные рынки

В публикации Muravyev и соавт. [2013] используется микроструктурная информация из американских акций и опционов для изучения событий, когда два рынка имеют разногласия. Авторы характеризуют такие разногласия, выводя базовый диапазон спроса-предложения, предполагаемый котировками с паритетом цен пут-колл, и сравнивая его с фактическим диапазоном спроса-предложения на акцию. Они приходят к выводу, что разногласия, как правило, разрешаются в пользу акционных котировок, а это означает, что опционные *котировки* не содержат экономически значимой информации. В то же время авторы все-таки обнаруживают, что опционные сделки содержат информацию, не включенную в акционную цену. Эти выводы не станут сюрпризом для портфельных менеджеров, привыкших торговать относительно неликвидными продуктами, включая опционы на акции. Котировки могут оставаться иррациональными в течение длительных периодов времени, даже если разреженные цены информативны.

В публикации Cremers и Weinbaum [2010] обнаружено, что акции с относительно дорогими ценами колл (акции с высоковолатильным спредом и высокой изменчивостью в спреде волатильности) опережают по результативности акции с относительно дорогими ценами пут (акции с низковолатильным спредом и низкой изменчивостью в спреде волатильности) на 50 базисных пунктов в неделю. Такая степень предсказуемости выше, когда опционная ликвидность высока, а акционная ликвидность низка.

В русле этих наблюдений полезные признаки могут быть извлечены из вычисления подразумеваемой ценами пут-колл цены акции, полученной из опционных сделок. Фьючерсные цены представляют только среднее или математическое ожидание в будущем. Но опционные цены позволяют нам получать полное распределение ценообразуемых исходов. Алгоритм МО может искать закономерности по грече-

ским буквам, зафиксированным по котировочным ценам-страйк и датам истечения срока действия.

19.6.5. Внутрирядовая корреляция ориентированного (по знаку) потока ордеров

В публикации Toth и соавт. [2011] изучается ориентированный (по знаку) поток ордеров на акции Лондонской фондовой биржи и обнаруживается, что знаки ордеров положительно автокоррелированы на протяжении многих дней. Они приписывают это наблюдение двум кандидатным объяснениям: скучиванию и дроблению ордеров. Они приходят к выводу, что в сроки менее нескольких часов устойчивость потока ордеров в подавляющем большинстве случаев связана с дроблением, а не со скучиванием.

С учетом того, что теория рыночной микроструктуры связывает устойчивость несбалансированности потока ордеров с наличием информированных трейдеров, имеет смысл измерять силу такой устойчивости через внутрирядовую корреляцию ориентированных (по знаку) объемов. Такой признак дополнил бы признаки, изученные нами в разделе 19.5.

19.7. Что такое микроструктурная информация?

Позвольте мне завершить эту главу, обратившись к тому, что я считаю основным недостатком в литературе по микроструктуре рынка. Большинство статей и книг по этой теме изучают асимметричную информацию и то, как стратегические агенты пользуются ею для получения прибыли от маркетмейкеров. Но каково именно определение понятия информации в контексте трейдинга? К сожалению, общепринятого определения информации в микроструктурном смысле не существует, и в литературе это понятие используется на удивление свободно и довольно неформально (Lopez de Prado [2017]). В этом разделе предлагается правильное определение понятия информации, основанное на обработке сигналов, которое может быть применено к микроструктурным исследованиям.

Рассмотрим признаковую матрицу $X = \{X_t\}_{t=1,\ldots,T}$, которая содержит информацию, обычно используемую маркетмейкерами для определения того, должны ли они предоставлять ликвидность на конкретном уровне или отменять свои пассивные котировки. Например, столбцы могут содержать все признаки, описанные в этой главе, такие как VPIN, лямбду Кайла, скорости отмены и т. д. Матрица X имеет одну строку для каждой точки принятия решений. Например, маркетмейкер может пересмотреть решение либо о предоставлении ликвидности, либо о выходе из рынка всякий раз, когда торгуется 10 000 контрактов или когда проис-

ходит значительное ценовое изменение (вспомните методы отбора из главы 2) и т. д. Во-первых, мы получаем массив $y = \{y_t\}_{t=1,\ldots,T}$, назначающий метку 1 наблюдению, которое привело к прибыли маркетмейкера, и маркирующий как 0 наблюдение, которое привело к убытку маркетмейкера (см. главу 3 для методов маркировки). Во-вторых, мы выполняем подгонку классификатора на тренировочном подмножестве (X, y). В-третьих, по мере прибытия новых вневыборочных наблюдений $t > T$ мы применяем подогнанный классификатор для предсказания метки $\hat{y}_\tau = E_\tau[y_\tau \mid X]$. В-четвертых, мы выводим перекрестно-энтропийную потерю этих предсказаний, L_τ, как описано в главе 9, раздел 9.4. В-пятых, мы выполняем подгонку оценщика ядерной плотности (kernel density estimator, KDE) на массиве отрицательных перекрестно-энтропийных потерь, $\{-L_t\}_{t=T+1,\ldots,\tau}$ для того чтобы получить его кумулятивную функцию распределения, F. В-шестых, мы оцениваем микроструктурную информацию в момент времени t как $\phi_\tau = F[-L_\tau]$, где $\phi_\tau \in (0,1)$.

Эту микроструктурную информацию можно понимать как сложность, с которой сталкиваются модели маркетмейкеров принятия решений. В условиях нормального рынка маркетмейкеры производят *информированные прогнозы* с низкой перекрестно-энтропийной потерей и могут получать прибыль от предоставления ликвидности для позиционеров. Вместе с тем в присутствии (асимметрично) информированных трейдеров маркетмейкеры производят *неинформированные прогнозы*, о которых свидетельствует высокая перекрестно-энтропийная потеря, и маркетмейкеры выбираются в неблагоприятных условиях. Другими словами, микроструктурная информация может быть определена и измерена только относительно предсказательной силы маркетмейкеров. Из этого вытекает, что $\{\phi_\tau\}$ должен стать важным признаком в вашем инструментарии финансового машинного обучения.

Рассмотрим события молниеносного обвала фондового рынка США 6 мая 2010 года. Маркетмейкеры ошибочно предсказывали, что их пассивные котировки, сидящие на цене спроса, могут быть исполнены и проданы обратно на более высоком уровне. Молниеносный обвал был вызван не одним неточным предсказанием, а накоплением тысяч предсказательных ошибок (Easley и соавт. [2011]). Если бы маркетмейкеры следили за ростом перекрестно-энтропийной потери своих предсказаний, то они бы признали наличие информированных трейдеров и опасно растущую вероятность неблагоприятного выбора. Это позволило бы им расширить спред между ценой спроса и ценой предложения до уровней, которые остановили бы несбалансированность потока ордеров, поскольку продавцы больше не были бы готовы продавать с теми скидками. Вместо этого маркетмейкеры продолжали предоставлять ликвидность продавцам на чрезвычайно щедрых уровнях, до тех пор пока в конечном итоге они не были остановлены принудительно, что вызвало кризис ликвидности, который шокировал рынки, регуляторов и привел в замешательство ученых в течение нескольких месяцев и лет.

Упражнения

19.1. На основе временно́го ряда фьючерсных тиковых данных E-mini S&P 500:

(а) Примените тиковое правило для получения ряда, состоящего из знаков сделок.

(б) Сравните со стороной агрессора, как это предусмотрено CME[1] (тег FIX 5797). Какова точность тикового правила?

(в) Отберите случаи, когда тег FIX 5797 расходится с тиковым правилом:

i) видите ли вы что-нибудь особенное, что могло бы объяснить это расхождение?

ii) связаны ли эти расхождения с большими ценовыми скачками? Или высокими скоростями отмены? Или тонкими котировочными размерами?

iii) когда эти разногласия более вероятны: в периоды высокой или низкой активности рынка?

19.2. Вычислите модель Ролла на временны́х рядах тиковых данных фьючерса E-mini S&P 500.

(а) Каковы оценочные значения σ_u^2 и c?

(б) Зная, что этот контракт является одним из самых ликвидных продуктов в мире и что он торгуется с максимально крутым спредом между ценой спроса и ценой предложения, соответствуют ли эти значения вашим ожиданиям?

19.3. Вычислите оценщика волатильности максимум-минимум (раздел 19.3.3) на фьючерсе E-mini S&P 500.

(а) Чем он отличается от среднеквадратического отклонения финансовых возвратов, рассчитанных от цены закрытия к цене закрытия, если использовать недельные значения?

(б) Чем он отличается от среднеквадратического отклонения финансовых возвратов, рассчитанных от цены закрытия к цене закрытия, если использовать суточные значения?

(в) Чем он отличается от среднеквадратического отклонения финансовых возвратов, рассчитанных от цены закрытия к цене закрытия, если использовать долларовые бары, в среднем 50 баров в день?

---

[1] CME Group Inc. — крупнейший североамериканский рынок финансовых деривативов, построенный путем объединения ведущих бирж Чикаго и Нью-Йорка. — *Примеч. ред.*

19.4. Примените оценщик Корвина—Шульца к суточному ряду фьючерса E-mini S&P 500.

(а) Каков ожидаемый спред между ценой спроса и ценой предложения?

(б) Какова предполагаемая волатильность?

(в) Совместимы ли эти оценки с более ранними результатами из упражнений 19.2 и 19.3?

19.5. Рассчитайте лямбду Кайла из:

(а) Тиковых данных.

(б) Временного ряда долларовых баров на фьючерсе E-mini S&P 500, где:

i) b_t — взвешенное по объему среднее значение знаков сделок;

ii) V_t — сумма объемов в этом баре;

iii) p_t — изменение в цене между двумя барами подряд.

19.6. Повторите упражнение 19.5, на этот раз применив лямбду Хасбрука. Совместимы ли результаты?

19.7. Повторите упражнение 19.5, на этот раз применив лямбду Амихуда. Совместимы ли результаты?

19.8. Сформируйте временной ряд объемных баров для фьючерса E-mini S&P 500.

(а) Вычислите ряд вероятностей VPIN на 6 мая 2010 года (молниеносный обвал фондового рынка США).

(б) Постройте график ряда VPIN и цен. Что вы видите?

19.9. Вычислите распределение размеров ордеров на фьючерс E-mini S&P 500:

(а) За весь период.

(б) За 6 мая 2010 года.

(в) Выполните статистическую проверку Колмогорова—Смирнова на обоих распределениях. Существенно ли они отличаются на 95 %-ном уровне достоверности?

19.10. Вычислите временны́е ряды суточных скоростей отмены котировок и доли рыночных ордеров на совокупности данных фьючерсов E-mini S&P 500.

(а) Какова корреляция между двумя рядами? Является ли она статистически значимой?

(б) Какова корреляция между двумя рядами и суточной волатильностью? Это то, чего вы ожидали?

19.11. На тиковых данных фьючерса E-mini S&P 500:

(а) Вычислите распределение объема, исполняемого в пределах первых 5 секунд каждой минуты.

(б) Вычислите распределение объема, исполняемого каждую минуту.

(в) Примените статистическую проверку Колмогорова—Смирнова на обоих распределениях. Существенно ли они отличаются на 95 %-ном уровне достоверности?

19.12. На тиковых данных фьючерса E-mini S&P 500:

(а) Вычислите внутрирядовую корреляцию первого порядка для ориентированных (по знаку) объемов.

(б) Является ли она статистически значимой на 95 %-ном уровне достоверности?

Часть 5

РЕЦЕПТЫ ВЫСОКО-ПРОИЗВОДИТЕЛЬНЫХ ВЫЧИСЛЕНИЙ

20 Мультиобработка и векторизация

20.1. Актуальность

Мультиобработка, или многопроцессорная обработка, имеет крайне важное значение для машинного обучения. Алгоритмы МО требуют больших вычислительных ресурсов и эффективного использования всех процессоров, серверов и кластеров. По этой причине большинство функций, представленных в этой книге, были спроектированы для асинхронной мультиобработки. Например, мы часто использовали таинственную функцию `mpPandasObj`, но ни разу ее не определили. В этой главе мы объясним, что эта функция делает. Более того, мы детально изучим способы разработки мультиобрабатывающих механизмов. Структура программ, представленных в этой главе, не зависит от аппаратной архитектуры, используемой для их выполнения, используем ли мы ядра одного сервера или же ядра, распределенные между несколькими взаимосвязанными серверами (к примеру, в высокопроизводительном вычислительном кластере или облаке).

20.2. Пример векторизации

Векторизация, также именуемая массиво-ориентированным программированием или программированием, оптимизированным для обработки массивов, является простейшим примером параллелизации, при котором операция применяется сразу ко всему множеству значений. В качестве минимального примера предположим, что вам нужно выполнить исчерпывающий поиск в трехмерном пространстве с двумя узлами на размерность. Невекторизованная реализация этого декартова произведения будет выглядеть как в листинге 20.1. Как бы выглядел этот исходный код, если бы вам пришлось выполнять поиск в 100 размерностях или если бы число размерностей определялось пользователем во время выполнения?

Листинг 20.1. Стандартное декартово произведение

```
# Декартово произведение словаря списков
dict0={'a':['1','2'],'b':['+','*'],'c':['!','@']}
for a in dict0['a']:
```

```
    for b in dict0['b']:
        for c in dict0['c']:
            print {'a':a,'b':b,'c':c}
```

Векторизованное решение заменит все явно заданные итераторы (например, циклы `for`) операциями матричной алгебры или скомпилированными итераторами либо генераторами. Листинг 20.2 реализует векторизованную версию листинга 20.1. Векторизованная версия предпочтительна по четырем причинам: 1) медленные вложенные циклы `for` заменяются быстрыми итераторами, 2) исходный код логически выводит размерность решетки из размерности `dict0`, 3) мы могли бы запустить 100 размерностей без необходимости модифицировать исходный код, иначе нам потребуется 100 циклов `for`, и 4) под капотом Python может выполнять операции на языках C или C++.

Листинг 20.2. Векторизированное декартово произведение

```
# Декартово произведение словаря списков
from itertoolsimport izip,product
dict0={'a':['1','2'],'b':['+','*'],'c':['!','@']}
jobs=(dict(izip(dict0,i)) for i in product(*dict0.values()))
for i in jobs:print i
```

20.3. Однопоточность против многопоточности и мультиобработки

Современный компьютер имеет многократное число процессорных разъемов. Каждый процессор имеет много ядер (процессоров), и каждое ядро имеет несколько потоков. Многопоточность — это метод, при котором несколько приложений выполняются параллельно в двух или более потоках на одном ядре. Одним из преимуществ многопоточности является то, что, поскольку приложения совместно используют одно и то же ядро, они совместно используют одно и то же пространство памяти. Это создает риск того, что несколько приложений могут одновременно записывать данные в один и тот же участок пространства памяти. Для того чтобы этого не произошло, глобальная блокировка интерпретатора (global interpreter lock, GIL) одновременно назначает доступ на запись одному потоку на ядро. В GIL многопоточность Python ограничена одним потоком на процессор. По этой причине Python решает задачу параллелизма через мультиобработку, а не через фактическую многопоточность. Процессоры не используют одно и то же пространство памяти совместно, следовательно, мультиобработка не рискует тем, что запись будет произведена в один и тот же участок пространства памяти; однако она так же затрудняет совместное использование объектов между процессами.

Функции Python, реализованные для работы в одном потоке, будут использовать только часть мощности современного компьютера, сервера или кластера. Давайте рассмотрим пример того, как простая задача может выполняться неэффективно при

реализации для однопотокового выполнения. Листинг 20.3 находит самое раннее время, когда 10 000 гауссовых процессов длиной 1000 касаются симметричного двойного барьера шириной 50-кратного среднеквадратического отклонения.

Листинг 20.3. Однопоточная реализация одноразового касания двойного барьера

```
import numpy as np
#--------------------------------------
def main0():
    # Траекторная зависимость: последовательная реализация
    r=np.random.normal(0,.01,size=(1000,10000))
    t=barrierTouch(r)
    return
#--------------------------------------
def barrierTouch(r,width=.5):
    # найти индекс самого раннего касания барьера
    t,p={},np.log((1+r).cumprod(axis=0))
    for j in xrange(r.shape[1]): # пройтись по столбцам
        for i in xrange(r.shape[0]): # пройтись по строкам
            if p[i,j]>=width or p[i,j]<=-width:
                t[j]=i
                continue
    return t
#--------------------------------------
if __name__=='__main__':
    import timeit
    print min(timeit.Timer('main0()',setup='from __main__ import main0').
              repeat(5,10))
```

Сравните эту реализацию с листингом 20.4. Теперь исходный код разбивает предыдущую задачу на 24 подзадачи, по одной на процессор. Затем подзадачи выполняются асинхронно-параллельно с использованием 24 процессоров. При выполнении того же исходного кода в кластере с 5000 процессорами истекшее время составит около 1/5000 однопоточной реализации.

Листинг 20.4. Мультиобрабатывающая реализация одноразового касания двойного барьера

```
import numpy as np
import multiprocessing as mp
#--------------------------------------
def main1():
    # Траекторная зависимость: многопоточная реализация
    r,numThreads=np.random.normal(0,.01,size=(1000,10000)),24
    parts=np.linspace(0,r.shape[0],min(numThreads,r.shape[0])+1)
    parts,jobs=np.ceil(parts).astype(int),[]
    for i in xrange(1,len(parts)):
        jobs.append(r[:,parts[i-1]:parts[i]]) # параллельные задания
    pool,out=mp.Pool(processes=numThreads),[]
    outputs=pool.imap_unordered(barrierTouch,jobs)
    for out_ in outputs: out.append(out_) # асинхронный отклик
```

```
    pool.close();pool.join()
    return
#-------------------------------------
if __name__=='__main__':
    import timeit
    print min(timeit.Timer('main1()',setup='from __main__ import main1').
              repeat(5,10))
```

Более того, вы можете реализовать тот же исходный код для мультиобработки векторизованной функции, как мы сделали с функцией `applyPtSlOnT1` в главе 3, где параллельные процессы исполняют подпрограммы, включающие векторизованные объекты библиотеки pandas. Благодаря этому вы достигнете двух уровней параллелизации сразу. Но зачем останавливаться на достигнутом? Вы могли бы достичь трех уровней параллелизации одновременно, запустив мультиобработочные экземпляры векторизованного кода в высокопроизводительном вычислительном кластере, где каждый узел кластера обеспечивает третий уровень параллелизации. В следующих далее разделах мы объясним, как работает мультиобработка.

20.4. Атомы и молекулы

При подготовке заданий для параллелизации полезно проводить грань между атомами и молекулами. Атомы — это неделимые задачи. Вместо того чтобы выполнять все эти задачи последовательно в одном потоке, мы хотим сгруппировать их в молекулы, которые могут обрабатываться параллельно с использованием нескольких процессоров. Каждая молекула представляет собой подмножество атомов, которые будут обрабатываться последовательно, с помощью функции обратного вызова, используя единственный поток. Параллелизация происходит на молекулярном уровне.

20.4.1. Линейные подразделы

Простейший способ формирования молекул состоит в подразделении списка атомов на подмножества одинакового размера, где число подмножеств — это минимум между числом процессоров и числом атомов. Для N подмножеств нам нужно найти $N+1$ индексов, которые охватывают подразделы. Эта логика продемонстрирована в листинге 20.5.

Листинг 20.5. Функция linParts

```
import numpy as np
#-------------------------------------
def linParts(numAtoms,numThreads):
    # подразделение атомов с помощью одного цикла
    parts=np.linspace(0,numAtoms,min(numThreads,numAtoms)+1)
    parts=np.ceil(parts).astype(int)
    return parts
```

Очень часто встречаются операции, включающие два вложенных цикла. Например, вычисление ряда SADF (глава 17), оценивание нескольких касаний барьера (глава 3) или вычисление ковариационной матрицы на невыровненном ряде. В таких ситуациях линейное подразделение атомарных задач было бы неэффективным, так как некоторые процессоры должны были бы решать гораздо большее число операций, чем другие, и время расчета будет зависеть от самой тяжелой молекулы. Частичное решение состоит в том, чтобы подразделить атомарные задачи на некоторое число заданий, кратных числу процессоров, а затем загрузить очередь заданий с тяжелыми молекулами в начале. Благодаря этому легкие молекулы будут назначены процессорам, которые сначала завершат обработку тяжелых молекул, оставляя все процессоры занятыми до тех пор, пока очередь заданий не будет исчерпана. В следующем далее разделе мы обсудим более полное решение. На рис. 20.1 показано линейное подразделение 20 атомарных задач одинаковой сложности на 6 молекул.

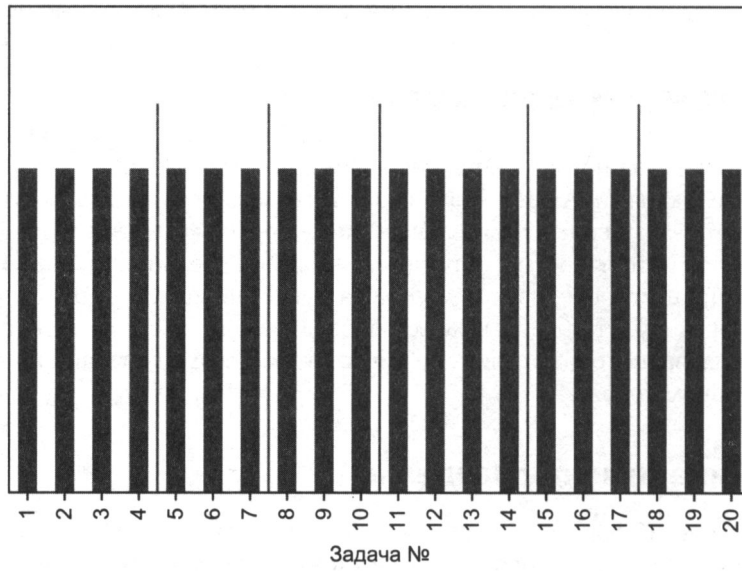


Рис. 20.1. Линейное подразделение 20 атомарных задач на 6 молекул

20.4.2. Подразделения с дважды вложенными циклами

Рассмотрим два вложенных цикла, где внешний цикл итеративно обходит $i = 1, ..., N$ и внутренний цикл итеративно обходит $j = 1, ..., i$. Мы можем упорядочить эти атомарные задачи $\{(i, j) | 1 \le j \le i, i = 1, ..., N\}$ в виде *нижней* треугольной матрицы (включая главную диагональ). Это влечет за собой $\frac{1}{2}N(N-1) + N = \frac{1}{2}N(N+1)$, где $\frac{1}{2}N(N-1)$ — внедиагональные и N — диагональные. Мы хотели бы паралле-

лизовать эти задачи, подразделив атомарные задачи на M подмножеств строк, M, $\{S_m\}_{m=1,...,M}$, каждая из которых составлена примерно из $\frac{1}{2M}N(N+1)$ задач. Следующий ниже алгоритм определяет строки, которые составляют каждое подмножество (молекулу).

Первое подмножество, S_1, состоит из первых r_1 строк, то есть $S_1 = \{1,..., r_1\}$ для общего числа элементов $\frac{1}{2}r_1(r_1+1)$. Тогда r_1 должно удовлетворять условию $\frac{1}{2}r_1(r_1+1) = \frac{1}{2M}N(N+1)$. Решив для r_1, мы получим положительный корень

$$r_1 = \frac{-1+\sqrt{1+4N(N+1)M^{-1}}}{2}.$$

Второе подмножество содержит строки $S_2 = \{r_1+1, ..., r_2\}$ для общего числа элементов $\frac{1}{2}(r_2+r_1+1)(r_2-r_1)$. Тогда r_2 должно удовлетворять условию $\frac{1}{2}(r_2+r_1+1)\times(r_2-r_1) = \frac{1}{2M}N(N+1)$. Решив для r_2, мы получим положительный корень

$$r_2 = \frac{-1+\sqrt{1+4(r_1^2+r_1+N(N+1)M^{-1}}}{2}.$$

Мы можем повторить тот же аргумент для будущего подмножества $S_m = \{r_{m-1}+1, ..., r_m\}$ с общим числом элементов $\frac{1}{2}(r_m+r_{m-1}+1)(r_m-r_{m-1})$. Тогда r_m должно удовлетворять условию $\frac{1}{2}(r_m+r_{m-1}+1)(r_m-r_{m-1}) = \frac{1}{2M}N(N+1)$. Решив для r_m, мы получим положительный корень

$$r_m = \frac{-1+\sqrt{1+4(r_{m-1}^2+r_{m-1}+N(N+1)M^{-1}}}{2}.$$

И легко видеть, что r_m сводится к r_1, где $r_{m-1} = r_0 = 0$. Поскольку номера строк являются целыми положительными числами, приведенные выше результаты округляются до ближайшего натурального числа. Это может означать, что размеры некоторых подразделов могут незначительно отличаться от целевого значения $\frac{1}{2M}N(N+1)$. Листинг 20.6 реализует эту логику.

Листинг 20.6. Функция nestedParts

```
def nestedParts(numAtoms,numThreads,upperTriang=False):
    # подразделить атомы с помощью внутреннего цикла
    parts,numThreads_=[0],min(numThreads,numAtoms)
    for num in xrange(numThreads_):
        part=1 + 4*(parts[-1]**2+parts[-1]+numAtoms*(numAtoms+1.)/numThreads_)
```

```
        part=(-1+part**.5)/2.
        parts.append(part)
    parts=np.round(parts).astype(int)
    if upperTriang: # первые строки - самые тяжелые
        parts=np.cumsum(np.diff(parts)[::-1])
        parts=np.append(np.array([0]),parts)
    return parts
```

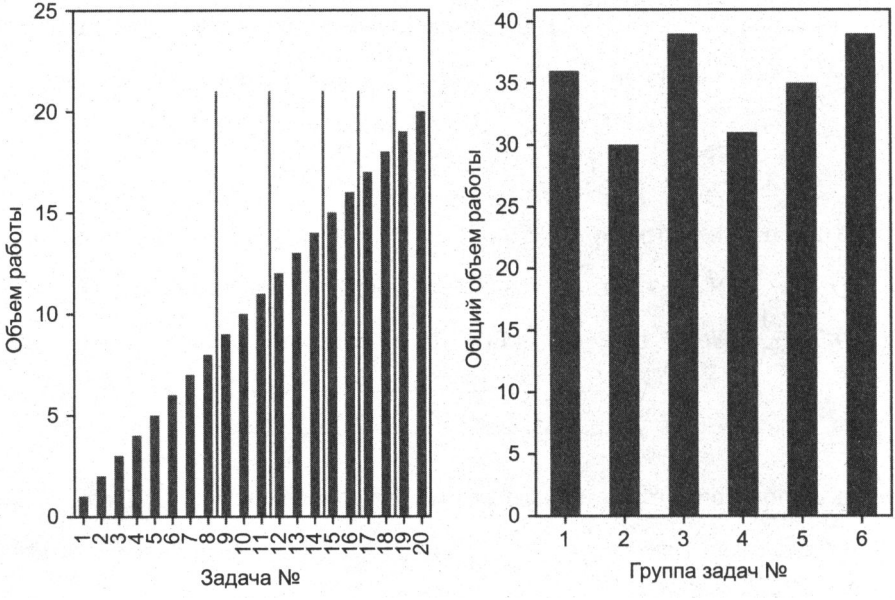


Рис. 20.2. Подразделение атомов на молекулы с помощью дважды вложенных циклов

Если внешний цикл итеративно обходит $i = 1, ..., N$, а внутренний цикл итеративно обходит $j = i, ..., N$, то мы можем упорядочить эти атомарные задачи $\{(i, j) \mid 1 \le i \le j, j = 1, ..., N\}$ в виде *верхней* треугольной матрицы (включая главную диагональ). В этом случае функции `nestedParts` должен быть передан аргумент `upperTriang=True`. Для любопытного читателя это частный случай задачи об упаковке контейнеров. На рис. 20.2 построен график подразделения атомов возрастающей сложности на молекулы в виде двух вложенных циклов. Каждая из полученных шести молекул требует аналогичного объема работы, хотя некоторые атомарные задачи до 20 раз сложнее, чем другие.

20.5. Мультиобрабатывающие механизмы

Было бы ошибкой писать распараллеливающую обертку для каждой мультиобрабатываемой функции. Вместо этого мы должны разработать библиотеку, которая может распараллеливать неизвестные функции, независимо от их аргументов

и структуры выходных данных. В этом состоит цель мультиобрабатывающего механизма. В данном разделе мы изучим один такой механизм, и как только вы поймете его логику, вы будете готовы разрабатывать свои собственные, включая в них всевозможные кастомизированные свойства.

20.5.1. Подготовка заданий

В предыдущих главах мы часто использовали функцию `mpPandasObj`. Эта функция получает шесть аргументов, четыре из которых являются необязательными:

- `func`: функция обратного вызова, которая будет исполняться параллельно;
- `pdObj`: кортеж, содержащий:
  - имя аргумента, используемого для передачи молекул в функцию обратного вызова;
  - список неразделимых задач (атомов), которые будут сгруппированы в молекулы;
- `numThreads`: число потоков, которые будут использоваться параллельно (один процессор на поток);
- `mpBatches`: число параллельных пакетов (заданий на ядро);
- `linMols`: будут ли подразделения линейными или двукратно вложенными;
- `kargs`: именованные аргументы, требуемые функцией `func`.

Листинг 20.7 показывает, как работает функция `mpPandasObj`. Во-первых, атомы группируются в молекулы с помощью функции `linParts` (равное количеству атомов на молекулу) или функции `nestedParts` (атомы распределены в нижнетреугольной структуре). Когда аргумент `mpBatches` больше 1, молекул будет больше, чем ядер. Предположим, что мы подразделяем задачу на 10 молекул, где молекула 1 занимает вдвое больше времени, чем остальные. Если мы запустим этот процесс в 10 ядрах, 9 ядер будут простаивать половину времени выполнения, ожидая, пока первое ядро обработает молекулу 1. В качестве альтернативы мы могли бы установить `mpBatches=10`, для того чтобы подразделить эту задачу на 100 молекул. При этом каждое ядро будет получать одинаковую нагрузку, хотя первые 10 молекул занимают столько же времени, сколько следующие 20 молекул. В этом примере выполнение с аргументом `mpBatches=10` займет половину времени, затрачиваемого при `mpBatches=1`.

Во-вторых, мы формируем список заданий. Задание представляет собой словарь, содержащий всю информацию, необходимую для обработки молекулы, то есть функцию обратного вызова, ее именованные аргументы и подмножество атомов, образующих молекулу. В-третьих, мы будем обрабатывать задания последовательно, если `numThreads==1` (см. листинг 20.8), и параллельно в противном случае (см. раздел 20.5.2). Причина, почему мы хотим, чтобы эта опция выполняла задания

последовательно, заключается в отладке. Отловить дефект нелегко, когда программы выполняются на нескольких процессорах[1]. После отладки кода мы захотим использовать numThreads > 1. В-четвертых, мы сшиваем выходные данные каждой молекулы в единый список, ряд или кадр данных.

