МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

МЕХАНИКО-МАТЕМАТИЧЕСКИЙ ФАКУЛЬТЕТ

Кафедра дифференциальных уравнений и системного анализа

ВОРОБЕЙ

Владислав Александрович

ПРОГНОЗИРОВАНИЕ КРЕДИТНЫХ РИСКОВ ПРИ ПОМОЩИ AVTH МОДЕЛЕЙ

Дипломная работа

Научный руководитель:
кандидат физ.-мат. наук,
доцент О. Л. Яблонский

Допущена к защите

«___» _____ 2018 г.

Зав. кафедрой дифференциальных уравнений и системного анализа

доктор физ.-мат. наук, профессор В. И. Громак

Минск, 2018

Оглавление

РЕФЕРАТ

В дипломной работе 70 страниц, 14 графиков, 7 рисунков, 4 таблицы, 26 источников и 2 приложения.

AVTH МОДЕЛИ, КРЕДИТНЫЕ РИСКИ, ПРОГНОЗИРОВАНИЕ КРЕДИТНЫХ РИСКОВ, РЕГРЕССИЯ, B-СПЛАЙНЫ, ВРЕМЕННЫЕ РЯДЫ.

В дипломной работе изучается прогнозирование и анализ кредитных рисков при помощи AVTH модели на основе B-сплайнов.

Целью дипломной работы является построение AVTH модели для прогнозирования и анализа кредитных рисков. Также исследуется целесообразность использования B-сплайнов в данной модели.

Для достижения поставленной задачи использовались: AVTH модели, B-сплайны, различные виды регрессии, временные ряды.

В дипломной работе получены следующие результаты:

1. Успешно построена AVTH модель, которая согласуется с историческими данными
2. Сглажены функции AVTH модели и увеличены возможности по их интерпретации
3. Проверена возможность построение прогнозов AVTH моделью
4. Увеличена скорость расчета параметров модели за счет уменьшения их количества.

Новизна заключается во введении в классическую AVT модель дополнительного параметра h — горизонта, а также в использовании B-сплайнов для создания модели.

Дипломная работа носит практический и теоретический характер. Ее результаты могут быть использованы для применения AVTH моделей на других данных, а также стать основой для дальнейший исследований модели.

Дипломная работа выполнена автором самостоятельно.

РЭФЕРАТ

У дыпломнай рабоце 70 старонак, 14 графікаў, 7 малюнкаў, 4 табліцы, 26 крыніц і 2 прыкладанні.

AVTH МАДЭЛЬ, КРЭДЫТНЫЕ РЫЗЫКІ, ПРАГНАЗАВАННЕ, КРЭДЫТНЫХ РЫЗЫКАУ, РЭГРЭСІЯ, ЧАСОВЫЯ ШЭРАГІ.

У дыпломнай рабоце вывучаюццая прагназаванне і аналіз крэдытных рызыкау пры дапамозе AVTH мадэлі на аснове B-сплайнаў.

Мэтай дыпломнай працы з'яўляецца пабудова AVTH мадэлі для прагназавання і аналізу крэдытных рызыкаў. Таксама даследуецца мэтазгоднасць выкарыстання B-сплайнаў ў дадзенай мадэлі.

Для дасягнення пастаўленай задачы выкарыстоўваліся: AVTH мадэлі, B-сплайны, розныя віды рэгрэсіі, часовыя шэрагі.

У дыпломнай рабоце атрыманы наступныя вынікі:

1. Паспяхова пабудавана AVTH мадэль, якая адпавядае гістарычным дадзеным
2. Згладжаныя функцыі AVTH мадэлі і павялічаны магчымасці па іх інтэрпрэтацыі
3. Праверана магчымасць пабудова прагнозаў для AVTH мадэлі
4. Павялічана хуткасць вылічэння параметраў мадэлі за кошт памяншэння іх колькасці.

Навізна заключаецца ва ўвядзенні ў класічную AVT мадэль дадатковага параметру h - гарызонту, а таксама ў выкарыстанні B-сплайнаў для стварэння мадэлі.

Дыпломная работа носіць тэарэтычны і практычны характар. Вынікі работы могуць быць выкарыстаны для прымянення AVTH мадэлі з іншымі дадзенымі, а таксама стаць асновай для далейшых даследаванняў мадэлі.

Дыпломная работа выканана аўтарам самастойна.

ABSTRACT

The research has 70 pages, 14 plots, 7 pictures, 4 tables, 26 references and 2 applications.

AVTH MODELS, LOAN RISKS, FORECASTING OF LOAN RISKS, REGRESSION, B-SPLINES, TIME SERIES

Forecasting and analysis of loan risks with AVTH model that is based on B-splines are considered in this research.

The purpose of this research is building AVTH model for forecasting and analysis of credit loans. Also the feasibility of using B-splines for this model is investigated.

The following methods and models were used during the research: AVTH models, B-splines, different regression techniques, time series.

The main obtained results in this research include:

1. Successfully build AVTH model, that is consistent with historical data
2. Functions for AVTH model were smoothed and thus possibility for their interpretation was increased
3. Forecasts of build AVTH model were validated
4. Increased speed for calculating model parameters due to reducing the number of unknown parameters

The novelty of the results lies in introducing new time parameter for classical AVT model — h, horizon; and in using B-splines for model.

The research is a practical and theoretical one. Its results can be used for application of AVTH model for different data. It can also form a basis for further research of this model.

The research was done solely by the author.

ВВЕДЕНИЕ

На протяжении около 90 лет в различных науках, таких, например, как те, что изучают демографические процессы, эпидемии и социальные процессы, люди пытались понять и проанализировать данные при помощи временных переменных: возраста (age) и времени (time/period). Позднее к ним добавили винтаж (vintage, cohort) — третью переменную.

Одной из главных целей AVT/APC-анализа является понимание эффектов, которые вносит одна из трех переменных под влиянием двух других.

В банковском деле используются AVT модели. В них время представляет собой влияние внешних факторов, например, экономики; эффект винтажа можно интерпретировать как влияние требований, предъявляемых к заемщику банку при получении кредита. AVT модели используются для прогнозирования кредитных рисков на основе вариантов развития экономики. Как правило государство предоставляет три таких варианта: плановый, плохой и очень плохой. Соответственно важно знать, как различные варианты развития экономики отразятся на платежеспособности граждан и предприятий. В некоторых случаях банку необходимо строить прогнозы без дополнительной информации, тогда можно использовать временные ряды для моделирования влияния внешних эффектов в будущем.

Целью дипломной работы является построение AVTH модели для прогнозирования и анализа кредитных рисков, а также исследование целесообразности использования B-сплайнов в данной модели.

Задачи исследования включают в себя:

- Построение AVTH модели на основе B-сплайнов
- Оценивание эффекта от добавления B-сплайнов
- Определение эффекта горизонта — нового параметра, добавленного в классическую AVT модель
- Прогнозирование кредитных рисков и оценивание качества данных прогнозов

В ходе дипломной работы использовались реальные данные американского банка по автомобильным кредитам.

Теоретической основой работы является опубликованные статьи на тему AVT/APC моделей, раздел математической статистики, относящейся к построению регрессионных моделей, а также интернет-ресурсы с информацией о B-сплайнах.

ГЛАВА 1
КЛАССИЧЕСКИЕ AVT МОДЕЛИ

Перед логическим обоснованием AVT-моделей (age, vintage, time), а также их свойств и ограничений, дадим определение каждой из указанных переменных.

Возраст относится ко времени, которое прошло с момента рождения, или, если рассматривать более общий случай, ко времени, когда субъект провел в некоторой системе. Под временем понимают календарный период, когда рассматривается некоторое событие. Наконец под винтажем будем понимать время, когда человек родился (например, в моделях популяции), или стал частью некоторой системы (например, в банковском деле). Таким образом винтаж отвечает за эффекты свойственные некоторому поколению.

Возраст как правило влияет на риск подверженности заболеваниям и оказывает влияние на экономическое положение человека. Кроме того, с возрастом у людей может менять роль в социальных процессах. Таким образом возраст зачастую оказывается очень важным фактором, который по возможности следует рассматривать при построении моделей или при проведении исследований.

Эффекты, связанные со временем, имеют тенденцию оказывать влияние на всех людей, в независимости от их возраста. Например, в случай распространения болезни по воздуху, у всех людей будет шанс ее заразиться. В банковском деле в качестве примера можно привести мировой финансовый кризис, который на многих людей оказал большое влияние.

Винтажные эффекты могут быть приписаны к факторам, связанным с годом рождения. Болезнь, связанная с плохим питанием, чаще может проявляться во время войны или эпидемии. Однако винтажные эффекты не обязательно будут связаны со временем, которое близко ко времени рождения. Рассмотрим следующий пример. Курить чаще всего начинают в подростковом возрасте или хотя бы в молодости. Таким образом, существенные изменения в маркетинговых кампаниях по продвижению сигарет оказывают влияние на людей, которые именно в этот момент оказываются в группе риска, хотя им уже достаточное количество лет, скажем 15-25, что явно намного больше, чем в первом примере.

APC-модели, которые были разработаны Мэйсоном (Mason) в 1973, на протяжении более чем 30 лет были ведущим статистическим инструментом для оценивания роли возраста, времени и винтажа в демографических и социальных исследованиях. Изначально модель применялась к возрастно-временным данным, для которых была предоставлена частота происшествий, например, заболеваний, преступлений, смертей и др. Главной проблемой в AVT-анализе является то, что три временные переменные линейно зависимы, а именно Время = Возраст + Винтаж. Именно этот факт и привлек внимание к этой проблеме и побудил исследовать ее различными статистическими методами. В прошлом при изучении

данной проблемы у исследователей было не очень много данных, а потому основное внимание были сосредоточено на том, как агрегировать данные, когда есть небольшое число возможностей выбора временных интервалов для возраста (age) и периода (period/time).

1.1 Графики временных трендов

Графики были первым шагом к пониманию эффектов возраста, времени и винтажа. Одним из самых простых считается график поверхности отклика от двух временных осей, как например график (1.1)[7]:

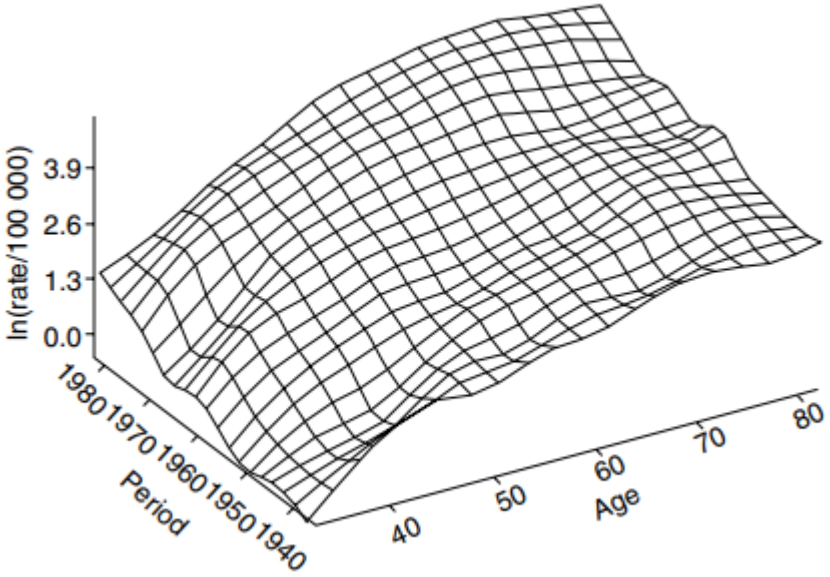

Рисунок 1.1 Заболевание раком легких женщин в Коннектикуте

Вертикальная ось отвечает за натуральный логарифм отклика на 100000 человек. Воспользуемся этой величиной, чтобы продемонстрировать различные графические методы. Хотя такие графики и несут в себе большое количество информации о различных факторах, бывает сложно выделить конкретные детали. Нелегко, а иногда и невозможно выделить величину инцидентов на заданном графике поверхности. Есть и другие вещи, которые остаются неясными при двумерном представлении трехмерной фигуры: меняется ли целевая величина с изменением значений по одной оси, в то время значения по другим осям фиксированы. Это особенно сложно для опущенных временных факторов, в данном случае винтажа. Очевидными альтернативами данному графику видятся поверхности для эффектов возраст-винтаж или же время-винтаж, хотя последняя зачастую не используется, поскольку обычно целесообразно полагать, что возраст оказывает очень сильное влияние на целевую величину.

Еще один тип графика можно получить, если спроецировать поверхность на плоскость возраст-отклик, как показано на рисунке (1.2) [7]:

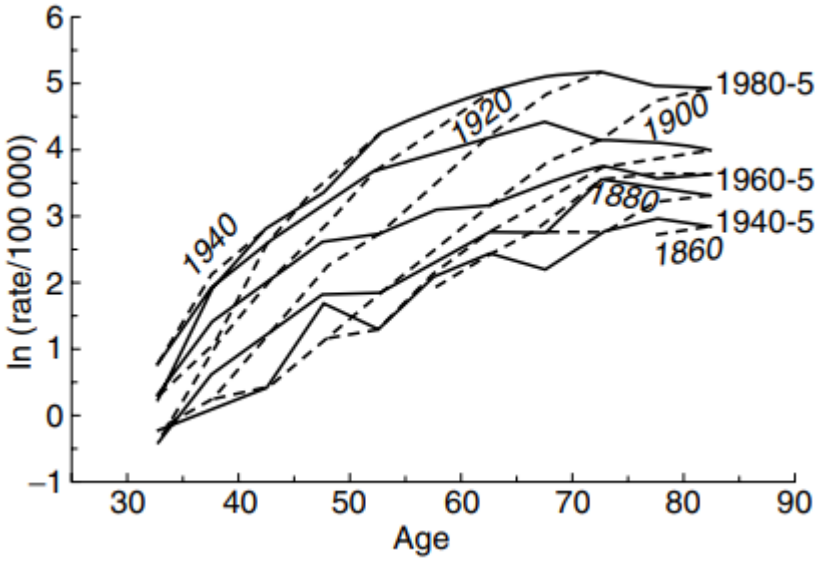

Рисунок 1.2 Проекция рисунка (1.1) на плоскость возраст-отклик

Такие графики использовались для распознавания эффектов винтажа на случаи заболевания. На данном графике толстые линии соединяют параметры возраста для одинакового периода (времени), а линии с черточками — для одинакового винтажа. Можно заметить, что параметры возраста по отношению к фиксированным периодам со временем уменьшаются. Константные линии для винтажа же монотонно возраста вместе с возрастом. С биологической точки зрения кажется неправдоподобным тот факт, что целевая переменная должна уменьшаться с возрастом, поэтому это ведет к тому, что можно убрать из рассмотрения модель возраст-время, а вместо этого считать возраст и винтаж объясняющими факторами для трендов болезни. Аналогичные рассуждения использовались в исследованиях других болезней, и оказывалось, что винтаж имеет существенную роль. Линии винтажа как правило оказывались более параллельны, нежели линии времени, а это в свою очередь очень важно для более формальных моделей, которые будут рассмотрены далее.