Листинг 20.7. Функция mpPandasObj, используемая в разных частях книги

```
def mpPandasObj(func,pdObj,numThreads=24,mpBatches=1,linMols=True,**kargs):
    '''
    Распараллелить задания, вернуть кадр данных DataFrame или ряд Series
    + func: параллелизуемая функция. Возвращает кадр данных
    + pdObj[0]: имя аргумента, используемого для передачи молекулы
    + pdObj[1]: список атомов, которые будут сгруппированы в молекулы
    + kargs: любой другой аргумент, необходимый для func
    Example: df1=mpPandasObj(func,('molecule',df0.index),24,**kargs)
    '''
    import pandas as pd
    if linMols: parts=linParts(len(pdObj[1]),numThreads*mpBatches)
    else: parts=nestedParts(len(pdObj[1]),numThreads*mpBatches)
    jobs=[]
    for i in xrange(1,len(parts)):
        job={pdObj[0]:pdObj[1][parts[i-1]:parts[i]],'func':func}
        job.update(kargs)
        jobs.append(job)
    if numThreads==1: out=processJobs_(jobs)
    else: out=processJobs(jobs,numThreads=numThreads)
    if isinstance(out[0],pd.DataFrame): df0=pd.DataFrame()
    elif isinstance(out[0],pd.Series): df0=pd.Series()
    else: return out
    for i in out: df0=df0.append(i)
    return df0.sort_index()
```

В разделе 20.5.2 мы увидим мультиобрабатывающий аналог функции processJobs_ листинга 20.8.

Листинг 20.8. Однопоточное исполнение с целью отладки

```
def processJobs_(jobs):
    # Выполнять задания последовательно с целью отладки
    out=[]
    for job in jobs:
        out_=expandCall(job)
        out.append(out_)
    return out
```

[1] Гейзенбаги, названные в честь принципа неопределенности Гейзенберга, описывают дефекты, которые меняют свое поведение при их рассмотрении. Дефекты многопроцессорной обработки являются ярким тому примером.

20.5.2. Асинхронные вызовы

В Python есть библиотека параллелизации под названием `multiprocessing`. Эта библиотека является основой для мультиобрабатывающих механизмов, таких как `joblib`[1], который используется многими алгоритмами библиотеки sklearn[2]. Листинг 20.9 иллюстрирует, как выполнять асинхронный вызов Python-овской библиотеки `multiprocessing`. Функция `reportProgress` информирует нас о проценте выполненных заданий.

Листинг 20.9. Пример асинхронного вызова Python-овской библиотеки multiprocessing

```
import multiprocessing as mp
#-------------------------------------
def reportProgress(jobNum,numJobs,time0,task):
    # Информировать о продвижении по мере завершения асинхронных задач
    msg=[float(jobNum)/numJobs,(time.time()-time0)/60.]
    msg.append(msg[1]*(1/msg[0]-1))
    timeStamp=str(dt.datetime.fromtimestamp(time.time()))
    msg=timeStamp+' '+str(round(msg[0]*100,2))+'% '+task+' done after '+ \
    str(round(msg[1],2))+' minutes. Remaining '+str(round(msg[2],2))+
                        ' minutes.'
    if jobNum<numJobs: sys.stderr.write(msg+'\r')
    else: sys.stderr.write(msg+'\n')
    return
#-------------------------------------
def processJobs(jobs,task=None,numThreads=24):
    # Выполнить параллельно.
    # задания должны содержать обратный вызов 'func' для функции expandCall
    if task is None: task=jobs[0]['func'].__name__
    pool=mp.Pool(processes=numThreads)
    outputs,out,time0=pool.imap_unordered(expandCall,jobs),[],time.time()
    # Обработать асинхронный результат, сообщать о продвижении
    for i,out_ in enumerate(outputs,1):
        out.append(out_)
        reportProgress(i,len(jobs),time0,task)
    pool.close();pool.join() # это нужно для предотвращения утечек памяти
    return out
```

20.5.3. Разворачивание функции обратного вызова

В листинге 20.9 инструкция `pool.imap_unordered()` параллелизовала вызов `expandCall` путем выполнения каждого элемента в `jobs` (молекуле) в одном потоке. Листинг 20.10 содержит функцию `expandCall`, которая разворачивает элементы (атомы) в задании (молекуле) и выполняет функцию обратного вызова. Эта ма-

[1] См. https://pypi.python.org/pypi/joblib.

[2] См. http://scikit-learn.org/stable/developers/performance.html#multi-core-parallelism-using-joblib-parallel.

ленькая функция представляет собой изюминку в основе мультиобрабатывающего механизма: она превращает словарь в задачу. Поняв, какую роль она играет, вы сможете разрабатывать свои собственные механизмы.

Листинг 20.10. Передача задания (молекулы) в функцию обратного вызова

```
def expandCall(kargs):
    # Развернуть аргументы функции обратного вызова, kargs['func']
    func=kargs['func']
    del kargs['func']
    out=func(**kargs)
    return out
```

20.5.4. Консервация/расконсервация объектов

Мультиобработка должна консервировать методы для того, чтобы назначить их разным процессорам. Проблема заключается в том, что связанные методы неконсервируемы[1]. Обходной путь состоит в том, чтобы добавить функциональность в ваш механизм, который сообщает библиотеке, как иметь дело с такого рода объектами. Листинг 20.11 содержит инструкции, которые должны быть приведены вверху нашей библиотеки мультиобработки. Если вам интересно узнать точную причину, почему необходима эта часть исходного кода, почитайте работу Ascher и соавт. [2005], раздел 7.5.

Листинг 20.11. Разместить этот исходный код в начале вашего механизма

```
def _pickle_method(method):
    func_name=method.im_func.__name__
    obj=method.im_self
    cls=method.im_class
    return _unpickle_method,(func_name,obj,cls)
#------------------------------------
def _unpickle_method(func_name,obj,cls):
    for cls in cls.mro():
        try: func=cls.__dict__[func_name]
        except KeyError: pass
        else: break
    return func.__get__(obj,cls)
#------------------------------------
import copy_reg,types,multiprocessing as mp
copy_reg.pickle(types.MethodType,_pickle_method,_unpickle_method)
```

20.5.5. Сокращение результата

Предположим, что вы подразделяете задачу на 24 молекулы с той целью, чтобы механизм назначал каждую молекулу одному имеющемуся ядру. Функция

[1] См. http://stackoverflow.com/questions/1816958/cant-pickle-type-instancemethod-when-using-pythonsmultiprocessing-pool-ma.

`processJobs` в листинге 20.9 зафиксирует эти 24 результата на выходе и сохранит их в списке. Такой подход эффективен в задачах, не связанных с крупными результатами. Если результаты должны быть объединены в один, то сначала мы будем ждать до тех пор, пока последняя молекула не будет завершена, а затем будем обрабатывать элементы в списке. Задержка, добавляемая этой постобработкой, не будет значительной до тех пор, пока выходные данные малы по размеру и количеству.

Однако когда результаты потребляют много оперативной памяти и они должны быть объединены в один результат, хранение всех этих результатов в списке может привести к ошибке памяти. Было бы лучше выполнять операцию сокращения результатов на лету, так как результаты возвращаются асинхронно функцией `func`, а не ждать завершения последней молекулы. Мы можем решить эту задачу, усовершенствовав функцию `processJobs`. В частности, мы будем передавать три дополнительных аргумента, которые определяют, каким образом результаты молекулы должны *сводиться* к одному результату. Листинг 20.12 содержит расширенную версию функции `processJobs`, которая содержит три новых аргумента:

- `redux`: это функция обратного вызова, которая выполняет сокращение. Например, `redux=pd.DataFrame.add`, если результирующие кадры данных должны быть просуммированы.
- `reduxArgs`: это словарь, содержащий именованные аргументы, которые должны быть переданы в `redux` (если таковые имеются). Например, если `redux=pd.DataFrame.join`, тогда возможен аргумент `reduxArgs={'how':'outer'}`.
- `reduxInPlace`: булево значение, указывающее, должна ли операция `redux` выполняться *прямо на месте* или нет. Например, `redux=dict.update` и `redux=list.append` требуют аргумента `reduxInPlace=True`, так как пополнение списка и обновление словаря являются операциями, выполняемыми прямо на месте.

Листинг 20.12. Усовершенствование функции processJobs с целью сокращения результата на лету

```
def processJobsRedux(jobs,task=None,numThreads=24,redux=None,reduxArgs={},
                     reduxInPlace=False):
    '''
    Выполнять параллельно
    Задания должны содержать обратный вызов 'func' для expandCall
    redux предотвращает излишний расход памяти, сокращая результат на лету
    '''
    if task is None: task=jobs[0]['func'].__name__
    pool=mp.Pool(processes=numThreads)
    imap,out,time0=pool.imap_unordered(expandCall,jobs),None,time.time()
    # Обработать асинхронный результат, сообщать о продвижении
    for i,out_ in enumerate(imap,1):
        if out is None:
            if redux is None:out,redux,reduxInPlace=[out_],list.append,True
            else: out=copy.deepcopy(out_)
        else:
```

```
            if reduxInPlace: redux(out,out_,**reduxArgs)
            else: out=redux(out,out_,**reduxArgs)
        reportProgress(i,len(jobs),time0,task)
    pool.close();pool.join() # это нужно для предотвращения утечек памяти
    if isinstance(out,(pd.Series,pd.DataFrame)): out=out.sort_index()
    return out
```

Теперь, когда функция `processJobsRedux` знает, что делать с результатами, мы также можем усовершенствовать функцию `mpPandasObj` из листинга 20.7. В листинге 20.13 новая функция `mpJobList` передает в функцию `processJobsRedux` три аргумента сокращения результата. Это избавляет от необходимости обрабатывать список `outputed`, как это делала функция `mpPandasObj`, тем самым экономя память и время.

Листинг 20.13. Усовершенствование функции mpPandasObj с целью сокращения результата на лету

```
def mpJobList(func,argList,numThreads=24,mpBatches=1,linMols=True,redux=None,
              reduxArgs={},reduxInPlace=False,**kargs):
    if linMols: parts=linParts(len(argList[1]),numThreads*mpBatches)
    else: parts=nestedParts(len(argList[1]),numThreads*mpBatches)
    jobs=[]
    for i in xrange(1,len(parts)):
        job={argList[0]:argList[1][parts[i-1]:parts[i]],'func':func}
        job.update(kargs)
        jobs.append(job)
    out=processJobsRedux(jobs,redux=redux,reduxArgs=reduxArgs,
                         reduxInPlace=reduxInPlace,numThreads=numThreads)
    return out
```

20.6. Пример мультиобработки

То, что мы представили до этого в данной главе, может быть использовано для ускорения на несколько порядков многих продолжительных и крупномасштабных математических операций. В этом разделе мы проиллюстрируем дополнительную мотивировку для мультиобработки: управление памятью.

Предположим, что вы провели спектральное разложение ковариационной матрицы вида $Z'Z$, как мы делали в главе 8, раздел 8.4.2, где Z имеет размер $T \times N$. В результате получилась матрица собственных векторов W и матрица собственных значений Λ такие, что $Z'ZW = W\Lambda$. Теперь вы хотели бы получить ортогональные главные компоненты, которые объясняют определенную пользователем порцию общей дисперсии, $0 \leq \tau \leq 1$. Для этого мы вычисляем $P = Z\tilde{W}$, где $\tilde{W}$ содержит первые $M \leq N$ столбцов W такие, что $(\sum_{m=1}^{M} \Lambda_{m,m})(\sum_{n=1}^{N} \Lambda_{n,n})^{-1} \geq \tau$. Вычисление $P = Z\tilde{W}$ может быть параллелизовано, отметив, что

$$P = Z\tilde{W} = \sum_{b=1}^{B} Z_b \tilde{W}_b ,$$

где Z_b — это разреженная $T \times N$-матрица только с $T \times N_b$ элементами (остальные пустые), $\tilde{W}_b$ — $N \times M$-матрица только с $N_b \times M$ элементами (остальные пустые) и $\sum_{b=1}^{B} N_b = N$. Эта разреженность создается путем деления множества столбцов на подраздел из B подмножеств столбцов, и погрузки в Z_b только b-го подмножества столбцов. На первый взгляд это понятие разреженности может показаться немного сложным, однако листинг 20.14 демонстрирует, как библиотека pandas позволяет реализовывать его в бесшовном виде. Функция `getPCs` получает $\tilde{W}$ через аргумент `eVec`. Аргумент `molecules` содержит подмножество имен файлов в `fileNames`, где каждый файл представляет Z_b. Ключевая идея заключается в том, что мы вычисляем скалярное произведение Z_b со срезом строк $\tilde{W}_b$, определенных столбцами в Z_b, и результаты этой молекулы агрегируются на лету (`redux=pd.DataFrame.add`).

Листинг 20.14. Главные компоненты для подмножества столбцов

```
pcs=mpJobList(getPCs,('molecules',fileNames),numThreads=24,mpBatches=1,
              path=path,eVec=eVec,redux=pd.DataFrame.add)
#-------------------------------------
def getPCs(path,molecules,eVec):
    # взять главные компоненты, загружая один файл за раз
    pcs=None
    for i in molecules:
        df0=pd.read_csv(path+i,index_col=0,parse_dates=True)
        if pcs is None: pcs=np.dot(df0.values,eVec.loc[df0.columns].values)
        else: pcs+=np.dot(df0.values,eVec.loc[df0.columns].values)
    pcs=pd.DataFrame(pcs,index=df0.index,columns=eVec.columns)
    return pcs
```

Такой подход дает два преимущества. Во-первых, поскольку функция `getPCs` загружает кадры данных Z_b последовательно, для достаточно большого B оперативная память не исчерпывается. Во-вторых, функция `mpJobList` исполняет молекулы параллельно, тем самым ускоряя вычисления.

В реальных приложениях МО мы часто сталкиваемся с совокупностями данных, где Z содержит миллиарды точек данных. Как показано в этом примере, параллелизация полезна не только с точки зрения сокращения времени выполнения. Многие задачи не могут быть решены без параллелизации, с точки зрения ограничений памяти, даже если бы мы были готовы ждать дольше.

Упражнения

20.1. Выполните листинги 20.1 и 20.2 с функцией хронометража `timeit`. Повторите 10 пакетов по 100 исполнений. Каково минимальное затраченное время для каждого листинга?

20.2. Инструкции в листинге 20.2 очень полезны для модульного тестирования, поиска методом полного перебора и сценарного анализа. Можете ли вы вспомнить, где еще в книге вы их видели? Где еще они могли бы быть использованы?

20.3. Скорректируйте листинг 20.4 так, чтобы формировать молекулы с помощью схемы с двукратно вложенными циклами, а не линейной схемы.

20.4. Сравните с помощью функции хронометража `timeit`:

(а) Листинг 20.4 путем повтора 10 пакетов из 100 исполнений. Каково минимальное затраченное время для каждого фрагмента?

(б) Модифицируйте листинг 20.4 (упражнение 20.3) путем повтора 10 пакетов из 100 исполнений. Каково минимальное затраченное время для каждого фрагмента?

20.5. Упростите листинг 20.4, используя функцию `mpPandasObj`.

20.6. Модифицируйте функцию `mpPandasObj` так, чтобы обрабатывать возможность формирования молекул с верхнетреугольной структурой, используя схему с двукратно вложенными циклами.

21 Метод полного перебора и квантовые компьютеры

21.1. Актуальность

Дискретная математика естественным образом появляется в многочисленных задачах машинного обучения, включая иерархическую кластеризацию, решеточный поиск, решения на основе пороговых значений и целочисленную оптимизацию. Иногда эти задачи не имеют известного аналитического решения (в закрытой форме) или даже аппроксимирующей его эвристики, и наша единственная надежда — разыскивать его с помощью полного перебора. В этой главе мы рассмотрим то, как финансовая задача, неразрешимая для современных суперкомпьютеров, может быть переформулирована как задача целочисленной оптимизации. Такое представление делает ее поддающейся для квантовых компьютеров. Из этого примера читатель может сделать вывод о том, как транслировать свою конкретную финансовую задачу машинного обучения в квантовый поиск по методу полного перебора.

21.2. Комбинаторная оптимизация

Комбинаторно-оптимизационные задачи можно описать как задачи, в которых существует конечное число допустимых решений, возникающих в результате сочетания дискретных значений конечного числа переменных. По мере роста числа возможных сочетаний исчерпывающий поиск становится непрактичным. Задача коммивояжера является примером комбинаторно-оптимизационной задачи, которая, как известно, является NP-трудной (Woeginger [2003]), то есть категорией задач, которые по крайней мере так же сложны, как и самые трудные задачи, разрешимые за недетерминированное полиномиальное время.

Что делает исчерпывающий поиск непрактичным, так это то, что стандартные компьютеры оценивают и хранят возможные решения последовательно. Но что, если бы мы могли оценивать и хранить все возможные решения сразу? В этом заключается цель квантовых компьютеров. В то время как биты стандартного компьютера одновременно могут принимать только одно из двух возможных состояний ($\{0,1\}$), квантовые компьютеры опираются на квантовые биты, или кубиты, представля-

ющие собой элементы памяти, которые могут содержать *линейную суперпозицию* обоих состояний. В теории квантовые компьютеры могут достигать этого благодаря квантово-механическим явлениям. В некоторых реализациях кубиты могут поддерживать токи, протекающие в двух направлениях одновременно, обеспечивая желаемую суперпозицию. Это линейное свойство суперпозиции делает квантовые компьютеры идеально подходящими для решения NP-трудных комбинаторно-оптимизационных задач. См. публикацию Williams [2010], в которой излагаются основы и демонстрируются возможности квантовых компьютеров.

Лучший способ понять этот подход — рассмотреть конкретный пример. Теперь мы посмотрим, как задача динамической оптимизации портфеля, являющаяся предметом обобщенных функций транзакционных издержек, может быть представлена как комбинаторно-оптимизационная задача, поддающаяся разрешению квантовыми компьютерами. В отличие от Garleanu and Pedersen [2012], мы не будем исходить из того, что финансовые возвраты берутся из гауссова распределения одинаково распределенных взаимно независимых случайных величин. Эта задача имеет прямое отношение к крупным управляющим активами, поскольку издержки, связанные с чрезмерным оборотом и дефицитом реализации, могут серьезно подорвать прибыльность их инвестиционных стратегий.

21.3. Целевая функция

Рассмотрим множество на активах $X = \{x_i\}$, $i = 1, ..., N$ с финансовыми возвратами, подчиняющимися многомерному нормальному распределению на каждом временном горизонте $h = 1, ..., H$, с варьирующимся средним значением и дисперсией. Мы будем исходить из допущения, что финансовые возвраты являются многомерными нормальными, независимыми от времени, не одинаково распределенными во времени. Мы определяем торговую траекторию как $N \times H$-матрицу w, которая идентифицирует долю капитала, размещенную в каждый из N активов по каждому из H горизонтов. На конкретном горизонте $h = 1, ..., H$ у нас есть спрогнозированное среднее μ_h, спрогнозированная дисперсия V_h и спрогнозированная функция транзакционных издержек $\tau_h[\omega]$. Это означает, что при заданной торговой траектории ω мы можем вычислить вектор ожидаемых инвестиционных возвратов r как

$$r = \text{diag}[\mu'\omega] - \tau[\omega],$$

где $\tau[\omega]$ может принимать любую функциональную форму. Не уменьшая общности, рассмотрим следующие выражения:

- $\tau_1[\omega] = \sum_{i=1}^{N} c_{n,1} \sqrt{|\omega_{n,1} - \omega_n^*|}$;
- $\tau_h[\omega] = \sum_{n=1}^{N} c_{n,h} \sqrt{|\omega_{n,h} - \omega_{n,h-1}|}$, для $h = 2, ..., H$;
- ω_n^* — это изначальное распределение в инструмент n, $n = 1,..., N$.

$\tau[\omega]$ — это $H \times 1$-вектор транзакционных издержек. На словах транзакционные издержки, связанные с каждым активом, представляют собой сумму квадратных корней из изменений в размещении капитала, прошкалированную на специфичный для актива коэффициент $C_h = \{c_{n,h}\}_{n=1,...,N}$, который изменяется вместе с h. Таким образом, C_h — это $N \times 1$-вектор, который определяет относительные транзакционные издержки по всем активам.

Связанный с r коэффициент Шарпа (глава 14) может быть вычислен как (μ_h за вычетом безрисковой ставки):

$$SR[r] = \frac{\sum_{h=1}^{H} \mu_h' \omega_h - \tau_h[\omega]}{\sum_{h=1}^{H} \omega_h' V_h \omega_h}.$$

21.4. Задача

Мы хотели бы вычислить оптимальную торговую траекторию, которая решает задачу

$$\max_{\omega} SR[r],$$

$$\text{так что: } \sum_{i=1}^{N} |\omega_{i,h}| = 1, \ \forall h = 1, \ldots, H.$$

Эта задача пытается вычислить глобальный динамический оптимум, в отличие от статического оптимума, выводимого среднедисперсными оптимизаторами (см. главу 16). Обратите внимание, что не-непрерывные транзакционные издержки встроены в r. По сравнению со стандартными приложениями портфельной оптимизации эта задача не является задачей выпуклого (квадратического) программирования как минимум по трем причинам: 1) финансовые возвраты не являются одинаково распределенными, потому что μ_h и V_h изменяются вместе с h; 2) транзакционные издержки $\tau_h[\omega]$ не непрерывны и изменяются вместе с h; 3) целевая функция $SR[r]$ не выпукла. Далее мы покажем, как вычислять решения без использования каких-либо аналитических свойств целевой функции (отсюда и обобщенный характер этого подхода).

21.5. Целочисленно-оптимизационный подход

Общность этой задачи делает ее не разрешимой стандартными методами выпуклой оптимизации. Наша стратегия решения состоит в том, чтобы дискретизировать ее так, чтобы она стала поддаваться целочисленной оптимизации. Это в свою очередь позволит использовать квантовые вычислительные технологии для нахождения оптимального решения.

21.5.1. Подразделения по методу голубиных клеток

Предположим, что мы подсчитываем число способов, которыми K единиц капитала могут быть размещены между N активами, где мы исходим из того, что $K > N$. Эта операция эквивалентна нахождению числа неотрицательных целочисленных решений $x_1 + ... + x_N = K$, которое имеет безупречное комбинаторное решение $\binom{K+N-1}{N-1}$. Оно похоже на классическую задачу целочисленного подразделения в теории чисел, для которой Харди и Рамануджан (а позже Радемахер) доказали асимптотическое выражение (см. Johansson [2012]). Хотя в данной задаче подразделения порядок не имеет значения, порядок очень важен для задачи, которую мы рассматриваем[1].

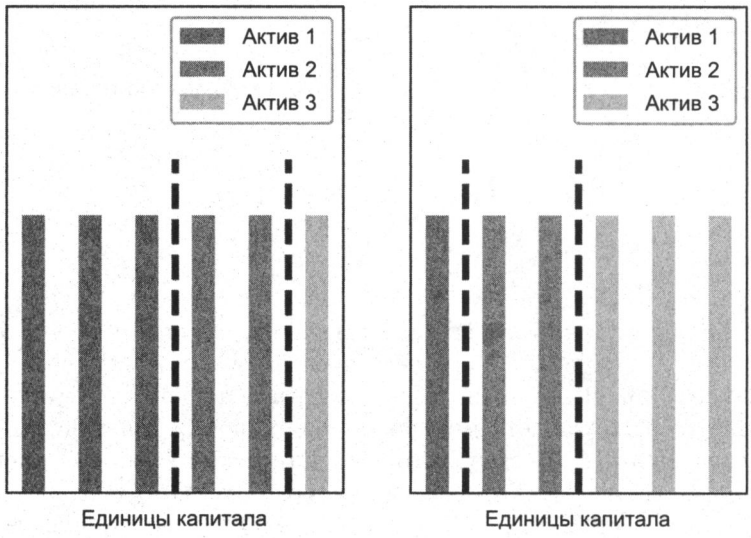


Рис. 21.1. Подразделения (1, 2, 3) и (3, 2, 1) должны рассматриваться как разные

Например, если $K = 6$ и $N = 3$, то подразделы (1, 2, 3) и (3, 2, 1) должны рассматриваться как разные (совершенно очевидно, что (2, 2, 2) переставлять не нужно). Рисунок 21.1 иллюстрирует то, насколько важен порядок при размещении шести единиц капитала в трех разных активах. Это означает, что мы должны рассмотреть все отдельные перестановки каждого подраздела. Хотя имеется безупречное комбинаторное решение, находящее число таких размещений, все же найти его может оказаться вычислительно трудоемким по мере того, как K и N вырастают

[1] Pigeonhole partitions (голубиные клетки). См. https://ru.wikipedia.org/wiki/Принцип_Дирихле_(комбинаторика). — *Примеч. науч. ред.*

до больших величин. Однако мы можем использовать аппроксимацию Стирлинга, которая легко получает его оценку.

Листинг 21.1 предоставляет эффективный алгоритм для генерирования множества всех подразделов, $p^{K,N}=\{\{p_i\}_{i=1,\ldots,N} | p_i \in \mathbb{W}, \sum_{i=1}^{N} p_i = K\}$, где $\mathbb{W}$ — это натуральные числа, включая нуль (целые числа).

Листинг 21.1. Подразделы *k* объектов на *n* клеток

```
from itertools import combinations_with_replacement
#---------------------------------------
def pigeonHole(k,n):
    # Задача голубиных клеток (организовать k объектов в n клеток)
    for j in combinations_with_replacement(xrange(n),k):
        r=[0]*n
        for i in j:
            r[i]+=1
        yield r
```

21.5.2. Допустимые статические решения

Мы хотели бы вычислить множество всех допустимых решений на любом заданном горизонте h, которое мы обозначим через Ω. Рассмотрим множество разделов из K единиц на N активов, $p^{K,N}$. Для каждого подраздела $\{p_i\}_{i=1,\ldots,N} \in p^{K,N}$ мы можем определить вектор абсолютных весов таких, что $|\omega_i| = \frac{1}{K} p_i$, где $\sum_{i=1}^{N} |\omega_i| = 1$ (полноинвестиционное ограничение). Из данного полноинвестиционного ограничения (без кредитного плеча) следует, что каждый вес может быть как положительным, так и отрицательным, поэтому для каждого вектора абсолютных весов $\{|\omega_i|\}_{i=1,\ldots,N}$ мы можем сгенерировать 2^N векторов весов (со знаком). Это достигается путем перемножения элементов в $\{|\omega_i|\}_{i=1,\ldots,N}$ с элементами декартова произведения $\{-1, 1\}$ с N повторами. Листинг 21.2 показывает, как генерировать множество Ω всех векторов весов, связанных со всеми подразделами,

$$\Omega = \left\{ \left\{ \frac{S_j}{K} p_i \right\} \middle| \{s_j\}_{j=1,\ldots,N} \in \underbrace{\{-1, 1\}x\ldots x\{-1, 1\}}_{N}, \{p_i\}_{i=1,\ldots,N} \in p^{K,N} \right\}.$$

Листинг 21.2. Множество Ω всех векторов, связанных со всеми подразделами

```
import numpy as np
from itertools import product
#---------------------------------------
def getAllWeights(k,n):
    #1) сгенерировать подразделы
    parts,w=pigeonHole(k,n),None
    #2) пройтись по подразделам
    for part_ in parts:
```

```
        w_=np.array(part_)/float(k) # вектор из abs(вес)
        for prod_ in product([-1,1],repeat=n): # добавить знак
            w_signed_=(w_*prod_).reshape(-1,1)
            if w is None: w=w_signed_.copy()
            else: w=np.append(w,w_signed_,axis=1)
    return w
```

21.5.3. Оценивание траекторий

При заданном множестве всех векторов Ω мы определяем множество всех возможных траекторий Φ как декартово произведение Ω с H повторами. Затем для каждой траектории мы можем оценить ее транзакционные издержки и коэффициент Шарпа SR и отобрать траекторию с оптимальной результативностью по всем Φ. Листинг 21.3 реализует эту функцию. Объект `params` представляет собой список словарей, содержащих значения C, μ, V.

Листинг 21.3. Оценивание всех траекторий

```
import numpy as np
from itertools import product
#---------------------------------------
def evalTCosts(w,params):
    # Вычислить транзакционные издержки конкретной траектории
    tcost=np.zeros(w.shape[1])
    w_=np.zeros(shape=w.shape[0])
    for i in range(tcost.shape[0]):
        c_=params[i]['c']
        tcost[i]=(c_*abs(w[:,i]-w_)**.5).sum()
        w_=w[:,i].copy()
    return tcost
#---------------------------------------
def evalSR(params,w,tcost):
    # Оценить SR по многочисленным горизонтам
    mean,cov=0,0
    for h in range(w.shape[1]):
        params_=params[h]
        mean+=np.dot(w[:,h].T,params_['mean'])[0]-tcost[h]
        cov+=np.dot(w[:,h].T,np.dot(params_['cov'],w[:,h]))
    sr=mean/cov**.5
    return sr
#---------------------------------------
def dynOptPort(params,k=None):
    # Динамический оптимальный портфель
    #1) сгенерировать подразделы
    if k is None: k=params[0]['mean'].shape[0]
    n=params[0]['mean'].shape[0]
    w_all,sr=getAllWeights(k,n),None
    #2) сгенерировать траектории как декартовы произведения
    for prod_ in product(w_all.T,repeat=len(params)):
        w_=np.array(prod_).T # конкатенировать произведение в траекторию
```

```
        tcost_=evalTCosts(w_,params)
        sr_=evalSR(params,w_,tcost_) # оценить траекторию
        if sr is None or sr<sr_: # сохранить траекторию, если она лучше
            sr,w=sr_,w_.copy()
    return w
```

Обратите внимание, что эта процедура выбирает глобально оптимальную траекторию, не опираясь на выпуклую оптимизацию. Решение будет найдено, даже если ковариационные матрицы плохо обусловлены, функции транзакционных издержек не непрерывны и т. д. Цена, которую мы платим за эту общность, заключается в том, что вычисление решения является чрезвычайно вычислительно интенсивным. Действительно, оценивание всех траекторий аналогично задаче коммивояжера. Для такого рода NP-полных или NP-трудных задач цифровые компьютеры неадекватны; однако квантовые компьютеры имеют преимущество оценивания многократного числа решений одновременно благодаря свойству линейной суперпозиции.

Подход, представленный в этой главе, заложил основу для работы Rosenberg и соавт. [2016], в которой была решена задача оптимальной торговой траектории с помощью квантового закаливателя. Та же логика может быть применена к широкому кругу финансовых задач, связанных с траекторной зависимостью, таких как торговая траектория. Неразрешимый алгоритм MO может быть дискретизирован и транслирован в исчерпывающий поиск по методу полного перебора, предназначенный для квантового компьютера.

21.6. Численный пример

Ниже мы проиллюстрируем, как глобальный оптимум можно найти на практике, используя цифровой компьютер. Квантовый компьютер будет оценивать все траектории сразу, в то время как цифровой компьютер делает это последовательно.

21.6.1. Случайные матрицы

Листинг 21.4 возвращает случайную матрицу гауссовых значений с известным рангом, которая широко применяется во многих приложениях (см. упражнения). Вы можете рассмотреть этот исходный код, когда в следующий раз будете выполнять многомерные эксперименты Монте-Карло или сценарный анализ.

Листинг 21.4. Произвести случайную матрицу заданного ранга

```
import numpy as np
#---------------------------------------
def rndMatWithRank(nSamples,nCols,rank,sigma=0,homNoise=True):
    # Произвести случайную матрицу X с заданным рангом
    rng=np.random.RandomState()
```

```
    U,_,_=np.linalg.svd(rng.randn(nCols,nCols))
    x=np.dot(rng.randn(nSamples,rank),U[:,:rank].T)
    if homNoise:
        x+=sigma*rng.randn(nSamples,nCols) # Добавление гомоскедастичного шума
    else:
        sigmas=sigma*(rng.rand(nCols)+.5) # Добавление гетероскедастичного
                                          # шума
        x+=rng.randn(nSamples,nCols)*sigmas
    return x
```

Листинг кода 21.5 генерирует H векторов средних значений, ковариационных матриц и коэффициентов транзакционных издержек, C, μ, V. Эти величины хранятся в списке параметров `params`.