Аналогично можно проецировать отклик и на две другие временные оси. Результат представлен на рисунках (1.3):

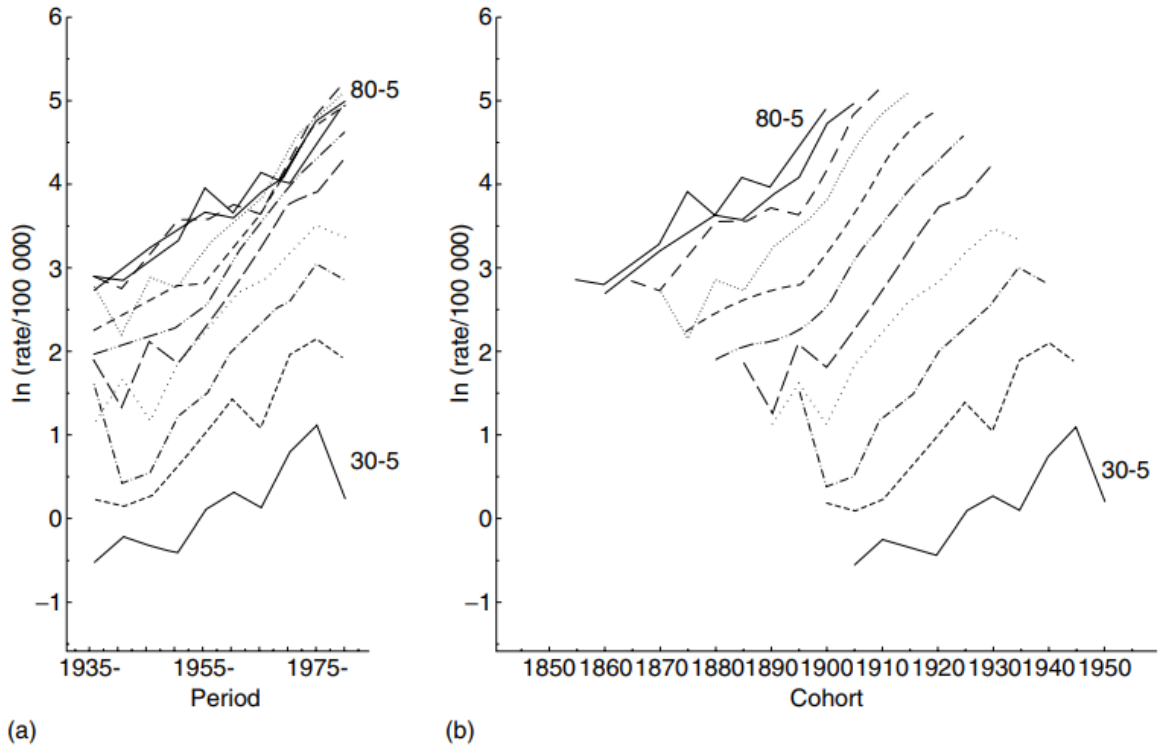

Рисунок 1.3 Проекция графика 1.1 на плоскость время (время)-отклик (a) и на плоскость винтаж(кохорт)-отклик (b)

Каждая линия обозначает или временной, или винтажный тренд для конкретной возрастной группы. Поскольку возраст преобладает над данными трендами, эти графики лучше подчеркивают более тонкие признаки о временных и винтажных трендах. Если данные линии более-менее параллельны осям времени или винтажа, тогда этот фактор предлагает более экономное описание возрастных параметров. Для винтажа и времени используется одинаковый масштаб, чтобы изгиб кривой имел одинаковое визуальное воздействие для винтажа и времени. Иначе ось периодов была бы более широкой, что визуально бы скрадывало некоторые изгибы. Такой эффект можно наблюдать на графике 5, который построен для точно таких же данных, но с сохранением пропорциональности. С растяжением временных осей кривые начинают казаться все более прямыми линиями, которые между собой параллельны. Поскольку зачастую винтажей больше, чем периодов, тренды для периодов будут казаться прямее, если только не использовать оси одинакового масштаба. Таким образом, для содействия сравнению эффектов времени и винтажа важно использовать одинаковый масштаб для временной оси.

Контурные графики предлагают еще один способ для изображения особенностей поверхности отклика путем проецирования линий, для которых отклик одинаков, на плоскость возраст-время., как показано на рисунке (1.4):

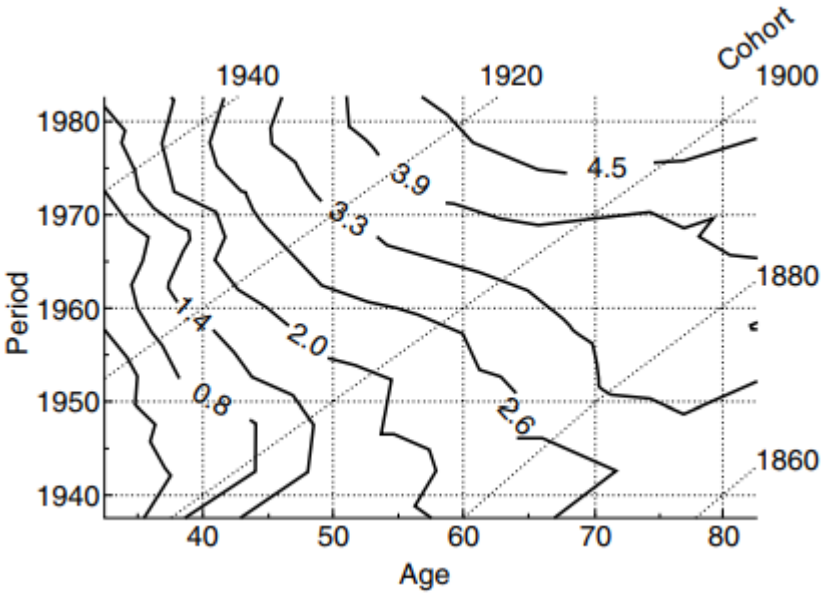

Рисунок 1.4 Пример контурного графика для AVT модели

Контуры представляют собой линии с константным значением случаев рака легких, и подписаны в соответствии с log(число инцидентов/100000) . Аналогичные графии можно получить, используя плоскости возраст-винтаж или время-винтаж. Следуя линиям параллельным оси возраста (времени), можно определить скорость, с которой поверхность увеличивается, в зависимости от того, насколько часто пересекаются контурные линии. Области, в которых контуры параллельны осям возраста (времени), не показывает никакого изменения в числе откликов с течением возраст (времени). Кроме того, рисунке (1.4) присутствуют линии винтажа (диагональные, в виде прерывистой линии из точек), и можно проинтерпретировать результаты аналогичным образом по отношению к винтажу, наблюдая, пересекаются ли контуры или они параллельны осям. Такие графики могут быть особенно полезными при попытке осмысления очень сложным шаблонов.

1.2 Моделирование временных эффектов

Из-за того, что переменные линейно зависимы, невозможно определить параметры моделей, которые основаны на линейно комбинации возраста, времени и винтажа. Целевая переменная зачастую представляет собой отношение успешных исходов ко всем исходам. Полагают, что она имеет распределение Пуассона, и ее логарифм — это линейная функция от приведенных выше регрессорных переменных. Модели такого вида принадлежат к обобщенным линейным моделям.

Сперва рассмотрим случаи, где данные разбиты по возрасту и времени одинаковой длины. Эта ситуация наиболее часто встречается на практике. Случаи,

когда для возраста и времени берутся интервалы разной длины также существуют, однако их исследование представляет еще большую сложность.

В демографических процессах данные располагаются в таблицах, где возрастные группы определяются строками, временные группы — столбцами, а винтаж — диагональю. Модель может быть записана в виде линейное регрессии следующим образом при помощи формулы (1.1)

$$M_{ij} = \frac{D_{ij}}{P_{ij}} = \mu + \alpha_i + \beta_j + \gamma_k + \varepsilon_{ij}, \quad (1.1)$$

Где M_{ij}— частота появления (заболеваний, смертей и т.д.) для i-й возрастной группы $i = 1, \dots, a$ и j-го периода времени $j = 1, \dots, J$; D_{ij} означает число происшествий в $ij-$ й группе; P_{ij} представляет собой общее число наблюдений в $ij-$ й группе; μ обозначает среднее частоты появления; α_i — эффект i-й строки на частоту появления, или коэффициент для i-й возрастной группы; β_j — эффект j-го столбца, или коэффициент для j-й временной группы; γ_k — k-ая диагональ эффекта винтажа, при этом $k = 1, \dots . (a + p - 1)$, где $k = a - i + j$; и наконец ε_{ij} означает случайную ошибку, причем $E(\varepsilon_{ij}) = 0$.

Как уже говорилось, модель, описанная уравнением (1), попадает в класс обобщенных линейных моделей (generalized linear models, GLMs). Они могут иметь различные альтернативные формы, такие как лого-линейная регрессия (log-linear regression) и логистическая регрессия (logistic regression). Такие модели можно считать обобщенными линейными моделями с фиксированным эффектом (fixed-effect generalized linear models) после репараметризации для центрирования параметров по формуле (1.2)

$$\sum_i \alpha_i = \sum_j \beta_j = \sum_k \gamma_k = 0 \quad (1.2)$$

После репараметризации как в уравнении (1.2), модель (1.1) может быть записана в удобной матричной форме по формуле (1.3)

$$Y = X\beta + \varepsilon, \quad (1.3)$$

где Y — вектор целевой частоты (или логарифм целевой частоты); X — матрица регрессии, которая состоит из индикаторов, или векторов-столбцов для вектора параметров модели β,

$$\beta = \left(\mu, \alpha_1, \dots, \alpha_{a-1}, \beta_1, \dots, \beta_{p-1}, \gamma_1, \dots, \gamma_{a+p-2}\right)^T \quad (1.4)$$

и ε — это вектор случайных ошибок со средним значением 0 и диагональной вариационной матрицей вида $\sigma^2 I$, где I — единичная матрица.

Решение уравнения (1.3) методом наименьших квадратов дает следующий ответ (1.5):

$$\beta = (X^T X)^{-1} X^T Y \quad (1.5)$$

Однако решение по формуле (1.5) получить не представляется возможным, поскольку матрица X сингулярна, с рангом на 1 меньше полного, и $(X^T X)$ не обратима, поскольку, как отмечалось выше три временные переменные связаны линейным отношением:

$$\text{Время} = \text{Возраст} + \text{Винтаж} \quad (1.6$$

Эта проблемы известна как model identification problem. Она означает, что есть бесконечное число возможных решений матричного уравнения (1.3), по одному для каждой возможной линейной комбинации векторов-столбцов, которые приводят к вектору идентичному одному из столбцов матрицы X. Таким образом невозможно определить отдельный вклад каждой временной переменной (возраст, время, винтаж) без дополнительных условий в дополнение к репараметризации (1.2).

1.3 Линейная зависимость в дизайн-матрице

Параметры при обычных ограничениях могут быть определены следующим образом. Пусть design-матрица для возраста имеет следующий вид (1.7):

$$A = (A_1, \dots, A_{I-1}), \quad (1.7)$$

где i-й столбец определяется по формуле (1.8):

$$A_i = \begin{cases} 1, \text{если } i - \text{я возрастная группа} \\ -1, \text{если } I - \text{я возрастная группа} \\ 0, \text{в остальных случаях} \end{cases} \quad (1.8)$$

Таким образом выдавая параметры $\alpha_i, i = 1, \dots, I-1$. Поскольку изначально мы требовали, чтобы сумма всех параметров α была равна нулю, то последний параметр α можно получить по следующей формуле (1.9):

$$\alpha_I = -\sum_{i=1}^{I-1} \alpha_i \quad (1.9)$$

Для времени (T) и винтажа (V) компоненты design-матрицы определяются аналогично. Работах [8, 9] показано, что столбцы общий design-матрицы, полученной при помощи конкатенации всех трех компонент удовлетворяет равенству (1.10):

$$\sum_{i=1}^{I-1}\left[i-\frac{I+1}{2}\right]A_i-\sum_{j=1}^{J-1}\left[i-\frac{J+1}{2}\right]T_j+\sum_{k=1}^{K-1}\left[k-\frac{K+1}{2}\right]V_k=0 \quad (1.10)$$

Таким образом, эти столбцы линейно зависимы. Условие, при котором существует уникальный набор параметров для данной модели состоит в том, что у design-матрицы должен быть полный ранг. Следовательно, модель, которая одновременно включает возраст, время и винтаж не позволяет установить уникальный набор параметров, что известно как проблема идентифицируемости (identifiability problem).

1.4 Выделение линейного тренда

Один из способов представления тренда для конкретного фактора заключается в использовании общего тренда. Для этого берется некий глобальный угловой коэффициент прямой, от которого в дальнейшем отталкиваемся, и корректируется при помощи дополнительного слагаемого. При таком подходе влияния возраста может быть выражено следующей формулой (1.11):

$$\phi_{ai}=\left(i-\frac{I+1}{2}\right)\times\beta_a+\gamma_{ai}, \quad (1.11)$$

где β_a — общий коэффициент наклона прямой, а γ_{ai} — кривизна (изгиб). Параметры для винтажа и времени могут быть выражены аналогичным образом. Они имеют следующий вид (1.12, 1.13):

$$\phi_{tj}=\left(j-\frac{J+1}{2}\right)\times\beta_t+\gamma_{tj}, \quad (1.12)$$

$$\phi_{ai}=\left(k-\frac{K+1}{2}\right)\times\beta_v+\gamma_{vk} \quad (1.13)$$

В работе [10] предложено использовать метод наименьших квадратов для оценки коэффициентов наклона, который может быть выражен линейным контрастом (contrast) между параметрами для возраста $\beta_a=C\times\phi_a$, где элементы вектора контрастов имеют вид (1.14):

$$C_i=\left[i-\frac{I+1}{2}\right]\times\frac{12}{I(I-1)(I+1)} \quad (1.14)$$

Для равных временных интервалов. Это ортогональный полиномиальный контраст первого порядка (first-order orthogonal polynomial contrast). β_a называют линейной компонентой наименьших квадратов.

Параметры γ_{ai} можно определить, взяв разность между оцененными параметрами и значениями, полученными при обучении линейной регрессии (residuals). Если используется линейная компонента наименьших квадратов, тогда остатки могут быть вычислены по формуле (1.15):

$$\gamma_{ai} = \phi_{ai} - \left(i - \frac{I+1}{2}\right) \times \beta_a \tag{1.15}$$

1.5 Проблема идентифицируемости для параметров

Поскольку три временные переменные линейно выражаются друг через друга, невозможно получить единственную оценку для параметров без дальнейших ограничений. Использование различных ограничений может изменить не только величину параметров, но и направление тренда для каждого временного фактора, таким образом значительно повлияв на сделанные после анализа выводы. Разделение временных эффектов на линейную составляющую и кривизну, позволяет сократить число параметров, которые линейно зависимы, что ведет к лучшему пониманию эффектов каждой переменной. Показано, что параметры, отвечающие за изгиб, такие как γ_{ai} инвариантны при различных параметризациях или ограничениях, которые накладываются на линейные компоненты [11,12]. Коэффициенты наклона (крутизна) же могут принимать любое значение, $\beta \in (-\infty, \infty)$. Хотя каждый из коэффициентов наклона может довольно сильно изменяться, их всех связывает некое отношение между собой. Данное ограничение предполагает использование оценочных функций от этих параметров, т.е. таких функций, которые не зависят от ограничений, которые мы накладываем для поиска конкретного набора параметров.

Для произвольной пары чисел (r, s) линейная функция $r\beta_a + s\beta_t + (s-r)\beta_v$ инвариантна к конкретному набору параметров, т.е. она является оценочной функцией для коэффициентов крутизны наклона. Например, полагая $r = s = 1$, мы видим, что можно оценить $\beta_a + \beta_t$. Аналогично, положив $r = 0$ и $s = 1$, можно оценить $\beta_t + \beta_v$. Действуя точно также, можно найти различные комбинации коэффициентов наклона, которые не зависят от ограничений, налагаемых для получения конкретного набора параметров.

Тот факт, что каждый отдельный коэффициент наклона может принимать любое значение, является серьезным недостатком данного анализа. Но можно заметить, что если нам удалось определить хотя бы один из коэффициентов наклона, то используя оценочные функции, два других коэффициента определяются сразу. Имея это ввиду, основное выражение для представления каждого отдельного коэффициента наклона может быть выражено по формулам (1.16-1.18):

$$\beta_a^* = \beta_a + \nu, \quad (1.16)$$
$$\beta_t^* = \beta_t - \nu, \quad (1.17)$$
$$\beta_v^* = \beta_v + \nu, \quad (1.18)$$

где β_a, β_t и β_v — истинные коэффициенты наклона, а ν — неопределенный параметр. Если, например, для нас особый интерес представляют временные тренды (time trends), β_t то очень неприятно осознавать, что оцениваемая крутизна может или возрастать или убывать в зависимости от неизвестной ν. Однако, β_a^* также зависит от той же неопределенной константы, так что если для основных групп убывание целевой частоты с возрастом выглядит неправдоподобно, тогда значения ν, при которых крутизна для возраста отрицательная, не должны подходить также и для β_t^*, β_v^*. Если каким-то образом удается показать, что ν лежит в некотором интервале, то должен быть соответствующий интервал значений и для времени, и для винтажа.

Мы можем наблюдать эффект неидентифицируемости линейных переменных, рассматривая модель, в которой мы отбрасываем компоненты изгибов:

$$Y = \mu + a \times \beta_a + t \times \beta_t + v \times \beta_v \quad (1.19)$$

Поскольку между тремя временными переменными есть линейная зависимость, мы можем добавить $0 = \nu \times (a - t + v)$ к правой стороне в предыдущем равенстве, что дает следующее уравнение (1.20):

$$Y = \mu + a \times (\beta_a + \nu) + t \times (\beta_t - \nu) + v \times (\beta_v + \nu) = \quad (1.20)$$
$$= \mu + a \times \beta_a^* + t \times \beta_t^* + v \times \beta_v^*$$

которое является моделью, основанной на параметрах полученных при использовании конкретных ограничений.

1.6 Подходы к идентифицируемости

Проблема идентифицируемости по своей природе не имеет решения в обычным смысле этого слова. Различные ограничения на модель могут привести к совершенно разным трендам параметров. Предложения для получения конкретного множества параметров могут быть произвольными, и поэтому должны подвергаться критической оценке при интерпретировании результатов, полученных при анализе. Есть большое число предложенных путей решения проблемы, но у каждого из них есть потенциально серьезные ограничения. В качестве альтернативы можно все ограничить оценочными функциями параметров, и таким образом избежать произвольности любого конкретного решения.

1.6.1 Ограничения на параметры

Уникальные значение оцениваемых параметров для модели временных трендов получаются при накладывании определенных ограничений на параметры. Иногда они произвольным образом выбираются программой, применяемой для решения задачи регрессии, которая использует некоторую обобщенную обратную матрицу для поиска оценок при помощи метода максимального правдоподобия. Однако при решении задач часто бывает полезно вручную задавать условия, чтобы лучше понимать влияние установленных ограничений.

1.6.2 Исключение фактора

Возможно, один из самых простых способов решения проблемы неидентифицируемости параметров модели — это попытка избегать рассматривания всех трех временных переменных одновременно. При обучении такой двухфакторной модели результаты довольно просто интерпретировать, и данный подход действительно может оказаться очень полезным, если такая модель хорошо подходит под данные. Однако модель, которая отбрасывает один из факторов, означает, что мы считаем, будто у фактор не оказывает никакого влияния, т.е. из-за него никак не меняется ни линейная, ни составляющая кривизны. При рассмотрении примеров на реальных данных можно убедиться, что последнее никак не согласуется с данными, и может иметь место затяжное смещения (bias) в параметрах — неидентифицируемая константа ν — что может оказывать очень большое влияние, даже если модель показывает хорошие результаты при обучении на данных.

1.6.3 Уравнивание двух эффектов

Второй подход к поиску уникального набора параметров заключается в уравнивании только двух эффектов для одного из факторов модели [13, 14, 15], вместо того, чтобы приравнивать все эффекты одного фактора к нулю. Например, два соседних эффекта для времени могут быть приравнены друг к другу, поскольку есть основания полагать, что никаких изменений за одну эпоху не возникло, т.е. $\phi_{t1} = \phi_{t2}$. Один из вариантов данного подхода — предположение, что среднее значение последовательных разностей равно нулю, что, раскрываю цепочку равенств, можно записать в виде $\phi_{t1} = \phi_{tJ}$ — при рассмотрении времени, для других факторов равенства аналогичные за исключением последнего индекса. Данное ограничение автоматически применяется некоторыми программами для последнего предоставленного фактора при решении задачи регрессии. Это

ограничение очень просто применить, и оно вынуждает параметры вернуться к их изначальному уровню, что на выходе может привести к параметрам идентичным тем, что получаются при задании временного (или любого другого) коэффициента наклона равным нулю.

Преимущество данного подхода к решении проблемы неидентифицируемости в том, что его интуитивно просто понять, и в тоже время совсем несложно применить на практике. В дополнение к этому, такой подход не приводит к равенству для всех эффектов, как это было бы при полном исключении одного из факторов из модели. К сожалению, нет достаточно весомой причины для уравнивания двух эффектов, мы лишь можем полагать, что вряд ли что-то изменится за один временной интервал, а раз так, то можно их уравнять. Часто при рассмотрении конкретного случая равенство, например, 4-го и 5-го периода кажется таким же логичным, как и равенство между 1-м и 2-м, но в тоже время параметры, полученные в этих случаях, могут очень сильно отличаться.

1.6.4 Минимизация евклидового расстояния по отношению к двухфакторным моделям

В работе [16] предлагается подход, который позволяет оценить неидентифицируемый параметр ν. Их критерий использует евклидово расстояние между параметрами от AVT-модели и соответствующей модели, которая исключает один из факторов, т.е. $\left\|\phi(\nu) - \phi_{(\nu)}\right\|$ в случае с моделью, которая отбрасывает винтаж. Поскольку возраст играет значительную роль в большинстве случаев, полностью он не исключается; но вместо того, чтобы использовать значения из модели, которая основана только на возрасте, используются смещенные значения. Критерий оценки заключается в минимизации выражения (1.21):

$$g(\nu) = \frac{\left\|\phi(\nu) - \phi_{(\nu)}\right\|}{\rho_v} + \frac{\left\|\phi(\nu) - \phi_{(t)}\right\|}{\rho_t} + \frac{\left\|\phi(\nu) - \phi_{(a)}\right\|}{\rho_a}, \qquad (1.21)$$

где ρ_v, ρ_t и ρ_a — остаточные средние квадраты (residual mean squares) для соответствующих моделей.

Хотя данный подход и позволяет получить уникальное значение параметров модели, остается открытым вопрос, является ли критерий для использования в общем случае. С одной стороны кажется логичным параметрам из двухфакторной модели, плохо описывающей данные (poor, high residual means squares), присваивать меньший вес, пока мы не вспоминаем, что можем только оценить кривизну, или изгиб, в трехфакторной модели. Фактор с большой кривизной приведет к относительно большому остатку средних квадратов для упрощенной модели, таки образом получив меньший вес согласно оценочному критерию. Но в

то же время неясно, почему параметр, который отвечает за линейный тренд должен как-то соотноситься с кривизной или же параметрами двухфакторной модели, которая плохо подходит для описания данных.

1.6.5 Принятие крутизны равной нулю

Альтернативой полному исключению фактора является предположение, что его крутизна равна нулю. Например, мы можем принять временную крутизну равной нулю, $\beta_t = 0$. Такой подход по сути по сути является расширением метода, в котором мы полностью исключаем фактор, поскольку в обоих случаях мы полагаем, что общая крутизна для одного фактора равна нулю; в то же время данный подход не требует, чтобы все слагаемые, которые отвечают за кривизну были равны нулю. Для всех трех кривых по-прежнему может быть неизвестно смещение (unidentifiable bias). Одной из вариаций данного подхода является фиксирование крутизны на коротком интервале времени, а не на всем сразу. Например, можно предположить, что для 1940-1969 гг. нет никакого тренда (3 десятилетних временных интервала), а в дальнейшем он появляется.

В качестве реального случая использования данного подхода можно привести работы [17, 18]. В них изучались случаи заболевания раком, используя данные из Реестра Опухолей Коннектикута (Connecticut Tumor Registry). В основном анализ фокусировался на изучении эффектов кривизны для каждого временного фактора, но в графиках в заключительной части временная крутизна полагалась равной нулю, $\beta_t = 0$. Тому было несколько рациональных причин: 1) есть серьезные биологические основания полагать, что возраст оказывает большое влияние на рак, и если один из временных факторов и должен быть неважен, то скорее всего, это или винтаж, или период (время); 2) эмпирические результаты позволяют сделать вывод, что винтаж оказывает куда большее влияния на случаи заболевания раком, чем временной период; 3) предположение, что $\beta_t = 0$ по смыслу налагает меньше ограничений, чем полное игнорирование временного фактора.

1.6.6 Ограничение границ крутизны

Еще один способ выбора ограничений на параметры заключается в использовании теоретических знаний о процессе по отношению к одной из временных переменных. В работе [19] анализировалось влияние эффектов возраста, времени и винтажа на риск возникновения депрессии в 5 общинах США. Несмотря на то что не было высказано никакого конкретного предположения об общих трендах для времени и винтажа, казалось разумным допустить, что и у времени, и у винтажа отсутствует убывающий тренд, т.е. $\beta_t \geq 0$ и $\beta_v \geq 0$. Добавляя

β_v к обеим частям первого неравенства и β_t ко второму, получаем $\beta_t + \beta_v \geq \beta_v \geq 0$ и $\beta_t + \beta_v \geq \beta_t \geq 0$. Можно заметить, что в каждом случае можно оценить и верхние границу. Точно также крутизна для возраста должна удовлетворять неравенству $\beta_a + \beta_t \geq \beta_a \geq \beta_a - \beta_v$, для которого также можно определить верхнюю и нижнюю границу. Используя данные границы Викрамаратну удалось получить качественные результаты, которые заключались в увеличивающемся тренде риска заболевания депрессией для людей, родившихся в 1935-1944 гг., хотя оказалось невозможным установить точечную оценку для тренда.

1.7 Оцениваемые функции

Для того, чтобы не прибегать к произвольным условиям, обратимся к оценочным функциям параметров, которые позволяют сделать выводы идентичные для любого полученного множества параметров модели. Рассмотрим некоторые оценочные функции, которые были признаны полезными.

1.7.1 Прогнозирование при помощи на AVT моделей

Проблема прогнозирования трендов является довольно сложной, поскольку необходимо делать предположения о трендах за пределами существующих данных, которые в общем случае нельзя проверить. Например, мы могли бы предположить, что тренды из прошлого сохранятся и в будущем, но в каком-то конкретном случае это может быть совсем необоснованно. Тем не менее его широко используют на практике, так как оно одно из тех, что кажутся разумными при отсутствии признаков, которые говорят об обратном. Если мы используем линейное экстраполирование для всех трех переменных, то частоты целевых событий вполне можно определить. Это свойство можно продемонстрировать на модели, которая включает только линейные переменные, помня про то, что более сложные модели, включающие кривизну, не несут новых проблем, поскольку параметры, отвечающие за кривизну, могут быть оценены. Итоговая модель для i-го возраста, j-го периода (времени) и k-го винтажа имеет вид (1.22):

$$Y_{ijk} = \mu + i \times \beta_a + j \times \beta_t + k \times \beta_v \quad (1.22)$$

Для аналогичного винтажа увеличение возраста и соответственно времени на одну единицу дает выражение (1.23):

$$Y_{i+1,j+1,k} = \mu + (i+1) \times \beta_a + (j+1) \times \beta_t + k \times \beta_v, \quad (1.23)$$

Тогда разность между двумя коэффициентами (1.24):

$$Y_{i+1,j+1,k} - Y_{i,j,k} = \beta_a + \beta_t, \quad (1.24)$$

является оцениваемой функцией коэффициентов наклона.

1.7.2 Авторегрессионные модели для временных эффектов

При рассмотрении вторых разностей мы также сталкиваемся с проблемой идентифицируемости. В работе [20] предлагается использовать авторегрессионные модели для временных эффектов, в которых последующие параметры определяются уравнением (1.25):

$$\phi_{v,k} = 2\phi_{v,k-1} - \phi_{v,k-2} + \varepsilon_{cv} \quad (1.25)$$

для винтажа. Для возраста и времени уравнения аналогичные. Каждое последующее значение явно зависит от двух предыдущих и случайной ошибки, $\varepsilon_{vk} \sim N(0, \sigma_v^2)$. Авторы описывают Байесовский подход для оценивания параметров модели, в котором применяются Марковская цепь и алгоритм Монте-Карло. Одно из интересных расширений этого подхода позволяет применять Байесовское прогнозирование целевой переменной, что также является оцениваемой функцией параметров модели.

1.7.3 Дизайн-матрица

Альтернативой использованию контрастов для оценки параметров модели является построение дизайн-матрицы с линейно независимыми столбцами, которая позволит получить единственное решение для параметров модели. Конечно, обычный подход построения дизайн-матрицы с использованием фиктивных переменных для каждого уровня возраста, времени и винтажа обязательно приведет к линейной зависимости в матрице, поскольку индексы, описанные выше, также линейно зависимы ($t = a + v$).

Распределенная дизайн-матрица, которая разделит эффекты может быть записана в виде (1.26):

$$X = [1|A_L|A_C|T_L|T_C|V_L|V_C], \quad (1.26)$$

Где каждая строка соответствует целевой переменной в конкретной возрастной и временной группе. В столбцах A_L и A_C содержатся компоненты кривизны и линейной части возраста, оставшиеся же элементы матрицы представляют собой аналогичные компоненты для времени и винтажа. Параметры регрессии, соответствующие дизайн-матрице, задаются вектором

$$\Theta = (\phi_0|\beta_a|\gamma_a|\beta_t|\gamma_t|\beta_v|\gamma_v)'$$

Вектор-столбец соответствующий линейной компоненте возраста, A_L, можно определить следующим набором элементов $A_{Li} = i - [I + 1]/2$ для $i -$ й вощрастной группы. Для времени и винтажа столбцы, которые также отвечают за линейную часть, T_L и V_L, , определяются аналогично. Линейная зависимость в дизайн-матрице очевидна, поскольку $T_L = A_L + V_L$. Таким образом модель, которая включает A_L и V_L, уже содержит информацию и T_L. Параметр, полученный в результате регрессии для оставшегося столбца A_L на самом деле будет оценивать сумму крутизны для возраста и времени, $\beta_a + \beta_t$. Точно так же параметр для V_L будет оценивать $\beta_v + \beta_t$. Это оцениваемые функции для коэффициентов наклона прямой, рассмотренные ранее.

Компоненты кривизны дизайн-матрицы задаются оставшимися регрессорными переменными, насыщающими эффект. Если убрать первый и последний столбцы для возраста в дизайн-матрице, которые содержат индикаторные переменные (0 или 1) для каждой группы, то это эквивалентно наложению условия $\phi_{a1} = \phi_{aI}$, что в конечном счете позволяет получить коэффициенты наклона прямых, которые аналогичны тем, что получаются при рассмотрении средних соответствующих разностей. Однако, если мы захотим представить кривизны, используя переменные, которые ортогональны линейной части, тогда мы получим коэффициенты наклона, которые соответствуют тем, что были бы получены при использовании метода наименьших квадратов.

1.7.4 Вывод об оцениваемых эффектах

Приведем некоторые выводы о проблеме неидентифицируемости. Не делая никаких строгих предположений, нельзя оценить общие изменения для возраста, времени и винтажа. Это является очень серьезным ограничением, поскольку наиболее интересные вопросы относятся к тому, возрастают тренды или же убывают. Тем не менее на некоторые возникающие вопросы можно ответить, используя величины, которые можно оценить, а именно:

1. Предсказанные значения;
2. Изменение коэффициента наклона прямой или отклонение от общего тренда;
3. Временные пики (сильно выбивающееся значение по сравнению с общим трендом);

В конечном счете, хотелось бы обоснованно оценивать параметры, и если это требуется делать без каких-либо предположений о наблюдаемом эффекте, то придется использовать оцениваемые функции.

1.8 Нелинейные эффекты

Сложности при работе с моделями, вызванные проблемой неидентифицируемости, привела к попыткам использования нелинейным моделей. Например, модель из работы [21] определяется формулой (1.27):

$$Y_{ijk} = \mu + \phi_{tj} + \phi_{tj} + \phi_{ai} \cdot \delta_j + \varepsilon_{ijk} \quad (1.27)$$

В данной модели ϕ_{tj} и ϕ_{tj} обозначают эффекты времени и винтажа. Эффект возраста, ϕ_{ai} включается совместно со множителем, δ_j, который зависит от времени и таким образом изменяем изначальный эффект. Несмотря на то, что данная модель позволяет получить единственное решение, оценить его параметры может быть довольно сложно, а кроме того, они могут быть очень нестабильными. Особый случай для данной модели мы получаем, если принять $\delta_j = 1 \; \forall j$, который совпадает с обычной моделью. В случаях, когда такая модель хорошо подходит под данные, параметры как правило оказываются нестабильными.

Еще один подход, который иногда ведет к нелинейным моделям заключается в использовании некой внешней информации для одного из параметров, и таким образом определяет его вид. Например, в работе [22] рассуждается об использовании многоступенчатых моделей канцерогенеза, описанных в работе [23] в применении к анализу раку легких. В таком случае эффект для возраста имеют вид (1.28):

$$A(a) \propto a^{\omega}, \quad (1.28)$$

где параметр ω означает число ступеней модели минус один. Если целевая переменная логарифмирована, тогда эффект для возраста принимает вид (1.29):

$$\phi_{ai} = \omega \times \ln(a_i), \quad (1.29)$$

который уже не является линейным по возрасту. Данная и связанные с ней нелинейные модели для возраста позволяют получить уникальный набор параметров модели, не прибегая к ограничениям. Однако в этом случае полученное решение сильно зависит от успешности выбранной математической модели для описания возраста. Даже в этом случае параметры могут быть крайне нестабильны и трудно оцениваемыми с использованием метода максимального правдоподобия. В дополнение к этому, оценки общего тренда могут сильно отличаться в зависимости от выбранной модели для возраста.

1.9 Замена временных переменных

В основе большинства исследований временных трендов лежит идея о том, что влияние времени связано с некоторыми факторами, которые влияют на целевую переменную. Если это верно, то в анализ лучше включать именно фактор, который оказывает непосредственное влияние, вместо того, чтобы заменять его временем. При работе с моделями, в которых есть возбудители болезни, необходимо иметь данные о населении, которое было подвержено влиянию вредных факторов на протяжении времени. Одним из примеров, для которого доступна такая информация, — рак легких, поскольку известно, что на сегодняшний момент курение является главной причиной заболевания. При этом при разработке модели важно помнить, что есть и несколько других фактов, которые стоит учитывать: 1) курить часто начинают в подростковом возрасте, или по крайней мере в молодости, — это довольно небольшой промежуток времени, и изменение потребления сигарет в первую очередь влияет на людей из этой возрастной группы; 2) может пройти довольно много времени между моментом, когда человек начал курить, и когда был диагностирован рак; 3) все курильщики по разному подвергаются воздействию никотина: кто-то курит больше сигарет, также сами сигареты содержат разное количество этого вещества; 4) эффект от курения накапливается, так что даже люди, которые курят одинаковые количество похожих сигарет, в конкретный момент времени имеют разное количество накопленного воздействия, если они начали курить в разное время; 5) у людей, которые бросили курить, риск заболеть находится где-то между риском курящего человека и того, кто никогда не курил.