Листинг 21.5. Сгенерировать параметры задачи

```
import numpy as np
#---------------------------------------
def genMean(size):
    # Сгенерировать случайный вектор средних
    rMean=np.random.normal(size=(size,1))
    return rMean
#---------------------------------------
#1) параметры
size,horizon=3,2
params=[]
for h in range(horizon):
    x=rndMatWithRank(1000,3,3,0.)
    mean_,cov_=genMean(size),np.cov(x,rowvar=False)
    c_=np.random.uniform(size=cov_.shape[0])*np.diag(cov_)**.5
    params.append({'mean':mean_,'cov':cov_,'c':c_})
```

21.6.2. Статическое решение

Листинг 21.6 вычисляет результативность траектории, которая является результатом локальных (статических) оптимумов.

Листинг 21.6. Вычислить и оценить статическое решение

```
import numpy as np
#---------------------------------------
def statOptPortf(cov,a):
    # Статический оптимальный портфель
    # Решение задачи "неограниченной" портфельной оптимизации
    cov_inv=np.linalg.inv(cov)
    w=np.dot(cov_inv,a)
    w/=np.dot(np.dot(a.T,cov_inv),a) # np.dot(w.T,a)==1
    w/=abs(w).sum() # перешкалировать полную инвестицию
    return w
#---------------------------------------
#2) статические оптимальные портфели
```

```
w_stat=None
for params_ in params:
    w_=statOptPortf(cov=params_['cov'],a=params_['mean'])
    if w_stat is None: w_stat=w_.copy()
    else: w_stat=np.append(w_stat,w_,axis=1)
tcost_stat=evalTCosts(w_stat,params)
sr_stat=evalSR(params,w_stat,tcost_stat)
print 'static SR:',sr_stat
```

21.6.3. Динамическое решение

Листинг 21.7 вычисляет результативность, связанную с глобально динамической оптимальной траекторией, применяя функции, описанные в данной главе.

Листинг 21.7. Вычислить и оценить динамическое решение

```
import numpy as np
#-------------------------------------
#3) динамические оптимальные портфели
w_dyn=dynOptPort(params)
tcost_dyn=evalTCosts(w_dyn,params)
sr_dyn=evalSR(params,w_dyn,tcost_dyn)
print 'dynamic SR:',sr_dyn
```

Упражнения

21.1. Используя аргумент голубиной клетки, докажите формулу $\sum_{n=1}^{N}\binom{N}{n}=2^N-1$.

21.2. Используйте листинг 21.4, чтобы произвести случайные матрицы размера (1000, 10), `sigma=1` и

(а) `rank=1`. Постройте график собственных значений ковариационной матрицы.

(б) `rank=5`. Постройте график собственных значений ковариационной матрицы.

(в) `rank=10`. Постройте график собственных значений ковариационной матрицы.

21.3. Выполните численный пример в разделе 21.6:

(а) Используйте `size=3` и вычислите время выполнения с помощью функции хронометража `timeit`. Повторите 10 пакетов по 100 исполнений. Сколько времени это заняло?

(б) Используйте `size=4` и вычислите время выполнения с помощью функции хронометража `timeit`. Повторите 10 пакетов по 100 исполнений. Сколько времени это заняло?

21.4. Просмотрите все листинги из данной главы.

(а) Сколько из них можно векторизировать?

(б) Сколько из них можно параллелизовать с использованием методов главы 20?

(в) Если вы оптимизируете исходный код, то насколько, по вашему мнению, вы сможете его ускорить?

(г) Какова размерность задачи, которую можно решить в течение года, используя оптимизированный код?

21.5. При каких обстоятельствах глобально динамическая оптимальная траектория будет соответствовать последовательности локальных оптимумов?

(а) Это реалистичный набор допущений?

(б) Если нет,

i) может ли это объяснить, почему наивные решения одержали верх над решением Марковица (глава 16)?

ii) как вы думаете, почему так много фирм тратят столько усилий на вычисление последовательностей локальных оптимумов?

22 Технологии высокопроизводительного вычислительного интеллекта и прогнозирования

Кишенг Ву и Хорст Д. Саймон

22.1. Актуальность

В этой главе представлено введение в проект технологий вычислительного интеллекта и прогнозирования (computational intelligence and forecasting technologies, CIFT) в Национальной лаборатории им. Лоуренса в Беркли (Lawrence Berkeley National Laboratory, LBNL). Основной целью проекта CIFT является содействие использованию инструментов и методов высокопроизводительных вычислений (high-performance computing, HPC) для анализа потоковых данных. После того как было замечено, что причиной пятимесячной задержки публикации отчета Комиссией по ценным бумагам и биржам (The United States Securities and Exchange Commission, SEC) и Комиссией по торговле товарными фьючерсами (Commodity Futures Trading Commission, CFTC) о молниеносном обвале 2010 года стал объем данных, Национальной лабораторией им. Лоуренса в Беркли был запущен проект CIFT по применению высокопроизводительных вычислительных технологий для управления и анализа финансовых данных. Своевременное принятие решений на основе потоковых данных является неотъемлемым требованием ко многим деловым приложениям, таким как предотвращение надвигающегося сбоя в электросети или кризиса ликвидности на финансовых рынках. Во всех этих случаях высокопроизводительные вычислительные (HPC) инструменты хорошо подходят для обработки сложных зависимостей данных и обеспечения своевременного решения. На протяжении многих лет проект CIFT работал над несколькими разными формами потоковых данных, в том числе получаемых из трафика транспортных средств, электросети, использования электроэнергии и т. д. В следующих ниже разделах объясняются ключевые особенности высокопроизводительных вычислительных систем, вводится несколько специальных инструментов, используемых в этих системах, и приводятся примеры анализа потоковых данных с использованием этих высокопроизводительных вычислительных инструментов.

22.2. Регулятивная реакция на молниеносный обвал 2010 года

6 мая 2010 года около 14:45 по восточному летнему времени фондовый рынок США испытал почти 10 %-ное падение промышленного индекса Доу—Джонса и лишь спустя несколько минут восстановил бóльшую часть убытка. Регулятивным агентствам потребовалось около пяти месяцев, чтобы составить отчет о расследовании. В присутствии членов комиссии на панели Конгресса, расследующей обвал на фондовом рынке, в качестве основной причины длительной задержки был указан объем данных (~20 терабайт). Поскольку системы HPC, такие как Национальный центр научных вычислений в области энергетических исследований (NERSC)[1], рутинным образом обрабатывают сотни терабайт в течение нескольких минут, у нас не должно было быть проблем с обработкой данных из финансовых рынков. Это привело к созданию проекта CIFT, призванного заняться применением высокопроизводительных вычислительных методов и инструментов для анализа финансовых данных.

Ключевым аспектом больших финансовых данных является то, что они состоят в основном из временны́х рядов. На протяжении многих лет команда CIFT, наряду с многочисленными сотрудниками, разрабатывала методы анализа различных форм потоков данных и временны́х рядов. В этой главе приводится краткое введение в высокопроизводительную вычислительную (HPC) систему, включая как аппаратное (раздел 22.4), так и ПО (раздел 22.5), и рассказывается о нескольких успешных вариантах ее использования (раздел 22.6). В заключение мы приводим краткое изложение нашего видения и проделанной к настоящему времени работы, а также предоставляем контактную информацию для заинтересованных читателей.

22.3. История вопроса

Достижения в вычислительной технике значительно облегчили поиск сложных закономерностей. Способность находить закономерности лежит в основе ряда недавних научных прорывов, таких как открытие бозона Хиггса (Aad и соавт. [2016]) и гравитационных волн (Abbot и соавт. [2016]). Эта же способность также лежит в основе многих интернет-компаний, например, для сопоставления пользователей с рекламодателями (Zeff and Aronson [1999], Yen и соавт. [2009]). Однако аппаратное обеспечение и ПО, используемое в науке и в трейдинге, совершенно разные. Высокопроизводительные вычислительные инструменты имеют некоторые важные преимущества, которые будут полезны в различных деловых приложениях.

[1] National Energy Research Scientific Computing (NERSC) — Национальный пользовательский фонд, финансируемый Министерством энергетики США, расположенный в Национальной лаборатории Лоуренса Беркли. Более подробную информацию о NERSC можно найти на http://nersc.gov/.

Инструменты для ученых, как правило, построены вокруг высокопроизводительных вычислительных (HPC) платформ, в то время как инструменты для коммерческих приложений построены вокруг облачных вычислительных платформ. Было показано, что для просеивания больших объемов данных с целью поиска полезных закономерностей оба подхода работают хорошо. Однако выдающимся применением высокопроизводительных вычислительных систем является широкомасштабная симуляция, такая как погодные модели, используемые для прогнозирования региональных ураганов в ближайшие несколько дней (Asanovic и соавт. [2006]). В отличие от этого, коммерческое облако изначально было мотивировано необходимостью обрабатывать большое число независимых объектов данных одновременно (задачи параллельной обработки данных).

Что касается нашей работы, то в первую очередь нас интересует анализ потоковых данных. В частности, высокоскоростные комплексные потоки данных, например, от сенсорных сетей, контролирующих электроэнергетические сети и системы автомобильных дорог нашей страны. Данная потоковая рабочая нагрузка не идеальна ни для высокопроизводительных вычислительных систем, ни для облачных систем, как мы покажем ниже, но мы считаем, что высокопроизводительная вычислительная экосистема может предложить больше для анализа потоковых данных, чем облачная экосистема.

Облачные системы изначально разрабатывались для задач параллельной обработки данных, в которых одновременно может обрабатываться большое число независимых объектов данных. Следовательно, система сконструирована для высокой пропускной способности, а не для порождения откликов в реальном режиме времени. Однако многие деловые приложения требуют откликов в режиме реального или почти реального времени. Например, событие нестабильности в электросети может развиваться и перерастать в катастрофу за считаные минуты; обнаружение контрольного сигнала достаточно быстро предотвратит такую катастрофу. Аналогичным образом, в литературе по финансовым исследованиям были выявлены признаки возникающих событий неликвидности; быстрое нахождение этих признаков в течение активных часов торговли на рынке может предложить варианты предотвращения шоков на рынке и избежать молниеносных обвалов. В этих случаях важно иметь возможность приоритизировать быстрое время цикла обработки.

Поток данных по определению доступен поступательно, поэтому может отсутствовать большое число объектов данных для параллельной обработки. Как правило, для анализа доступно только фиксированное число последних записей данных. В этом случае эффективным способом использования вычислительной мощности многочисленных ядер центральных процессоров (CPU) является разделение аналитической работы на одном объекте данных (или одном временном шаге) на несколько ядер CPU. Высокопроизводительная вычислительная экосистема имеет более продвинутые инструменты для такого рода работы, чем облачная экосистема.

Это основные моменты, которые мотивировали нашу работу. Для более тщательного сравнения высокопроизводительных вычислительных систем и облачных систем

мы отсылаем заинтересованных читателей к публикации Asanovic и соавт. [2006]. В частности, в публикации Fox и соавт. [2015] была создана обширная таксономия для описания сходств и различий для любого прикладного сценария.

Одним словом, мы считаем, что сообщество высокопроизводительных вычислений может многое предложить для продвижения современного состояния потоковой аналитики. Проект CIFT был создан с целью передачи экспертных знаний лаборатории LBNL в области высокопроизводительных вычислений деловым потоковым приложениям. Мы следуем этой миссии, осуществляя сотрудничество, демонстрацию и разработку инструментов.

Чтобы оценить потенциальные возможности использования технологии высокопроизводительных вычислений, мы потратили много времени на работу с различными приложениями. Этот процесс не только побуждает наших специалистов по высокопроизводительным вычислениям обращаться к разнообразным областям, но также позволяет нам накапливать финансовую поддержку для построения демонстрационного оборудования.

Благодаря щедрым подаркам от ряда ранних сторонников этой работы мы создали значительный вычислительный кластер, посвященный этой работе. Этот выделенный компьютер (под именем dirac1) позволяет пользователям задействовать высокопроизводительную вычислительную систему и самостоятельно оценивать свои приложения.

Мы также занимаемся разработкой инструментов, которые позволили бы сделать высокопроизводительные вычислительные системы более удобными для анализа потоковых данных. В следующих далее разделах мы опишем аппаратное и программное обеспечение специальной машины CIFT, а также некоторые демонстрационные и инструментальные наработки. Основные моменты включают 21-кратное улучшение скорости обработки данных и 720-кратное увеличение скорости вычисления индикатора раннего предупреждения.

22.4. Аппаратное обеспечение для высокопроизводительных вычислений

Легенда гласит, что первое поколение систем больших данных было построено с помощью запасных компьютерных компонентов, почерпнутых из университетского городка. По всей видимости, это городская легенда, но она подчеркивает важный момент о разнице между высокопроизводительными вычислительными (HPC) системами и облачными системами. Теоретически, система HPC строится из кастомизированных дорогостоящих компонентов, в то время как облачные системы строятся с использованием стандартных малобюджетных товарных компонентов. На практике, поскольку во всем мире инвестиции в системы HPC гораздо меньше, чем инвестиции в персональные компьютеры, нет никаких возможностей для того, чтобы производители изготавливали по индивидуальному заказу компоненты

специально для рынка HPC. Следует признать, что системы HPC в основном собираются из обычных компонентов, как и облачные системы. Однако вследствие их разнящихся целевых приложений есть некоторые различия в выборе компонентов.

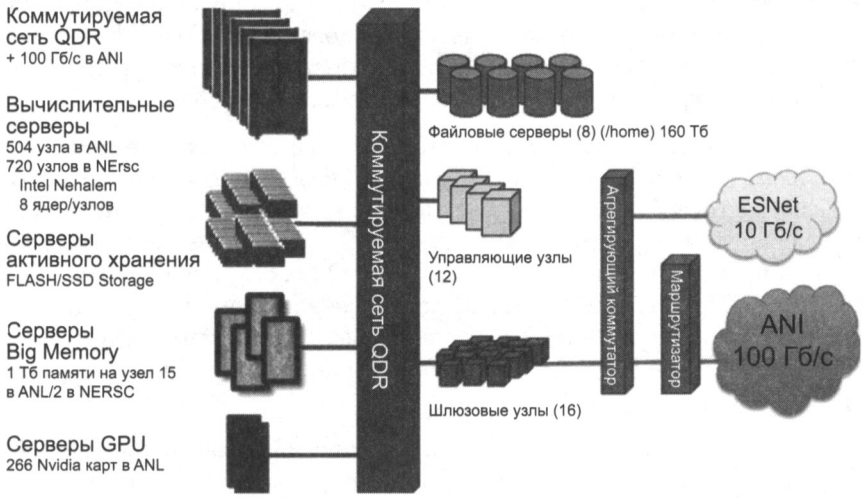


Рис. 22.1. Схема кластера Магеллана (около 2010 г.), пример компьютерного кластера ВВ

Опишем по очереди вычислительные элементы, систему хранения и сетевую систему. Рисунок 22.1 представляет собой высокоуровневую схематичную диаграмму, иллюстрирующую ключевые компоненты кластера Magellan в 2010 году (Jackson и соавт. [2010], Yelick и соавт. [2011]). Компьютерные элементы включают в себя как процессоры (CPU), так и графические процессоры (GPU). Почти во всех случаях эти CPU и GPU являются коммерческими продуктами. Например, в узлах на dirac1 используется 24-ядерный процессор Intel 2.2 ГГц, который общепринято использовать для облачных вычислительных систем. В настоящее время dirac1 не содержит графических процессоров.

Сетевая система состоит из двух частей: коммутируемой сети InfiniBand, соединяющей компоненты кластера, и коммутируемого сетевого соединения с внешним миром. В данном конкретном примере внешние соединения имеют метки *ESNet* и *ANI*. Сетевые коммутаторы InfiniBand также распространены в облачных вычислительных системах.

Система хранения на рис. 22.1 включает как вращающиеся диски, так и флэш-память. Данное сочетание также общепринято. Другое дело, что подсистема хранения системы HPC обычно сосредоточена за пределами компьютерных узлов, в то время как подсистема хранения типичной облачной вычислительной системы распределена между вычислительными узлами. Эти два подхода имеют свои преимущества и недостатки. Например, концентрированное хранилище обычно экс-

портируется в виде глобальной файловой системы на все узлы компьютера, что упрощает работу с данными, хранящимися в файлах. Однако для этого требуется сеть с высокой пропускной способностью, соединяющая CPU и диски. В отличие от этого распределенный подход может использовать сеть меньшей мощности, поскольку существует некоторое хранилище, близкое к каждому CPU. Как правило, распределенная файловая система, такая как файловая система Google (Ghemawat, Gobioff and Leung [2003]), располагается поверх облачной вычислительной системы, чтобы сделать хранилище доступным для всех процессоров.

Одним словом, нынешнее поколение систем HPC и облачных систем использует в значительной степени те же самые коммерческие аппаратные компоненты. Их отличия в первую очередь заключаются в расположении систем хранения и сетевых систем. Очевидно, что разница в конструкции системы хранения данных может повлиять на производительность приложения. Однако уровень виртуализации облачных систем, по всей видимости, является основной причиной разницы в производительности приложений. В следующем далее разделе мы обсудим еще один фактор, который может оказать еще большее влияние, а именно программные средства и библиотеки.

В среде облачных вычислений обычно используется виртуализация, которая служит для того, чтобы сделать одно и то же оборудование доступным для нескольких пользователей и изолировать одну программную среду от другой. Это одна из наиболее характерных особенностей, отличающих облачную вычислительную среду от среды HPC. В большинстве случаев все три основных компонента компьютерной системы — CPU, хранилище и сеть — виртуализированы. Эта виртуализация имеет много преимуществ. Например, существующее приложение может работать на чипе CPU без перекомпиляции; многие пользователи могут использовать одно и то же оборудование; аппаратные сбои можно исправлять с помощью виртуализационного программного обеспечения, а приложения на отказавшем вычислительном узле легче переносятся на другой узел. Однако этот уровень виртуализации также добавляет некоторые накладные расходы времени выполнения и может снизить производительность приложения. Для приложений, зависящих от времени, это снижение производительности может стать критической проблемой.

Тесты показывают, что различия в производительности могут быть довольно большими. Далее мы кратко опишем исследование производительности, опубликованное в работе Jackson и соавт. [2010]. На рис. 22.2 показано замедление производительности при использовании разных компьютерных систем. Имена под горизонтальной осью — это различные пакеты программ, обыкновенно используемые в Национальном научно-исследовательском вычислительном центре по энергетике NERSC. Левая полоса соответствует коммерческому облаку, средняя — кластеру Magellan, а правая (иногда отсутствует) — системе EC2-Beta-Opt. Неоптимизированные коммерческие облачные экземпляры работают с этими пакетами программ в 2–10 раз медленнее, чем на суперкомпьютере NERSC. Даже на более дорогих высокопроизводительных экземплярах наблюдаются заметные замедления.

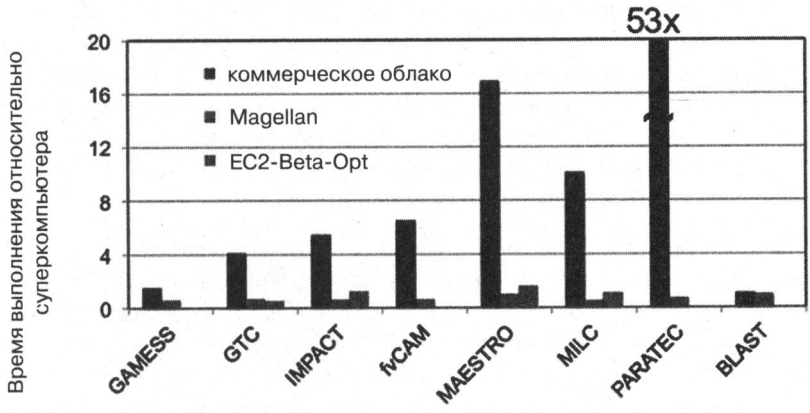


Рис. 22.2. В облаке научные приложения работали значительно медленнее, чем в системах APC (около 2010 г.)

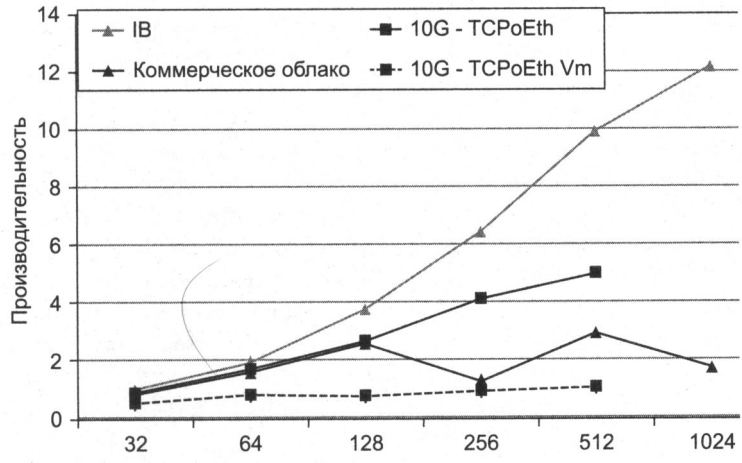


Рис. 22.3. По мере увеличения ядер (горизонтальная ось) потребление ресурсов из-за виртуализации становится более заметным

На рис. 22.3 показано исследование главного фактора, вызывающего замедление работы с программным пакетом PARATEC. На рис. 22.2 мы видим, что пакету PARATEC потребовалось в 53 раза больше времени для завершения задачи в коммерческом облаке, чем в системе HPC. Из рис. 22.3 мы видим, что по мере увеличения числа ядер (горизонтальная ось) разница между измеренными производительностями (измеренными в терафлопах/с) становятся больше. В частности, строка с надписью «10G-TCPoEth Vm» едва увеличивается по мере роста числа ядер. Это тот случай, когда сетевой экземпляр использует виртуализированную сеть (TCP через Ethernet). Данный результат ясно показывает, что виртуализационные накладные расходы сети значительны, вплоть до бесполезности облака.

Проблема виртуализационных накладных расходов общепризнана (Chen и соавт. [2015]). Были проведены значительные исследования, направленные на устранение виртуализационных накладных расходов ввода-вывода (Gordon и соавт. [2012]), а также сетевые виртуализационные накладные расходы (Dong и соавт. [2012]). По мере того как эти современные методы постепенно перемещаются в коммерческие продукты, мы ожидаем, что накладные расходы будут уменьшаться в будущем, но некоторые накладные расходы неизбежно останутся.

В завершение данного раздела мы кратко коснемся экономики HPC в сравнении с облаком. Как правило, системы HPC управляются некоммерческими исследовательскими организациями и университетами, а облачные системы — коммерческими компаниями. На стоимость облачной системы влияют прибыль, удержание клиентов и многие другие факторы (Armburst и соавт. [2010]). В 2011 году в отчете о проекте Magellan было указано, что «анализ затрат показывает, что центры Министерства энергетики DOE0 конкурентоспособны по стоимости, как правило, в 3–7 раз дешевле, по сравнению с коммерческими поставщиками облачных услуг» (Yelick и соавт. [2010]).

Группа физиков высоких энергий посчитала, что их вариант использования хорошо подходит для облачных вычислений, и провела подробное сравнительное исследование (Holzman и соавт. [2017]). Их сравнение стоимости все же показывает, что коммерческие облачные предложения примерно на 50 % дороже, чем выделенные системы HPC для сопоставимых вычислительных задач; однако авторы работали с серьезными ограничениями на внесение и извлечение данных, чтобы избежать потенциально непомерных затрат на перемещение данных. Для сложных рабочих нагрузок, таких как анализ потоковых данных, обсуждаемый в этой книге, мы ожидаем, что это преимущество затрат HPC останется и в будущем. Исследование Национальной академии наук 2016 года пришло к такому же выводу, что даже долгосрочная аренда у Amazon, вероятно, будет в 2–3 раза дороже, чем системы HPC, для обработки ожидаемой научной нагрузки от Национального научного фонда NSF (Box 6.2 от Национальной академии наук [2016]).

22.5. Программное обеспечение высокопроизводительных вычислений

По иронии судьбы, реальная мощь суперкомпьютера заключается в его специализированном программном обеспечении. Как для систем HPC, так и для облачных систем имеется широкий выбор пакетов программ. В большинстве случаев один и тот же пакет ПО доступен на обеих платформах. Поэтому мы решили сосредоточиться на пакетах программного обеспечения, которые являются уникальными для систем HPC и имеют потенциал для улучшения технологий вычислительного интеллекта и прогнозирования.

Одной из заметных особенностей экосистемы ПО HPC является то, что подавляющая часть прикладного программного обеспечения выполняет свою собственную

межпроцессорную связь через интерфейс передачи сообщений (message passing interface, MPI). В самом деле, основой большинства научно-вычислительных книг является MPI (Kumar и соавт. [1994], Gropp, Lusk and Skjellum [1999]). Соответственно, наше обсуждение программных средств HPC начнется с MPI. Поскольку эта книга основана на алгоритмах обработки данных, мы сосредоточимся на инструментах управления данными (Shoshami and Rotem [2010]).

22.5.1. Интерфейс передачи сообщений

Интерфейс передачи сообщений — это коммуникационный протокол для параллельных вычислений (Gropp, Lusk and Skjellum [1999], Snir и соавт. [1988]). Он определяет ряд операций межпунктового (из точки в точку) обмена данными, а также некоторые операции коллективной связи. Стандарт MPI был установлен на основе нескольких ранних попыток создания портативных коммуникационных библиотек. Ранняя реализация от Argonne National Lab, названная MPICH, имела высокую производительность, масштабируемость и портативность. Это помогло протоколу MPI получить широкое признание среди научных пользователей.

Успех протокола MPI частично обусловлен его отделением языково-независимых спецификаций (LIS) от языковых привязок. Это позволяет предоставлять одну и ту же стержневую функцию многим разным языкам программирования, что также способствует его принятию. Первый стандарт протокола MPI определил привязки ANSI C и Fortran-77 вместе с LIS. Черновая спецификация была представлена сообществу пользователей на конференции по суперкомпьютерам 1994 года.

Еще одним ключевым фактором успеха протокола MPI является лицензия с открытым исходным кодом, используемая в библиотеке MPICH. Эта лицензия позволяет поставщикам использовать исходный код для создания своих собственных версий, а это в свою очередь позволяет поставщикам систем HPC быстро создавать свои собственные библиотеки MPI. По сей день все системы HPC поддерживают взаимно распознаваемый протокол MPI на своих компьютерах. Это широкое внедрение также гарантирует, что протокол MPI будет оставаться любимым протоколом связи среди пользователей систем HPC.

22.5.2. Иерархический формат данных 5 (HDF5)

Описывая аппаратные компоненты HPC, мы отметили, что системы хранения данных на платформе HPC обычно отличаются от систем на облачной платформе. Соответственно, библиотеки программного обеспечения, используемого большинством пользователей для доступа к системам хранения, также разнятся. Это различие можно проследить по разнице в концептуальных моделях данных. Как правило, приложения HPC рассматривают данные как многомерные массивы, и поэтому наиболее популярные библиотеки ввода-вывода в системах HPC предназначены для работы с многомерными массивами. Здесь мы опишем наиболее широко используемую библиотеку массивоподобного формата HDF5 (Folk и соавт. [2011]).

HDF5 — это пятая итерация иерархического формата данных, созданная некоммерческой корпорацией HDF Group[1]. Основной единицей данных в формате HDF5 является массив плюс связанная с ним информация, такая как атрибуты, размерности и тип данных. Вместе они называются совокупностью данных. Совокупности данных могут быть сгруппированы в большие единицы, называемые группами, а группы могут быть организованы в высокоуровневые группы. Эта гибкая иерархическая организация позволяет пользователям выражать сложные отношения между совокупностями данных.

Помимо базовой библиотеки для организации пользовательских данных в файлы, некоммерческая корпорация HDF Group также предоставляет набор инструментов и специализацию формата HDF5 для различных приложений. Например, формат HDF5 содержит средство профилирования производительности. NASA имеет специализацию формата HDF5, именуемую HDF5-EOS, для данных из их системы наблюдения Земли (EOS); и сообщество следующего поколения секвенирования ДНК выпустило для своих биоинформационных данных специализацию под названием BioHDF.

Формат HDF5 обеспечивает эффективный способ доступа к системам хранения данных на платформе HPC. В тестах мы продемонстрировали, что использование формата HDF5 для хранения данных фондовых рынков значительно ускоряет операции анализа. Это во многом связано с его эффективными алгоритмами сжатия/разжатия, которые минимизируют сетевой трафик и операции ввода-вывода, что подводит нас к следующему пункту.

22.5.3. Обработка прямо на месте

За последние несколько десятилетий производительность CPU примерно удваивалась каждые 18 месяцев (по закону Мура), в то время как производительность диска увеличивалась менее чем на 5 % в год. Это различие приводило к тому, что ему требовалось все больше и больше времени на запись содержимого памяти процессора. Для решения этой проблемы ряд исследовательских усилий был сосредоточен на обеспечении возможности анализа прямо на месте (Ayachit и соавт. [2016]).

Среди обрабатывающих систем текущего поколения наиболее широко используется адаптируемая система ввода/вывода (adaptable i/o system, ADIOS) (Liu и соавт. [2014]). В ней используется несколько транспортных механизмов передачи данных, которые позволяют потребителям подключаться к потоку ввода/вывода и выполнять аналитические операции. Это полезно тем, что малозначительные данные могут отбрасываться на лету, что позволяет избегать медленного и объемного хранения. Этот же механизм с обработкой прямо на месте также позволяет очень быстро завершать операции записи. По сути дела, изначально он привлек

[1] См. веб-сайт некоммерческой организации HDF Group: https://www.hdfgroup.org/.

внимание именно из-за своей скорости записи. С тех пор разработчики системы ADIOS работали с несколькими очень большими командами над усовершенствованием своих конвейеров ввода-вывода и возможностей их анализа.

Поскольку система ADIOS поддерживает потоковый доступ к данным, она также очень важна для работы проекта CIFT. В ряде демонстраций системы ADIOS с транспортным механизмом ICEE смог завершить распределенный анализ потоковых данных в режиме реального времени (Choi и соавт. [2013]). В следующем далее разделе мы опишем один из вариантов с использованием пятен в термоядерной плазме.

Подводя итог, можно сказать, что возможность обработки данных прямо на месте является еще одним весьма полезным инструментом экосистемы HPC.

22.5.4. Конвергенция

Мы упоминали ранее, что рынок оборудования для высокопроизводительных вычислений (HPC) является крошечной частью совокупного рынка компьютерного оборудования. Рынок программного обеспечения HPC еще меньше по сравнению с совокупным рынком программного обеспечения. До сих пор экосистема программного обеспечения HPC в основном поддерживалась рядом небольших поставщиков вместе с несколькими участниками, работающими с открытым исходным кодом. Таким образом, пользователи системы HPC находятся под огромным давлением, толкающим их мигрировать в более качественно поддерживаемые системы облачного программного обеспечения. Это важный драйвер для конвергенции между программным обеспечением для HPC и программным обеспечением для облака (Fox и соавт. [2015]).

Несмотря на то что конвергенция кажется неизбежной, мы выступаем за вариант конвергенции, который сохраняет преимущество упомянутых выше программных средств. Одним из мотивов проекта CIFT является поиск способа переноса вышеуказанных инструментов в вычислительные среды будущего.