Вклад в риск подверженности заболеванию с момента, когда человек начал курить, очевидно, зависит от винтажа (см. п 1 выше). Тем не менее, изобретение фильтров и другие изменения в технологии производства сигарет связаны с эффектом периода (времени). Другие факторы могут быть связаны и с винтажем, и с временем одновременно. Например, определенные поколения могут более серьезно относиться к своему здоровью и поэтому более охотно бросать курить, эффект винтажа, но в тоже время вся популяция может быть подвержена антитабачной кампании, исходящей из средств массовой информации, временной эффект.

В работе [24] авторы применили модель к данным по смерти от рака легких в США, которые использовали данные США по составу сигарет с течением времени, что, по ожиданиям, должно было оказать наибольшее влияние на параметры периода (времени). Эффект периода был выражен линейной функцией от количества смолы, в итоге логарифмированная смертность описывалась моделью (1.30):

$$Y_{ijk} = \phi_0 + \phi_{ai} + \beta X_j + \phi_{vk} + \varepsilon_{ijk}, \qquad (1.30)$$

где X_j представляет собой воздействие количества смолы на население для j-го периода. Хотя оценки по распространения курения и не были включены в модель, шаблон в оцененных эффектах винтажа был очень похож на шаблон распространения курени, полученный на основании выборочного обследования мужчин и женщин. Успешное использование информации об изменении состава сигарет в данном конкретном примере не обязательно подразумевает, что такой подход будет одинаково успешным. Если бы был сильный линейный тренд в средних содержания смолы по времени, тогда X_j линейно бы зависело от I и k, что вновь бы привело к проблеме неидентифицируемости.

1.10 Взаимодействия с временными факторами

Каждая из ранее рассмотренных моделей подразумевает, что эффект временного фактора никаким образом не зависит от уровня других, т.е. что между ними отсутствуют взаимодействия. Частично проблема с рассмотрением взаимодействия между временными переменными проявляется, поскольку в любой двухфакторной модели включение взаимодействия приводит к насыщенной модели. Например, если мы будем рассматривать только возраст и время, модель примет вид (1.31):

$$Y_{ij} = \mu + \phi_{ai} + \phi_{tj} + \phi_{at,ij} + \varepsilon_{ij}, \qquad (1.31)$$

Где ϕ_{ai} и ϕ_{tj} — главные эффекты возраста и времени, а $\phi_{at,ij}$ — эффект от взаимодействия. Сравнивая эту модель с AVT-моделью, можно видеть, что единственная разница между моделями состоит в том, что классическая модель включает эффект винтажа, ϕ_{vk}, а эта — взаимодействие возраста-времени, $\phi_{at,ij}$. Поскольку взаимодействия насыщают модель, которая включает только главные эффекты возраста и времени, можно считать эффект винтажа конкретным типом взаимодействия возраст-время. Аналогичным образом можно описать эффект времени, как конкретный тип взаимодействия возраст-винтаж, или эффект возраста, как взаимодействие время-винтаж, — обе эти модели также являются насыщенными. В работе [25] изучаются полиномиальные модели для трех временных факторов; более того указывается, какие взаимодействия могут быть идентифицированы. Даже при лучших обстоятельствах интерпретация взаимодействий в моделях высших порядков затруднительна, не говоря уже о том, что по-прежнему актуальна проблема неидентифицируемости. Следовательно, нужно очень аккуратно вводить взаимодействия в данных моделях.

В качестве альтернативы полиномиальным взаимодействиям между временными переменными, можно просто разделить время на части, как например, при рассмотрении рака груди у женщин, которые младше и старше 50 лет. В работе

[18] использовалось именно такое разделение, поскольку использовалось предположение о том, что тренды для рака груди могут отличаться у женщин, у которых наступила или не наступила менопауза. В таком случае модель может быть записана в виде (1.32):

$$\log \lambda_{ijk} = \begin{cases} \mu + \phi_{ai} + \phi_{tj} + \phi_{vk} + \phi_{av,k} + \varepsilon_{ijk}, i < i_0 \\ \mu + \phi_{ai} + \phi_{tj} + \phi_{vk} - \phi_{av,k} + \varepsilon_{ijk}, i \geq i_0' \end{cases} \quad (1.32)$$

где $\phi_{av,k}$ представляет разность между средней логарифмированной частотой (log rate), а i_0 — это та возрастная группа, по которой проводится разбиение. Такой тип взаимодействия не добавил бы дополнительных сложностей, если бы рассматривалась только модель возраст-время, но дополнительная трудность возникает, когда в рассмотрение включается винтаж. Если следовать диагонали винтажа в обычной таблице частот, можно видеть, что, изменяя строку возрастной группы, с которой идет взаимодействие, в то же время изменяется и набор винтажей, которые используются для построения выводов о взаимодействии.

1.11 Выводы по главе

AVT модели нашли свое применение в анализе различных процессов, где важную роль играют временные переменные: банковское дело, биологические модели

При использовании AVT модели не существует единственного значения для неизвестных параметров, поэтому крайне полезно иметь некоторое представление о влиянии хотя бы одного фактора на целевую переменную.

ГЛАВА 2
ИНСТРУМЕНТЫ ДЛЯ ПОСТРОЕНИЯ И АНАЛИЗА МОДЕЛЕЙ

2.1 Линейная регрессия

Пусть дан набор объектов $D = \{x_i, y_i\}_{i=1}^{n}$, где $x_i = (x_1, x_2, \dots, x_m) \in \mathbb{R}^m$ — признаки, которые описывает объект, а $y_i \in \mathbb{R}$ — целевая (предикторная) переменная.

Гипотеза состоит в том, что целевую переменную можно представить в виде формулы (2.1):

$$h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + \cdots \theta_m x_m = \theta x, \tag{2.1}$$

т.е. мы считаем, что целевая переменная линейно зависит от каждого своего признака. Для удобства полагаем, что $x_0 = 1$.

Наша задача найти такой вектор параметров $\theta^T = (\theta_0, \theta_1, \dots, \theta_m) \in \mathbb{R}^{m+1}$, который позволяет наиболее точно приблизить предикторную переменную по ее признакам. Для этого введем функцию стоимости по формуле (2.2):

$$J(\theta) = \frac{1}{2n}\sum_{i=1}^{n}(h_\theta(x_i) - y_i)^2, \tag{2.2}$$

Тогда мы получаем следующую задачу оптимизации (2.3):

$$\min_\theta J(\theta) = \min_\theta \frac{1}{2n}\sum_{i=1}^{n}(h_\theta(x_i) - y_i)^2 \tag{2.3}$$

Рассмотрим два варианта ее решения: точный и приближенный.

2.1.1 Нормальное уравнение

Введем обозначения (2.4, 2.5)

$$Y = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \in \mathbb{R}^n, \qquad X_i = \begin{pmatrix} x_{1,i} \\ \vdots \\ x_{n,i} \end{pmatrix} \in \mathbb{R}^n, \qquad X_0 = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \tag{2.4}$$

$$X = [X_0, X_1, \dots, X_{m+1}] = \begin{pmatrix} x_{1,0} & \cdots & x_{1,m+1} \\ \vdots & \ddots & \vdots \\ x_{n,0} & \cdots & x_{n,m+1} \end{pmatrix} \in \mathbb{R}^{n\times(m+1)} \tag{2.5}$$

Также полагаем, что $n > m$ — число наблюдений больше, чем число признаков, что естественно. На практике так обычно и есть, иначе бы пришлось оценивать число параметров большее, чем число уравнений. Кроме того считаем, что $rankX = m + 1$, т.е. у нас нет наблюдений, для которых одинаковые признаки дают разные значения целевых переменных.

Итак, в новых обозначениях наша модель имеет вид (2.6):

$$Y = X\theta \tag{2.6}$$

Задачу же (2.3) можно представить в виде (2.7):

$$\|Y - X\theta\|^2 = (Y - X\theta)^T(Y - X\theta) = \sum_{i=1}^{n}\left(y_i - \sum_j x_{i,j}\theta_j\right)^2 \to \min \tag{2.7}$$

При такой функции стоимости $J(\theta)$ решение задачи (2.3) эквивалентно решению переопределенной системы (2.6) методом наименьших квадратов. Для этого уравнение (2.6) в обоих частях умножим слева на X^T, откуда уже легко сможем найти вектор параметров θ. Получаем формулу (2.8):

$$X^TY = X^TX\theta \Rightarrow \theta = (X^TX)^{-1}X^TY \tag{2.8}$$

Уравнение (здесь номер) называют Нормальным уравнением (Normal Equation). В данном случае значение вектора θ находится аналитически. Такая оценка является несмещенной и состоятельной.

2.1.2 Нормальное уравнение на практике

На практике при решении Нормального уравнения могут возникать сложности связанные с обратимостью матрицы X^TX. Эта матрица будет вырожденной. Наиболее частые случаи:

1. Избыточные признаки. Существуют два или более признаков, которые очень сильно коррелируют, фактически они линейно зависимы. Например, сумма кредита, выданная клиенту в долларах, евро и фунтах. В этом случае нужно оставить один любой признак из данного набора.
2. Слишком большое количество признаков ($m \geq n$). Хотя выше мы и считаем, что такой ситуации нет, иногда такое все же происходит. В этом случае нужно уменьшить число признаков или же воспользоваться регуляризацией.

2.1.3 Градиентный спуск

В случае когда n очень велико, например, $n = 10000$, найти решение, используя его аналитическое выражение, становится затруднительно, поскольку операция нахождения обратной матрицы требует $O(n^3)$ операций, что очень затратно. В этом случае искать параметры θ можно численно. Для этого воспользуемся градиентным спуском: возьмем частные производные функции стоимости $J(\theta)$ по параметрам θ, и будем идти против градиента.

Общее правило обновление каждой компоненты вектора θ имеет вид (2.9):

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta) \tag{2.9}$$

Где α — скорость обучения, гиперпараметр, который необходимо задать. $0 < \alpha \leq 1$. Как правило хорошим начальным значением α можно считать $\alpha = 0.01$.

Теперь распишем подробно частные производные по формулам (2.10):

$$\begin{aligned} &\frac{\partial}{\partial \theta_j} J(\theta) = \frac{\partial}{\partial \theta_j} \frac{1}{2} (h_\theta(x) - y)^2 = \\ &= 2 \frac{1}{2} (h_\theta(x) - y)^2 \frac{\partial}{\partial \theta_j} (h_\theta(x) - y) = \\ &= 2 \frac{1}{2} (h_\theta(x) - y)^2 \frac{\partial}{\partial \theta_j} \left(\sum_{i=0}^{m} \theta_i x_i - y \right) = \\ &= (h_\theta(x) - y) x_j \end{aligned} \tag{2.10}$$

С учетом полученного выражение градиента итоговое правило обновление компонент вектора θ для каждого отдельного наблюдения можно задать по формуле (2.11):

$$\theta_j := \theta_j - \alpha (h_\theta(x_i) - y_i) x_j \tag{2.11}$$

Используя формулу (2.11) общий алгоритм градиентного спуска можно представить в виде (2.12) [1]:

$$\begin{aligned} &\text{Повторять, пока не будет сходимости } \{ \\ &\quad \theta_j := \theta_j - \alpha (h_\theta(x_i) - y_i) x_j, \qquad j = 0, 1 \dots m \\ &\} \end{aligned} \tag{2.12}$$

Используя матричные обозначения, алгоритм (2.12) можно упростить.

Градиентный спуск имеет вид (2.13):

$$\theta = \theta - \alpha \nabla J(\theta), \tag{2.13}$$

где

$$\nabla J(\theta) = \begin{pmatrix} \frac{\partial J(\theta)}{\partial \theta_0} \\ \frac{\partial J(\theta)}{\partial \theta_1} \\ \vdots \\ \frac{\partial J(\theta)}{\partial \theta_m} \end{pmatrix} \quad (2.14)$$

Подробнее распишем изменение каждой компоненты вектора θ по формулам (2,15):

$$\begin{gathered} \theta_0 := \theta_0 - \alpha(h_\theta(x_i) - y_i)x_0, \\ \theta_1 := \theta_1 - \alpha(h_\theta(x_i) - y_i)x_1 \\ \theta_2 := \theta_2 - \alpha(h_\theta(x_i) - y_i)x_2 \\ \dots \\ \theta_m := \theta_m - \alpha(h_\theta(x_i) - y_i)x_m, \end{gathered} \quad (2.15)$$

заметно, что множители после α можно представить в матричном виде как $X^T(X\theta - Y)$. Далее возьмем среднее обновление параметров для каждой компоненты θ, и в итоге получим следующую векторизованную формулу (2.16):

$$\theta := \theta - \frac{\alpha}{n} X^T(X\theta - Y) \quad (2.16)$$

2.1.4 Градиентный спуск на практике

Градиентный спуск можно значительно ускорить, если все наши признаки будут примерно в одном диапазоне. Если этого не делать, то в зависимости от величины признаков x параметры θ будут слишком быстро или же наоборот слишком медленно изменяться, т.е. алгоритм будет совсем не эффективно сходиться к оптимальному значению.

Чтобы этого не допустить, мы должны отобразить все признаки в интервал

$$-1 \le x_i \le 1 \text{ или, например, } -0.5 \le x_i \le 0.5$$

Не обязательно в эти интервалы, мы лишь пытаемся ускорить алгоритм. Наша цель, чтобы признаки были примерно в одном масштабе.

Для этого воспользуемся двумя способами: центрированием (mean normalization) и масштабированием (feature scaling). Для этого сначала вычтем среднее значение признака для всех объектов (таким образом их среднее станет равным нулю), а затем разделим на среднеквадратичное отклонение. Запишем это в виде формулы (2.17):

$$x_i := \frac{x_i - \mu_i}{\sigma_i}, \tag{2.17}$$

где μ_i — среднее значение, а σ_i — среднеквадратичное отклонение.

2.1.5 Регуляризация

Зачастую бывает полезно вводить в модель регуляризацию: это может помочь ей лучше работать на новых данных, и сделать ее “проще”. Под этим понимается тот факт, что составляющие вектора θ не будут слишком большими по модулю числами.

Для этого в функцию стоимости, которую мы оптимизируем по θ, добавим слагаемые, которые будут содержать θ. Например, модули. В этом случае функция для минимизации будет определяться формулой (2.18):

$$\min_{\theta} \frac{1}{2n}\left(\sum_{i=1}^{n}(h_\theta(x_i) - y_i)^2 + \lambda \sum_{i=1}^{n}|\theta_i|\right) \tag{2.18}$$

Получаем l_1— регуляризацию, если же добавить квадраты компонент θ, то получим l_2—регуляризацию, которая определяет задачу оптимизации по формуле (2.19):

$$\min_{\theta} \frac{1}{2n}\left(\sum_{i=1}^{n}(h_\theta(x_i) - y_i)^2 + \lambda \sum_{i=1}^{n}\theta_i^2\right) \tag{2.19}$$

В Нормальном уравнении можно использовать единичную матрицу, где на первом месте стоит 0. Она имеет вид (2.20):

$$L = \begin{pmatrix} 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ 0 & 0 & \ddots & \vdots & \vdots \\ \vdots & \vdots & \dots & 1 & 0 \\ 0 & 0 & \dots & 0 & 1 \end{pmatrix}, \tag{2.20}$$

после этого ее нужно умножить на параметр регуляризации λ.

В итоге Нормальное уравнение можно записать с использованием формулы (2.21):

$$\theta = (X^T X + \lambda L)^{-1} X^T Y, \tag{2.21}$$

Доказано, что матрица $X^T X + \lambda L$ всегда обратима, даже если для матрицы $X^T X$ не существует обратной (при условии, что L имеет вид, приведенный выше).

2.1.6 Сравнение аналитического и численного решения

Приведем таблицу, в котором отметим основные плюсы и минусы Нормального уравнения и градиентного спуска:

Нормальное уравнение	**Градиентный спуск**
Необходимость выбирать скорост обучения α	Не нужно выбирать скорость обучения
Нужно много итераций для сходимости	Не нужно итераций
Достаточно быстро работает, даже есл очень много признаков	Медленно работает, если числ признаков очень большое (например 10000)

Таблица 2.1 Сравнение Нормального уравнения и градиентного спуска

Также можно отметить, что при использовании Нормального уравнения не нужно масштабировать признаки.

Градиентный спуск не единственный метод для численного решения. Существуют и другие алгоритмы, которые зачастую работают быстрее и в которых не нужно выбирать скорость обучения α, например Метод сопряженный градиентов (Conjugate gradients), BFGS, L-BFGS, однако реализовывать их намного сложнее.