22.6. Примеры использования

Обработка данных является настолько важной частью современных научных исследований, что некоторые исследователи называют ее четвертой парадигмой науки (Hey, Tansley and Tolle [2009]). В экономике те же направляемые данными исследования привели к дико популярной поведенческой экономике (Camerer and Loewenstein [2011]). Подавляющая часть последних достижений в области направляемых данными исследований основана на применении машинного обучения (Qiu и соавт. [2016], Rudin and Wagstaff [2014]). Их успехи, например в планетологии и биоинформатике, вызвали значительный интерес среди исследователей из различных областей. В остальной части этого раздела мы опишем несколько

примеров применения передовых методов анализа данных в различных областях, где многие из этих примеров были рождены в рамках проекта CIFT.

22.6.1. Поиск сверхновой звезды

В астрономии определение многих важных фактов, таких как скорость расширения Вселенной, выполняется путем измерения света от взрывающихся сверхновых типа Ia (Bloom и соавт. [2012]). Процесс поиска в ночном небе взрывающихся сверхновых называется синоптической съемкой. Примером такого синоптического обследования является Паломарская транзиентная фабрика (Palomar transient factory, PTF) (Nicholas и соавт. [2009]). Телескопы PTF сканируют ночное небо и производят набор изображений каждые 45 минут. Новое изображение сравнивается с предыдущим наблюдением одинакового участка неба с целью определения изменений и классифицирования этих изменений. Такие задачи идентификации и классификации выполнялись астрономами вручную. Однако текущее число поступающих изображений с телескопов PTF слишком велико для ручной проверки. Для этих задач обработки изображений был разработан автоматизированный рабочий процесс и затем развернут в целом ряде разных компьютерных центров.

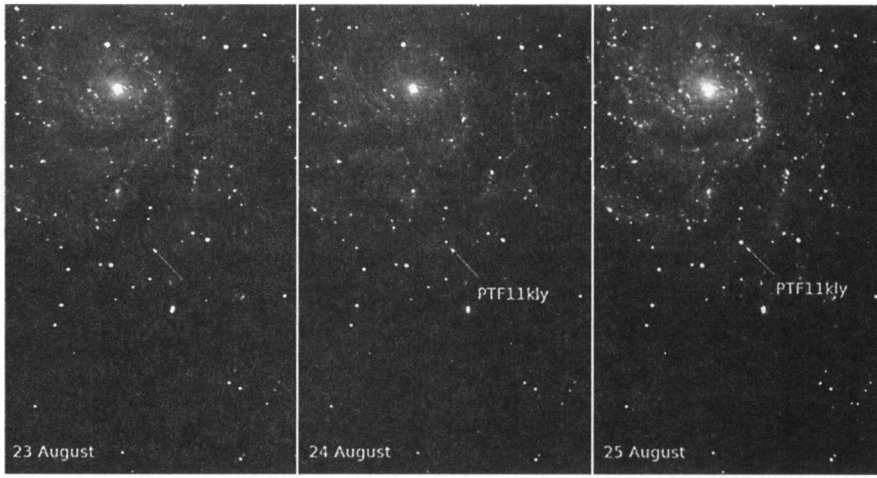


Рис. 22.4. Сверхновая SN2011fe была обнаружена 11 часов спустя после первых свидетельств взрыва благодаря расширенной автоматизации в классифицировании астрономических наблюдений

На рис. 22.4 показана сверхновая, которая была идентифицирована самой ранней, в процессе взрыва. 23 августа 2011 года участок неба не показывал никаких признаков этой звезды, но слабый свет появился 24 августа. Эта быстрая оборачиваемость позволила астрономам по всему миру выполнять подробные последовавшие на-

блюдения, которые важны для определения параметров, связанных с расширением Вселенной.

Быстрая идентификация этой сверхновой является важной демонстрацией способностей МО автоматизированного рабочего процесса. Данный рабочий процесс обрабатывает входящие изображения с целью извлечения объектов, которые изменились с момента последнего наблюдения. Затем он классифицирует измененный объект, для того чтобы определить предварительный тип на основе предварительной тренировки. Поскольку последующие ресурсы для извлечения новых научных знаний из быстро меняющихся транзиентов являются ценными, классификация должна не только указывать предполагаемый тип, но и вероятность, и достоверность классификации. Используя классификационные алгоритмы, натренированные на данных PTF, искаженная маркировка транзиентов и переменных звезд имеет совокупную частоту ошибок 3,8 %. Дополнительная работа, как ожидается, позволит достичь более высоких уровней точности в предстоящих исследованиях, таких как большой синоптический обзорный телескоп.

22.6.2. Пятна в термоядерной плазме

Крупномасштабные научные исследования в таких областях, как физика и климатология, являются предметом огромного международного сотрудничества, в котором участвуют тысячи ученых. По мере того как общие усилия производят все больше и больше данных с поступательно увеличивающимися темпами, существующие системы управления потоком операций с трудом поспевают их обрабатывать. Необходимым решением является обработка, анализ, резюмирование и сокращение объема данных до того, как они достигнут относительно медленной дисковой системы хранения. Такой процесс называется обработкой в пути (или анализом в полете). Работая с разработчиками системы ADIOS, мы внедрили транспортный механизм ICEE с целью значительного увеличения возможности обработки данных в системах совместного потока операций (Choi и соавт. [2013]). Эта новая функциональная возможность значительно улучшила управление потоками данных для распределенных потоков операций. Испытания показали, что механизм ICEE позволил ряду крупных международных совместных усилий принимать совместные решения почти в режиме реального времени. Здесь мы кратко описываем совместные усилия в области термоядерной реакции с участием реактора KSTAR.

KSTAR — это термоядерный реактор с полностью сверхпроводящими магнитами. Он расположен в Южной Корее, но помимо него существует ряд соответствующих исследовательских групп по всему миру. Во время пуска термоядерного эксперимента некоторые исследователи управляют физическим устройством в реакторе KSTAR, а другие вполне могут захотеть принять участие в совместном анализе предыдущих пусков эксперимента, чтобы дать советы по поводу того, как настроить устройство для следующего пуска. Во время анализа экспериментальных данных замеров ученые могут провести симулирования или изучить предыдущие

симулирования с целью изучения выбранных параметров. Как правило, между двумя пусками подряд может пройти от 10 до 30 минут, и все совместные анализы должны быть завершены в течение этого промежутка времени, чтобы повлиять на следующий пуск.

Мы продемонстрировали функциональность системы управления потоком операций на основе транспортного механизма ICEE с двумя различными типами данных: один тип поступает из электронной циклотронной эмиссионной визуализации (ECEI), измеряемой в реакторе KSTAR, а другой связан с использованием синтетических диагностических данных из моделирования XGC. Механизм распределенного потока операций должен собирать данные из этих двух источников, извлекать признак, именуемый пятнами, отслеживать перемещение этих пятен, прогнозировать перемещение пятен в экспериментальных замерах, а затем предоставлять рекомендации по выполняемым действиям. На рис. 22.5 показано, как обрабатываются данные ECEI. Поток операций для симуляционных данных XGC подобен тому, что показано на рис. 22.5, за исключением того, что данные XGC расположены в Национальном научно-исследовательском вычислительном центре по энергетике NERSC.

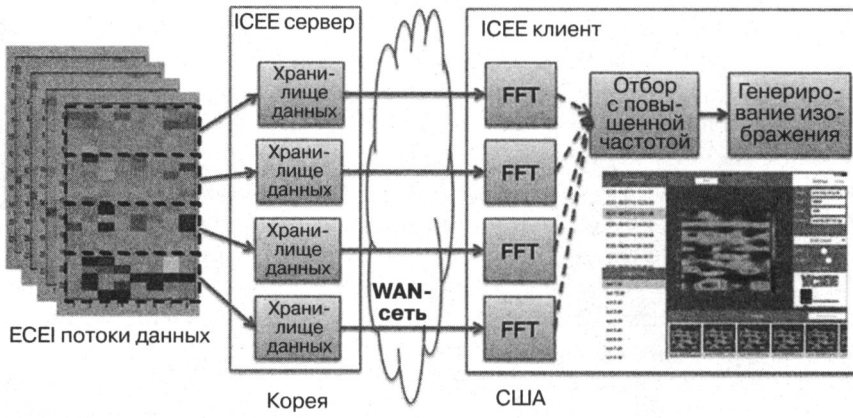


Рис. 22.5. Распределенный поток данных для изучения динамики термоядерной плазмы

Для того чтобы иметь возможность выполнять вышеуказанные аналитические задачи в режиме реального времени, эффективное управление данными с помощью транспортного механизма ICEE системы ADIOS — это только часть истории. Вторая часть — эффективное обнаружение пятен (Wu и соавт. [2016]). В этой работе нам необходимо сократить объем данных, передаваемых по территориально распределенным сетям (WAN), путем отбора только необходимых фрагментов. Затем мы выделяем все ячейки внутри пятен и группируем эти ячейки в связных областях в пространстве, где каждая связная область образует пятно. Новый разработанный нами алгоритм подразделяет работу на различные ядра CPU, используя все преиму-

щества протокола MPI для обмена данными между узлами и общей памятью между ядрами CPU на одном узле. Кроме того, мы также обновили алгоритм маркировки связных компонент, чтобы он правильно определял пятна на краю, которые часто пропускались более ранними алгоритмами обнаружения. В целом, наш алгоритм смог идентифицировать пятна за несколько миллисекунд для каждого временного шага, используя все преимущества параллелизма, доступного в системе HPC.

22.6.3. Внутридневное пиковое потребление электроэнергии

Коммунальные предприятия развертывают современную инфраструктуру учета (advanced metering infrastructure, AMI) с целью фиксирования потребления электроэнергии с беспрецедентной пространственной и временной детализацией. Этот обширный и быстрорастущий поток данных является важным полигоном для тестирования возможностей прогнозирования на основе платформ анализа больших данных (Kim и соавт. [2015]). Эти передовые методы науки о данных, наряду с поведенческими теориями, позволяют поведенческой аналитике получить новое представление о моделях потребления электроэнергии и лежащих в их основе факторах (Todd и соавт. [2014]).

Поскольку электричество не может быть легко сохранено, его производство должно соответствовать потреблению. Когда спрос превышает генерирующие мощности, происходит отключение электроэнергии и, как правило, в то время, когда потребители больше всего в ней нуждаются. Поскольку увеличение генерирующих мощностей является дорогостоящим и требует многих лет, регулирующие и коммунальные компании разработали ряд схем тарификации, призванных препятствовать ненужному потреблению в периоды пикового спроса.

Для измерения эффективности ценовой политики при пиковом спросе можно анализировать данные об использовании электроэнергии, генерируемые инфраструктурой AMI. Наша работа направлена на извлечение базовых моделей потребления электроэнергии в домашних хозяйствах с целью анализа их поведения. В идеале базовые модели должны отражать структуру потребления электроэнергии домашними хозяйствами, включая все признаки, за исключением новых схем тарификации. Создание такой модели сопряжено с многочисленными трудностями. Например, существует ряд признаков, которые могут влиять на потребление электроэнергии, но для которых не записывается никакой информации, например, заданная температура кондиционера или покупка нового прибора. Другие признаки, такие как наружная температура, известны, но их влияние трудно уловить в простых функциях.

В ходе нашей работы был разработан ряд новых базовых моделей, которые могли бы удовлетворить вышеупомянутые потребности. В настоящее время базовым золотым стандартом является хорошо спроектированная рандомизированная контрольная группа. Мы показали, что наши новые направляемые данными ба-

зовые модели могут точно предсказывать среднее потребление электроэнергии контрольной группой. Для его оценивания мы используем хорошо разработанное исследование из региона Соединенных Штатов, где потребление электроэнергии является самым высоким во второй половине дня и вечером в течение месяцев с мая по август.

Хотя эта работа сосредоточена на демонстрации того, что новые базовые модели эффективны для групп, мы считаем, что в будущем эти новые модели также будут полезны для изучения индивидуальных домохозяйств.

Мы разведали ряд стандартных черноящичных подходов. Среди методов машинного (само)обучения мы обнаружили, что градиентное бустирование деревьев (gradient tree boosting, GTB) эффективнее, чем другие. Однако наиболее точные модели GTB требуют наличия в качестве признаков лаговых (запаздывающих) переменных (например, потребление электроэнергии днем ранее и неделей ранее). В нашей работе нам необходимо использовать данные года T – 1 для установления базового потребления для года T и года T + 1. Лаговая переменная за день до этого и за неделю до этого будет включать в себя последнюю информацию, которая не относится к году T – 1. Мы попытались модифицировать процедуру предсказания для того, чтобы использовать последние предсказания вместо фактических измеренных значений за день до этого и за неделю до этого; однако наши тесты показывают, что предсказательные ошибки накапливаются с течением времени, что приводит к нереалистичным предсказаниям в течение месяца или около того в летний сезон. Этот тип накопления предсказательных ошибок широко распространен среди непрерывных предсказательных процедур для временных рядов.

Для решения вышеуказанной проблемы мы разработали ряд белоящичных подходов, наиболее эффективный из которых, именуемый LTAP, рассматривается далее. Подход LTAP основывается на том, что агрегированная переменная потребления электроэнергии в сутки точно описывается кусочно-линейной функцией среднесуточной температуры. Этот факт позволяет делать прогнозы об общем суточном потреблении электроэнергии. Далее, исходя из того, что потребительский профиль каждого домохозяйства остается неизменным во время исследования, мы можем назначать почасовые значения потребления из суточного совокупного потребления. Этот подход является самосогласованным, то есть предсказательная процедура точно воспроизводит потребление электроэнергии в году T – 1, и предсказания для контрольной группы в году T и T + 1 очень близки к измеренным значениям. Обе обрабатываемые группы сократили потребление электричества во время часов с пиковым спросом, и активная группа сократила потребление больше, чем пассивная группа. Это наблюдение согласуется с другими исследованиями.

Хотя новая базовая направляемая данными модель LTAP точно предсказывает средний уровень потребления контрольной группой, существуют некоторые различия в предсказываемом влиянии новой зависящей от времени потребления тарификации, предназначенной для сокращения потребления в часы пикового спроса (см. рис. 22.6). Например, при использовании контрольной группы

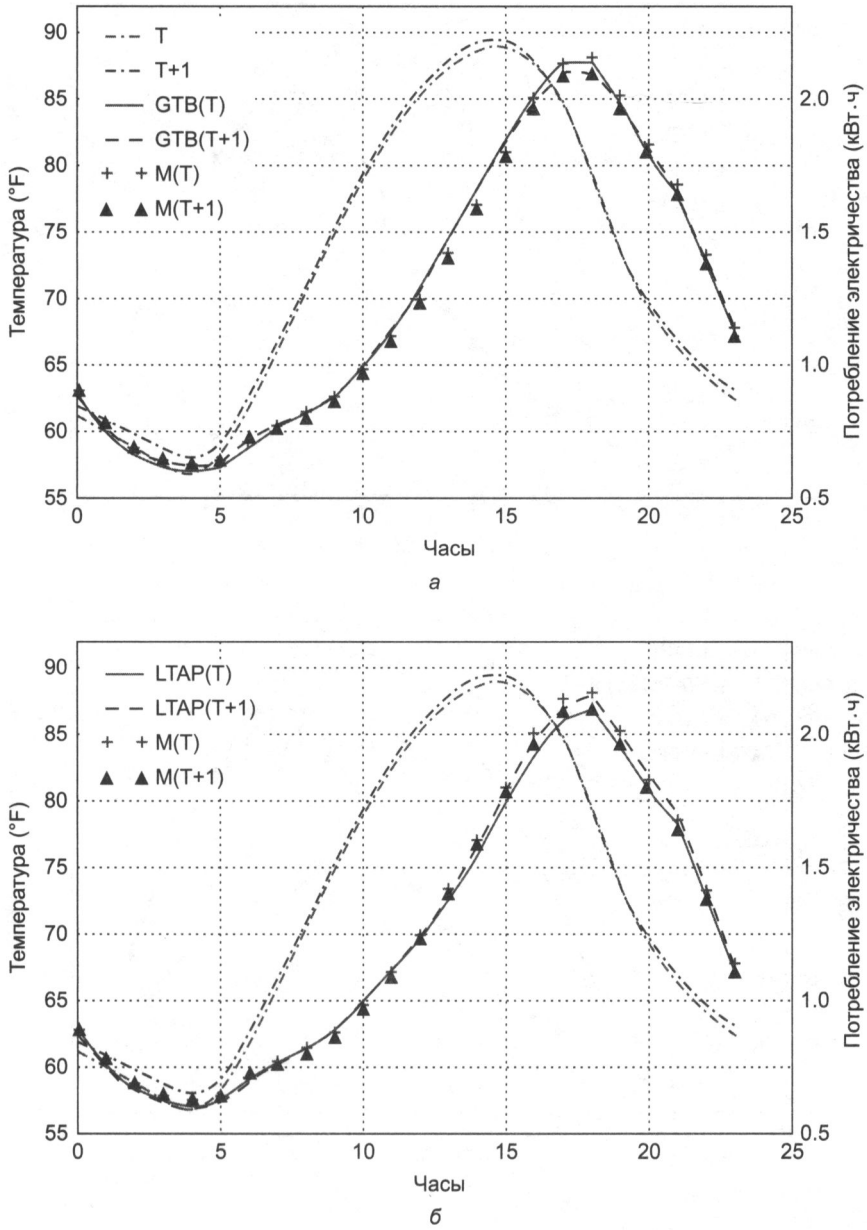
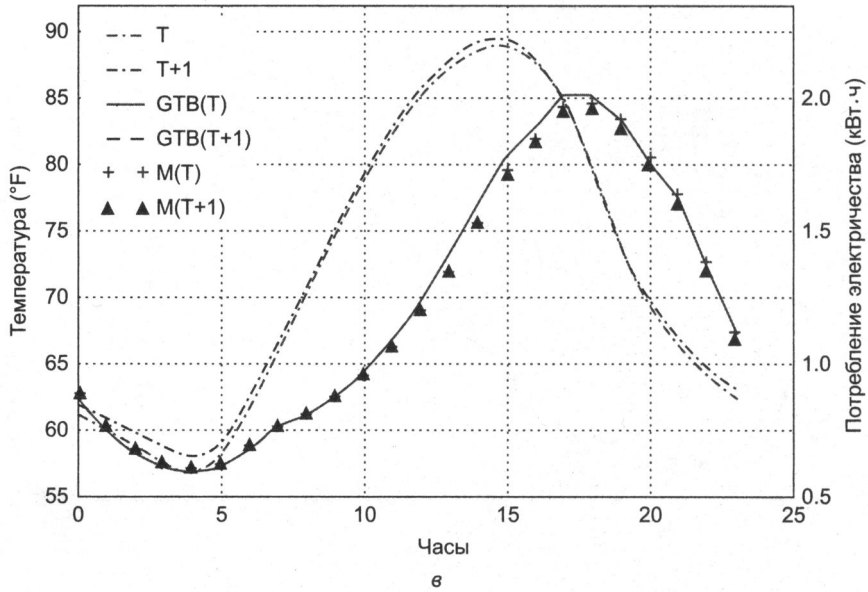
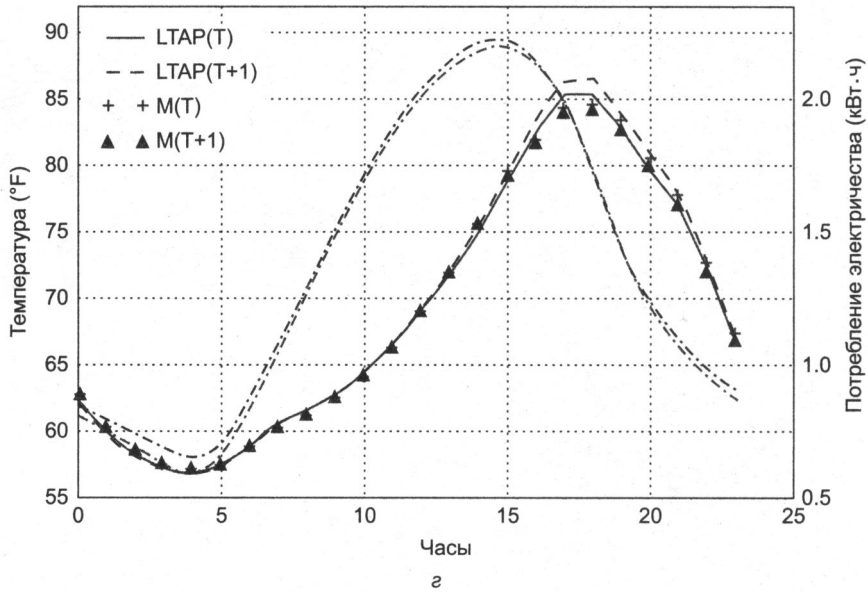


Рис. 22.6. Градиентное бустирование деревьев (GBT), по всей видимости, слишком близко следует за недавним потреблением и поэтому не может предсказать базовое потребление, а также новый разработанный метод LTAP. *а*) GBT на контрольной группе; *б*) LTAP на контрольной группе; *в*) GBT на пассивной группе; *г*) LTAP на пассивной группе; *д*) GBT на активной группе; *е*) LTAP на активной группе

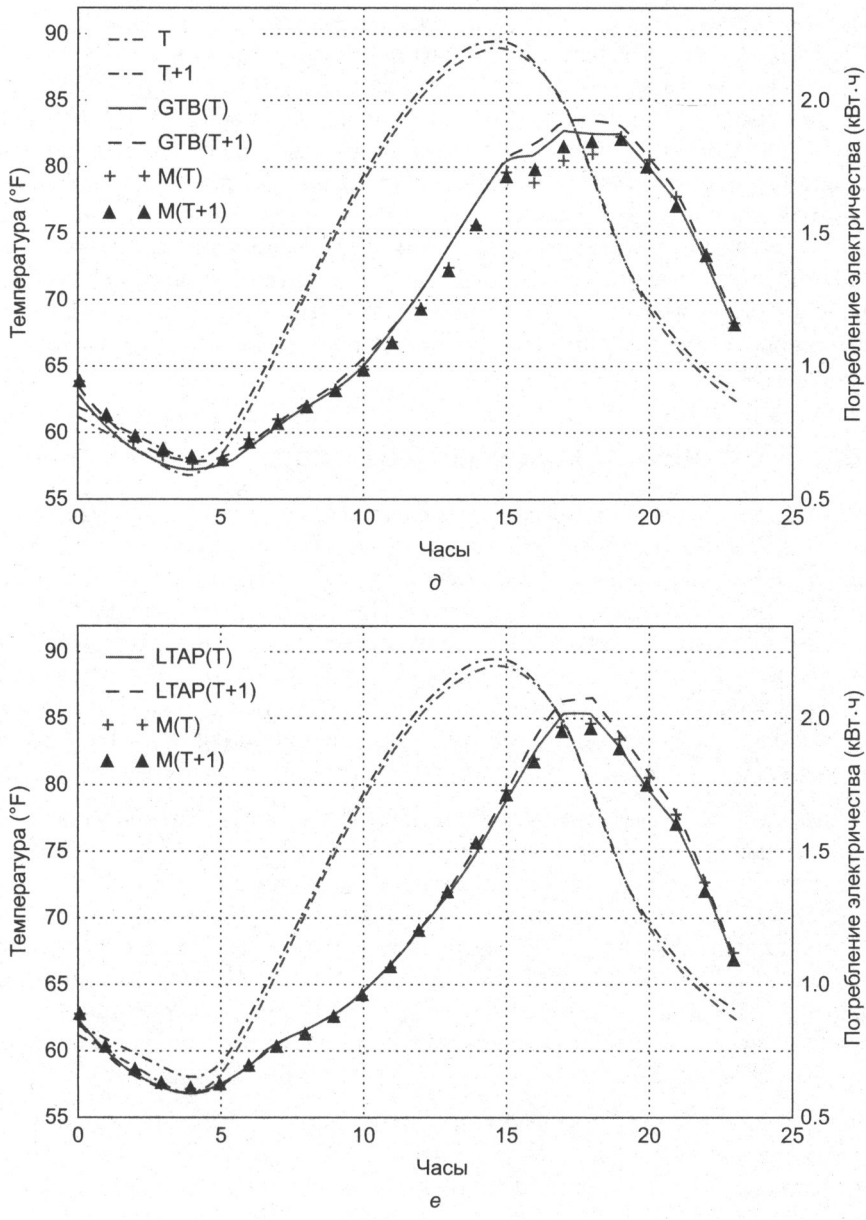


в



г

Рис. 22.6 (*продолжение*)



д

е

Рис. 22.6 (*окончание*)

в качестве базовой активная группа сокращает свое потребление на 0,277 кВт · ч (из примерно 2 кВт · ч), усредненно по часам пикового спроса в первый год с новой ценой и 0,198 кВт · ч во второй год. Используя модель LTAP в качестве базовой, средние сокращения составляют только 0,164 кВт · ч в обоих годах. Отчасти эта разница может быть обусловлена систематическим смещением при самоотборе в обрабатываемых группах, в особенности в активной группе, где домохозяйствам приходится явным образом давать свое согласие на участие в испытании. Вполне вероятно, что домохозяйства, которые решили присоединиться к активной группе, вполне могут воспользоваться предлагаемой новой структурой тарификации. Мы считаем, что базовая модель LTAP является способом устранения систематического смещения при отборе, и планируем провести дополнительные исследования, чтобы в этом убедиться.

22.6.4. Молниеносный обвал 2010 года

Дополнительное время, которое потребовалось комиссиям SEC и CFTC для расследования молниеносного обвала 2010 года, было первоначальным мотивом для работы проекта CIFT. Федеральным следователям нужно было просеять десятки терабайт данных, чтобы найти первопричину обвала. Поскольку комиссия CFTC публично обвинила объем данных как источник продолжительной задержки, мы начали нашу работу с поиска инструментов HPC, которые могли бы легко обрабатывать десятки терабайт. Поскольку HDF5 является наиболее часто используемой библиотекой ввода/вывода, мы начали нашу работу с применения библиотеки HDF5 с целью организовать большую совокупность биржевых данных (Bethel и соавт. [2011]).

Давайте быстро проведем обзор того, что произошло во время молниеносного обвала 2010 года. 6 мая, около 14:45 по восточному летнему времени США, промышленный индекс Доу—Джонса (Dow Jones Industrial Average) упал почти на 10 %, и многие акции торговались по цене одного процента за акцию, минимальной цене для любой возможной сделки. На рис. 22.7 показан пример еще одного экстремального случая, когда акции Apple (символ AAPL) торговались по цене $100 000 за акцию, максимально возможной цене, разрешенной биржей. Понятно, что это были необычные события, которые подорвали доверие инвесторов к нашим финансовым рынкам. Инвесторы требовали объяснений, чем эти события были вызваны.

Для нашей работы мы реализовали две версии программы: одна использует данные, организованные в файлы в формате HDF5, а другая считывает данные из часто используемых текстовых файлов в формате ASCII. На рис. 22.8 показано время, необходимое для обработки торговых записей всех акций S&P 500 в течение 10-летнего периода. Поскольку размер 10-летних торговых данных все еще относительно невелик, мы также реплицировали данные 10 раз. На одном ядре процессора (обозначенном как «Serial» на рис. 22.8) потребовалось около 3,5 часа с данными ASCII, но только 603,98 секунды с файлами HDF5. При использовании 512 ядер процессора это время уменьшается до 2,58 секунды при использовании файлов HDF5, что приводит к ускорению в 234 раза.

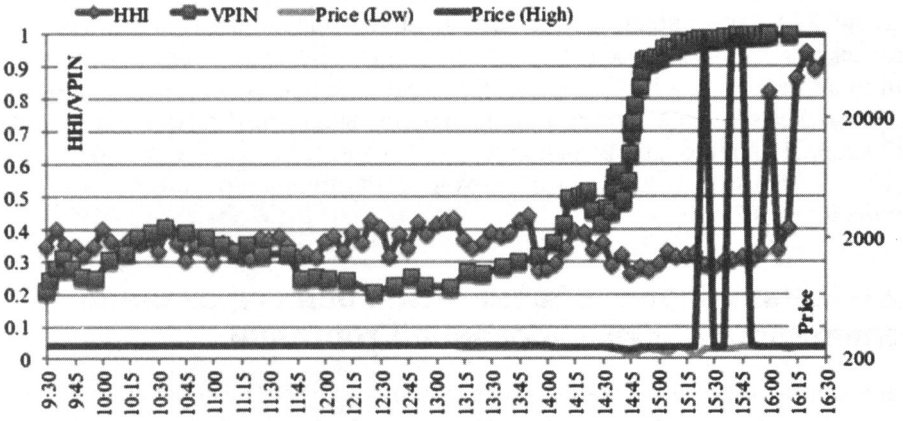


Рис. 22.7. Цены на акции Apple на момент 6 мая 2010 года вместе со значениями индекса Херфиндаля—Хиршмана и вероятностью информированного трейдинга, синхронизированного с объемом, рассчитываемыми каждые 5 минут в часы работы рынка

На более крупной (реплицированной) совокупности данных преимущество программного кода HPC для вычисления этих индексов еще более выражено. С данными, в 10 раз большими по объему, компьютеру потребовалось всего примерно в 2,3 раза больше времени, чтобы завершить задачи, показав сублинейное увеличение задержки. Использование большего числа CPU делает систему HPC еще более масштабируемой.

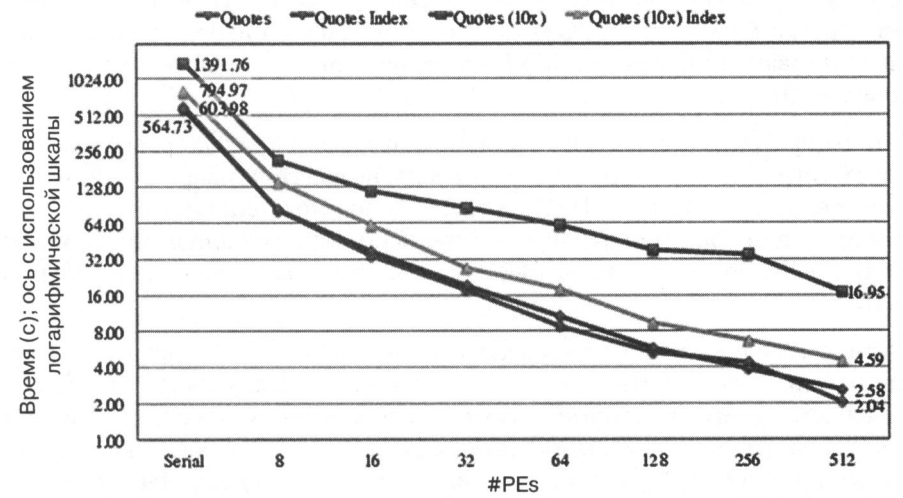


Рис. 22.8. Время на обработку 10-летних данных котировок S&P500 в файлах в формате HDF5: файлы HDF5 обрабатываются в 21 раз быстрее, чем файлы ASCII (603,98 секунды против 3,5 часа)

На рис. 22.8 также показано, что при наличии большой совокупности данных мы можем воспользоваться дальнейшими преимуществами методов индексирования, имеющихся в HDF5, которые сокращают время доступа к данным (что, в свою очередь, сокращает суммарное время вычислений). При использовании 512 ядер CPU общее время выполнения сокращается с 16,95 секунды до 4,59 секунды, то есть благодаря этому методу HPC индексирования мы получаем ускорение в 3,7 раза.

22.6.5. Калибровка объемно-синхронизированной вероятности информированной торговли

Понимание волатильности финансового рынка требует обработки огромного количества данных. Мы применяем методы из предназначенных для этой задачи научных приложений с интенсивной обработкой данных и демонстрируем их эффективность, вычисляя индикатор раннего предупреждения под названием объемно синхронизированная вероятность информированной торговли (Volume Synchronized Probability of Informed Trading, VPIN) на массивном наборе фьючерсных контрактов. Тестовые данные содержат 67 месяцев торгов по сотне наиболее часто торгуемых фьючерсных контрактов. В среднем обработка одного контракта за 67 месяцев занимает около 1,5 секунды. До этой высокопроизводительной вычислительной реализации на завершение такой же задачи требовалось около 18 минут. Наша реализация HPC достигает ускорения в 720 раз.

Обратите внимание, что указанное ускорение было получено исключительно на основе алгоритмического улучшения без преимущества параллелизации. Программный код HPC может выполняться на параллельных машинах с использованием протокола MPI, что позволяет еще больше сократить время вычислений.

Методы программного обеспечения, используемые в нашей работе, включают более быстрый доступ при операциях ввода/вывода благодаря описанной выше библиотеке для формата HDF5, а также более модернизированную структуру данных для хранения баров и корзин, используемых для вычисления индикатора VPIN. Более подробная информация доступна в публикации Wu и соавт. [2013].

С помощью более быстрой программы для вычисления индикатора VPIN мы также смогли подробнее изучить варианты параметров. Например, мы смогли определить значения параметров, которые снижают частоту ложных утверждений индикатора VPIN на сотне контрактов с 20 до 7 %, см. рис. 22.9. Варианты параметров для достижения этой эффективности были следующими: 1) оценивание объемного бара с помощью медианных цен сделок (в отличие от цены закрытия, которая, как правило, используется в анализе), 2) 200 корзин в день, 3) 30 баров на корзину, 4) поддержка окна для вычисления VPIN = 1 день, продолжительность события = 0,1 дня, 5) классификация суммарного объема с помощью t-распределения

Стьюдента[1] с v = 0,1 и 6) порог кумулятивной функции распределения (CDF) показателя VPIN = 0,99. Опять же, эти параметры обеспечивают низкую частоту ложных утверждений на совокупности фьючерсных контрактов и не являются результатом индивидуальной подгонки.

Рис. 22.9. Средние частоты ложных утверждений (α) относительно разных классов фьючерсных контрактов, упорядоченных согласно их средним значениям

Для достижения еще более низких частот ложных утверждений на разных классах фьючерсных контрактов можно выбрать разные параметры. В некоторых случаях частоты ложных утверждений могут упасть значительно ниже 1 %. Исходя из рис. 22.9, процентные ставки и индексные фьючерсные контракты, как правило, имеют более низкие частоты ложных утверждений. Фьючерсные контракты на биржевые товары, такие как энергоносители и металлы, как правило, имеют более высокие частоты ложных утверждений.

Вдобавок, более быстрая программа для вычисления индикатора VPIN позволяет нам проверять, что события, определенные индикатором VPIN, являются «внутренними» в том смысле, что изменение параметров, таких как порог на кумулятивную функцию распределения VPIN CDF, только немного изменяет число обнаруженных событий. Если бы события были случайными, изменение этого порога с 0,9 до 0,99 уменьшало бы число событий в 10 раз. Одним словом, более быстрая программа VPIN также позволяет подтвердить реально-временную эффективность индикатора VPIN.

[1] t-распределение (t-distribution) — эталонное распределение (в частности, полученное из нулевой гипотезы), с которым может быть сопоставлено наблюдаемое t-значение. — *Примеч. науч. ред.*

22.6.6. Выявление высокочастотных событий с помощью неравномерного быстрого преобразования Фурье

Высокочастотная торговля широко распространена на всех электронных финансовых рынках. По мере того как алгоритмы все чаще выполняют задачи, ранее выполнявшиеся людьми, каскадные эффекты, подобные молниеносному обвалу 2010 года, могут стать более вероятными. В нашей работе (Song и соавт. [2014]) мы объединили ряд высокопроизводительных инструментов обработки сигналов, с тем чтобы улучшить наше понимание этих торговых операций. В качестве иллюстрации мы подведем итоги Фурье-анализа торговых цен фьючерсов на природный газ.

Как правило, анализ Фурье применяется к равномерно расположенным данным. Поскольку рыночная активность возникает всплесками, мы вполне можем захотеть выполнять отбор из финансовых временных рядов в соответствии с индексом торговой активности. Например, индикатор VPIN выполняет отбор из финансового ряда в зависимости от торгуемого объема. Однако анализ Фурье финансовых рядов в хронологическом порядке все же может оказаться поучительным. Для этой цели мы используем неравномерную процедуру быстрого преобразования Фурье (FFT).

Из Фурье-анализа фьючерсного рынка природного газа мы видим сильные доказательства высокочастотной торговли на рынке. Фурье-компоненты, соответствующие высоким частотам: 1) в последние годы становятся все более заметными, и 2) гораздо сильнее, чем можно было бы ожидать от структуры рынка. Кроме того, значительная величина торговой активности происходит в первую секунду каждой минуты, что является контрольным признаком торговли, вызванной алгоритмами, которые нацелены на средневзвешенную по времени цену (TWAP).

Фурье-анализ на торговых данных показывает, что активность на частоте раз в минуту значительно выше, чем на соседних частотах (рис. 22.10). Обратите внимание, что вертикальная ось имеет логарифмическую шкалу. Сила действий на частоте раз в минуту более чем в 10 раз выше соседних частот. Кроме того, активность очень точно определяется раз в минуту, что указывает на то, что эти сделки запускаются намеренно созданными автоматическими событиями. Мы считаем это убедительным доказательством того, что на этом рынке значительное присутствие имеют алгоритмы TWAP.

Мы ожидали, что анализ частотности покажет ярко выраженные дневные циклы. На рис. 22.10 амплитуда частотности 365 должна была быть широкой. Однако, как мы видим, самая высокая амплитуда зафиксирована для частотности 366. Это можно объяснить тем, что 2012 год был годом скачков. Это подтверждает тот факт, что неравномерное БПФ фиксирует ожидаемые сигналы. Вторая и третья по величине амплитуды имеют частотности 732 и 52, что соотносится с ежедневным и еженедельным показателем. В этом также нет ничего удивительного.

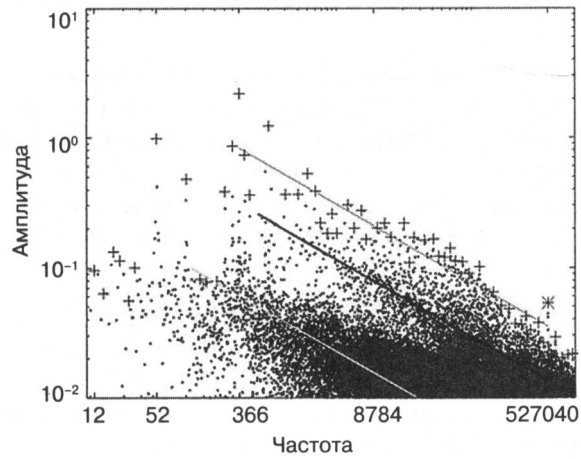


Рис. 22.10. Фурье-спектр торговых цен фьючерсных контрактов на природный газ в 2012 году. Неравномерное быстрое преобразование Фурье (FFT) определяет сильное присутствие действий, происходящих один раз в день (частота = 366), два раза в день (частота = 732) и один раз в минуту (частота = 527 040 = 366 · 24 · 60)

Мы дополнительно применили неравномерное быстрое преобразование Фурье (FFT) на торговых объемах и нашли дополнительные доказательства алгоритмической торговли. Более того, сигналы указывали на более сильное присутствие алгоритмической торговли в последние годы. Очевидно, что алгоритм неравномерного быстрого преобразования Фурье (FFT) полезен для анализа сильно нерегулярных временных рядов.

22.7. Итоги и призыв к сотрудничеству

В настоящее время существует два основных способа построения крупномасштабных вычислительных платформ: высокопроизводительный вычислительный подход и облачный подход. Большинство научных вычислений используют высокопроизводительный вычислительный подход (HPC), в то время как большинство потребностей деловых вычислений удовлетворяются с помощью облачного подхода. Расхожее мнение, что высокопроизводительный вычислительный подход занимает небольшую нишу, приносящую мало пользы. Это не соответствует действительности. Системы HPC необходимы для продвижения научных исследований. Они сыграли важную роль в новых интересных научных открытиях, включая бозон Хиггса и гравитационные волны. Они стимулировали разработку новых учебных предметов, таких как поведенческая экономика, и новые способы ведения торговли через интернет. Польза чрезвычайно больших

систем HPC привела к Национальной стратегической вычислительной инициативе 2015 года[1].

Предпринимаются усилия, направленные на то, чтобы сделать инструменты HPC еще более полезными за счет ускорения их внедрения в деловые приложения. Инициатива HPC4Manufacturing является первопроходцем в этой передаче знаний в обрабатывающую промышленность США и привлекла значительное внимание. Настало время предпринять более согласованные усилия, для того чтобы высокопроизводительные вычисления могли удовлетворять другие критически важные потребности бизнеса.

В последние годы мы разработали проект CIFT как широкий класс деловых приложений, которые могут извлечь выгоду из инструментов и методов HPC. В таких решениях, как реагирование на колебания напряжения в силовом трансформаторе и сигнал раннего предупреждения о надвигающемся событии волатильности рынка, программные средства HPC способны помочь определять сигналы достаточно рано для лиц, принимающих решения, обеспечивать достаточную достоверность в предсказании и предвидеть последствия до наступления катастрофического события. Эти приложения имеют сложные вычислительные потребности и часто также имеют строгое требование ко времени на ответ. Инструменты HPC лучше подходят для удовлетворения этих требований, чем облачные инструменты[2].

В нашей работе мы продемонстрировали, что библиотека HDF5 ввода/вывода для HPC может быть использована с целью увеличения скорости доступа к данным в 21 раз, а методы HPC могут ускорить вычисление индикатора раннего предупреждения VPIN о молниеносных обвалах в 720 раз. Мы разработали дополнительные алгоритмы, позволяющие предсказывать ежедневный пик потребления электроэнергии в будущем. Мы ожидаем, что применение инструментов и методов HPC к другим приложениям может достичь столь же значительных результатов.

В дополнение к упомянутым выше преимуществам производительности ряд опубликованных исследований (Yelick и соавт. [2011], Holzman и соавт. [2017]) показывает, что системы HPC также имеют значительное ценовое преимущество. В зависимости от требований рабочей нагрузки к CPU, хранилищу и сети использование облачной системы может стоить на 50 % дороже, чем использование системы HPC, а в некоторых случаях и в 7 раз дороже. В случае описанных в этой книге сложных аналитических задач с их постоянной потребностью в потреблении данных для анализа мы ожидаем, что преимущество по затратам будет оставаться большим.

---

[1] План Национальной стратегической вычислительной инициативы доступен онлайн по адресу https://www.whitehouse.gov/sites/whitehouse.gov/files/images/NSCI%20Strategic%20Plan.pdf. Страница «Википедии» по данной теме (https://en.wikipedia.org/wiki/National_Strategic_Computing_Initiative) также имеет несколько полезных ссылок на дополнительную информацию.

[2] Информация об инициативе HPC4Manufacturing имеется онлайн по адресу https://hpc4mfg.llnl.gov/.

Проект CIFT расширяет усилия по передаче технологии HPC частным компаниям, с тем чтобы они могли также извлекать выгоду из ценовых и эксплуатационных преимуществ крупномасштабного исследовательского оборудования. Наши предыдущие участники предоставили средства для запуска специальной системы HPC для нашей работы. Этот ресурс должен значительно облегчить заинтересованным сторонам опробование своих приложений в системе HPC. Мы открыты для различных форм сотрудничества. Для получения дополнительной информации о проекте CIFT, пожалуйста, посетите его веб-страницу на http://crd.lbl.gov/cift/.

22.8. Благодарности

Проект CIFT является детищем доктора Дэвида Лейнвебера. Доктор Хорст Саймон принес его в LBNL в 2010-м. Доктор Э. В. Вефиль и Д. Бейли руководили проектом в течение четырех лет.

Проект CIFT получил щедрые подарки от ряда инвесторов. Эта работа частично поддерживается Управлением перспективных научных исследований в области вычислительной техники Управления науки Министерства энергетики США по контракту No. DE-AC02-05CH11231. Это исследование также использует ресурсы Национального энергетического научно-исследовательского вычислительного центра, поддерживаемого тем же контрактом.

ПРИЛОЖЕНИЯ

А О книге «Машинное обучение: алгоритмы для бизнеса» Лопеза де Прадо

Энтони Кавалларо, 23 августа 2018 г.

Машинное обучение[1] — модное словцо, которым часто разбрасываются при обсуждении будущего финансов и всего мира. Возможно, вы слышали о нейронных сетях, решающих задачи распознавания лиц, обработки естественного языка и даже связанные с финансовыми рынками[2], но без особых объяснений. Легко рассматривать эту область как черный ящик, волшебную машину, которая каким-то образом производит решения, но никто не знает, почему она работает. Надо признать, что методы машинного обучения (в частности, нейронные сети) улавливают смутные и труднообъяснимые признаки, однако для исследований, кастомизации и анализа существует больше возможностей, чем может показаться на первый взгляд.

В этой статье мы обсудим на высоком уровне различные факторы, которые необходимо учитывать при исследовании вопроса инвестирования через призму машинного обучения. Содержание этой статьи и дальнейшие дискуссии по этой теме в значительной степени вдохновлены книгой Маркоса Лопеза де Прадо «Достижения в финансовом машинном обучении». Если вы хотите изучить его исследовательские работы подробнее, то его веб-сайт приведен в сноске[3].

А.1. Структуры данных

Мусор на входе → мусор на выходе — это мантра информатики, и она вдвойне относится к моделированию. Модель хороша настолько, насколько хороши данные, которые она принимает, поэтому очень важно, чтобы исследователи понимали природу своих данных. Это основа алгоритма, и он будет успешным или неуспешным, опираясь на достоинства своих данных.

---

[1] Адрес статьи в интернете: https://www.quantopian.com/posts/introduction-to-advances-in-financial-machine-learning-by-lopez-de-prado. — *Примеч. пер.*

[2] См. https://www.qplum.co/.

[3] См. http://www.quantresearch.info/.

В целом, неструктурированные и уникальные данные полезнее, чем предупакованные данные от поставщика, так как они не были взяты уже очищенными от альфа-коэффициентов другими менеджерами активов. Данные могут различаться по тому, что они описывают (фундаментальные[1], ценовые[2]) и по частоте (помесячные, поминутные, тиковые и т. д.). Ниже перечислены основные типы данных в порядке возрастания многообразия:

1. Фундаментальные данные — финансовые показатели компаний, обычно публикуемые ежеквартально.
2. Рыночные данные — вся торговая деятельность на торговой бирже или площадке.
3. Аналитика — производные данные, анализ различных факторов (включая все остальные типы данных), которые приобретаются у поставщика.
4. Альтернативные данные — первичная информация, не полученная из других источников. Спутниковые изображения, движения нефтяных танкеров, погода и т. д.

Структуры данных, используемые для хранения торговой информации, часто называются барами. Они могут сильно различаться в том, как они построены, хотя есть общие характеристики. Общими величинами являются цена открытия, максимальная цена, минимальная цена и цена закрытия, дата/время сделки и индексирующая переменная. Общепринятый индекс — это время; суточные бары — это структура, каждая из которых представляет один торговый день, минутные бары представляют одну минуту биржевой торговли и т. д. Время остается постоянным. Объем биржевых торгов — это еще один вариант, где каждый бар индексируется постоянным числом торгуемых акций (скажем, 200K-объемные бары). Третий вариант — торгуемая стоимость, где индексом являются доллары (торгуемые акции × цена за акцию).

- Временные бары:

  Бары, индексированные по временным интервалам, поминутно, ежедневно и т. д. Стандартом является OHLCV (Open, High, Low, Close, Volume), то есть цена открытия, максимальная цена, минимальная цена, цена закрытия, объем.

- Тиковые бары:

  Бары, индексированные по ордерам, при этом номер каждого множества ордеров (обычно только один) создает отдельный бар. Обычно используется цена и размер заказа, а также биржа, на которой ордер был исполнен.

- Объемные бары:

  Бары, индексированные по общему объему, при этом номер каждого множества торгуемых акций создает отдельный бар. Мы можем преобразовать минутные бары в аппроксимацию объемных баров, но в идеале для поддержания информации по всем параметрам на барах мы используем тиковые бары.

---

[1] См. https://www.quantopian.com/data/morningstar/fundamentals.

[2] См. https://www.quantopian.com/data/quantopian/us_equity_pricing.

- Долларовые бары:

  Аналогично объемным барам, за исключением измерения общей стоимости (в долларах), поменявшей собственника. Примером могут служить 100 000-долларовые бары, каждый из которых содержит максимально точное долларовое значение.

Альтернативные структуры данных демонстрируют статистические свойства в разной степени, причем объемные[1] и долларовые бары обычно выражают бóльшую стационарность результатов, чем временны́е и тиковые бары. Эти свойства играют большую роль при рассмотрении того, какие бары использовать в рамках машинного обучения, что мы и обсудим далее.

A.2. Статистические свойства и преобразования стационарности

Подавляющая часть литературы по машинному обучению и статистическому выводу в целом исходит из допущения[2] об одинаковой распределенности и взаимной независимости наблюдений. Независимость означает, что возникновение одного наблюдения не влияет ни на какие другие и равным образом означает, что наши переменные получены из одного и того же распределения вероятностей (например, имеют ту же дисперсию, среднее, асимметрию и т. д.).

К сожалению, эти свойства редко встречаются в финансовых данных временны́х рядов. Рассмотрим ценообразование: сегодняшняя цена сильно зависит от вчерашней, средняя цена в течение некоторого интервала времени постоянно меняется, и волатильность цен может быстро изменяться при публикации важной информации. С другой стороны, финансовые возвраты устраняют большинство из этих связей. Тем не менее дисперсия (то есть волатильность) финансовых возвратов по-прежнему изменяется с течением времени, поскольку рынок проходит через разные режимы волатильности, и, следовательно, не распределяется одинаково.

Разные типы баров (и дополнительные структуры данных) демонстрируют разные статистические свойства. Это важно учитывать при применении методов машинного обучения или других методов статистического вывода, так как в них принимается допущение, что входные данные отобраны как одинаково распределенные взаимно независимые случайные величины (или стационарны во временны́х рядах). Использование долларовых баров вместо временны́х баров

[1] См. https://www.jstor.org/stable/1913889?seq=1#page_scan_tab_contents.

[2] См. https://en.wikipedia.org/wiki/Independent_and_identically_distributed_random_variables.

может стать тем фактором, который определит разницу между слабым и переподогнанным алгоритмом и тем, который последовательно прибылен. Однако это лишь один шаг в поисках стационарности, и в нашем арсенале должны быть другие инструменты.

Приведенное выше примечание о независимости ценового ряда относительно ряда финансового возврата освещает важную идею: компромисс между памятью и стационарностью. Последняя является необходимым атрибутом для статистического вывода, но не имеет значения без первого. В крайнем случае, можно рассмотреть возможность преобразования любого ряда строго в единицы — вы успешно достигнете стационарности, но за счет всей информации, содержащейся в исходном ряде. Полезной интуитивной идеей является рассмотрение степеней дифференциации от исходного ряда, где большие степени увеличивают стационарность и понижают память. Финансовые возвраты дифференцированы одношагово, пример всех единиц полностью дифференцирован, ценовой ряд имеет нулевую дифференциацию. Лопез де Прадо предлагает альтернативный метод, именуемый *дробным дифференцированием*, который направлен на поиск оптимального баланса между нашими противоположными факторами; минимальное нецелое дифференцирование, необходимое для достижения стационарности. При этом сохраняется максимальный объем информации в наших данных. Для полного понимания прочитайте главу 5 книги. С реализованными таким образом и достаточно подготовленными нами данными мы *практически* готовы к применению алгоритмов машинного обучения. Нам осталось только промаркировать наши данные.

A.3. Маркировка для самообучения

A.3.1. Тройной барьерный метод

Большинство классификаторов МО требуют промаркированных данных (непромаркированные данные мощны, но сложны для технологической обработки и имеют высокий риск переподгонки). Мы намерены предсказать будущую финансовую результативность ценной бумаги, поэтому представляется справедливым маркировать каждое наблюдение, основываясь на его последующей ценовой результативности. Заманчиво просто задействовать тот факт, были ли финансовые возвраты положительными или отрицательными на протяжении фиксированного временно́го окна. Этот метод, однако, приводит к тому, что много меток ссылается на несущественные, малые ценовые изменения. Кроме того, реальная торговля часто осуществляется с помощью лимитных ордеров для взятия прибыли или остановки убытка. Маркос Лопез де Прадо предложил стратегию маркировки, которую он называет *тройным барьерным методом*, который сочетает наши желания выполнить маркировку с реальным поведением рынка. Когда совершается сделка, инвесторы могут решить упреждающе устанавливать ордера для исполнения по

определенным ценам. Если текущая цена ценной бумаги S составляет $5.00 и мы хотим контролировать наш риск, мы можем задать остановку убытка (стоп-лосс) на уровне $4.50. Если мы хотим взять (зафиксировать) прибыль до того, как она исчезнет, мы можем задать ордер на взятие прибыли на уровне $5.50. Эти ордера настроены на автоматическое закрытие позиции при достижении ценой любого из лимитов. Ордера взятия прибыли (профит-тейк) и остановки убытка (стоп-лосс) представляют собой два горизонтальных барьера тройного барьерного метода, в то время как третий, вертикальный, барьер просто основывается на времени: если сделка останавливается, вы можете закрыть ее в течение t дней, независимо от результативности.

Классификатор выводит значение −1 или 1 для каждой указанной даты покупки и заданной ценной бумаги, в зависимости от того, какой барьер был выбран первым. Если верхний барьер достигнут первым, значение устанавливается равным 1, так как была получена прибыль. Если вместо этого пробивается нижний барьер, то убытки были зафиксированы, и значение устанавливается равным −1. Если время покупки истекает до того, как какой-либо из лимитов будет нарушен, и вертикальный барьер достигнут, то значение устанавливается в диапазоне (−1, 1), который шкалируется тем, насколько близко конечная цена была к барьеру (в качестве альтернативы, если вы хотите обозначить строго достаточные ценовые изменения, здесь можно вывести 0).

A.3.2. Метамаркировка

После того как у вас есть модель, натренированная для установки стороны сделки (маркируемой тройным барьерным методом), вы можете обучить вторичную модель для того, чтобы установить размер сделки. Данная модель принимает первичную модель в качестве входных данных. Научиться одновременно узнавать направление и размер сделки намного сложнее, чем научиться это делать для каждого отдельно, плюс этот подход допускает модульность (та же модель проставления размеров может работать для длинных/коротких версий сделки). Мы должны снова промаркировать наши данные с помощью метода, который де Прадо называет *метамаркировкой*. Эта стратегия назначает метки сделкам 0 или 1 (1 — если сделка состоялась, 0 — если нет) с приписываемой им вероятностью. Эта вероятность используется для расчета размера сделки.

Полезными соображениями для проверок бинарных классификаций (таких, как метод тройного барьера и метамаркировка) являются чувствительность и специфичность[1]. Существуют компромисс между ошибками 1-го рода (ложными утверждениями, FP) и ошибками 2-го рода (ложными отрицаниями, FN), а также

[1] См. https://en.wikipedia.org/wiki/Sensitivity_and_specificity.

истинные утверждения (TP) и истинные отрицания (TN). Балльная оценка F1 служит мерой эффективности классификатора как гармоническое среднее между точностью (отношение между TP и TP+FP) и полнотой (отношение между TP и FN). Метамаркировка помогает максимизировать оценки F1. Сначала мы строим модель с высокой полнотой, независимо от точности (модель научилась узнавать направление, но со многими излишними сделками). Затем мы корректируем низкую точность путем применения метамаркировки к предсказаниям первичной модели. Тем самым ложные утверждения отфильтровываются, и масштаб наших истинных утверждений увеличивается на их вычисленную точность.

A.4. Обучающиеся алгоритмы для направления и размера ставки

Теперь, когда мы обсудили соображения, учитываемые при структурировании финансовых данных, мы, наконец, готовы обсудить, как программы на самом деле учатся торговать!

Архетип МО, известный как ансамблевое обучение, неоднократно подтверждает свою надежность и эффективность. В этих алгоритмах используется большее число слабых учеников (например, деревья решений), объединенных для создания более сильного сигнала. Примеры: случайные леса, другие бэггированные (бутстрап-агрегированные) классификаторы и бустированные (форсированные) классификаторы. Они производят признаковое пространство, которое может быть подрезано с целью сократить преобладание переподгонки. В этой статье предполагается, что вы на каком-то уровне знакомы с методами машинного обучения (в частности, с ансамблевыми учениками). Если это не так, то руководства библиотеки scikit-learn[1] являются сказочной отправной точкой.

A.4.1. Агрегирование бутстраповских выборок (бэггированные классификаторы)

Это популярный метод ансамблевого обучения, объединяющий большое число отдельных учеников, которые склонны к переподгонке, если они используются изолированно (деревья решений являются характерным примером), в более низкодисперсный «пакет» учеников. Грубый рецепт выглядит следующим образом:

1. Сгенерировать N тренировочных подмножеств данных путем случайного отбора с возвратом взятых образцов назад в исходную совокупность.

[1] См. http://scikit-learn.org/stable/tutorial/index.html.

2. Выполнить подгонку N оценщиков, по одному на каждое тренировочное подмножество, причем они подгоняются независимо друг от друга и тренируются параллельно.
3. Взять простое среднее предсказание каждой из N моделей, и вуаля! У вас есть ансамблевый прогноз. (При заданной классификационной задаче с дискретными вариантами исхода используется не простое среднее значение, а мажоритарное голосование, то есть голосование большинством голосов. Если задействована вероятность предсказания, то ансамблевый прогноз использует среднее значение вероятностей.)

Главное преимущество бэггирования (пакетирования) состоит в сокращении дисперсии, что помогает решать проблему переподгонки. Дисперсия является функцией от числа N упакованных классификаторов, средней дисперсии предсказания одного оценщика и средней корреляции между их предсказаниями.

Лопез де Прадо также представляет последовательное бутстрапирование, новый метод бэггирования (пакетирования), который производит образцы с более высокой степенью независимости друг от друга (в надежде приблизиться к совокупности одинаково распределенных взаимно независимых данных). Благодаря данному методу дисперсия упакованных классификаторов сокращается еще больше.

A.4.2. Случайные леса

Они предназначены для сокращения дисперсии/переподгонки деревьев решений. Случайные леса представляют собой реализацию бэггирования (пакетирования) («лес» является агрегацией большого числа деревьев) с дополнительным слоем случайности: во время оптимизации каждого узлового расщепления будет оцениваться только случайная подвыборка (взятая без возврата) атрибутов с целью дальнейшего декоррелирования оценок.

A.4.3. Важность признаков

Анализ важности признаков позволяет подрезать признаки в нашей зашумленной совокупности данных временных рядов, которые не влияют на результативность. После того как признаки обнаружены, мы можем на них поэкспериментировать. Всегда ли они важны или только в определенных условиях? Что вызывает изменение важности с течением времени? Можно ли предсказать переключатели режимов? Имеют ли эти важные признаки отношение и к другим финансовым инструментам? Относятся ли они к другим классам активов? Каковы наиболее важные признаки всех финансовых инструментов? Каково подмножество признаков с наивысшим рангом корреляции по всему инвестиционному универсуму? Подрезание нашего пространства признаков является важной частью оптимизации

наших моделей для повышения их результативности и снижения риска их переподгонки, как и любое другое соображение выше.

A.4.4. Перекрестная проверка

После того как наша модель уловила некие признаки, мы хотим оценить, как она работает. Перекрестная проверка[1] является стандартным методом для выполнения этого анализа, но требует некоторого здорового надувательства для применения в области финансов, такой, как эта тема статьи. Перекрестная проверка (cross-validation, CV) — это метод, который подразделяет наблюдения, полученные из одинаково распределенного взаимно независимого случайного процесса, на тренировочное и тестовое подмножества, последнее из которых никогда не используется для тренировки алгоритма (по довольно очевидным причинам), а только для его оценивания. Одним из наиболее популярных методов является k-блочный метод, при котором данные разбиваются на k одинаковых по размеру корзин (или блоков), одна из которых используется для проверки результатов тренировки на остальных $k - 1$ корзинах. Этот процесс повторяется k раз так, что каждая корзина используется в качестве тестируемой ровно один раз. Все сочетания повторяются в цикле.

Этот метод, однако, имеет проблемы в финансах, так как данные не являются одинаково распределенными и взаимно независимыми. Ошибки также могут быть результатом множественного тестирования и систематического смещения при отборе образцов из-за того, что многочисленные подмножества используются как для тренировки, так и для тестирования. Утечка информации происходит между корзинами потому, что наши наблюдения коррелированы (если X_{t+1} зависит от X_t, потому что они внутрирядово коррелированы, и они разбиты в корзины отдельно, то корзина, содержащая последнее из двух значений, будет содержать информацию из первого). Это усиливает мнимую результативность признака, который описывает X_t и X_{t+1}, независимо от того, ценен он или незначителен. Это приводит к ложным открытиям и завышенной оценке финансовых возвратов. Лопез де Прадо представляет решения этих проблем, которые также позволяют алгоритму больше обучаться на том же объеме данных. В частности, один из них, который он называет *прочищенной k-блочной перекрестной проверкой*, просто удаляет любые значения в тренировочном подмножестве, метки которых накладываются во времени с тестовым подмножеством. Еще одна стратегия удаления, которую он называет *эмбарго*, используется для устранения из тренировочных данных любых сигналов, которые были получены на тестовом подмножестве (по существу, просто удаляя некоторое число баров, непосредственно следующих после тестовых данных). Он также обеспечивает рамки для поиска оптимального значения k, исходя из малой утечки, или полного ее отсутствия, между тренировочными и тестовыми данными.

[1] См. http://scikit-learn.org/stable/modules/cross_validation.html.

Итак, наконец, у нас есть все, что нам нужно. Подрежьте свои данные, промаркируйте их направление, натренируйте обучающийся ансамблевый алгоритм, промаркируйте размер ставок, натренируйте на всем этом *еще один* алгоритм и объедините. Это может показаться чрезмерно сложным, и это так, но значительная часть программирования является модульным. С заложенной основой вы сможете найти более впечатляющие стратегии, чем когда-либо прежде.

A.5. Дальнейшие исследования

Материал данной статьи в значительной степени опирался на книгу «Машинное обучение: алгоритмы для бизнеса» Маркоса Лопеза де Прадо. Это отличный ресурс, если вы уже знакомы на высоком уровне с инвестиционным менеджментом, машинным обучением и наукой о данных. Если это не так, то серия лекций, к примеру, на Quantopian, Quandl, Quantra или Stackexchange[1] — это отличные места для старта, в особенности в сочетании с конкурсами Kaggle[2] и пособиями по машинному обучению библиотеки scikit-learn[3]. Не бойтесь просто поэкспериментировать, это может быть забавно!