2.2 Построение графиков при помощи библиотеки Matplotlib

На основе [6] рассмотрим библиотеку Matplotlib, также сразу оговоримся, что для улучшения «стиля» графиков установим тему пакета Seaborn. Это можно сделать при помощи команды import seaborn as sns; sns.set(). Пример кода также возьмем с [6].

С библиотекой Matplotlib обычно требуется меньше ста строк кода, чтобы нарисовать графики, которые по качеству подходят для использования в продакшене.

И тем не менее, Matplotlib очень большая библиотека, применение которой обычно заключается в рисовании графиков, которые выглядит “достаточно хорошо”. Однострочные команды для создания базовых графиков в Matplotlib довольно прозрачны, но оставшиеся 98% возможностей библиотеки выглядят обескураживающими.

В данной работе приведены приемы, которые позволяют повысить уровень владения библиотекой с начального до среднего. Присутствует и теория, и

практика. Тщательное рассмотрение конкретного примера может быть чрезвычайно полезным, поскольку оно помогает хотя бы на поверхностном уровне понять внутренние процессы, которые происходят в библиотеке.

Рассмотрены следующие вещи:

- Pylab и pyplot: что есть что?
- Ключевые концепты в дизайне Matplotlib
- Рисование совместных графиков
- Визуализация массивов в Matplotlib

2.2.1 Почему Matplotlib может сбивать с толку?

Временами изучение Matplotlib может разочаровывать. Проблема не в том, что не хватает документации, ее то как раз с избытком. Вот, что может представлять основные сложности:

- Библиотека сама по себе огромна, в общей сложности примерно 70000 строк кода
- Matplotlib объединяет в себе несколько интерфейсов (способы построения графиков) и способен взаимодействовать большим наборов бэкендов (Бэкенды отвечают за рендеринг графиков).
- Удивительно, но часть документации, которая находится в открытом доступе серьёзно устарела. https://matplotlib.org/users/shell.html. Библиотека все еще развивается. И для многих старых примеров, приведенных в сети, может потребоваться на 70% меньше строк кода в новой версии.

Прежде чем перейти к невероятным примерам, полезно будет ознакомиться с концептами, при помощи которых был создан Matplotlib.

2.2.2 Pylab: что это такое и стоит ли это использовать?

Создатель Matplotlib при начале его разработки в 2003 году, был вдохновлен тем, как выполняются команды в Matlab. Одна из черт Matlab это глобальный стиль. Концепция импорта в Python сильно отличается от той, что используется в Matlab, где большинство функций доступны пользователя уже на верхнем уровне.

Знание того, что Matplotlib корнями уходит в Matlab, позволяет понять. Почему существует pylab. Pylab является модулем внутри Matplotlib, который был создан, чтобы имитировать глобальный стиль Matlab. Он существует только для того, чтобы перенести некоторые функции и классы из Numpy и Matplotlib в пространство имен, обеспечив тем самым безболезненный переход бывшим пользователям Matlab, которые не привыкли к использованию выражений с import. Экс-пользователям Matlab нравилось эта функциональность, поскольку после

команды from pylab import * они могли напрямую вызвать функции plot() или array(), как они бы сделали это в Matlab.

Некоторые люди. использующие Python, могли заметить, в чем здесь проблема: использование команды from pylab import * в скрипте как правило является плохой практикой. Внутри может быть огромное количество потенциально конфликтующих импортов. Также полезно помнить, что использование ipython –pylab (в терминале или командной строке) или %pylab (Ipython/Jupyter) просто вызывает from pylab import *. Matplotlib прекратила поддержку этого модуля и явно рекомендует его не использовать, следуя одной из концепций Python: “явное лучше, чем неявное”.

Вместо вызова pylab обычно можно обойтись одной строчкой импорта [6]:

```
Python

>>> import matplotlib.pyplot as plt
```

2.2.3 Иерархия объектов в Matplotlib

Одним из самых важных концептов в Matplotlib является иерархия объектов.

При освоении Matplotlib многие, наверняка, использовали команды вида plt.plot([1,2,3]). Уже одна эта строка показывает, что plot на самом деле представляет собой иерархию вложенных объектов Python. В данном контексте «иерархия» означает, что существует древовидная структура объектов, лежащая в основе каждого графика.

Figure является самым отдаленным контейнером для графики Matplotlib, который может содержать множество Axes объектов. Название может вносить некую путаницу, поскольку Axes на самом деле преобразуется в то, что мы считаем отдельным графиком (вместо «axis» во множественном числе, как можно было бы ожидать).

Можно считать Figure объектом наподобие ящика, контейнера, содержащим один или больше Axes (графиков). Ниже Axes в иерархии располагаются меньшие объекты, такие как tick marks (подписи осей), individual lines (отдельные линии), legends (легенды) и text boxes (участки для текста), Почти каждый элемент графика является собственным манипулируемым Python-объектом, вплоть до подписи и маркировки осей [6]:

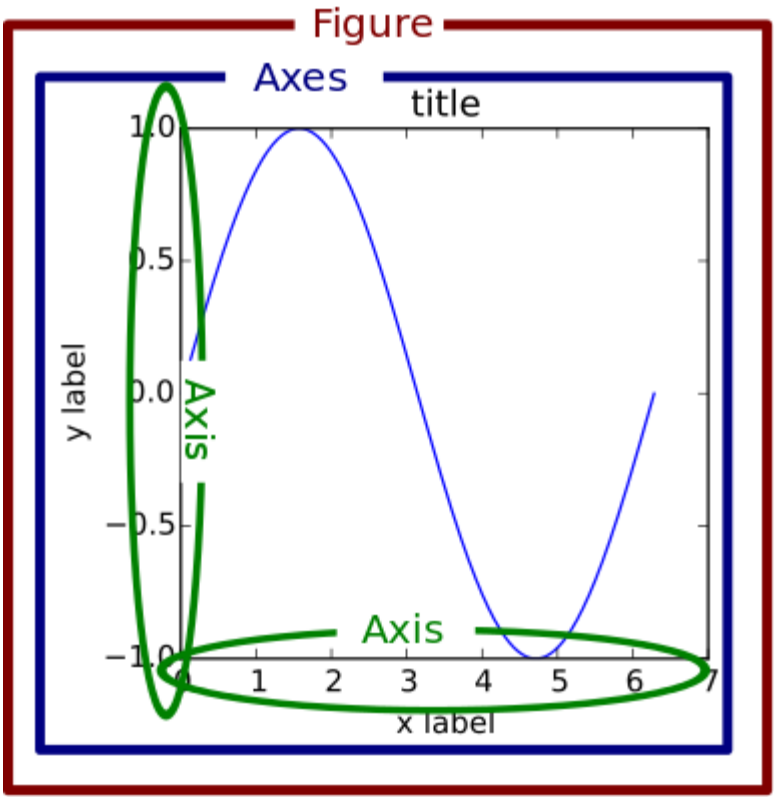

Рисунок 2.1 Иерархия в графике Matplotlib

Ниже приведен пример, показывающий иерархию в действии [6].

Python

```
>>> fig, _ = plt.subplots()
>>> type(fig)
<class 'matplotlib.figure.Figure'>
```

Выше при помощи команды plt.subplots() были созданы две переменные. Первая из них является Figure — объектом высшего уровня; вторая же переменная нам пока не понадобится, поэтому вместо нее используется простое подчеркивание. Используя атрибуты, можно легко двигаться вниз по иерархии, и увидеть первый tick оси y у первого Axes-объекта [6]:

Python

```
>>> one_tick = fig.axes[0].yaxis.get_major_ticks()[0]
>>> type(one_tick)
<class 'matplotlib.axis.YTick'>
```

Выше, fig (объект класса Figure) имеет несколько Axes (список, у которого берется первый элемент). У каждого Axes-объекта есть yaxis и xaxis, каждый из которых имеет набор «major ticks», из которых извлекается первый элемент.

Для того, чтобы лучше понять строение Figure, полезна будет следующая картинка, [6]:

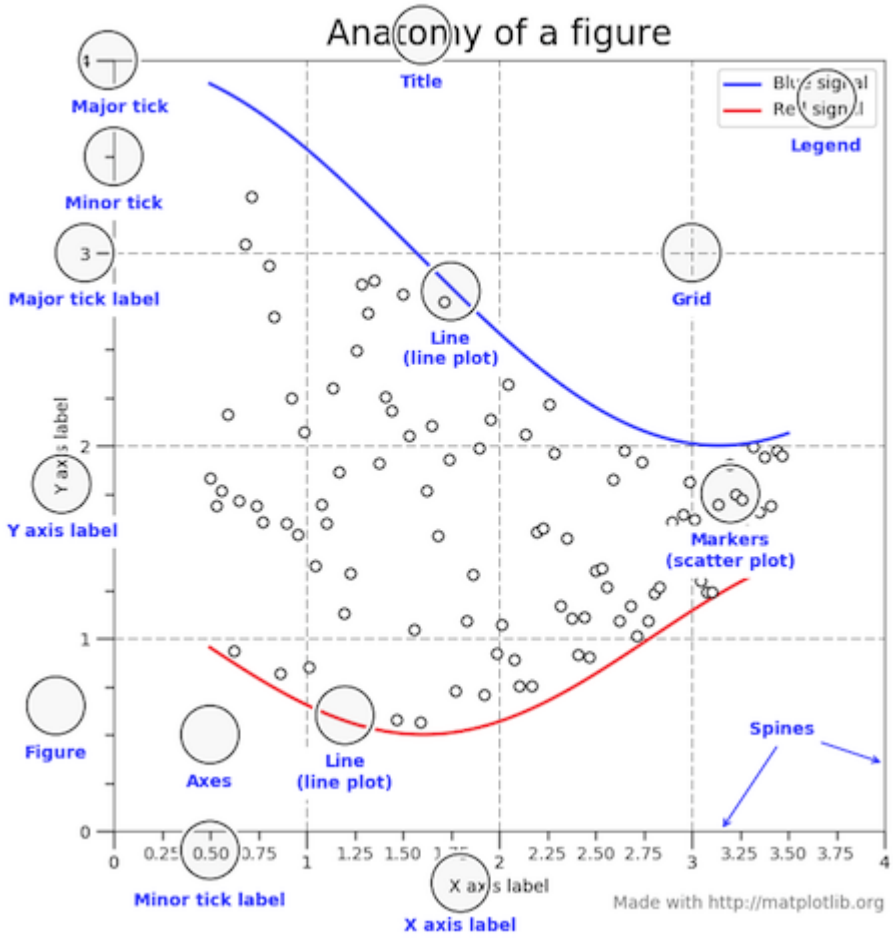

Рисунок 2.2 Строение Figure в Matplotlib

2.2.4 Фиксированный и объектно-ориентированный интерфейсы

Остался последний кусочек теории, перед тем как можно будет приступить к созданию прекрасных графиков: различия между stateful (с фиксированным состоянием) и stateless (объектно-ориентированный) интерфейсами.

Выше использовалась команда import matplotlib.pyplot as plt, чтобы импортировать pyplot модуль из matplotlib и назвать его plt.

Почти все функции из pyplot, такие как plt.plot(), неявно или относятся к существующим и текущим Figure и Axes, или создают их, если таковых не существует. В документации Matplotlib есть полезная подсказка, которая говорит,

что простые функции используются для добавления элементов графика (линии, текст и т.д.) *к текущим Axes и Figure*.

Что это значит:

- Stateful интерфейс вызывается с plt.plot() и другими функциями pyplot высшего уровня. В конкретный момент можно манипулировать лишь одним Figure или Axes, без необходимости явно к ним обращаться.
- Модификация базовых объектов напрямую является объектно-ориентированным подходом. Обычно это делается при помощи методов Axes объекта, который является объектом, представляющим график.

Объединяя эти утверждения, можно сказать, что большинство функций из pyplot также являются методами класса matplotlib.axes.Axes.

Чтобы лучше это понять, приведем реализацию функции plt.plot() [6]:

Python

```
# matplotlib/pyplot.py
>>> def plot(*args, **kwargs):
...     """An abridged version of plt.plot()."""
...     ax = plt.gca()
...     return ax.plot(*args, **kwargs)

>>> def gca(**kwargs):
...     """Get the current Axes of the current Figure."""
...     return plt.gcf().gca(**kwargs)
```

Можно видеть, что вызов plt.plot() является по сути удобным способом получения текущего Axes текущей Figure, а затем для него вызывается метод plot(). Это то, что понимается под фразой, что stateful-интерфейс всегда «неявно отслеживает» график, к которому он будет обращаться.

Pyplot является домом для большого числа функций, которые на самом деле являются всего лишь обертками (wrappers) над объектно-ориентированным интерфейсом matplotlib. Например, для команды plt.title() есть соответствующие setter и getter в ОО-подходе: ax.set_title() и ax.get_title().

При вызове plt.title() трансформируется в одну строчку: gca().set_title(s, *args, **kwargs), а делает она следующее:

- gca() захватывает и возвращает текущий Axes
- set_title() является setter-методом, который устанавливает название для данного Axes объекта. Удобство здесь заключается в том, что нам не нежно явно указывать объект Axes при вызове plt.title().

Точно так же, если взглянуть на код в исходниках для функций высшего уровня, например, plt.grid(), plt.legend() и plt.ylabels(), можно заметить, что все из

них следуют такой же структуре получение текущего Axes при помощи gca(), а затем вызывая некоторые методы для текущего Axes.

2.2.5 Подграфики

Приведенной выше теории достаточно для того, чтобы начать строить красивые графики. С этого момента будем полагаться на объектно-ориентированный подход, который является более кастомизируемым, а также становится более удобным, в то время как графики становятся сложнее.

Правильно создавать Figure с одним Axes при ОО-подходе с использованием plt.subplots(). (Это единственный раз, когда ОО подход использует pyplot для создания Figure и Axes) [6].

```
Python

>>> fig, ax = plt.subplots()
```

Выше использовалось преимущество множественной распаковки для присваивания разным переменным значений двух результатов команды plt.subplots(). В subplots не передавались никакие переменные; по умолчанию вызывается subplots(nrows=1, ncols=1). Следовательно, ax — один объект типа AxesSubplot [6]:

```
Python

>>> type(ax)
<class 'matplotlib.axes._subplots.AxesSubplot'>
```

и можно вызвать его методы для манипулирования графиком, точно так же, как вызывались бы функции pyplot. Это можно проиллюстрировать при помощи графика типа stacked-area для трех временных рядов [6]:

```
Python

>>> rng = np.arange(50)
>>> rnd = np.random.randint(0, 10, size=(3, rng.size))
>>> yrs = 1950 + rng

>>> fig, ax = plt.subplots(figsize=(5, 3))
>>> ax.stackplot(yrs, rng + rnd, labels=['Eastasia', 'Eurasia', 'Oceania'])
>>> ax.set_title('Combined debt growth over time')
>>> ax.legend(loc='upper left')
>>> ax.set_ylabel('Total debt')
>>> ax.set_xlim(xmin=yrs[0], xmax=yrs[-1])
>>> fig.tight_layout()
```

Что же делает данный код?

- После создания трех случайных временных рядов, мы определяем один объект Figure (fig), который содержит один Axes (график, ax)
- Мы вызываем методы ax для создания графика и добавления легенды, названия и подписи y-оси. При использовании OO-подхода ясно, что все это является атрибутами ax.
- tight_layout() применяется к Figure объекту в целом для удаления ненужных пробелов.