Как только вы освоитесь со всем этим, просмотрите книгу Лопеза де Прадо и поработайте над реализацией этих методов со структурой данных, которую вы создали, и подключите результат к нескольким разным специально откалиброванным алгоритмам МО. Если у вас есть предсказательная модель, протестируйте ее вне выборки в течение продолжительного времени (хотя ваша методология должна была предотвратить переподгонку) и посмотрите, приживется ли она. Если это так, то примите мои поздравления! Удачи вам!

[1] См. https://quant.stackexchange.com/.
[2] См. https://www.kaggle.com/.
[3] См. http://scikit-learn.org/stable/tutorial/index.html.

Б Глоссарий

ARMA (autoregressive moving-average) — авторегрессионное скользящее среднее. Модель ARMA обобщает две более простые модели временных рядов — авторегрессионную модель (AR) и модель на основе скользящего среднего (moving average, MA).

Bloomberg L.P. — один из двух ведущих поставщиков финансовой информации для профессиональных участников финансовых рынков.

BWIC (bid wanted in competition, требуется заявка на конкурентной основе) — ситуация, когда институциональный инвестор представляет свой список заявителей на облигации различным дилерам ценных бумаг, дилеры делают свои заявки на указанные бумаги, и с теми, кто предлагает самые высокие заявки, заключаются контракты.

E-Mini S&P 500 — фьючерсный контракт на фондовом рынке, который торгуется на электронной торговой платформе Chicago Globex Чикагской товарной биржи (Chicago Mercantile Exchange); часто приводится в сокращенном написании «E-mini» и обозначается символом товарного тикера ES.

EURO STOXX 50 — фондовый индекс ценных бумаг в еврозоне, разработанный поставщиком индекса STOXX, принадлежащим Deutsche Börse Group.

PnL (net mark-to-market value of profits and losses) — чистая стоимость прибылей и убытков по текущим рыночным ценам.

***p*-значение (*p*-value)** — при заданной случайной модели, которая воплощает в себе нулевую гипотезу, *p*-значение является вероятностью получения столь же необычных или предельных результатов, что и наблюдаемые результаты.

***R*-квадрат (*R*-squared)** — доля дисперсии, объясненная моделью, со значениями в диапазоне от 0 до 1.

***t*-значение (*t*-value)** — стандартизованная версия проверочной статистики, используемой в качестве критерия при проверке статистической гипотезы.

***t*-распределение (*t*-distribution), или распределение Стьюдента**, — эталонное распределение (в частности, полученное из нулевой гипотезы), с которым может быть сопоставлено наблюдаемое *t*-значение.

Агрессор (aggressor) — трейдер, который забирает ликвидность с рынка. Вместо того чтобы делать ставку на акции, агрессор покупает на рынке по текущей цене предложения. Он также продает по текущим рыночным ценам и не указывает цену продажи. Покупая доступные акции или контракты по текущим рыночным ценам, агрессор размещает ордера, которые имеют немедленное исполнение.

Айсберговый ордер (iceberg order) — тип ордера, размещаемого на публичной бирже. Общая сумма ордера делится на видимую часть, которая сообщается другим участникам рынка, и скрытую часть, о которой ничего не сообщается.

Активы в управлении (assets under management, AUM) — мера общей рыночной стоимости всех финансовых активов (ценных бумаг), которыми финансовое учреждение, такое как паевой инвестиционный фонд, венчурная компания или брокерский дом, управляет от имени своих клиентов и себя.

Альтернативная гипотеза (alternative hypothesis) — обратная нулевой гипотезе (то, что вы надеетесь доказать).

Альфа-коэффициент (alpha), или альфа, — показатель, рассчитываемый для ценной бумаги или портфеля ценных бумаг, связывающий доходность ценной бумаги (портфеля) с доходностью близкого фондового индекса. Высокое значение альфа означает, что ценная бумага (портфель) работает лучше, чем ожидалось при его заданном бета (волатильности). Термин заимствован из статистики, где альфа (уровень значимости) — это вероятностный порог «необычности», который случайные результаты должны превзойти, чтобы фактические исходы считались статистически значимыми.

Асимметрия (skew), или скошенность, — состояние, когда один хвост распределения длиннее другого.

Биномиальное испытание (binomial trial) — испытание с двумя результатами.

Биномиальное распределение (binomial distribution) — распределение набора успехов в n испытаниях. Синоним: бернуллиево распределение.

Биржевой инвестиционный фонд (exchange-traded fund, ETF) — торгуемый на рынке финансовый актив, который отслеживает фондовый индекс, товар, облигации или корзину активов. Хотя во многих отношениях ETF похожи, они отличаются от паевых инвестиционных фондов, потому что акции торгуются как обычные акции на бирже. Цена акций ETF будет меняться в течение дня по мере их покупки и продажи. Крупнейшие ETF обычно имеют более высокий средний суточный объем и более низкие комиссии, чем акции паевых инвестиционных фондов, что делает их привлекательной альтернативой для индивидуальных инвесторов.

Бустирование (boosting) — общая методика подгонки последовательности моделей путем предоставления большего веса записям с большими остатками для каждого последующего цикла.

Бутстраповская выборка (bootstrap sample) — подвыборка, взятая с возвратом из совокупности наблюдаемых данных. Процесс бутстрапирования можно концеп-

туально представить как многократное взятие образцов с их возвратом в совокупность данных для того, чтобы получить синтетическую выборку.

Бэггирование (bagging), или бутстрап-агрегирование, пакетирование, — общая методика формирования набора моделей путем взятия бутстраповских подвыборок из данных, в данном случае из тренировочных данных. Синонимы: агрегирование бутстраповских выборок, бутстрап-агрегирование. Термин «бэггинг» — это своего рода технический каламбур, который, с одной стороны, является аббревиатурой для бутстрап-агрегирования и, с другой, означает упаковывание в пакеты (выборки), поскольку, в сущности, характер работы алгоритма бэггинга в этом и состоит — он отбирает бутстраповские пакеты данных и затем их агрегирует. Отсюда возникли термины «внутрипакетный» и «внепакетный». В бэггинге предикторы строятся путем взятия бутстраповских образцов из тренировочного набора, а затем они агрегируются для формирования бэггированного (упакованного) предиктора. Получившееся подмножество образцов называется внутрипакетным (in-bag), а те образцы, которые в пакет не попали, — внепакетными (out-of-bag).

Бэктестирование (backtesting), или ретроактивное тестирование, — тестирование на исторических данных с целью получения результатов и анализа риска и возвратности инвестиций, прежде чем рисковать любым фактическим капиталом. В юриспруденции «ретроактивный» относится к законам и постановлениям, имеющим обратную силу.

Внутри выборки и вне выборки (in-sample and out-of-sample) — статистические проверки результативности модельного предсказания обычно проводятся путем разбиения заданной совокупности данных на внутривыборочный (in-sample) период, используемый для первоначального оценивания параметров и отбора модели, и вневыборочный (out-of-sample) период, используемый для оценивания результативности предсказания. Эмпирические данные, основанные на результатах предсказания вне выборки, как правило, считаются более достоверными, чем данные, основанные на результатах в выборке. В частности, если имеются данные, скажем, за три года, необходимые для расчета волатильности, модель, используемая в течение этого периода, будет «в выборке». Но если использовать исторические данные для предсказания вперед, то оценивание будет выполняться за период времени, для которых нет данных (вне выборки). Таким образом, обычно «вне выборки» — это понятие для «предсказания там, где у нас нет данных». Технически, даже использование модели для оценивания сегодняшней волатильности на основе исторической выборки является прогнозом «вне выборки», потому что у нас нет мгновенной волатильности.

Внутрирядовая корреляция (serial correlation) — это взаимосвязь между наблюдениями одной и той же переменной в течение определенных периодов времени. Если величина внутрирядовой корреляции переменной равна нулю, то это означает, что корреляция отсутствует, и все наблюдения друг от друга не зависят.

Выборка (sample) — подмножество большей по размеру совокупности данных.

Гетероскедастичность (heteroskedasticity) — ситуация, когда некоторые диапазоны исхода показывают остатки с более высокой дисперсией (что может говорить о предикторе, который в уравнении отсутствует).

Гиперпараметры (hyperparameters) — параметры, которые необходимо установить перед подгонкой алгоритма.

Главная компонента (principal component) — линейная комбинация предикторных переменных.

Градиентное бустирование (gradient boosting) — более общая форма бустирования, которая создана с точки зрения минимизации функции стоимости.

Гэп (gap), или скачок — разрыв в цене, в частности между ценой старого контракта и ценой нового контракта.

Дефицит реализации (implementation shortfall) — разница между ценой решения и окончательной ценой исполнения (включая комиссии, налоги и т. д.) для торговой сделки. Он также известен под названием «соскальзывание». Агентская торговля в значительной степени связана с минимизацией дефицита реализации и поиском ликвидности.

Дисперсия (variance) — сумма квадратических отклонений от среднего, деленная на $n - 1$, где n — число значений данных. Синоним: среднеквадратическое отклонение.

Длинная позиция (long), или лонг — покупка ценной бумаги в ожидании, что она вырастет в цене. Разница между ценой продажи и ценой покупки приносит прибыль или убыток. Инвесторы входят в длинную позицию (покупают, лонгуют) в ожидании, что ценные бумаги возрастут в цене, а трейдеры входят в короткую позицию (продают, шортят) в ожидании, что ценные бумаги упадут в цене.

Долларовый финансовый возврат (dollar return) — возврат на портфельные инвестиции в течение любого периода оценивания, включающего изменение рыночной стоимости портфеля. Он также включает любые распределения, начисленные из портфеля в течение этого периода.

Инвестиционная стратегия (strategy) — систематический план размещения инвестируемых активов среди инвестиционных вариантов, таких как облигации, депозитные сертификаты, биржевые товары, недвижимость, акции. Эти планы учитывают такие факторы, как экономические тенденции, инфляция и процентные ставки. Другие факторы включают возраст инвестора, уровень толерантности к риску и краткосрочные или долгосрочные цели роста. Корпоративные инвестиционные стратегии определяют средства, необходимые для достижения конкурентного преимущества, и денежные результаты (прибыль), ожидаемые от таких решений.

Квант (quant) — биржевой специалист по квантитативным (количественным) методам.

Квантиль (quantile) — число x_p такое, что заданная случайная величина x превышает его лишь с фиксированной вероятностью p.

Квантоментальный (quantamental) — неологизм, который образован из двух терминов — «квантитативный подход» и «фундаментальный подход» и означает сочетание новейших квантитативных методов, в том числе на основе машинного обучения, с классическими методами на основе фундаментальных величин.

Ковариация (covariance), или совместная дисперсия — метрический показатель, который показывает степень, с которой переменная варьируется совместно с другой (то есть имеет аналогичную величину и направление).

Контанго (contango, CGO) — надбавка в цене, взимаемая продавцом за отсрочку расчета по сделке.

Короткая позиция (short), или шорт — продажа ценной бумаги в ожидании, что она упадет в цене. Короткая позиция сначала продается, а затем покупается по более низкой цене. Разница между ценой продажи и ценой покупки приносит прибыль или убыток. Инвесторы входят в длинную позицию (покупают, лонгуют) в ожидании, что ценные бумаги возрастут в цене, а трейдеры входят в короткую позицию (продают, шортят) в ожидании, что ценные бумаги упадут в цене.

Котировка (quote) — последняя цена, по которой торгуется биржевой актив — ценная бумага или товар, имея в виду самую недавнюю цену, с которой согласились покупатель и продавец и по которой была проведена сделка с определенной суммой актива.

Лимитный ордер — поручение клиента брокеру исполнить заявку в заданном диапазоне цен.

Логит (logit), или логит-преобразование — функция, которая увязывает вероятность принадлежности классу с диапазоном $\pm\infty$ (вместо диапазона от 0 до 1).

Лонговать (going long) — открывать позицию на покупку; трейдер «лонгует», когда цена растет, что дает ему возможность получить прибыль от покупки.

Лямбда (lambda) — интенсивность (в расчете на единицу времени или пространства), с которой события происходят.

Маржин колл (margin call) — сигнал уведомления инвестора о приближении к критическому уровню, который обязывает вносить средства для поддержания кредита и покрытия дальнейших возможных убытков. В отличие от него, стоп-аут (stop out) — это более низкий, чем «маржин колл», уровень, помогающий избежать потери финансовых средств, предоставленных через кредитное плечо. В этом случае, если количество убытков продолжает увеличиваться, то сделка, или даже несколько открытых позиций, закрывается принудительно, причем основные убытки в этом случае несет только инвестор.

Маркетмейкер (market maker) — брокер, дилер или инвестиционная компания, которая принимает на себя рыночный риск (системный риск), вступая во владение финансовым активом и торгуя им в качестве принципала. Маркетмейкеры обязаны постоянно выдавать котировки цен спроса и предложения, а также гарантировать полную продажу или поглощение финансового актива по определенной цене.

Мартингейл (martingale) — тип инвестиционной стратегии, используемой трейдерами для капитализации убытков. По мере того как цены на акции снижаются, инвестор покупает больше инвестиций, чтобы расширить свой портфель по более низкой цене. Инвестор покупает больше акций по более низкой цене с верой в то, что цена в конечном итоге увеличится и принесет чистую прибыль. Например, инвестор может приобрести акции Google по цене 500 долларов за акцию, а затем развернуться и купить больше акций, если цена упадет до 450 долларов за акцию. Когда цена восстанавливается, инвестор получает больший финансовый возврат на инвестиции.

Матрица ошибок (confusion matrix), или несоответствий — отображение в табличной форме (2×2 в бинарном случае) количеств записей по их предсказанному и фактическому состоянию, или результату, классификации.

Медиана (median) — это значение, которое расположено ровно посередине отсортированных данных, в результате чего половина данных находится выше или ниже него.

Мультиколлинеарность (multicollinearity) — когда предикторные переменные имеют идеальную, или почти идеальную, корреляцию, регрессия может быть нестабильной либо ее невозможно вычислить.

Нулевая гипотеза (null hypothesis) — гипотеза о том, что виной всему является случайность.

Ордер (order) — подтвержденная заявка одной стороны другой стороне на покупку, продажу, доставку или получение товаров или услуг в соответствии с указанными условиями. Когда ордер принимается принимающей стороной, он становится юридически обязательным договором.

Относительный финансовый возврат (relative return) — соотношение между финансовым возвратом, реализованным на фонде или активе, и финансовым возвратом, реализованным установленным эталоном. Относительные финансовые возвраты часто используются для оценивания менеджеров фондов, от которых ожидается, что они достигнут финансовых возвратов выше движения индексов неуправляемых активов.

Оценщик (estimator) — это правило (формула) для расчета оценки заданной величины на основе наблюдаемых данных: таким образом, различаются правило (оценщик), интересующая величина (эстиманд) и его результат (оценка).

Ошибка (error) — разница между точкой данных и предсказанным либо средним значением.

Паритет цен put и call (put-call parity) — связь между ценой колл и ценой пут для опциона с одинаковыми характеристиками (цена исполнения, дата истечения срока действия, базовый). Она используется в теории арбитражного ценообразования. Если разные портфели, состоящие из колл и пут, имеют одинаковое значение во время истечения срока действия, то из этого следует, что они будут иметь оди-

наковое значение вплоть до истечения срока действия. Таким образом, значения портфелей перемещаются строго в параллельно-шаговом режиме.

Перенос вперед (roll forward), или накат — продление срока действия фьючерсного контракта путем закрытия первоначального контракта и открытия нового контракта с более долгим сроком на тот же базовый актив по текущей рыночной цене. Позволяет трейдеру поддерживать позицию после первоначального истечения контракта, так как фьючерсные контракты имеют конечные даты истечения. Обычно проводится незадолго до истечения первоначального контракта и требует урегулирования прибыли или убытка по первоначальному контракту.

Перенос назад (roll backward), или откат — укорочение срока действия фьючерсного контракта путем выхода из одной позиции и входа в новую позицию с более близким сроком действия. Обычно проводится незадолго до истечения первоначального контракта и требует урегулирования прибыли или убытка по первоначальному контракту.

Повторный отбор (resampling) — процесс многократного взятия выборок из наблюдаемых данных.

Подстановочный оценщик энтропии (plug-in entropy estimator) — это формулировка энтропии распределения, где вероятности символов или блоков были заменены их относительными частотами в выборке.

Позиционер (position-taker), или покупатель позиций — физическое или юридическое лицо, которое должно принимать преобладающие позиции на рынке, не располагая долей рынка, чтобы влиять на рыночную позицию самостоятельно.

Позиция (position) — сумма ценной бумаги, товара или валюты, которыми владеет физическое лицо, дилер, учреждение или другая налогооблагаемая сущность. Они бывают двух типов: короткие позиции, которые заимствуются, а затем продаются, и длинные позиции, которыми владеют, а затем продаются. В зависимости от рыночных тенденций, движений и колебаний позиция может быть прибыльной или убыточной. Пересчет стоимости позиции для отражения ее фактической текущей стоимости на открытом рынке в отрасли называется «маркировкой по рынку».

Покупатель цен (price-taker) — физическое или юридическое лицо, которое должно принимать преобладающие цены на рынке, не располагая долей рынка, чтобы влиять на рыночную цену самостоятельно. В большинстве конкурентных рынков фирмы являются покупателями цен. Если фирмы устанавливают на свою продукцию более высокие цены, чем преобладающие рыночные цены, то потребители просто покупают ее у другого продавца по более низкой цене. На фондовом рынке индивидуальные инвесторы считаются покупателями цен, а маркетмейкерами — те, кто устанавливает цену на ценную бумагу и предлагает ее на рынке.

Полураспад (half-life) — мера измерения скорости распада конкретного вещества или время, затрачиваемое данным количеством вещества на распад до половины его массы. Применительно к данной теме полураспад показывает медленность процесса либо время достижения ожидаемого значения.

Правильность (accuracy) — частота (или доля) случаев, классифицированных правильно.

Примесность (impurity), или разнородность — степень смешанности классов в подразделе данных (чем больше смешанность, тем больше примесность).

Проверочный статистический показатель (test statistic) — метрический показатель целевой разницы или эффекта, используемый в качестве критерия при проверке статистической гипотезы.

Протокол FIX (financial information exchange, FIX) — протокол обмена сообщениями, разработанный для обмена актуальной информацией по сделкам. Находится в свободном доступе и использовании (http://www.fixprotocol.org).

Прямая оптимизация (walk-farward optimization) — торговая стратегия оптимизируется с помощью внутривыборочных данных внутри временного окна во временном ряде финансовых данных. Остальная часть данных резервируется для тестирования вне выборки. Небольшая порция зарезервированных данных, следующих после внутривыборочных, тестируется с использованием зафиксированных результатов. Затем внутривыборочное временное окно переносится вперед на период, охватываемый вневыборочным тестом, и процесс повторяется. В конце все зафиксированные результаты используются для выявления торговой стратегии. См. также **Внутри выборки и вне выборки**.

Размещение активов (asset allocation) — реализация инвестиционной стратегии, которая пытается сбалансировать риск и вознаграждение, корректируя процент каждого актива в инвестиционном портфеле в соответствии с толерантностью к риску со стороны инвестора, его целями и сроками инвестирования.

Распределение данных (data distribution) — частотное распределение индивидуальных значений в совокупности данных.

Распределение Пуассона (Poisson distribution) — частотное распределение числа событий в отобранных единицах времени или пространства.

Решеточный поиск (grid search), или поиск в решетке гиперпараметров — предусматривает, что для каждого гиперпараметра заранее подбирается список значений, которые могут оказаться для него хорошими, затем пишется вложенный цикл *for*, который пробует все комбинации этих значений с целью найти их контрольные точности и отслеживает те, которые показывают наилучшую результативность.

Рыночный ордер — поручение клиента брокеру немедленно купить или продать товар по текущей лучшей цене. Наиболее распространенный способ исполнения ордеров.

Систематическое смещение при отборе образцов (selection bias) — ошибка, выражающаяся в появлении у изучаемой выборки признаков, не свойственных генеральной совокупности; возникает в результате применения неподходящего метода отбора. Также называется систематической ошибкой отбора.

Случайный отбор (random sampling) — взятие элементов в выборку в произвольном порядке.

Соскальзывание (slippage) — разница в цене, по которой брокер получает указание от принципала выполнить ордер, и цене, по которой ордер фактически исполнен.

Спред (spread) — разность между лучшими ценами заявок на продажу и на покупку в один и тот же момент времени на какой-либо актив. Словом «спред» также называют разность цен двух различных сходных товаров, торгуемых на открытых рынках.

Спред между ценой спроса и ценой предложения (bid-ask spread) — сумма, на которую цена предложения (ask) превышает цену спроса (bid) на ценную бумагу на рынке. Спред между ценами спроса и предложения представляет собой, по сути, разницу между самой высокой ценой, которую покупатель готов заплатить за актив, и самой низкой ценой, которую продавец готов принять, чтобы продать его.

Срединная цена (mid-price) — цена между лучшей ценой продавцов актива, или ценой предложения (ask), и лучшей ценой покупателей актива, или ценой спроса (bid).

Среднеквадратическое отклонение (standard deviation) — квадратный корень из дисперсии.

Стандартное нормальное распределение (standard normal) — нормальное распределение со средним = 0 и стандартным отклонением = 1.

Стоп-аут (stop out), или принудительная остановка — более низкий, чем «маржин колл», уровень, помогающий избежать потери финансовых средств, предоставленных через кредитное плечо. Если количество убытков продолжает увеличиваться, то сделка или даже несколько открытых позиций закрываются принудительно, причем основные убытки в этом случае несет только инвестор.

Сторона ставки (the side of the bet) — показывает, в какую сторону пойдет движение цены — в длинную (вверх) или короткую (вниз). Этот термин имеет синоним «позиция», то есть соответственно длинная или короткая позиция.

Структурный сдвиг (structural break), или изменение, или разрыв — это неожиданный сдвиг во временном ряде, который может привести к огромным ошибкам предсказания и ненадежности модели в целом.

Тик (tick) — мера минимального восходящего или нисходящего движения цены ценной бумаги. Тик также может относиться к изменению цены ценной бумаги от сделки к сделке. С 2001 года, с появлением децимализации, минимальный размер тика для торговли акциями выше 1 доллара составляет 1 цент. Тик представляет собой стандарт, на котором стоимость ценной бумаги может колебаться. Тик обеспечивает определенное приращение цены, отраженное в местной валюте, связанной с рынком, на котором торгуется ценная бумага, на которое может измениться общая цена ценной бумаги.

Точность (precision) — частота (или доля) предсказанных единиц, которые фактически являются нулями.

Транзакционные издержки (transaction cost) — стоимость осуществления любой экономической сделки, сопровождающая участие в рынке. В инвестировании — затраты, понесенные при покупке или продаже активов, такие как комиссии и спред.

Финансовый возврат (return) — деньги, сделанные или потерянные на инвестиции. Возврат выражается номинально, как изменение денежной стоимости инвестиции с течением времени, либо в процентах из соотношения прибыли к инвестициям. Также называется возвратом на инвестицию или финансовой отдачей.

Фьючерсный контракт (futures contract) — юридическое соглашение о покупке или продаже определенного товара или актива по заранее определенной цене в определенное время в будущем. Фьючерсные контракты стандартизированы по качеству и количеству для облегчения торговли на фьючерсной бирже. Покупатель фьючерсного контракта берет на себя обязательство купить базовый актив по истечении срока действия фьючерсного контракта. Продавец фьючерсного контракта берет на себя обязательство предоставить базовый актив на дату истечения срока действия.

Хвост (tail) — длинная узкая часть частотного распределения, где относительно предельные значения встречаются с низкой частотой.

Цена предложения (ask) — цена, которую продавец готов принять за ценную бумагу или другой финансовый инструмент. Также упоминается как предложение (offer).

Цена спроса (bid), или заявка — самая высокая цена, которую любой покупатель готов заплатить за данную ценную бумагу в данный момент времени. Как правило, заявка ниже, чем цена предложения (ask), и разница между ними называется спредом между ценой спроса (bid) и ценой предложения (ask).

Ценные бумаги (securities) — финансовые или инвестиционные инструменты (некоторые оборотные, другие нет), купленные и проданные на финансовых рынках, такие как облигации, долговые обязательства, опционы, акции и купоны.

Частное распределение (marginal distribution), или маржинальное распределение — распределение вероятностей подмножества совокупности случайных величин. Например, при заданных двух случайных величинах X и Y, совместное распределение которых известно, частное распределение X — это простое распределение вероятности X, усредненное по информации о Y.

Шкалирование (scaling) — сплющивание или расширение данных, обычно для приведения многочисленных переменных к одинаковой шкале измерения.

Шортить (selling short) — открывать позицию на продажу; трейдер шортит, когда цена падает, что дает ему возможность получить прибыль от продажи.

B Справочные материалы и библиография

Глава 1

Справочные материалы

Bailey, D., P. Borwein, and S. Plouffe (1997): «On the rapid computation of various polylogarithmic constants». Mathematics of Computation, Vol. 66, No. 218, pp. 903–913.

Calkin, N. and M. López de Prado (2014a): «Stochastic flow diagrams». Algorithmic Finance, Vol. 3, No. 1, pp. 21–42.

Calkin, N. and M. López de Prado (2014b): «The topology of macro financial flows: An application of stochastic flow diagrams.» Algorithmic Finance, Vol. 3, No. 1, pp. 43–85.

Easley, D., M. Lopez de Prado, and M. O'Hara (2013): High-Frequency Trading, 1st ed. Risk Books.

López de Prado, M. (2014): «Quantitative meta-strategies.» Practical Applications, Institutional Investor Journals, Vol. 2, No. 3, pp. 1–3.

López de Prado, M. (2015): «The Future of Empirical Finance.» Journal of Portfolio Management, Vol. 41, No. 4, pp. 140–144.

Stigler, Stephen M. (1981): «Gauss and the invention of least squares.» Annals of Statistics, Vol. 9, No. 3, pp. 465–474.

Библиография

Abu-Mostafa, Y., M. Magdon-Ismail, and H. Lin (2012): Learning from Data, 1st ed. AMLBook.

Akansu, A., S. Kulkarni, and D. Malioutov (2016): Financial Signal Processing and Machine Learning, 1st ed. John Wiley & Sons-IEEE Press.

Aronson, D. and T. Masters (2013): Statistically Sound Machine Learning for Algorithmic Trading of Financial Instruments: Developing Predictive-Model-Based Trading Systems Using TSSB, 1st ed. CreateSpace Independent Publishing Platform.

Boyarshinov, V. (2012): Machine Learning in Computational Finance: Practical Algorithms for Building Artificial Intelligence Applications, 1st ed. LAP LAMBERT Academic Publishing.

Cerniglia, J., F. Fabozzi, and P. Kolm (2016): «Best practices in research for quantitative equity strategies.» Journal of Portfolio Management, Vol. 42, No. 5, pp. 135–143.

Chan, E. (2017): Machine Trading: Deploying Computer Algorithms to Conquer the Markets, 1st ed. John Wiley & Sons.

Gareth, J., D. Witten, T. Hastie, and R. Tibshirani (2013): An Introduction to Statistical Learning: with Applications in R, 1st ed. Springer.

Geron, A. (2017): Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems, 1st ed. O'Reilly Media.

Gyorfi, L., G. Ottucsak, and H. Walk (2012): Machine Learning for Financial Engineering, 1st ed. Imperial College Press.

Hackeling, G. (2014): Mastering Machine Learning with Scikit-Learn, 1st ed. Packt Publishing.

Hastie, T., R. Tibshirani, and J. Friedman (2016): The Elements of Statistical Learning, 2nd ed. Springer-Verlag.

Hauck, T. (2014): Scikit-Learn Cookbook, 1st ed. Packt Publishing.

McNelis, P. (2005): Neural Networks in Finance, 1st ed. Academic Press.

Raschka, S. (2015): Python Machine Learning, 1st ed. Packt Publishing.

Глава 2

Справочные материалы

Ane, T. and H. Geman (2000): «Order flow, transaction clock and normality of asset returns.» Journal of Finance, Vol. 55, pp. 2259–2284.

Bailey, David H., and M. Lopez de Prado (2012): «Balanced baskets: A new approach to trading and hedging risks.» Journal of Investment Strategies (Risk Journals), Vol. 1, No. 4 (Fall), pp. 21–62.

Clark, P. K. (1973): «A subordinated stochastic process model with finite variance for speculative prices.» Econometrica, Vol. 41, pp. 135–155.

Easley, D., M. Lopez de Prado, and M. O'Hara (2011): «The volume clock: Insights into the high frequency paradigm.» Journal of Portfolio Management, Vol. 37, No. 2, pp. 118–128.

Easley, D., M. Lopez de Prado, and M. O'Hara (2012): «Flow toxicity and liquidity in a high frequency world.» Review of Financial Studies, Vol. 25, No. 5, pp. 1457–1493.

Fama, E. and M. Blume (1966): «Filter rules and stock market trading.» Journal of Business, Vol. 40, pp. 226–241.

Kolanovic, M. and R. Krishnamachari (2017): «Big data and AI strategies: Machine learning and alternative data approach to investing.» White paper, JP Morgan, Quantitative and Derivatives Strategy. May 18.

Lam, K. and H. Yam (1997): «CUSUM techniques for technical trading in financial markets.» Financial Engineering and the Japanese Markets, Vol. 4, pp. 257–274.

Lopez de Prado, M. and D. Leinweber (2012): «Advances in cointegration and sub-set correlation hedging methods.» Journal of Investment Strategies (Risk Journals), Vol. 1, No. 2 (Spring), pp. 67–115.

Mandelbrot, B. and M. Taylor (1967): «On the distribution of stock price differences.» Operations Research, Vol. 15, No. 5, pp. 1057–1062.

Глава 3

Библиография

Ahmed, N., A. Atiya, N. Gayar, and H. El-Shishiny (2010): «An empirical comparison of machine learning models for time series forecasting.» Econometric Reviews, Vol. 29, No. 5–6, pp. 594–621.

Ballings, M., D. van den Poel, N. Hespeels, and R. Gryp (2015): «Evaluating multiple classifiers for stock price direction prediction.» Expert Systems with Applications, Vol. 42, No. 20, pp. 7046–7056.

Bontempi, G., S. Taieb, and Y. Le Borgne (2012): «Machine learning strategies for time series forecasting.» Lecture Notes in Business Information Processing, Vol. 138, No. 1, pp. 62–77.

Booth, A., E. Gerding and F. McGroarty (2014): «Automated trading with performance weighted random forests and seasonality.» Expert Systems with Applications, Vol. 41, No. 8, pp. 3651–3661.

Cao, L. and F. Tay (2001): «Financial forecasting using support vector machines.» Neural Computing & Applications, Vol. 10, No. 2, pp. 184–192.

Cao, L., F. Tay and F. Hock (2003): «Support vector machine with adaptive parameters in financial time series forecasting.» IEEE Transactions on Neural Networks, Vol. 14, No. 6, pp. 1506–1518.

Cervello-Royo, R., F. Guijarro, and K. Michniuk(2015): «Stock market trading rule based on pattern recognition and technical analysis: Forecasting the DJIA index with intraday data.» Expert Systems with Applications, Vol. 42, No. 14, pp. 5963–5975.

Chang, P., C. Fan and J. Lin (2011): «Trend discovery in financial time series data using a case-based fuzzy decision tree.» Expert Systems with Applications, Vol. 38, No. 5, pp. 6070–6080.

Kuan, C. and L. Tung (1995): «Forecasting exchange rates using feedforward and recurrent neural networks.» Journal of Applied Econometrics, Vol. 10, No. 4, pp. 347–364.

Creamer, G. and Y. Freund (2007): «A boosting approach for automated trading.» Journal of Trading, Vol. 2, No. 3, pp. 84–96.

Creamer, G. and Y. Freund (2010): «Automated trading with boosting and expert weighting.» Quantitative Finance, Vol. 10, No. 4, pp. 401–420.

Creamer, G., Y. Ren, Y. Sakamoto, and J. Nickerson (2016): «A textual analysis algorithm for the equity market: The European case.» Journal of Investing, Vol. 25, No. 3, pp. 105–116.

Dixon, M., D. Klabjan, and J. Bang (2016): «Classification-based financial markets prediction using deep neural networks.» Algorithmic Finance, forthcoming (2017). Available at SSRN: https://ssrn.com/abstract=2756331.

Dunis, C., and M. Williams (2002): «Modelling and trading the euro/US dollar ex-change rate: Do neural network models perform better?» Journal of Derivatives & Hedge Funds, Vol. 8, No. 3, pp. 211–239.

Feuerriegel, S. and H. Prendinger (2016): «News-based trading strategies.» Decision Support Systems, Vol. 90, pp. 65–74.

Hsu, S., J. Hsieh, T. Chih, and K. Hsu (2009): «A two-stage architecture for stock price forecasting by integrating self-organizing map and support vector regression.» Expert Systems with Applications, Vol. 36, No. 4, pp. 7947–7951.

Huang, W., Y. Nakamori, and S. Wang (2005): «Forecasting stock market movement direction with support vector machine.» Computers & Operations Research, Vol. 32, No. 10, pp. 2513–2522.

Kara, Y., M. Boyacioglu, and O. Baykan(2011): «Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul Stock Exchange.» Expert Systems with Applications, Vol. 38, No. 5, pp. 5311–5319.

Kim, K. (2003): «Financial time series forecasting using support vector machines.» Neurocomputing, Vol. 55, No. 1,pp. 307–319.

Krauss, C., X. Do, and N. Huck (2017): «Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500.» European Journal of Operational Research, Vol. 259, No. 2, pp. 689–702.

Laborda, R. and J. Laborda (2017): «Can tree-structured classifiers add value to the investor?» Finance Research Letters, Vol. 22 (August), pp. 211–226.

Nakamura, E. (2005): «Inflation forecasting using a neural network.» Economics Letters, Vol. 86, No. 3,pp. 373–378.

Olson, D. and C. Mossman (2003): «Neural network forecasts of Canadian stock returns using accounting ratios.» International Journal of Forecasting, Vol. 19, No. 3, pp. 453–465.

Patel, J., S. Sha, P. Thakkar, and K. Kotecha (2015): «Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques.» Expert Systems with Applications, Vol. 42, No. 1, pp. 259–268.

Patel, J., S. Sha, P. Thakkar, and K. Kotecha (2015): «Predicting stock market index using fusion of machine learning techniques.» Expert Systems with Applications, Vol. 42, No. 4, pp. 2162–2172.

Qin, Q., Q. Wang, J. Li, and S. Shuzhi (2013): «Linear and nonlinear trading models with gradient boosted random forests and application to Singapore Stock Market.» Journal of Intelligent Learning Systems and Applications, Vol. 5, No. 1, pp. 1–10.

Sorensen, E., K. Miller, and C. Ooi (2000): «The decision tree approach to stock selection.» Journal of Portfolio Management, Vol. 27, No. 1, pp. 42–52.

Theofilatos, K., S. Likothanassis, and A. Karathanasopoulos (2012): «Modeling and trading the EUR/USD exchange rate using machine learning techniques.» Engineering, Technology & Applied Science Research, Vol. 2, No. 5, pp. 269–272.

Trafalis, T. and H. Ince (2000): «Support vector machine for regression and applications to financial forecasting.» Neural Networks, Vol. 6, No. 1, pp. 348–353.

Trippi, R. and D. DeSieno (1992): «Trading equity index futures with a neural network.» Journal of Portfolio Management, Vol. 19, No. 1, pp. 27–33.

Tsai, C. and S. Wang (2009): «Stock price forecasting by hybrid machine learning techniques.» Proceedings of the International Multi-Conference of Engineers and Computer Scientists, Vol. 1, No. 1, pp. 755–760.

Tsai, C., Y. Lin, D. Yen, and Y. Chen (2011): «Predicting stock returns by classifier ensembles.» Applied Soft Computing, Vol. 11, No. 2, pp. 2452–2459.

Wang, J. and S. Chan (2006): «Stock market trading rule discovery using two-layer bias decision tree.» Expert Systems with Applications, Vol. 30, No. 4, pp. 605–611.

Wang, Q., J. Li, Q. Qin, and S. Ge (2011): «Linear, adaptive and nonlinear trading models for Singapore Stock Market with random forests.» Proceedings of the 9th IEEE International Conference on Control and Automation, pp. 726–731.

Wei, P. and N. Wang (2016): «Wikipedia and stock return: Wikipedia usage pattern helps to predict the individual stock movement.» Proceedings of the 25th International Conference Companion on World Wide Web, Vol. 1, pp. 591–594.

Zbikowski, K. (2015): «Using volume weighted support vector machines with walk forward testing and feature selection for the purpose of creating stock trading strategy.» Expert Systems with Applications, Vol. 42, No. 4, pp. 1797–1805.

Zhang, G., B. Patuwo, and M. Hu (1998): «Forecasting with artificial neural networks: The state of the art.» International Journal of Forecasting, Vol. 14, No. 1, pp. 35–62.

Zhu, M., D. Philpotts and M. Stevenson (2012): «The benefits of tree-based models for stock selection.» Journal of Asset Management, Vol. 13, No. 6, pp. 437–448.

Zhu, M., D. Philpotts, R. Sparks, and J. Stevenson, Maxwell (2011): «A hybrid approach to combining CART and logistic regression for stock ranking.» Journal of Portfolio Management, Vol. 38, No. 1,pp. 100–109.

Глава 4

Справочные материалы

Rao, C., P. Pathak and V. Koltchinskii (1997): «Bootstrap by sequential resampling.» Journal of Statistical Planning and Inference, Vol. 64, No. 2, pp. 257–281.

King, G. and L. Zeng (2001): «Logistic Regression in Rare Events Data.» Working paper, Harvard University. Available at https://gking.harvard.edu/files/0s.pdf.

Lo, A. (2017): Adaptive Markets, 1st ed. Princeton University Press.

Библиография

Взвешивание выборки является распространенной темой в литературе по машинному обучению. Однако практические задачи, обсуждаемые в этой главе, характерны для инвестиционных

приложений, по которым академическая литература крайне скудна. Ниже приведено несколько публикаций, которые вскользь затрагивают некоторые вопросы, обсуждаемые в этой главе.

Efron, B. (1979): «Bootstrap methods: Another look at the jackknife.» Annals of Statistics, Vol. 7, pp. 1–26.

Efron, B. (1983): «Estimating the error rote of a prediction rule: Improvement on cross-validation.» Journal of the American Statistical Association, Vol. 78, pp. 316–331.

Bickel, P. and D. Freedman (1981): «Some asymptotic theory for the bootstrap.» Annals of Statistics, Vol. 9, pp. 1196–1217.

Gine, E. and J. Zinn (1990): «Bootstrapping general empirical measures.» Annals of Probability, Vol. 18, pp. 851–869.

Hall, P. and E. Mammen (1994): «On general resampling algorithms and their performance in distribution estimation.» Annals of Statistics, Vol. 24, pp. 2011–2030.

Mitra, S. and P. Pathak (1984): «The nature of simple random sampling.» Annals of Statistics, Vol. 12, pp. 1536–1542.

Pathak, P. (1964): «Sufficiency in sampling theory.» Annals of Mathematical Statistics, Vol. 35, pp. 795–808.

Pathak, P. (1964): «On inverse sampling with unequal probabilities.» Biometrika, Vol. 51, pp. 185–193.

Praestgaard, J. and J. Wellner (1993): «Exchangeably weighted bootstraps of the general empirical process.» Annals of Probability, Vol. 21, pp. 2053–2086.

Rao, C., P. Pathak and V. Koltchinskii (1997): «Bootstrap by sequential resampling.» Journal of Statistical Planning and Inference, Vol. 64, No. 2, pp. 257–281.

Глава 5

Справочные материалы

Alexander, C. (2001): Market Models, 1st edition. John Wiley & Sons.

Hamilton, J. (1994): Time Series Analysis, 1st ed. Princeton University Press.

Hosking, J. (1981): «Fractional differencing.» Biometrika, Vol. 68, No. 1, pp. 165–176.

Jensen, A. and M. Nielsen (2014): «A fast fractional difference algorithm.» Journal of Time Series Analysis, Vol. 35, No. 5, pp. 428–436.

Lopez de Prado, M. (2015): «The Future of Empirical Finance.» Journal of Portfolio Management, Vol. 41, No. 4, pp. 140–144. Available at https://ssrn.com/abstract=2609734.

Библиография

Cavaliere, G., M. Nielsen, and A. Taylor (2017): «Quasi-maximum likelihood estimation and bootstrap inference in fractional time series models with heteroskedasticity of unknown form.» Journal of Econometrics, Vol. 198, No. 1, pp. 165–188.

Johansen, S. and M. Nielsen (2012): «A necessary moment condition for the fractional functional central limit theorem.» Econometric Theory, Vol. 28, No. 3, pp. 671–679.

Johansen, S. and M. Nielsen (2012): «Likelihood inference for a fractionally cointegrated vector autoregressive model.» Econometrica, Vol. 80, No. 6, pp. 2267–2732.

Johansen, S. and M. Nielsen (2016): «The role of initial values in conditional sum-of-squares estimation of nonstationary fractional time series models.» Econometric Theory, Vol. 32, No. 5, pp. 1095–1139.

Jones, M., M. Nielsen and M. Popiel (2015): «A fractionally cointegrated VAR analysis of economic voting and political support.» Canadian Journal of Economics, Vol. 47, No. 4, pp. 1078–1130.

Mackinnon, J. and M. Nielsen, M. (2014): «Numerical distribution functions of fractional unit root and cointegration tests.» Journal of Applied Econometrics, Vol. 29, No. 1, pp. 161–171.

Глава 6

Справочные материалы

Geron, A. (2017): Hands-on Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems, 1st edition. O'Reilly Media.

Kearns, M. and L. Valiant (1989): «Cryptographic limitations on learning Boolean formulae and finite automata.» In Proceedings of the 21st Annual ACM Symposium on Theory of Computing, pp. 433–444, New York. Association for Computing Machinery.

Schapire, R. (1990): «The strength of weak learnability.» Machine Learning. Kluwer Academic Publishers. Vol. 5 No. 2, pp. 197–227.

Библиография

Gareth, J., D. Witten, T. Hastie, and R. Tibshirani (2013): An Introduction to Statistical Learning: With Applications in R, 1st ed. Springer-Verlag.

Hackeling, G. (2014): Mastering Machine Learning with Scikit-Learn, 1st ed. Packt Publishing. Hastie, T., R. Tibshirani and J. Friedman (2016): The Elements of Statistical Learning, 2nd ed. Springer-Verlag.

Hauck, T. (2014): Scikit-Learn Cookbook, 1st ed. Packt Publishing.

Raschka, S. (2015): Python Machine Learning, 1st ed. Packt Publishing.

Глава 7

Библиография

Bharat Rao, R., G. Fung, and R. Rosales (2008): «On the dangers of cross-validation: An experimental evaluation.» White paper, IKM CKS Siemens Medical Solutions USA. Available at http://people.csail.mit.edu/romer/papers/CrossVal_SDM08.pdf.

Bishop, C. (1995): Neural Networks for Pattern Recognition, 1st ed. Oxford University Press.

Breiman, L. and P. Spector (1992): «Submodel selection and evaluation in regression: The X-random case.» White paper, Department of Statistics, University of California, Berkeley. Available at http://digitalassets.lib.berkeley.edu/sdtr/ucb/text/197.pdf.

Hastie, T., R. Tibshirani, and J. Friedman (2009): The Elements of Statistical Learning, 1st ed. Springer.

James, G., D. Witten, T. Hastie and R. Tibshirani (2013): An Introduction to Statistical Learning, 1st ed. Springer.

Kohavi, R. (1995): «A study of cross-validation and bootstrap for accuracy estimation and model selection.» International Joint Conference on Artificial Intelligence. Available at http://web.cs.iastate.edu/~jtian/cs573/Papers/Kohavi-IJCAI-95.pdf.

Ripley, B. (1996): Pattern Recognition and Neural Networks, 1st ed. Cambridge University Press.

Глава 8

Справочные материалы

American Statistical Association (2016): «Ethical guidelines for statistical practice.» Committee on Professional Ethics of the American Statistical Association (April). Available at http://www.amstat.org/asa/files/pdfs/EthicalGuidelines.pdf.

Belsley, D., E. Kuh, and R. Welsch (1980): Regression Diagnostics: Identifying Influential Data and Sources of Collinearity, 1st ed. John Wiley & Sons.

Goldberger, A. (1991): A Course in Econometrics. Harvard University Press, 1st edition.

Hill, R. and L. Adkins (2001): «Collinearity.» In Baltagi, Badi H. A Companion to Theoretical Econometrics, 1st ed. Blackwell, pp. 256–278.

Louppe, G., L. Wehenkel, A. Sutera, and P. Geurts (2013): «Understanding variable importances in forests of randomized trees.» Proceedings of the 26th International Conference on Neural Information Processing Systems, pp. 431–439.

Strobl, C., A. Boulesteix, A. Zeileis, and T. Hothorn (2007): «Bias in random forest variable importance measures: Illustrations, sources and a solution.» BMC Bioinformatics, Vol. 8, No. 25, pp. 1–11.

White, A. and W. Liu (1994): «Technical note: Bias in information-based measures in decision tree induction.» Machine Learning, Vol. 15, No. 3, pp. 321–329.

Глава 9

Справочные материалы

Bergstra, J., R. Bardenet, Y. Bengio, and B. Kegl (2011): «Algorithms for hyper-parameter optimization.» Advances in Neural Information Processing Systems, pp. 2546–2554.

Bergstra, J. and Y. Bengio (2012): «Random search for hyper-parameter optimization.» Journal of Machine Learning Research, Vol. 13, pp. 281–305.

Библиография

Chapelle, O., V. Vapnik, O. Bousquet, and S. Mukherjee (2002): «Choosing multiple parameters for support vector machines.» Machine Learning, Vol. 46, pp. 131–159.

Chuong, B., C. Foo, and A. Ng (2008): «Efficient multiple hyperparameter learning for log-linear models.» Advances in Neural Information Processing Systems, Vol. 20. Available at http://ai.stanford.edu/~chuongdo/papers/learn_reg.pdf.

Gorissen, D., K. Crombecq, I. Couckuyt, P. Demeester, and T. Dhaene (2010): «A surrogate modeling and adaptive sampling toolbox for computer based design.» Journal of Machine Learning Research, Vol. 11, pp. 2051–2055.

Hsu, C., C. Chang, and C. Lin (2010): «A practical guide to support vector classification.» Technical report, National Taiwan University.

Hutter, F., H. Hoos, and K. Leyton-Brown (2011): «Sequential model-based optimization for general algorithm configuration.» Proceedings of the 5th international conference on Learning and Intelligent Optimization, pp. 507–523.

Larsen, J., L. Hansen, C. Svarer, and M. Ohlsson (1996): «Design and regularization of neural networks: The optimal use of a validation set.» Proceedings of the 1996 IEEE Signal Processing Society Workshop.

Maclaurin, D., D. Duvenaud, and R. Adams (2015): «Gradient-based hyperparameter optimization through reversible learning.» Working paper. Available at https://arxiv.org/abs/1502.03492.

Martinez-Cantin, R. (2014): «BayesOpt: A Bayesian optimization library for nonlinear optimization, experimental design and bandits.» Journal of Machine Learning Research, Vol. 15, pp. 3915–3919.

Глава 10

Справочные материалы

Lopez de Prado, M. and M. Foreman (2014): «A mixture of Gaussians approach to mathematical portfolio oversight: The EF3M algorithm.» Quantitative Finance, Vol. 14, No. 5, pp. 913–930.

Wu, T., C. Lin and R. Weng (2004): «Probability estimates for multi-class classification by pairwise coupling.» Journal of Machine Learning Research, Vol. 5, pp. 975–1005.

Библиография

Allwein, E., R. Schapire, and Y. Singer (2001): «Reducing multiclass to binary: A unifying approach for margin classifiers.» Journal of Machine Learning Research, Vol. 1, pp. 113–141.

Hastie, T. and R. Tibshirani (1998): «Classification by pairwise coupling.» The Annals of Statistics, Vol. 26, No. 1,pp. 451–471.

Refregier, P. and F. Vallet (1991): «Probabilistic approach for multiclass classification with neural networks.» Proceedings of International Conference on Artificial Networks, pp. 1003–1007.

Глава 11

Справочные материалы

Arlot, S. and A. Celisse (2010): «A survey of cross-validation procedures for model selection.» Statistics Surveys, Vol. 4, pp. 40–79.

Bailey, D., J. Borwein, M. Lopez de Prado, and J. Zhu (2014): «Pseudo-mathematics and financial charlatanism: The effects of backtest overfitting on out-of-sample performance.» Notices of the American Mathematical Society, Vol. 61, No. 5 (May), pp. 458–471. Available at https://ssrn.com/abstract=2308659.

Bailey, D., J. Borwein, M. Lopez de Prado, and J. Zhu (2017a): «The probability of backtest overfitting.» Journal of Computational Finance, Vol. 20, No. 4, pp. 39–70. Available at http://ssrn.com/abstract=2326253.

Bailey, D. and M. Lopez de Prado (2012): «The Sharpe ratio efficient frontier.» Journal of Risk, Vol. 15, No. 2 (Winter). Available at https://ssrn.com/abstractM821643.

Bailey, D. and M. Lopez de Prado (2014b): «The deflated Sharpe ratio: Correcting for selection bias, backtest overfitting and non-normality.» Journal of Portfolio Management, Vol. 40, No. 5, pp. 94–107. Available at https://ssrn.com/abstract=2460551.

Harvey, C., Y. Liu, and H. Zhu (2016): "...and the cross-section of expected returns.» Review of Financial Studies, Vol. 29, No. 1, pp. 5–68.

López de Prado, M. (2017): «Finance as an industrial science.» Journal of Portfolio Management, Vol. 43, No. 4, pp. 5–9. Available at http://www.iijournals.com/doi/pdfplus/10.3905/jpm.2017.43.4.005.

Luo, Y., M. Alvarez, S. Wang, J. Jussa, A. Wang, and G. Rohal (2014): «Seven sins of quantitative investing.» White paper, Deutsche Bank Markets Research, September 8.

Sarfati, O. (2015): «Backtesting: A practitioner's guide to assessing strategies and avoiding pitfalls.» Citi Equity Derivatives. CBOE 2015 Risk Management Conference. Available at https://www.cboe.com/rmc/2015/olivier-pdf-Backtesting-FuH.pdf.

Библиография

Bailey, D., J. Borwein, and M. Lopez de Prado (2016): «Stock portfolio design and backtest overfitting.» Journal of Investment Management, Vol. 15, No. 1, pp. 1–13. Available at https://ssrn.com/abstract=2739335.

Bailey, D., J. Borwein, M. Lopez de Prado, A. Salehipour, and J. Zhu (2016): «Backtest overfitting in financial markets.» Automated Trader, Vol. 39. Available at https://ssrn.com/abstract=2731886.

Bailey, D., J. Borwein, M. Lopez de Prado, and J. Zhu (2017b): «Mathematical appendices to: 'The probability of backtest overfitting.'» Journal of Computational Finance (Risk Journals), Vol. 20, No. 4. Available at https://ssrn.com/abstract=2568435.

Bailey, D., J. Borwein, A. Salehipour, and M. Lopez de Prado (2017): «Evaluation and ranking of market forecasters.» Journal of Investment Management, forth-coming. Available at https://ssrn.com/abstract=2944853.

Bailey, D., J. Borwein, A. Salehipour, M. Lopez de Prado, and J. Zhu (2015): «Online tools for demonstration of backtest overfitting.» Working paper. Available at https://ssrn.com/abstract=2597421.

Bailey, D., S. Ger, M. Lopez de Prado, A. Sim and, K. Wu (2016): «Statistical overfitting and backtest performance.» In Risk-Based and Factor Investing, Quantitative Finance Elsevier. Available at ttps://ssrn.com/abstract=2507040.

Bailey, D. and M. Lopez de Prado (2014a): «Stop-outs under serial correlation and 'the triple penance rule.'» Journal of Risk, Vol. 18, No. 2, pp. 61–93. Available at https://ssrn.com/ abstract=2201302.

Bailey, D. and M. Lopez de Prado (2015): «Mathematical appendices to: 'Stop-outs under serial correlation.'» Journal of Risk, Vol. 18, No. 2. Available at https://ssrn.com/abstract=2511599.

Bailey, D., M. Lopez de Prado, and E. del Pozo (2013): «The strategy approval decision: A Sharpe ratio indifference curve approach.» Algorithmic Finance, Vol. 2, No. 1, pp. 99–109. Available at https://ssrn.com/abstract=2003638.

Carr, P. and M. Lopez de Prado (2014): «Determining optimal trading rules without backtesting.» Working paper. Available at https://ssrn.com/abstract=2658641.

Lopez de Prado, M. (2012a): «Portfolio oversight: An evolutionary approach.» Lec-ture at Cornell University. Available at https://ssrn.com/abstract=2172468.

Lopez de Prado, M. (2012b): «The sharp razor: Performance evaluation with non-normal returns.» Lecture at Cornell University. Available at https://ssrn.com/abstract=2150879.

Lopez de Prado, M. (2013): «What to look for in a backtest.» Lecture at Cornell University. Available at https://ssrn.com/abstract=2308682.

Lopez de Prado, M. (2014a): «Optimal trading rules without backtesting.» Lecture at Cornell University. Available at https://ssrn.com/abstract=2502613.

Lopez de Prado, M. (2014b): «Deflating the Sharpe ratio.» Lecture at Cornell Uni-versity. Available at https://ssrn.com/abstract=2465675.

Lopez de Prado, M. (2015a): «Quantitative meta-strategies.» Practical Applications, Institutional Investor Journals, Vol. 2, No. 3, pp. 1–3. Available at https://ssrn.com/abstract=2547325.

Lopez de Prado, M. (2015b): «The Future of empirical finance.» Journal of Portfolio Management, Vol. 41, No. 4, pp. 140–144. Available at https://ssrn.com/abstract=2609734.

Lopez de Prado, M. (2015c): «Backtesting.» Lecture at Cornell University. Available at https://ssrn.com/abstract=2606462.

Lopez de Prado, M. (2015d): «Recent trends in empirical finance.» Journal of Portfolio Management, Vol. 41, No. 4, pp. 29–33. Available at https://ssrn.com/abstract=2638760.

Lopez de Prado, M. (2015e): «Why most empirical discoveries in finance are Likely wrong, and what can be done about it.» Lecture at University of Pennsylvania. Available at https://ssrn.com/abstract=2599105.

Lopez de Prado, M. (2015f): «Advances in quantitative meta-strategies.» Lecture at Cornell University. Available at https://ssrn.com/abstract=2604812.

Lopez de Prado, M. (2016): «Building diversified portfolios that outperform out-of-sample.» Journal of Portfolio Management, Vol. 42, No. 4, pp. 59–69. Available at https://ssrn.com/ abstract=2708678.

Lopez de Prado, M. and M. Foreman (2014): «A mixture of Gaussians approach to mathematical portfolio oversight: TheEF3M algorithm.» Quantitative Finance, Vol. 14, No. 5, pp. 913–930. Available at https://ssrn.com/abstract=1931734.

Lopez de Prado, M. and A. Peijan (2004): «Measuring loss potential of hedge fund strategies.» Journal of Alternative Investments, Vol. 7, No. 1, pp. 7–31, Summer 2004. Available at https://ssrn.com/abstract=641702.

Lopez de Prado, M., R. Vince, and J. Zhu (2015): «Risk adjusted growth portfolio in a finite investment horizon.» Lecture at Cornell University. Available at https://ssrn.com/abstract=2624329.

Глава 12

Справочные материалы

Bailey, D. and M. Lopez de Prado (2012): «The Sharpe ratio efficient frontier.» Journal of Risk, Vol. 15, No. 2 (Winter). Available at https://ssrn.com/abstract=1821643.

Bailey, D. and M. Lopez de Prado (2014): «The deflated Sharpe ratio: Correcting for selection bias, backtest overfitting and non-normality.» Journal of Portfolio Management, Vol. 40, No. 5, pp. 94–107. Available at https://ssrn.com/abstract=2460551.

Bailey, D., J. Borwein, M. Lopez de Prado, and J. Zhu (2014): «Pseudo-mathematics and financial charlatanism: The effects of backtest overfitting on out-of-sample performance.» Notices of the American Mathematical Society, Vol. 61, No. 5, pp. 458–471. Available at http://ssrn.com/abstract=2308659.

Bailey, D., J. Borwein, M. Lopez de Prado, and J. Zhu (2017): «The probability of backtest overfitting.» Journal of Computational Finance, Vol. 20, No. 4, pp. 39–70. Available at https://ssrn.com/abstract=2326253.

Глава 13

Справочные материалы

Bailey, D. and M. Lopez de Prado (2012): «The Sharpe ratio efficient frontier.» Journal of Risk, Vol. 15, No. 2, pp. 3–44. Available at http://ssrn.com/abstract=1821643.

Bailey, D. and M. Lopez de Prado (2013): «Drawdown-based stop-outs and the triple penance rule.» Journal of Risk, Vol. 18, No. 2, pp. 61–93. Available at http://ssrn.com/abstract=2201302.

Bailey, D., J. Borwein, M. Lopez de Prado, and J. Zhu (2014): «Pseudo-mathematics and financial charlatanism: The effects of backtest overfitting on out-of-sample performance.» Notices of the American Mathematical Society, 61(5), pp. 458–471. Available at http://ssrn.com/abstract=2308659.

Bailey, D., J. Borwein, M. Lopez de Prado, and J. Zhu (2017): «The probability of backtest overfitting.» Journal of Computational Finance, Vol. 20, No. 4, pp. 39–70. Available at http://ssrn.com/abstract=2326253.

Bertram, W. (2009): «Analytic solutions for optimal statistical arbitrage trading.» Working paper. Available at http://ssrn.com/abstract=1505073.

Easley, D., M. Lopez de Prado, and M. O'Hara (2011): «The exchange of flow-toxicity.» Journal of Trading, Vol. 6, No. 2, pp. 8–13. Available at http://ssrn.com/abstract=1748633.

Глава 14

Справочные материалы

Bailey, D. and M. Lopez de Prado (2012): «The Sharpe ratio efficient frontier.» Journal of Risk, Vol. 15, No. 2, pp. 3–44.

Bailey, D. and M. Lopez de Prado (2014): «The deflated Sharpe ratio: Correcting for selection bias, backtest overfitting and non-normality.» Journal of Portfolio Management, Vol. 40, No. 5. Available at https://ssrn.com/abstract=2460551.

Barra (1998): Risk Model Handbook: U.S. Equities, 1st ed. Barra. Available at http://www.alacra.com/alacra/help/barra_handbook_US.pdf.

Barra (2013): MSCI BARRA Factor Indexes Methodology, 1st ed. MSCI Barra. Available at https://www.msci.com/eqb/methodology/meth_docs/MSCI_Barra_Factor%20Indices_ Methodology_Nov13.pdf.

CFA Institute (2010): «Global investment performance standards.» CFA Institute, Vol. 2010, No. 4, February. Available at https://www.gipsstandards.org/.

Zhang, Y. and S. Rachev (2004): «Risk attribution and portfolio performance measurement — An overview.» Working paper, University of California, Santa Barbara. Available at http://citeseerx.ist.psu.edu/viewdoc/summary?doi= 10.1.1.318.7169.

Библиография

American Statistical Society (1999): «Ethical guidelines for statistical practice.» Available at http://www.amstat.org/committees/ethics/index.html.

Bailey, D., J. Borwein, M. Lopez de Prado, and J. Zhu (2014): «Pseudo-mathematics and financial charlatanism: The effects of backtest overfitting on out-of-sample performance.» Notices of the American Mathematical Society, Vol. 61, No. 5. Available at http://ssrn.com/abstract= 2308659.

Bailey, D., J. Borwein, M. Lopez de Prado, and J. Zhu (2017): «The probability of backtest overfitting.» Journal of Computational Finance, Vol. 20, No. 4, pp. 39–70. Available at http://ssrn.com/abstract=2326253.

Bailey, D. andM. Lopez de Prado (2012): «Balanced baskets: Anew approach to trading and hedging risks.» Journal of Investment Strategies (Risk Journals), Vol. 1, No. 4, pp. 21–62.

Beddall, M. and K. Land (2013): «The hypothetical performance of CTAs.» Working paper, Winton Capital Management.

Benjamini, Y. and Y. Hochberg (1995): «Controlling the false discovery rate: A practical and powerful approach to multiple testing.» Journal of the Royal Statistical Society, Series B (Methodological), Vol. 57, No. 1, pp. 289–300.

Bennet, C., A. Baird, M. Miller, and G. Wolford (2010): «Neural correlates of interspecies perspective taking in the post-mortem Atlantic salmon: An argument for proper multiple comparisons correction.» Journal of Serendipitous and Unexpected Results, Vol. 1, No. 1, pp. 1–5.

Bruss, F. (1984): «A unified approach to a class of best choice problems with an unknown number of options.» Annals of Probability, Vol. 12, No. 3, pp. 882–891.

Dmitrienko, A., A.C. Tamhane, and F. Bretz (2010): Multiple Testing Problems in Pharmaceutical Statistics, 1st ed. CRC Press.

Dudoit, S. and M.J. van der Laan (2008): Multiple Testing Procedures with Applications to Genomics, 1st ed. Springer.

Fisher, R.A. (1915): «Frequency distribution of the values of the correlation coefficient in samples of an indefinitely large population.» Biometrika (Biometrika Trust), Vol. 10, No. 4, pp. 507–521.

Hand, D. J. (2014): The Improbability Principle, 1st ed. Scientific American/Farrar, Straus and Giroux.

Harvey, C., Y. Liu, and H. Zhu (2013): "... And the cross-section of expected returns.» Working paper, Duke University. Available at http://ssrn.com/abstract=2249314.

Harvey, C. and Y. Liu (2014): «Backtesting.» Working paper, Duke University. Available at http://ssrn.com/abstract=2345489.

Hochberg Y. and A. Tamhane (1987): Multiple Comparison Procedures, 1st ed. John Wiley and Sons.

Holm, S. (1979): «A simple sequentially rejective multiple test procedure.» Scandinavian Journal of Statistics, Vol. 6, pp. 65–70.

Ioannidis, J.P.A. (2005): «Why most published research findings are false.» PloS Medicine, Vol. 2, No. 8, pp. 696–701.

Ingersoll, J., M. Spiegel, W. Goetzmann, and I. Welch (2007): «Portfolio performance manipulation and manipulation-proof performance measures.» Review of Financial Studies, Vol. 20, No. 5, pp. 1504–1546.

Lo, A. (2002): «The statistics of Sharpe ratios.» Financial Analysts Journal, Vol. 58, No. 4 (July/August), pp. 36–52.

Lopez de Prado M., and A. Peijan (2004): «Measuring loss potential of hedge fund strategies.» Journal of Alternative Investments, Vol. 7, No. 1 (Summer), pp. 7–31. Available at http://ssrn .com/abstract=641702.