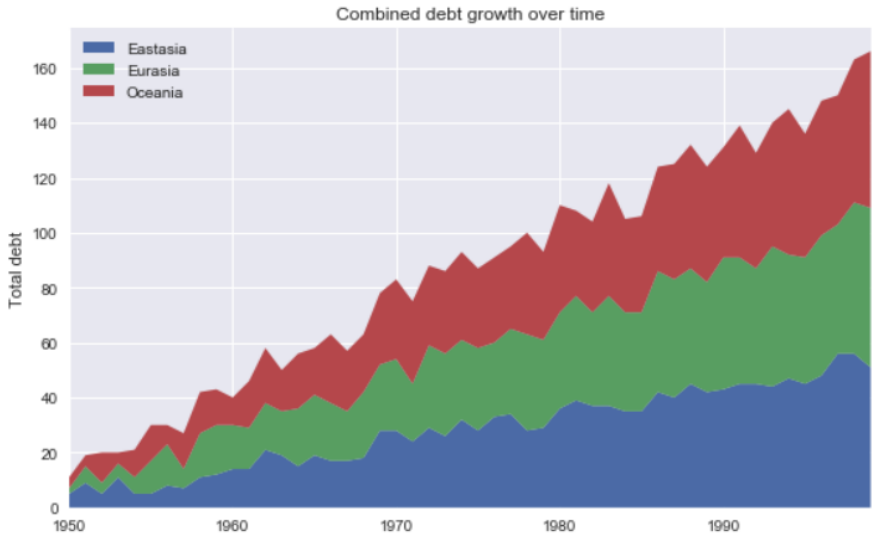

График 2.1 Пример stackplot-графика в Matplotlib

Давайте посмотрим на пример с несколькими subplot (Axes) внутри одной Figure, нарисовав два коррелирующих массива, полученных из дискретного равномерного распределения [6]:

Python

```
>>> x = np.random.randint(low=1, high=11, size=50)
>>> y = x + np.random.randint(1, 5, size=x.size)
>>> data = np.column_stack((x, y))

>>> fig, (ax1, ax2) = plt.subplots(nrows=1, ncols=2,
                                   figsize=(8, 4))

>>> ax1.scatter(x=x, y=y, marker='o', c='r', edgecolor='b')
>>> ax1.set_title('Scatter: $x$ versus $y$')
>>> ax1.set_xlabel('$x$')
>>> ax1.set_ylabel('$y$')

>>> ax2.hist(data, bins=np.arange(data.min(), data.max()),
             label=('x', 'y'))
>>> ax2.legend(loc=(0.65, 0.8))
>>> ax2.set_title('Frequencies of $x$ and $y$')
>>> ax2.yaxis.tick_right()
```

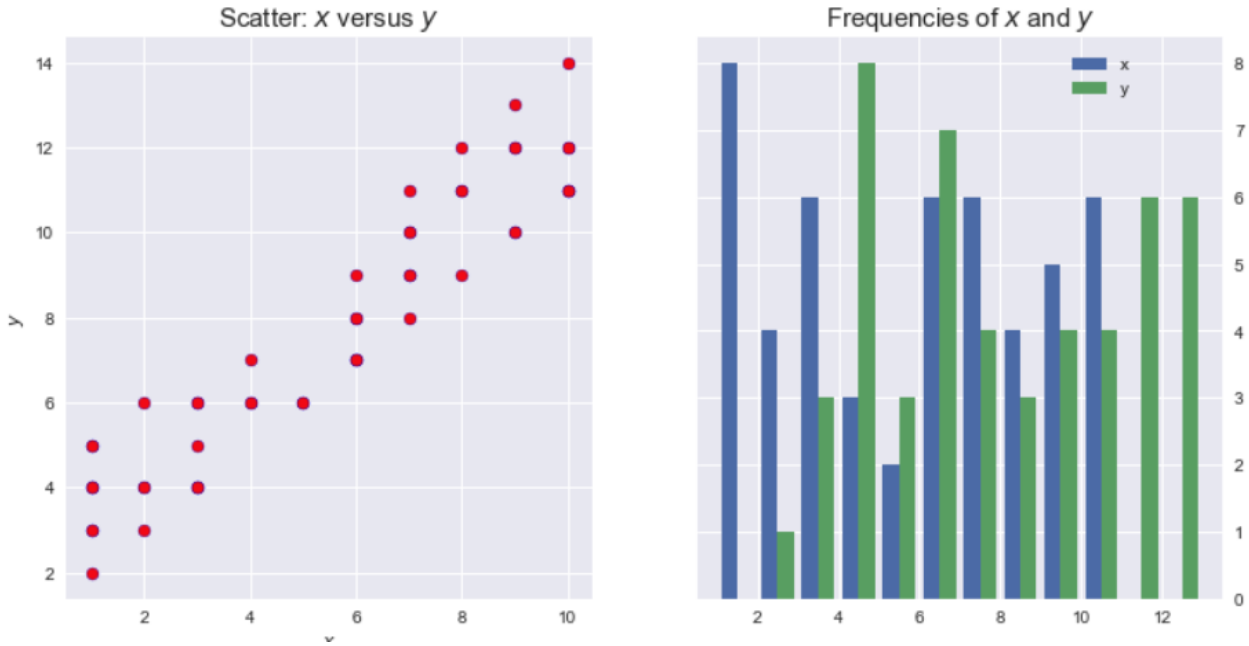

График 2.2 Пример scatter plot и hist в Matplotlib

В данном примере происходит немного больше:

- Поскольку мы создаем Figure размера 1x2, возвращаемый результат для команды plt.subplots(1,2) является объект Figure и Numpy массив из Axes объектов.

- Мы работаем с ax1 и ax2 по отдельности. (Было бы сложно делать это при stateful-подходе.) Последняя строка является хорошим примером иерархии объектов, где мы изменяем yaxis, принадлежащий ax2, помещая его справа.
- Текст внутри знака доллара использует TeX разметку, чтобы сделать переменные курсивными.

Полезно помнить, что множественные Axes могут быть заключены в данную фигуру; в данном случае, fix.ages получает список всех Axes объектов [6]:

Python

```
>>> (fig.axes[0] is ax1, fig.axes[1] is ax2)
(True, True)
```

На следующем шаге мы могли бы создать фигуру, которая содержит сетку размера 2x2 из Axes объектов [6]:

Python

```
fig, ax = plt.subplots(nrows=2, ncols=2, figsize=(7, 7))
```

Что же теперь из себя представляет ax? Это больше не один объект Axes, а двумерный numpy массив из них [6]:

Python

```
>>> type(ax)
numpy.ndarray

>>> ax
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x1106daf98>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x113045c88>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x11d573cf8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x1130117f0>]],
      dtype=object)

>>> ax.shape
(2, 2)
```

Нам нужно вызывать методы для каждого Axes объекта. Общепринятым способом сделать это является использование множественной распаковки после превращения массива в одномерный [6]:

Python

```
>>> fig, ax = plt.subplots(nrows=2, ncols=2, figsize=(7, 7))
>>> ax1, ax2, ax3, ax4 = ax.flatten()  # flatten a 2d NumPy array to 1d
```

Можно было сделать это и при помощи ((ax1, ax2), (ax3, ax4)) = ax, но первый способ выглядит удобнее.

Чтобы продемонстрировать еще больше особенностей subplot, используем данные о домах в Калифорнии: среднюю цену, количество проживающих людей, средний возраст.

Определим также функцию, которая будет располагать текст внутри графика и будет своего рода внутренней подписью [6]:

```
Python

>>> def add_titlebox(ax, text):
...     ax.text(.55, .8, text,
...         horizontalalignment='center',
...         transform=ax.transAxes,
...         bbox=dict(facecolor='white', alpha=0.6),
...         fontsize=12.5)
...     return ax
```

Модуль для matplotlib gridspec добавляет больше возможностей кастомизации подграфиков; subplot2grid() из pyplot отлично с ним взаимодействует. Допустим, нужно создать графики в таком виде [6]:

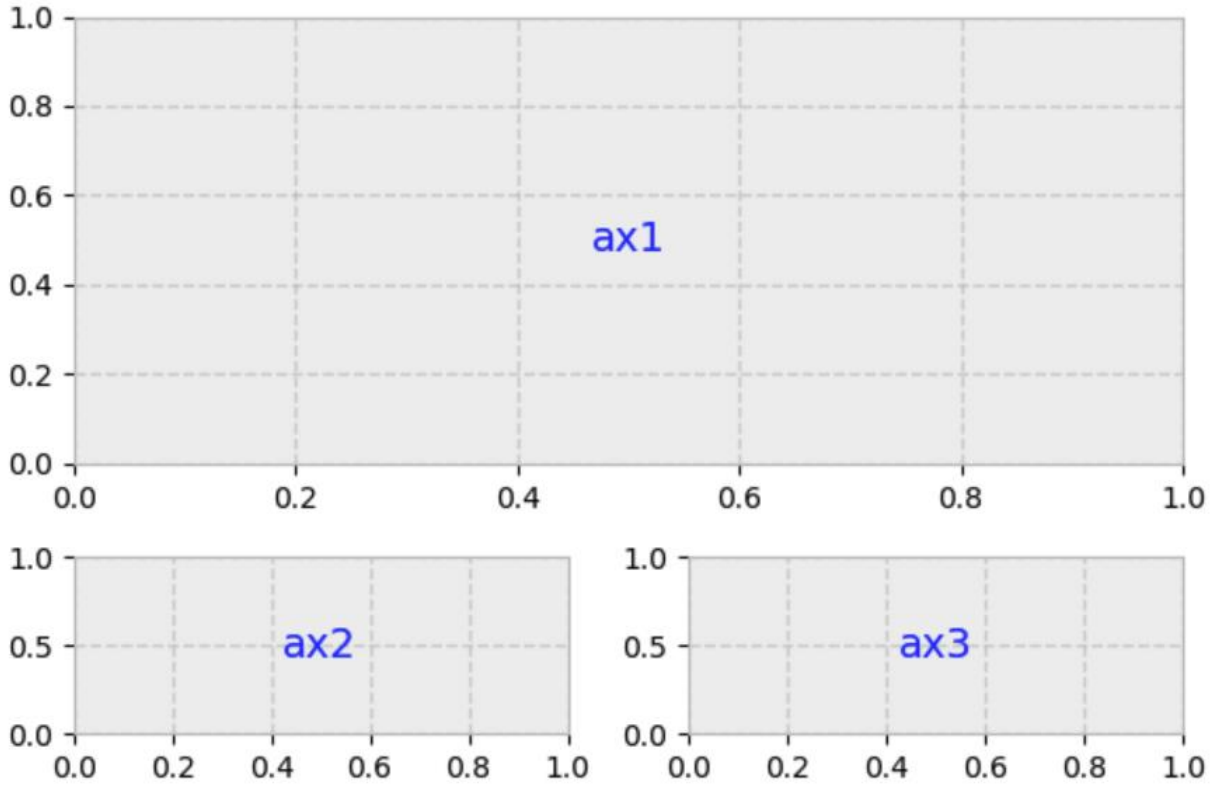

Рисунок 2.3 Пример расположения графиков в Matplotlib

Заготовка для графиков по сути представляет собой сетку 3x2, где ax1 в два раза выше и шире ax2/ax3 [6]:

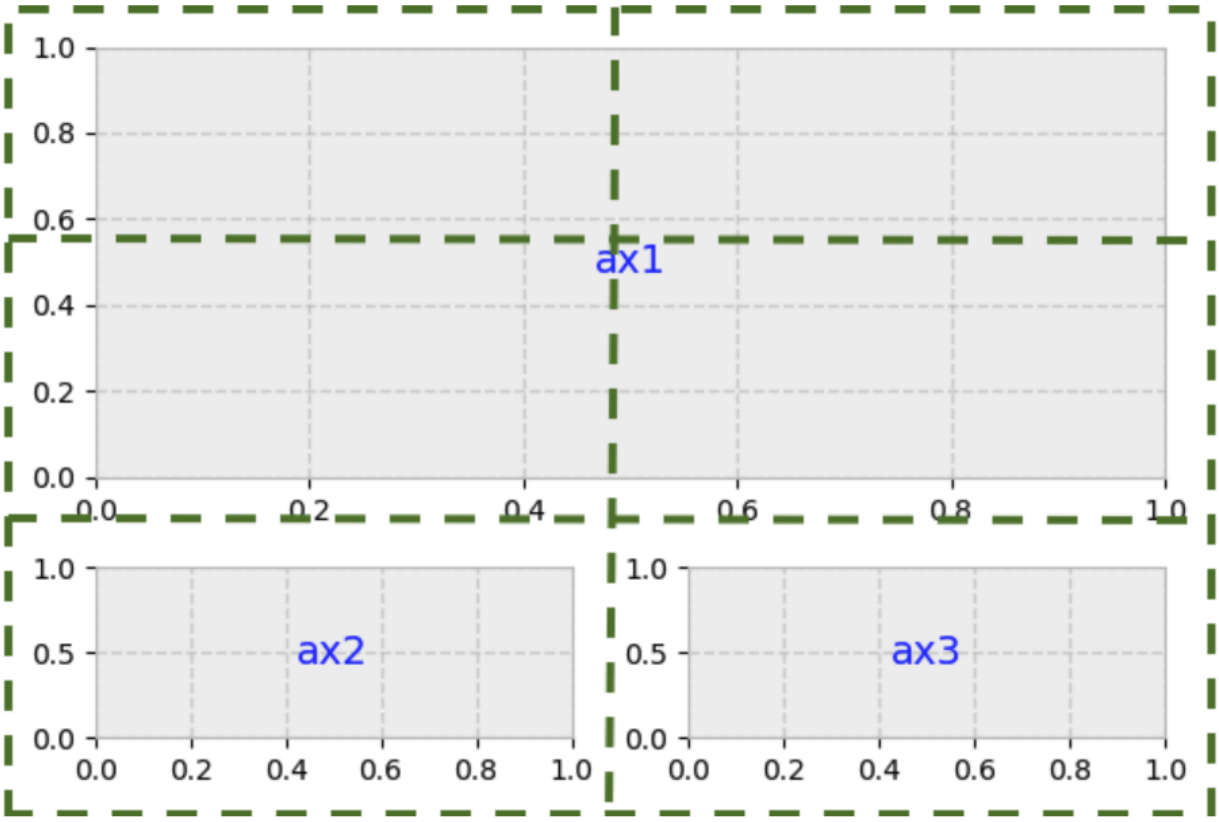

Рисунок 2.4 Разделение области графиков в Matplotlib

Второй аргумент для subplot2grid() — это (row, column) положение Axes внутри сетки [6]:

Python

```
>>> gridsize = (3, 2)
>>> fig = plt.figure(figsize=(12, 8))
>>> ax1 = plt.subplot2grid(gridsize, (0, 0), colspan=2, rowspan=2)
>>> ax2 = plt.subplot2grid(gridsize, (2, 0))
>>> ax3 = plt.subplot2grid(gridsize, (2, 1))
```

Теперь можно действовать, как и раньше: работая с каждый графиком по отдельности [6]:

Python

```
>>> ax1.set_title('Home value as a function of home age & area population',
...               fontsize=14)
>>> sctr = ax1.scatter(x=age, y=pop, c=y, cmap='RdYlGn')
>>> plt.colorbar(sctr, ax=ax1, format='$%d')
>>> ax1.set_yscale('log')
>>> ax2.hist(age, bins='auto')
>>> ax3.hist(pop, bins='auto', log=True)

>>> add_titlebox(ax2, 'Histogram: home age')
>>> add_titlebox(ax3, 'Histogram: area population (log scl.)')
```

График 2.3 Использование subplot в Matplotlib на примере scatter plot и hist

В данном примере colorbar() вызывается прямо на фигуре (Figure),а не на графике (Axes). Первый аргумент — результат ax1.scatter(), который передает значение переменной y в ColorMap.

Визуально не наблюдается особой разницы в цвете (переменная y) при движении вдоль y-оси, — это говорит о том, что цена сильнее зависит от возраста дома.

2.2.6 Imshow() и matshow()

Хотя ax.plot() и является одним из самых популярных методов для графика (Axes), есть еще и много других, например, bar(), pie(), scatter(), hist() и др.

Одной из групп методов, которые также очень широко используются являются imshow() и matshow(), причем второй является оберткой для первого. Они оказываются полезными, когда числовой массив нужно визуализировать в виде цветной сетки.

Сперва создадим две отчетливые сетки с интересной numpy-индексацией [6]:

```
Python
>>> x = np.diag(np.arange(2, 12))[::-1]
>>> x[np.diag_indices_from(x[::-1])] = np.arange(2, 12)
>>> x2 = np.arange(x.size).reshape(x.shape)
```

На следующем шаге мы можем отобразить их. В данном конкретном случае мы «выключим» все оси и подписи к ним, используя словари, а результат передадим в ax.tick_params() [6]:

```
Python
>>> sides = ('left', 'right', 'top', 'bottom')
>>> nolabels = {s: False for s in sides}
>>> nolabels.update({'label%s' % s: False for s in sides})
>>> print(nolabels)
{'left': False, 'right': False, 'top': False, 'bottom': False, 'labelleft': False,
 'labelright': False, 'labeltop': False, 'labelbottom': False}
```

Далее можно использовать контекстный менеджер, чтобы отключить сетку на графике, и вызвать matshow() для каждого Axes объекта. И наконец добавим colorbar [6]:

```
Python

>>> from mpl_toolkits.axes_grid1.axes_divider import make_axes_locatable

>>> with plt.rc_context(rc={'axes.grid': False}):
...     fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(8, 4))
...     ax1.matshow(x)
...     img2 = ax2.matshow(x2, cmap='RdYlGn_r')
...     for ax in (ax1, ax2):
...         ax.tick_params(axis='both', which='both', **nolabels)
...     for i, j in zip(*x.nonzero()):
...         ax1.text(j, i, x[i, j], color='white', ha='center', va='center')
...
...     divider = make_axes_locatable(ax2)
...     cax = divider.append_axes("right", size='5%', pad=0)
...     plt.colorbar(img2, cax=cax, ax=[ax1, ax2])
...     fig.suptitle('Heatmaps with `Axes.matshow`', fontsize=16)
```

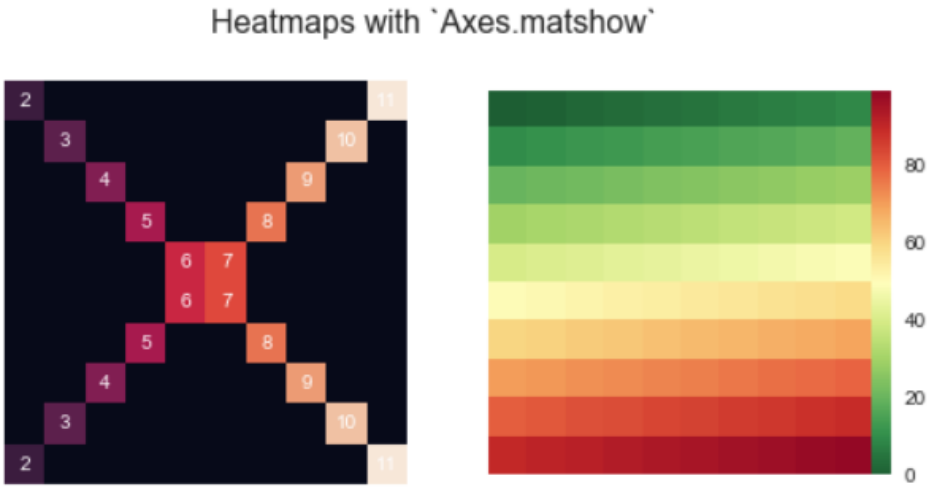

График 2.4 Пример matshow графиков в Matplotlib

2.3 Выводы по главе

Линейная регрессия и различные ее модификации являются достаточно простыми, но в то же время очень мощными алгоритмами, которые можно использовать для решения широкого класса задач.