Mertens, E. (2002): «Variance of the IID estimator in Lo (2002).» Working paper, University of Basel.

Roulston, M. and D. Hand (2013): «Blinded by optimism.» Working paper, Winton Capital Management.

Schorfheide, F. and K. Wolpin (2012): «On the use of holdout samples for model selection.» American Economic Review, Vol. 102, No. 3, pp. 477–481.

Sharpe, W. (1966): «Mutual fund performance.» Journal of Business, Vol. 39, No. 1, pp. 119–138.

Sharpe, W. (1975): «Adjusting for risk in portfolio performance measurement.» Journal of Portfolio Management, Vol. 1, No. 2 (Winter), pp. 29–34.

Sharpe, W. (1994): «The Sharpe ratio.» Journal of Portfolio Management, Vol. 21, No. 1 (Fall), pp. 49–58.

Studeny M. andVejnarova J. (1999): «The multiinformation function as a tool for measuring stochastic dependence,» in M. I. Jordan, ed., Learning in Graphical Models. MIT Press, pp. 261–296.

Wasserstein R., and Lazar N. (2016) «The ASA's statement on p-values: Context, process, and purpose.» American Statistician, Vol. 70, No. 2, pp. 129–133. DOI: 10.1080/00031305.2016.1154108.

Watanabe S. (1960): «Information theoretical analysis of multivariate correlation.» IBM Journal of Research and Development, Vol. 4, pp. 66–82.

Глава 15

Справочные материалы

Bailey, D. and M. Lopez de Prado (2014): «The deflated Sharpe ratio: Correcting for selection bias, backtest overfitting and non-normality.» Journal of Portfolio Management, Vol. 40, No. 5. Available at https://ssrn.com/abstract=2460551.

Bailey, D. and M. Lopez de Prado (2012): «The Sharpe ratio efficient frontier.» Journal of Risk, Vol.

15, No. 2, pp. 3–44. Available at https://ssrn.com/abstract=1821643.

Lopez de Prado, M. and M. Foreman (2014): «A mixture of Gaussians approach to mathematical portfolio oversight: TheEF3M algorithm.» Quantitative Finance, Vol. 14, No. 5, pp. 913–930. Available at https://ssrn.com/abstract=1931734.

Lopez de Prado, M. and A. Peijan (2004): «Measuring loss potential of hedge fund strategies.» Journal of Alternative Investments, Vol. 7, No. 1 (Summer), pp. 7–31. Available at http://ssrn.com/abstract=641702.

Глава 16

Справочные материалы

Bailey, D. and M. Lopez de Prado (2012): «Balanced baskets: A new approach to trading and hedging risks.» Journal of Investment Strategies, Vol. 1, No. 4, pp. 21–62. Available at http://ssrn.com/abstract=2066170.

Bailey, D. and M. Lopez de Prado (2013): «An open-source implementation of the critical-line algorithm for portfolio optimization.» Algorithms, Vol. 6, No. 1, pp. 169–196. Available at http://ssrn.com/abstract=2197616.

Bailey, D., J. Borwein, M. Lopez de Prado, and J. Zhu (2014) «Pseudo-mathematics and financial charlatanism: The effects of backtest overfitting on out-of-sample performance.» Notices of the American Mathematical Society, Vol. 61, No. 5, pp. 458–471. Available at http://ssrn.com/abstract=2308659.

Bailey, D. and M. Lopez de Prado (2014): «The deflated Sharpe ratio: Correcting for selection bias, backtest overfitting and non-normality.» Journal of Portfolio Man-agement, Vol. 40, No. 5, pp. 94–107.

Black, F. and R. Litterman (1992): «Global portfolio optimization.» Financial Analysts Journal, Vol. 48, pp. 28–43.

Brualdi, R. (2010): «The mutually beneficial relationship of graphs and matrices.» Conference Board of the Mathematical Sciences, Regional Conference Series in Mathematics, Nr. 115.

Calkin, N. and M. Lopez de Prado (2014): «Stochastic flow diagrams.» Algorithmic Finance, Vol. 3, No. 1, pp. 21–42. Availble at http://ssrn.com/abstract=2379314.

Calkin, N. and M. Lopez de Prado (2014): «The topology of macro financial flows: An application of stochastic flow diagrams.» Algorithmic Finance, Vol. 3, No. 1, pp. 43–85. Available at http://ssrn.com/abstract=2379319.

Clarke, R., H. De Silva, and S. Thorley (2002): «Portfolio constraints and the funda-mental law of active management.» Financial Analysts Journal, Vol. 58, pp. 48–66.

De Miguel, V., L. Garlappi, and R. Uppal (2009): «Optimal versus naive diversification: How inefficient is the 1/N portfolio strategy?» Review of Financial Studies, Vol. 22, pp. 1915–1953.

Jurczenko, E. (2015): Risk-Based and Factor Investing, 1st ed. Elsevier Science.

Kolanovic, M., A. Lau, T. Lee, andR. Krishnamachari (2017): «Cross asset portfolios of tradable risk premia indices. Hierarchical risk parity: Enhancing returns at target volatility.» White paper, Global Quantitative & Derivatives Strategy. J.P. Morgan, April 26.

Kolm, P., R. Tutuncu and F. Fabozzi (2014): «60 years of portfolio optimization.» European Journal of Operational Research, Vol. 234, No. 2, pp. 356–371.

Kuhn, H. W. and A. W. Tucker (1951): «Nonlinear programming.» Proceedings of 2nd Berkeley Symposium. Berkeley, University of California Press, pp. 481–492.

Markowitz, H. (1952): «Portfolio selection.» Journal of Finance, Vol. 7, pp. 77–91.

Merton, R. (1976): «Option pricing when underlying stock returns are discontinuous.» Journal of Financial Economics, Vol. 3, pp. 125–144.

Michaud, R. (1998): Efficient Asset Allocation: A Practical Guide to Stock Portfolio Optimization and Asset Allocation, 1st ed. Harvard Business School Press.

Ledoit, O. and M. Wolf (2003): «Improved estimation of the covariance matrix of stock returns with an application to portfolio selection.» Journal of Empirical Finance, Vol. 10, No. 5, pp. 603–621.

Raffinot, T. (2017): «Hierarchical clustering based asset allocation.» Journal of Portfolio Management, forthcoming.

Rokach, L. and O. Maimon (2005): «Clustering methods,» in Rokach, L. and O. Maimon, eds., Data Mining and Knowledge Discovery Handbook. Springer, pp. 321–352.

Глава 17

Справочные материалы

Andrews, D. (1993): «Tests for parameter instability and structural change with unknown change point.» Econometrics, Vol. 61, No. 4 (July), pp. 821–856.

Breitung, J. and R. Kruse (2013): «When Bubbles Burst: Econometric Tests Based on Structural Breaks.» Statistical Papers, Vol. 54, pp. 911–930.

Breitung, J. (2014): «Econometric tests for speculative bubbles.» Bonn Journal of Economics, Vol. 3, No. 1,pp. 113–127.

Brown, R.L., J. Durbin, and J.M. Evans (1975): «Techniques for Testing the Constancy of Regression Relationships over Time.» Journal of the Royal Statistical Society, Series B, Vol. 35, pp. 149192.

Chow, G. (1960). «Tests of equality between sets of coefficients in two linear regressions.» Econometrica, Vol. 28, No. 3, pp. 591–605.

Greene, W. (2008): Econometric Analysis, 6th ed. Pearson Prentice Hall.

Homm, U. and J. Breitung (2012): «Testing for speculative bubbles in stock markets: A comparison of alternative methods.» Journal of Financial Econometrics, Vol. 10, No. 1, 198–231.

Maddala, G. and I. Kim (1998): Unit Roots, Cointegration and Structural Change, 1st ed. Cambridge University Press.

Phillips, P., Y. Wu, and J. Yu (2011): «Explosive behavior in the 1990s Nasdaq: When did exuberance escalate asset values?» International Economic Review, Vol. 52, pp. 201–226.

Phillips, P. and J. Yu (2011): «Dating the timeline of financial bubbles during the subprime crisis.» Quantitative Economics, Vol. 2, pp. 455–491.

Phillips, P., S. Shi, and J. Yu (2013): «Testing for multiple bubbles 1: Historical episodes of exuberance and collapse in the S&P 500.» Working paper 8–2013, Singapore Management University.

Глава 18

Справочные материалы

Bailey, D. and M. Lopez de Prado (2012): «Balanced baskets: A new approach to trading and hedging risks.» Journal of Investment Strategies, Vol. 1, No. 4, pp. 21–62. Available at https://ssrn.com/abstract=2066170.

Easley D., M. Kiefer, M. O'Hara, and J. Paperman (1996): «Liquidity,information, and infrequently traded stocks.» Journal of Finance, Vol. 51, No. 4, pp. 1405–1436.

Easley D., M. Kiefer and, M. O'Hara (1997): «The information content of the trading process.» Journal of Empirical Finance, Vol. 4, No. 2, pp. 159–185.

Easley, D., M. Lopez de Prado, and M. O'Hara (2012a): «Flow toxicity and liquidity in a high frequency world.» Review of Financial Studies, Vol. 25, No. 5, pp. 1547–1493.

Easley, D., M. Lopez de Prado, and M. O'Hara (2012b): «The volume clock: Insights into the high frequency paradigm.» Journal of Portfolio Management, Vol. 39, No. 1, pp. 19–29.

Gao, Y., I. Kontoyiannis and E. Bienestock (2008): «Estimating the entropy of binary time series: Methodology, some theory and a simulation study.» Working paper, arXiv. Available at https://arxiv.org/abs/0802.4363v1.

Fiedor, Pawel (2014a): «Mutual information rate-based networks in financial markets.» Working paper, arXiv. Available at https://arxiv.org/abs/1401.2548.

Fiedor, Pawel (2014b): «Information-theoretic approach to lead-lag effect on financial markets.» Working paper, arXiv. Available at https://arxiv.org/abs/1402.3820.

Fiedor, Pawel (2014c): «Causal non-linear financial networks.» Working paper, arXiv. Available at https://arxiv.org/abs/1407.5020.

Hausser, J. and K. Strimmer (2009): «Entropy inference and the James-Stein estimator, with application to nonlinear gene association networks,» Journal of Machine Learning Research, Vol. 10, pp. 1469–1484. http://www.jmlr.org/papers/volume10/hausser09a/hausser09a.pdf.

Kolmogorov, A. (1965): «Three approaches to the quantitative definition of information.» Problems in Information Transmission, Vol. 1, No. 1, pp. 1–7.

Kontoyiannis, I. (1997): «The complexity and entropy of literary styles», NSF Technical Report # 97.

Kontoyiannis (1998): «Asymptotically optimal lossy Lempel-Ziv coding,» ISIT, Cambridge, MA, August 16–August 21.

MacKay, D. (2003): Information Theory, Inference, and Learning Algorithms, 1st ed. Cambridge University Press.

Meucci, A. (2009): «Managing diversification.» Risk Magazine, Vol. 22, pp. 74–79.

Norwich, K. (2003): Information, Sensation and Perception, 1st ed. Academic Press.

Ornstein, D.S. and B. Weiss (1993): «Entropy and data compression schemes.» IEEE Transactions on Information Theory, Vol. 39, pp. 78–83.

Shannon, C. (1948): «A mathematical theory of communication.» Bell System Technical Journal, Vol. 27, No. 3, pp. 379–423.

Ziv, J. and A. Lempel (1978): «Compression of individual sequences via variable-rate coding.» IEEE Transactions on Information Theory, Vol. 24, No. 5, pp. 530–536.

Библиография

Easley, D., R. Engle, M. O'Hara, and L. Wu (2008): «Time-varying arrival rates of informed and uninformed traders.» Journal of Financial Econometrics, Vol. 6, No. 2, pp. 171–207.

Easley, D., M. Lopez de Prado, and M. O'Hara (2011): «The microstructure of the flash crash.» Journal of Portfolio Management, Vol. 37, No. 2, pp. 118–128.

Easley, D., M. Lopez de Prado, and M. O'Hara (2012c): «Optimal execution horizon.» Mathematical Finance, Vol. 25, No. 3, pp. 640–672.

Gnedenko, B. and I. Yelnik (2016): «Minimum entropy as a measure of effective dimensionality.» Working paper. Available at https://ssrn.com/abstract=2767549.

Глава 19

Справочные материалы

Abad, D. and J. Yague (2012): «From PIN to VPIN.» The Spanish Review of Financial Economics, Vol. 10, No. 2, pp.74–83.

Aitken, M. and A. Frino (1996): «The accuracy of the tick test: Evidence from the Australian Stock Exchange.» Journal of Banking and Finance, Vol. 20, pp. 1715–1729.

Amihud, Y. and H. Mendelson (1987): «Trading mechanisms and stock returns: An empirical investigation.» Journal of Finance, Vol. 42, pp. 533–553.

Amihud, Y. (2002): «Illiquidity and stock returns: Cross-section and time-series effects.» Journal of Financial Markets, Vol. 5, pp. 31–56.

Andersen, T. and O. Bondarenko (2013): «VPIN and the Flash Crash.» Journal of Financial Markets, Vol. 17, pp.1–46.

Beckers, S. (1983): «Variances of security price returns based on high, low, and closing prices.» Journal of Business, Vol. 56, pp. 97–112.

Bethel, E. W., Leinweber. D., Rubel, O., and K. Wu (2012): «Federal market information technology in the post-flash crash era: Roles for supercomputing.» Journal of Trading, Vol. 7, No. 2, pp. 9–25.

Carlin, B., M. Sousa Lobo, and S. Viswanathan (2005): «Episodic liquidity crises. Cooperative and predatory trading.» Journal ofFinance, Vol. 42, No. 5 (October), pp. 2235–2274.

Cheung, W., R. Chou, A. Lei (2015): «Exchange-traded barrier option and VPIN.» Journal of Futures Markets, Vol. 35, No. 6, pp. 561–581.

Corwin, S. and P. Schultz (2012): «A simple way to estimate bid-ask spreads from daily high and low prices.» Journal of Finance, Vol. 67, No. 2, pp. 719–760.

Cremers, M. and D. Weinbaum (2010): «Deviations from put-call parity and stock return predictability.» Journal of Financial and Quantitative Analysis, Vol. 45, No. 2 (April), pp. 335–367.

Donefer, B. (2010): «Algos gone wild. Risk in the world of automated trading strategies.» Journal of Trading, Vol. 5, pp. 31–34.

Easley, D., N. Kiefer, M. O'Hara, and J. Paperman (1996): «Liquidity, information, and infrequently traded stocks.» Journal of Finance, Vol. 51, No. 4, pp. 1405–1436.

Easley, D., R. Engle, M. O'Hara, and L. Wu (2008): «Time-varying arrival rates of informed and uninformed traders.» Journal of Financial Econometrics, Vol. 6, No. 2, pp. 171–207.

Easley, D., M. Lopez de Prado, and M. O'Hara (2011): «The microstructure of the flash crash.» Journal of Portfolio Management, Vol. 37, No. 2 (Winter), pp. 118–128.

Easley, D., M. Lopez de Prado, and M. O'Hara (2012a): «Flow toxicity and liquidity in a high frequency world.» Review of Financial Studies, Vol. 25, No. 5, pp. 1457–1493.

Easley, D., M. Lopez de Prado, and M. O'Hara (2012b): «The volume clock: Insights into the high frequency paradigm.» Journal of Portfolio Management, Vol. 39, No. 1, pp. 19–29.

Easley, D., M. Lopez de Prado, and M. O'Hara (2013): High-Frequency Trading: New Realities for Traders, Markets and Regulators, 1st ed. Risk Books.

Easley, D., M. Lopez de Prado, and M. O'Hara (2016): «Discerning information from trade data.» Journal of Financial Economics, Vol. 120, No. 2, pp. 269–286.

Eisler, Z., J. Bouchaud, and J. Kockelkoren (2012): «The impact of order book events: Market orders, limit orders and cancellations.» Quantitative Finance, Vol. 12, No. 9, pp. 1395–1419.

Fabozzi, F., S. Focardi, and C. Jonas (2011): «High-frequency trading. Methodologies and market impact.» Review of Futures Markets, Vol. 19, pp. 7–38.

Hasbrouck, J. (2007): Empirical Market Microstructure, 1st ed. Oxford University Press.

Hasbrouck, J. (2009): «Trading costs and returns for US equities: Estimating effective costs from daily data.» Journal ofFinance, Vol. 64, No. 3, pp. 1445–1477.

Jarrow, R. and P. Protter (2011): «A dysfunctional role of high frequency trading in electronic markets.» International Journal of Theoretical and Applied Finance, Vol. 15, No. 3.

Kim, C., T. Perry, and M. Dhatt (2014): «Informed trading and price discovery around the clock.» Journal of Alternative Investments, Vol 17, No. 2, pp. 68–81.

Kyle, A. (1985): «Continuous auctions and insider trading.» Econometrica, Vol. 53, pp. 1315–1336.

Lee, C. and M. Ready (1991): «Inferring trade direction from intraday data.» Journal of Finance, Vol. 46, pp. 733–746.

Lopez de Prado, M. (2017): «Mathematics and economics: A reality check.» Journal of Portfolio Management, Vol. 43, No. 1, pp. 5–8.

Muravyev, D., N. Pearson, and J. Broussard (2013): «Is there price discovery in equity options?» Journal of Financial Economics, Vol. 107, No. 2, pp. 259–283.

NANEX (2011): «Strange days: June 8, 2011—NatGas Algo.» NANEX blog. Available at www.nanex.net/StrangeDays/06082011.html.

O'Hara, M. (1995): Market Microstructure, 1st ed. Blackwell, Oxford.

O'Hara, M. (2011): «What is a quote?» Journal of Trading, Vol. 5, No. 2 (Spring), pp. 10–15.

Parkinson, M. (1980): «The extreme value method for estimating the variance of the rate of return.» Journal of Business, Vol. 53, pp. 61–65.

Patzelt, F. and J. Bouchaud (2017): «Universal scaling and nonlinearity of aggregate price impact in financial markets.» Working paper. Available at https://arxiv.org/abs/1706.04163.

Roll, R. (1984): «A simple implicit measure of the effective bid-ask spread in an efficient market.» Journal of Finance, Vol. 39, pp. 1127–1139.

Stigler, Stephen M. (1981): «Gauss and the invention of least squares.» Annals of Statistics, Vol. 9, No. 3, pp. 465–474.

Song, J, K. Wu and H. Simon (2014): «Parameter analysis of the VPIN (volume synchronized probability of informed trading) metric.» In Zopounidis, C., ed., Quantitative Financial Risk Management: Theory and Practice, 1st ed. Wiley.

Toth, B., I. Palit, F. Lillo, and J. Farmer (2011): «Why is order flow so persistent?» Working paper. Available at https://arxiv.org/abs/1108.1632.

Van Ness, B., R. Van Ness, and S. Yildiz (2017): «The role of HFTs in order flow toxicity and stock price variance, and predicting changes in HFTs' liquidity provisions.» Journal of Economics and Finance, Vol. 41, No. 4, pp. 739–762.

Wei, W., D. Gerace, and A. Frino (2013): «Informed trading, flow toxicity and the impact on intraday trading factors.» Australasian Accounting Business and Finance Journal, Vol. 7, No. 2, pp. 3–24.

Глава 20

Справочные материалы

Ascher, D., A. Ravenscroft, and A. Martelli (2005): Python Cookbook, 2nd ed. O'Reilly Media.

Библиография

Gorelick, M. and I. Ozsvald (2008): High Performance Python, 1st ed. O'Reilly Media.

Lopez de Prado, M. (2017): «Supercomputing for finance: A gentle introduction.» Lecture materials, Cornell University. Available at https://ssrn.com/abstract=2907803.

McKinney, W. (2012): Python for Data Analysis, 1st ed. O'Reilly Media.

Palach, J. (2008): Parallel Programming with Python, 1st ed. Packt Publishing.

Summerfield, M. (2013): Python in Practice: Create Better Programs Using Concurrency, Libraries, and Patterns, 1st ed. Addison-Wesley.

Zaccone, G. (2015): Python Parallel Programming Cookbook, 1st ed. Packt Publishing.

Глава 21

Справочная литература

Garleanu, N. and L. Pedersen (2012): «Dynamic trading with predictable returns and transaction costs.» Journal of Finance, Vol. 68, No. 6, pp. 2309–2340.

Johansson, F. (2012): «Efficient implementation of the Hardy-Ramanujan-Rademacher formula,» LMS Journal of Computation and Mathematics, Vol. 15, pp. 341–359.

Rosenberg, G., P. Haghnegahdar, P. Goddard, P. Carr, K. Wu, and M. Lopez de Prado (2016): «Solving the optimal trading trajectory problem using a quantum annealer.» IEEE Journal of Selected Topics in Signal Processing, Vol. 10, No. 6 (September), pp. 1053–1060.

Williams, C. (2010): Explorations in Quantum Computing, 2nd ed. Springer.

Woeginger, G. (2003): «Exact algorithms for NP-hard problems: A survey.» In Junger, M., G. Reinelt, and G. Rinaldi: Combinatorial Optimization—Eureka, You Shrink! Lecture notes in computer science, Vol. 2570, Springer, pp. 185–207.

Глава 22

Справочные материалы

Aad, G., et al. (2016): «Measurements of the Higgs boson production and decay rates and coupling strengths using pp collision data at yfs = 7 and 8 TeV in the ATLAS experiment.» The European Physical Journal C, Vol. 76, No. 1, p. 6.

Abbott, B.P. et al. (2016): «Observation of gravitational waves from a binary black hole merger.» Physical Review Letters, Vol. 116, No. 6, p. 061102.

Armbrust, M., et al. (2010): «A view of cloud computing.» Communications of the ACM, Vol. 53, No. 4, pp. 50–58.

Asanovic, K. et al. (2006): «The landscape of parallel computing research: A view from Berkeley.» Technical Report UCB/EECS-2006–183, EECS Department, University of California, Berkeley.

Ayachit, U. et al. «Performance analysis, design considerations, and applications of extreme-scale in situ infrastructures.» Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. IEEE Press.

Bethel, E. W. et al. (2011): «Federal market information technology in the post Flash Crash era: Roles for supercomputing.» Proceedings of WHPCF'2011. ACM. pp. 23–30.

Bloom, J. S. et al. (2012): «Automating discovery and classification of transients and variable stars in the synoptic survey era.» Publications of the Astronomical Society of the Pacific, Vol. 124, No. 921, p. 1175.

Camerer, C.F. and G. Loewenstein (2011): «Behavioral economics: Past, present, future.» In Advances in Behavioral Economics, pp. 1–52.

Chen, L. et al. (2015): «Profiling and understanding virtualization overhead in cloud.» Parallel Processing (ICPP), 2015 44th International Conference. IEEE.

Choi, J.Y. et al. (2013): ICEE: «Wide-area in transit data processing framework for near real-time scientific applications.» 4th SC Workshop on Petascale (Big) Data Analytics: Challenges and Opportunities in Conjunction with SC13.

Dong, Y. et al. (2012): «High performance network virtualization with SR-IOV.» Journal of Parallel and Distributed Computing, Vol. 72, No. 11, pp. 1471–1480.

Easley, D., M. Lopez de Prado, and M. O'Hara (2011): «The microstructure of the 'Flash Crash': Flow toxicity, liquidity crashes and the probability of informed trading.» Journal of Portfolio Management, Vol. 37, No. 2, pp. 118–128.

Folk, M. et al. (2011): «An overview of the HDF5 technology suite and its applications.» Proceedings of the EDBT/ICDT 2011 Workshop on Array Databases. ACM.

Fox, G. et al. (2015): «Big Data, simulations and HPC convergence, iBig Data benchmarking»: 6th International Workshop, WBDB 2015, Toronto, ON, Canada, June 16–17, 2015; and 7th International Workshop, WBDB 2015, New Delhi, India, December 14–15, 2015, Revised Selected Papers, T. Rabl, et al., eds. 2016, Springer International Publishing: Cham. pp. 3–17. DOI: 10.1007/978–3–319–49748–8_1.

Ghemawat, S., H. Gobioff, and S.-T. Leung (2003): «The Google file system,» SOSP '03: Proceedings of the nineteenth ACM symposium on operating systems principles. ACM. pp. 29–43.

Gordon, A. et al. (2012): «ELI: Bare-metal performance for I/O virtualization.» SIGARCH Comput. Archit. News, Vol. 40, No. 1, pp. 411–422.

Gropp, W., E. Lusk, and A. Skjellum (1999): Using MPI: Portable Parallel Programming with the Message-Passing Interface. MIT Press.

Hey, T., S. Tansley, and K.M. Tolle (2009): The Fourth Paradigm: Data-Intensive Scientific Discovery. Vol. 1. Microsoft research Redmond, WA.

Hirschman, A. O. (1980): National Power and the Structure of Foreign Trade. Vol. 105. University of California Press.

Holzman, B. et al. (2017): «HEPCloud, a new paradigm for HEP facilities: CMS Amazon Web Services investigation. Computing and Software for Big Science, Vol. 1, No. 1, p. 1.

Jackson, K. R., et al. (2010): «Performance analysis of high performance computing applications on the Amazon Web Services Cloud. Cloud Computing Technology and Science (CloudCom). 2010 Second International Conference. IEEE.

Kim, T. et al. (2015): «Extracting baseline electricity usage using gradient tree boosting.» IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity). IEEE.

Kumar, V. et al. (1994): Introduction to Parallel Computing: Design and Analysis of Algorithms. Benjamin/Cummings Publishing Company.

Liu, Q. et al., (2014): «Hello ADIOS: The challenges and lessons of developing leadership class I/O frameworks.» Concurrency and Computation: Practice and Experience, Volume 26, No. 7, pp. 1453–1473.

National Academies of Sciences, Engineering and Medicine (2016): Future Directions for NSF Advanced Computing Infrastructure to Support U.S. Science and Engineering in 2017–2020. National Academies Press.

Nicholas, M. L. et al. (2009): «The Palomar transient factory: System overview, performance, and first results.» Publications of the Astronomical Society of the Pacific, Vol. 121, No. 886, p. 1395.

Qiu, J. et al. (2016): «A survey of machine learning for big data processing.» EURASIP Journal on Advances in Signal Processing, Vol. 2016, No. 1, p. 67. DOI: 10.1186/s13634–016–0355–x.

Rudin, C. and K. L. Wagstaff (2014) «Machine learning for science and society.» Machine Learning, Vol. 95, No. 1, pp. 1–9.

Shoshani, A. and D. Rotem (2010): «Scientific data management: Challenges, technology, and deployment.» Chapman & Hall/CRC Computational Science Series. CRC Press.

Snir, M. et al. (1998): MPI: The Complete Reference. Volume 1, The MPI-1 Core. MIT Press.

Song, J. H. et al. (2014): «Exploring irregular time series through non-uniform fast Fourier transform.» Proceedings of the 7th Workshop on High Performance Computational Finance, IEEE Press.

Todd, A. et al. (2014): «Insights from Smart Meters: The potential for peak hour savings from behavior-based programs.» Lawrence Berkeley National Laboratory. Available at https://www4.eere.energy.gov/seeaction/system/files/documents/smart_meters.pdf.

Wu, K. et al. (2013): «A big data approach to analyzing market volatility.» Algorithmic Finance. Vol. 2, No. 3, pp. 241–267.

Wu, L. et al. (2016): «Towards real-time detection and tracking of spatio-temporal features: Blob-filaments in fusion plasma. IEEE Transactions on Big Data, Vol. 2, No. 3, pp. 262–275.

Yan, J. et al. (2009): «How much can behavioral targeting help online advertising?» Proceedings of the 18th international conference on world wide web. ACM. pp. 261–270.

Yelick, K., et al. (2011): «The Magellan report on cloud computing for science.» U.S. Department of Energy, Office of Science.

Zeff, R.L. and B. Aronson (1999): Advertising on the Internet. John Wiley & Sons.

Маркос Лопез де Прадо

Машинное обучение: алгоритмы для бизнеса

Перевели с английского А. Логунов, М. Панин

Заведующая редакцией	*Ю. Сергиенко*
Ведущий редактор	*К. Тульцева*
Научный редактор	*А. Логунов*
Литературный редактор	*А. Руденко*
Художественный редактор	*В. Мостипан*
Корректоры	*С. Беляева, М. Молчанова*
Верстка	*Л. Егорова*

Изготовлено в России. Изготовитель: ООО «Прогресс книга».
Место нахождения и фактический адрес: 194044, Россия, г. Санкт-Петербург,
Б. Сампсониевский пр., д. 29А, пом. 52. Тел.: +78127037373.

Дата изготовления: 03.2019. Наименование: книжная продукция. Срок годности: не ограничен.

Налоговая льгота — общероссийский классификатор продукции ОК 034-2014, 58.11.12 —
Книги печатные профессиональные, технические и научные.

Импортер в Беларусь: ООО «ПИТЕР М», 220020, РБ, г. Минск, ул. Тимирязева, д. 121/3, к. 214, тел./факс: 208 80 01.

Подписано в печать 20.03.19. Формат 70x100/16. Бумага офсетная. Усл. п. л. 34,830. Тираж 1700. Заказ № ВЗК-02018-19.

Отпечатано в АО «Первая Образцовая типография», филиал «Дом печати — ВЯТКА»
610033, г. Киров, ул. Московская, 122.

$\{\hat{\omega}_t\}$

[МАРКОС ЛОПЕЗ ДЕ ПРАДО]

Маркос Лопез де Прадо делится тем, что обычно скрывают, — самыми прибыльными алгоритмами машинного обучения, которые он использовал на протяжении двух десятилетий, чтобы управлять большими пулами средств самых требовательных инвесторов.

Машинное обучение меняет практически каждый аспект нашей жизни, алгоритмы МО выполняют задачи, которые до недавнего времени доверяли только проверенным экспертам. В ближайшем будущем машинное обучение будет доминировать в финансах, гадание на кофейной гуще уйдет в прошлое, а инвестиции перестанут быть синонимом азартных игр.

Воспользуйтесь шансом поучаствовать в «машинной революции», для этого достаточно познакомиться с первой книгой, в которой приведен полный и систематический анализ методов машинного обучения применительно к финансам, начиная со структур финансовых данных, маркировки финансового ряда, взвешиванию выборки, дифференцированию временного ряда и заканчивая целой частью, посвященной правильному бэктестированию инвестиционных стратегий.

$u_{t,j}^{(2)}$

$$L[f_t, \omega, m] = f_t - m\sqrt{\frac{\omega}{1-m^2}}$$

ОБ АВТОРЕ:

Маркос Лопез де Прадо управляет многомиллиардными фондами, используя алгоритмы МО и суперкомпьютеры. Он основал компанию Guggenheim Partners' Quantitative Investment Strategies (QIS), где разработал высокоэффективные стратегии, позволяющие гарантировать максимальные возвраты на вложенный капитал с поправкой на риск, затем выкупил QIS и успешно развернул этот бизнес в 2018 году.

Маркос Лопез де Прадо входит в топ-10 самых читаемых авторов в области финансов благодаря десяткам научных статей, посвященных машинному обучению.

$$\left\{\left\{\frac{S_i}{K}p_i\right\}_{i=1,\ldots N}, \underbrace{\{-1,1\}x\ldots x\{-1,1\}}_{N}, \{p_i\}_{i=1,\ldots N} \in p^{K\cdot N}\right\}$$

$\hat{f}[x]$

ISBN: 978-5-4461-1154-1