Matplotlib является библиотекой с небольшим порогом входа, однако при должном использовании позволяет добиться впечатляющих результатов в визуализации различных видов данных.

ГЛАВА 3
СОЗДАНИЕ AVTH МОДЕЛИ И ПОСТРОЕНИЕ ПРОГНОЗОВ НА ЕЁ ОСНОВЕ

3.1 Определение модели

Моделирование кредитных рисков связано с понятием вероятности дефолта кредитов при заданных значениях некоторых параметров. Определим вероятность дефолта по формуле (3.1):

$$PD(t,h,v) = \frac{Def(t,v)}{Active(t-h,v)} \tag{3.1}$$

где

- PD — вероятность дефолта при заданных значениях t, h, v
- Def — количество дефолтов
- $Active$ — количество активных счетов
- t — время
- v — винтаж, время, когда были открыт счета
- h — горизонт, параметр, показывающий на сколько отсчетов (горизонтов) вперед строится прогноз

Будем строить декомпозицию формулы (3.1) в виде (3.2):

$$PD(t,h,v) \approx F(a) + G(v) + H(t) + K(h) \tag{3.2}$$

где

- a — возраст аккаунтов,
- t, v, h — время, винтаж, горизонт
- $F(a), G(v), H(t), K(h)$ — функции возраста, винтажа, времени и горизонта соответственно

3.2 Классическая модель

Классическую AVT-модель в матричном виде имеет вид, определяемый формулами (3.3-3.6):

$$Xb = y \tag{3.3}$$

$$b^T = [\alpha_1 \dots \alpha_{a_{max}} | \beta_1 \dots \beta_{t_{max}} | \gamma_1 \dots \gamma_{v_{max}}] \tag{3.4}$$

$$X = [D_{age} | D_{time} | D_{vintage}] \tag{3.5}$$

где матрица D_{age} имеет вид:

$$D_{age,i,j} = \begin{cases} 1, age_j = j \\ 0, \text{иначе} \end{cases} \quad (3.6)$$

$i = 1 \dots n$, n — количество наблюдений, $j = 1 \dots a_{max}$. a_{max}— максимальный возраст аккаунта. Матрицы $D_{time}, D_{vintage}$ определяются аналогичным образом.

Данную модель легко обобщить на случай нового параметра горизонта: достаточно добавить к матрице X матрицу для горизонта $D_{horizon}$, которая определяется аналогично матрицам для других параметров, а именно по формуле (3.7):

$$D_{horizon,i,j} = \begin{cases} 1, horizon_j = j \\ 0, \text{иначе} \end{cases} \quad (3.7)$$

В итоге AVTH-модель в матричном виде можно записать в виде (3.8-3.10):

$$Xb = y \quad (3.8$$

$$b^T = [\alpha_1 \dots \alpha_{a_{max}} | \beta_1 \dots \beta_{t_{max}} | \gamma_1 \dots \gamma_{v_{max}} | \varphi_1 \dots \varphi_{h_{max}}] \quad (3.9$$

$$X = [D_{age} | D_{time} | D_{vintage} | D_{horizon}] \quad (3.10$$

В которой матрицы $D_{age}, D_{time}, D_{vintage}, D_{horizon}$ определяются по формулам выше, а a_{max} — максимальный возраст аккаунта, t_{max} — максимальное время наблюдения, v_{max} — максимальный винтаж аккаунта, h_{max} − максимальный горизонт.

3.3 Модель на основе B-сплайнов

В классической модели значения функций $F(a), G(v), H(t), K(h)$ определяются очень точно: для каждого значения параметра, который присутствует в данных, вычисляется их значения. Например, для возраста функция $F(a)$ определяется по формуле (3.11):

$$F(a) = \begin{cases} \alpha_1, a = 1 \\ \alpha_2, a = 2 \\ \vdots \\ \alpha_m, a = m \end{cases} = \sum_i F(a_i) I(a = a_i) \quad (3.11$$

Однако у данного подхода есть недостатки: из-за неравномерности данных полученные значения для соседних значений параметров могут сильно отличаться, в итоге их графики становится довольно сложно анализировать и

интерпретировать. Чтобы избавиться от этих проблем представим функции $F(a), G(v), H(t), K(h)$ в виде линейно комбинации B-сплайнов соответствующей степени (формулы 3.12-3.15):

$$F(a) = \sum_{i=1}^{n_{age}} \alpha_i S_i(a) \quad (3.12$$

$$G(v) = \sum_{i=1}^{n_{vintage}} \beta_i S_i(v) \quad (3.13$$

$$H(t) = \sum_{i=1}^{n_{time}} \gamma_i S_i(t) \quad (3.14$$

$$K(h) = \sum_{i=1}^{n_{horizon}} \varphi_i S_i(h) \quad (3.15$$

где

- n_{age} — количество сплайнов для $F(a)$
- $n_{vintage}$ — количество сплайнов для $G(v)$
- n_{time} — количество сплайнов для $H(t)$
- количество сплайнов для $K(h)$
- $\alpha_i, \beta_i, \gamma_i, \varphi_i$ — параметры, которые необходимо определить

Такое представление дает следующий эффект: во-первых, как говорилось выше, мы сглаживаем функции, что упрощает их анализ, а во-вторых мы существенно уменьшаем размер матрицы X.

3.4 Модель на основе B-сплайнов в матричном виде

При переходе к B-сплайнам модель внешне не сильно меняется, ее по-прежнему удобно записывать в матричном виде (формулы 3.16-3.18):

$$Xb = y \quad (3.16)$$

$$b^T = \left[\alpha_1 \dots \alpha_{n_{age}} | \beta_1 \dots \beta_{n_{time}} | \gamma_1 \dots \gamma_{n_{vintage}} | \varphi_1 \dots \varphi_{h_{horizon}}\right] \quad (3.17)$$

$$X = [D_{age} | D_{time} | D_{vintage} | D_{horizon}] \quad (3.18)$$

Основное же различие кроется в построении матриц $D_{age}, D_{time}, D_{vintage}, D_{horizon}$. Приведем пример для матрицы D_{age} (формула 3.19), остальные определяются аналогичным образом:

$$D_{age} = \begin{pmatrix} S_1(a_1) & S_2(a_1) & \cdots & S_{n_{age}}(a_1) \\ S_1(a_2) & S_2(a_2) & \cdots & S_{n_{age}}(a_2) \\ \vdots & \vdots & \ddots & \vdots \\ S_1(a_n) & S_2(a_n) & \cdots & S_{n_{age}}(a_n) \end{pmatrix} \quad (3.19)$$

Или в более краткой форме по формуле (3.20):

$$D_{age,i,j} = S_i(age_j) \quad (3.20)$$

Т.е. каждый столбец представляет собой вектор значений одного сплайна, вычисленный для всех значений параметра (в данном случае возраста) из обучающей выборки.

3.5 Нормировка функций

После того, как мы решим задачу, определяемую формулами (1-3), на функции $F(a), G(v), H(t), K(h)$ необходимо будет наложить некоторые дополнительные условия, поскольку они определяются неоднозначно, например, к $F(a)$ можно добавить 1, а из $G(v)$ вычесть 1 — новые функции по-прежнему останутся решениями задачи. Накладывание таких ограничений назовём нормировкой.

Дополнительные условие будут следующими:

1) Функция $H(t)$, должна быть ортогональна линейной функции, т.е. по сути мы удаляем из нее линейный тренд
2) Линейная часть из функции $H(t)$ идет в функции $F(a), G(v)$, используя формулу $t = a + v$
3) Среднее значение функций $G(v), H(t), K(h)$ должно быть равно 0 — все константы идут в функцию $F(a)$

При помощи уравнений эти ограничения можно в виде систем формул (3.21-3.23):

$$\begin{cases} H(t) = H_1(t) + \alpha + \beta t \\ \sum_{t_k} H_1(t_k) t_k = 0 \\ \sum_{t_k} H_1(t_k) = 0 \end{cases} \quad (3.21)$$

$$\begin{cases} G_1(v) = G(v) + \beta v - c_{vintage} \\ \sum_{v_k} G_1(v_k) = 0 \end{cases} \quad (3.22)$$

$$\begin{cases} K_1(h) = K(h) - c_{horizon} \\ \sum_{h_k} K_1(h_k) = 0 \end{cases} \quad (3.23)$$

В итоге для вероятности дефолта (формула 3.1) получаем следующее представление (формула 3.24):

$$\begin{aligned} PD(t,h,v) &= F(a) + G(v) + H(t) + K(h) \quad (3.24) \\ &= F(a) + G(v) + (H_1(t) + \alpha + \beta t) + K(h) = \\ &= \left(F(a) + \alpha + \beta a + c_{time} + c_{vintage} + c_{horizon}\right) + \\ &+ \left(G(v) + \beta v - c_{vintage}\right) + H_1(t) + (K(h) - c_{horizon}) = \\ &= F_1(a) + G_1(v) + H_1(t) + K_1(h) \end{aligned}$$

3.6 Описание и группировка данных

Изначально данные имели следующий вид:

	Date	Vintage.Date	StateC	Term.segs	Default	N.Accts	Snapshot.Date	Delq.Bucket	Product
1	Feb 2003	Dec 1996	AK	(48,60]	0	1	Jan 2003	0	Auto
2	Feb 2003	Jan 1997	AK	(48,60]	0	1	Jan 2003	0	Auto
3	Feb 2003	Feb 1997	AK	(48,60]	0	4	Jan 2003	0	Auto
4	Feb 2003	Mar 1997	AK	(48,60]	0	2	Jan 2003	0	Auto
5	Feb 2003	Apr 1997	AK	(48,60]	0	6	Jan 2003	0	Auto

Таблица 3.1 Пример изначальных данных

всего было доступно 19155561 записей.

В обозначениях данной работы

- Time — Date
- Vintage — Vintage.Date
- Age — разность Date – Vintage.Date
- Horizon — разность Date – Snapshot.Date
- Def — Default
- Active — N.Accts
- Временные переменные будет измерять в месяцах

Соответственно данные необходимо было предварительно сгруппировать, поскольку изначально была слишком мелкая сегментация. Группируем следующим образом: фиксируем Date, Vintage.Date, Snapshot.Date и в каждой полученной группе считаем сумму аккаунтов и дефолтов. Поскольку данных достаточно много

удобно это делать параллельно при помощи пакета joblib, а в конце создать один dataframe с новыми параметрами. Отметим, что если использовать среду Jupyter, то функцию применения к группе (подсчет суммы дефолтов и аккаунтов) необходимо определять в отдельном модуле, и затем импортировать ее в Jupyter-ноутбуке, иначе код может не работать.

После группировки удобно привести данные к одному виду: а именно найти минимальные даты для времени и винтажа, и вычислить, насколько текущая дата в записи больше минимального значения. Также на этом этапе полезно создать столбцы для возраста и горизонта. Минимальное время — февраль 2003, минимальный винтаж — февраль 1986 года.

В итоге получаем следующие данные:

	Accounts	Defaults	Age	Horizon	Time	Vintage
0	8	0	72	2	2	134
1	8	0	72	3	2	134
2	8	0	72	1	2	134
3	17	0	60	2	2	146
4	20	0	60	3	2	146

Таблица 3.2 Пример предобработанных данных

Всего после группировки доступно 1627898 записей, т.е. количество записей уменьшилось больше, чем в 10 раз.

3.7 Разведочный анализ

Перед построением модели не лишним будет ознакомиться с доступными данными, например, нарисовать гистограмму признаков и целевой переменной.

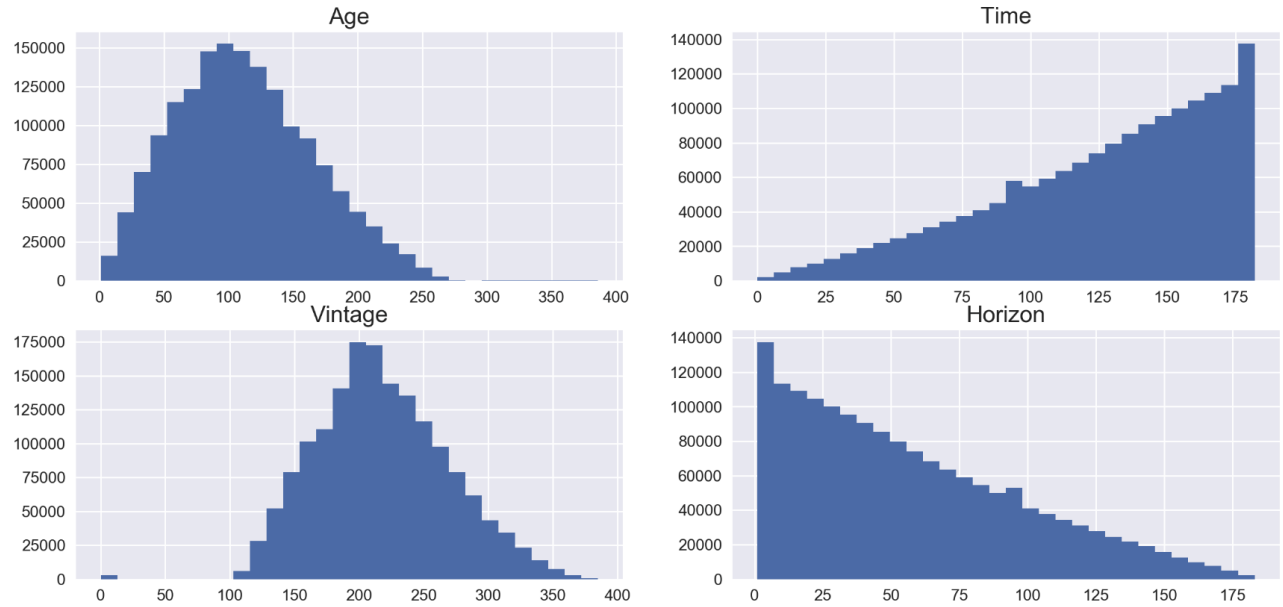

График 3.1 Гистограммы признаков в исходных данных

Поскольку существует очень много записей, где вероятность дефолта равна 0, то полезно также нарисовать график вероятности дефолта на отрезке, где сконцентрировано больше всего значений, но не считая 0:

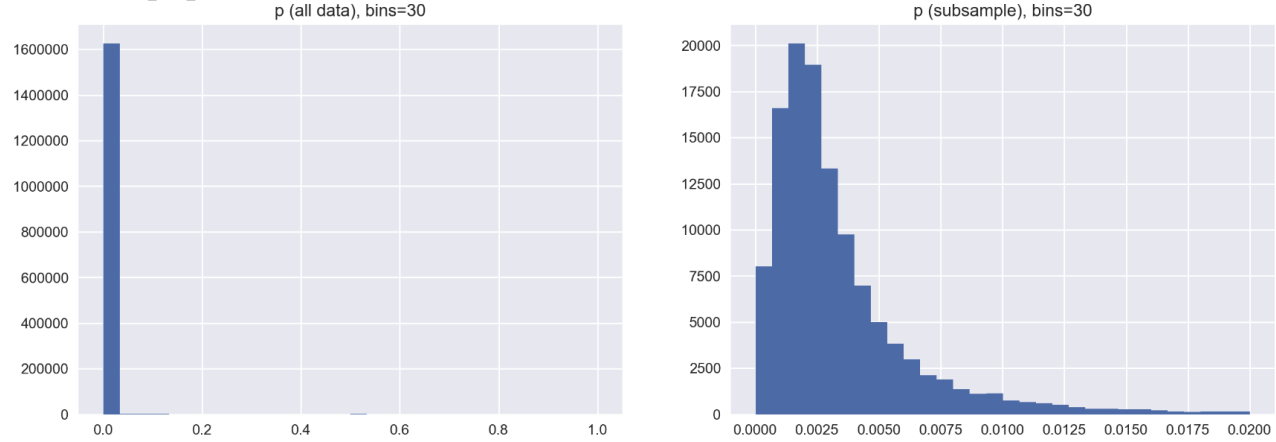

График 3.2 Гистограммы целевой переменной в исходных данных

Также графики с количеством дефолтов и аккаунтов:

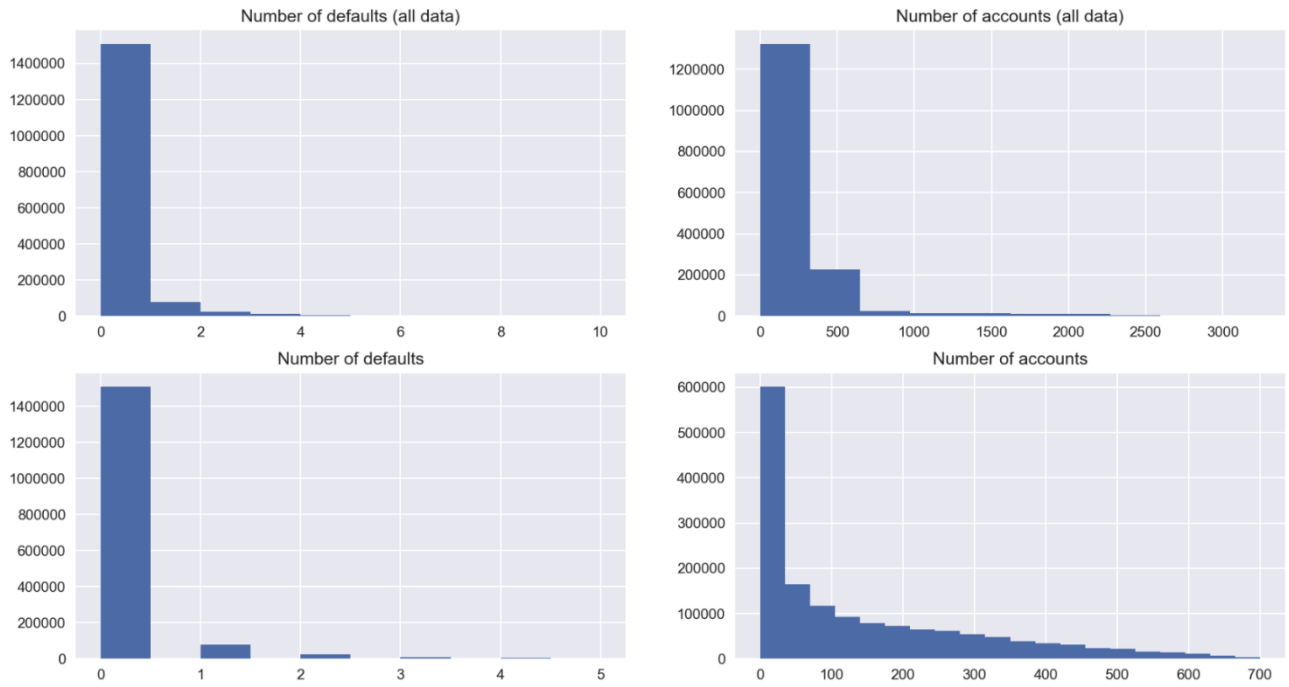

График 3.3 Гистограммы количества аккаунтов и дефолтов для исходных данных и данных, использовавшихся при построении модели

Поскольку мы будем строить достаточно актуальную модель, нам не нужно учитывать слишком старые данные. Для этого ограничимся рассмотрением записей, для которых винтаж начинается с сентября 2005 г. (винтаж больше 235), время — с июня 2006 г. (время больше 40), возраст аккаунтов меньше 100, и горизонт меньше 115 (потом данных становится очень мало, и модель может учиться на шуме).

После данных ограничений мы получаем 427358 записей, гистограммы признаков и целевой вероятности выглядят следующим образом:

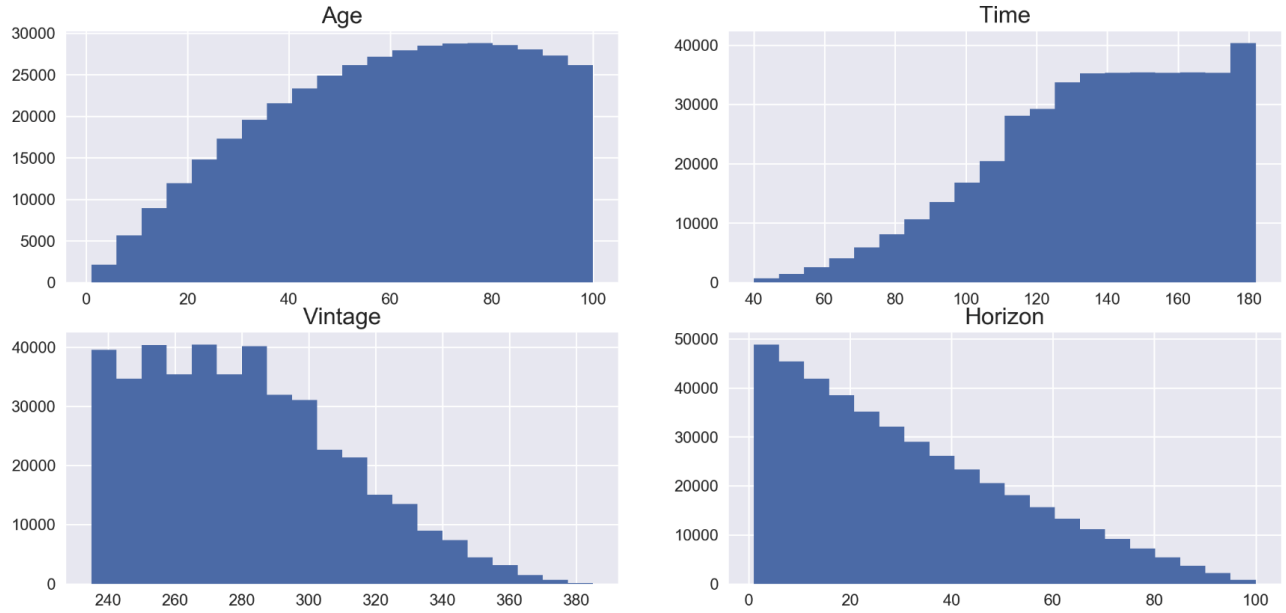

График 3.4 Гистограммы признаков в данных, использовавшихся для построения модели

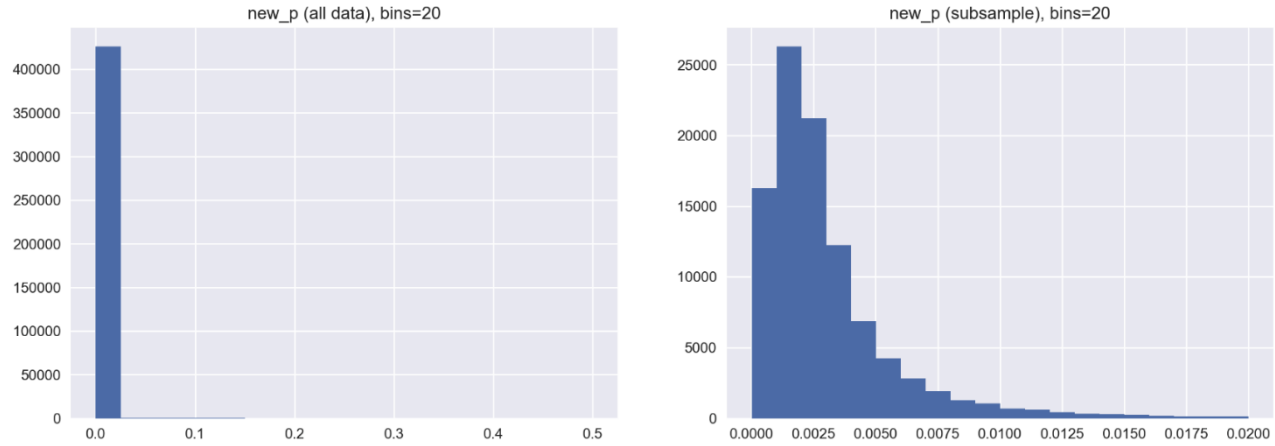

График 3.5 Гистограммы целевой переменной в данных, использовавшихся для построения модели

3.8 Описание модели

При построении модели используем B-сплайны 3-й степени, т.к. они очень хорошо себя показывают. 4-я степень не дает никаких видимых улучшений (графики и так получаются достаточно гладкие), 2-я же степень работает не очень хорошо.

Сплайны строились при помощи библиотеки scipy (модуль interpolate.BSpline): в ней реализовано векторное вычисление значений сплайна, кроме того 1 сплайн нужно использовать лишь один раз, поэтому считается все очень быстро. Например, если взять 13 сплайнов 3-й степени для возраста, матрица D_{age} вычисляется меньше, чем за 1 секунду. Важно помнить, что здесь можно использовать разряженные матрицы, хотя ненулевых элементов и несколько меньше, чем в классической модели (в ней на одну строку матрицы X четыре 1, а остальные элементы — 0).

Приведенная таблица показывает количество уникальных значений для параметров, а также сколько сплайнов использовалось для соответствующей функции, кроме того приведены размеры матриц X для классической модели и модели на основе сплайнов:

	Количество уникальны значений / параметров классической модели	Количество B-сплайнов параметров в сплайн модели
Age (возраст)	100	13
Time (время)	143	15
Vintage (винтаж)	151	15
Horizon (горизонт)	100	11
Размер матрицы X	427358 x 494	427358 x 54

Таблица 3.3 Сравнение количества параметров в моделях

Можно видеть, что количество столбцов уменьшилось почти в 10 раз, а значит регрессия будет намного быстрее считаться. Особенно прирост должен быть заметен на этапе вычисления обратной матрицы (LinearRegression в SKLearn), хотя нужно учитывать что в матрице много нулей, поэтому прирост по скорости может быть меньше ожидаемого. В Ridge-регрессии (из SKLearn) используется градиентный спуск (напрямую не считается обратная матрица).

3.9 Анализ полученных результатов

Полученный набор функций для линейной регрессии представлен на графике (….). Функция для Ridge-регрессии — на графике (…). Для Ridge-регрессии также необходимо выбрать параметр регуляризации, в данном случае он принимался равным 1,0.

По отдельности проанализируем графики функций $F(a), G(v), H(t), K(h)$.

Обе модели получили схожие графики для функции возраста, которые согласуются с результатами применения данных моделей: сначала функция некоторое время возрастает, потом возможны относительно небольшие колебания, и в конце она убывает. Это можно интерпретировать следующим образом: первые пару месяцев клиент почти всегда платит по кредиту (иначе бы ему его просто не выдали), а дальше в силу различных обстоятельств вероятность дефолта увеличивается, и наконец по прошествии определенного времени она должна начать убывать, поскольку естественно предположить, что если за кредит вносятся платежи, скажем, 10 лет, то скорее всего, они и дальше будут вноситься.

Для функции времени, которая отражает воздействие внешних факторов на вероятность дефолта, можно видеть пик в 2008-2009 гг., что согласуется с известными данными, поскольку в это время был мировой финансовый кризис, и больше счетов уходили в дефолт. Также наблюдается немного повышенное значение функции в период для 2014-2017 гг., однако оно по-прежнему меньше, чем во время кризиса.

Функции винтажа стоит более осторожно анализировать на конце, поскольку там было очень мало данных, однако если смотреть на средне-ранний период, то можно видеть, что функция довольно слабо колеблется — это согласуется с естественными предположениями, поскольку вряд ли банки очень сильно меняли политику выдачи кредитов. Интересно наблюдать за графиком, начиная с 2014 года — первое серьезное убывание функции может быть связано с тем, что банки начали применять более продвинутые метода машинного обучения для анализа платежеспособности клиента, и в дальнейшем требования немного ослабли, но

опять же для этого периода намного меньше данных, поэтому анализировать его нужно осторожно.

Функция горизонта выглядит примерно одинаковой у обеих моделей, и в целом можно сделать вывод о том, что она обучилась под то, чтобы занижать вероятность дефолта с увеличением количества горизонтов, на которое строится прогноз, т.е. она скорее считает, что люди более склонны платить по кредитам с течением времени.

В целом обе модели показали сопоставимые результаты, однако модель на основе Ridge-регрессии немного более предпочтительна, поскольку мы можем регулировать, как сильно мы хотим подстраиваться под данные — а это может оказаться достаточно полезным, учитывая то, что регрессия склонна к переобучению.

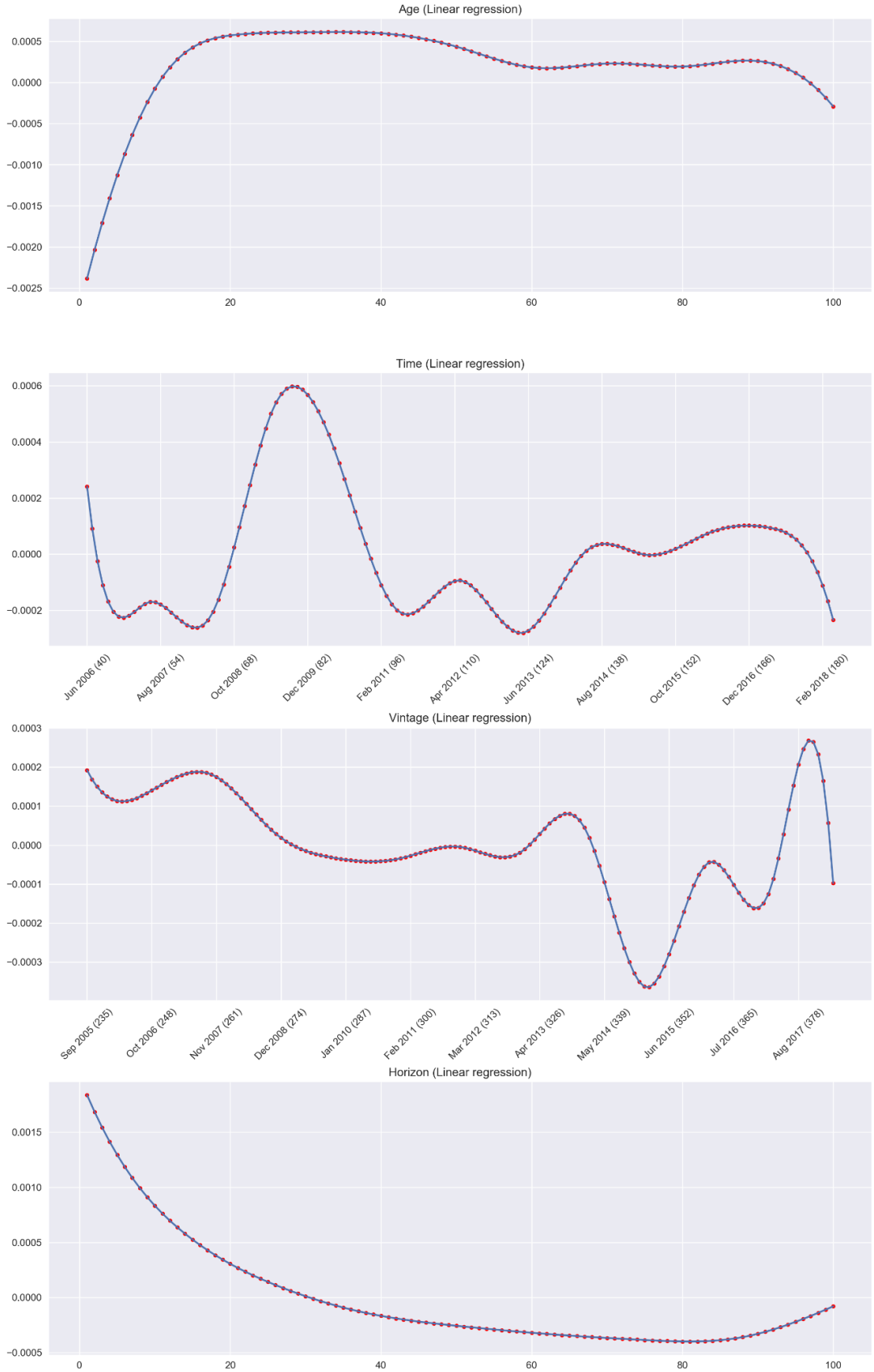

График 3.6 Графики функций F(a), H(t), G(v), K(h) для линейной регрессии

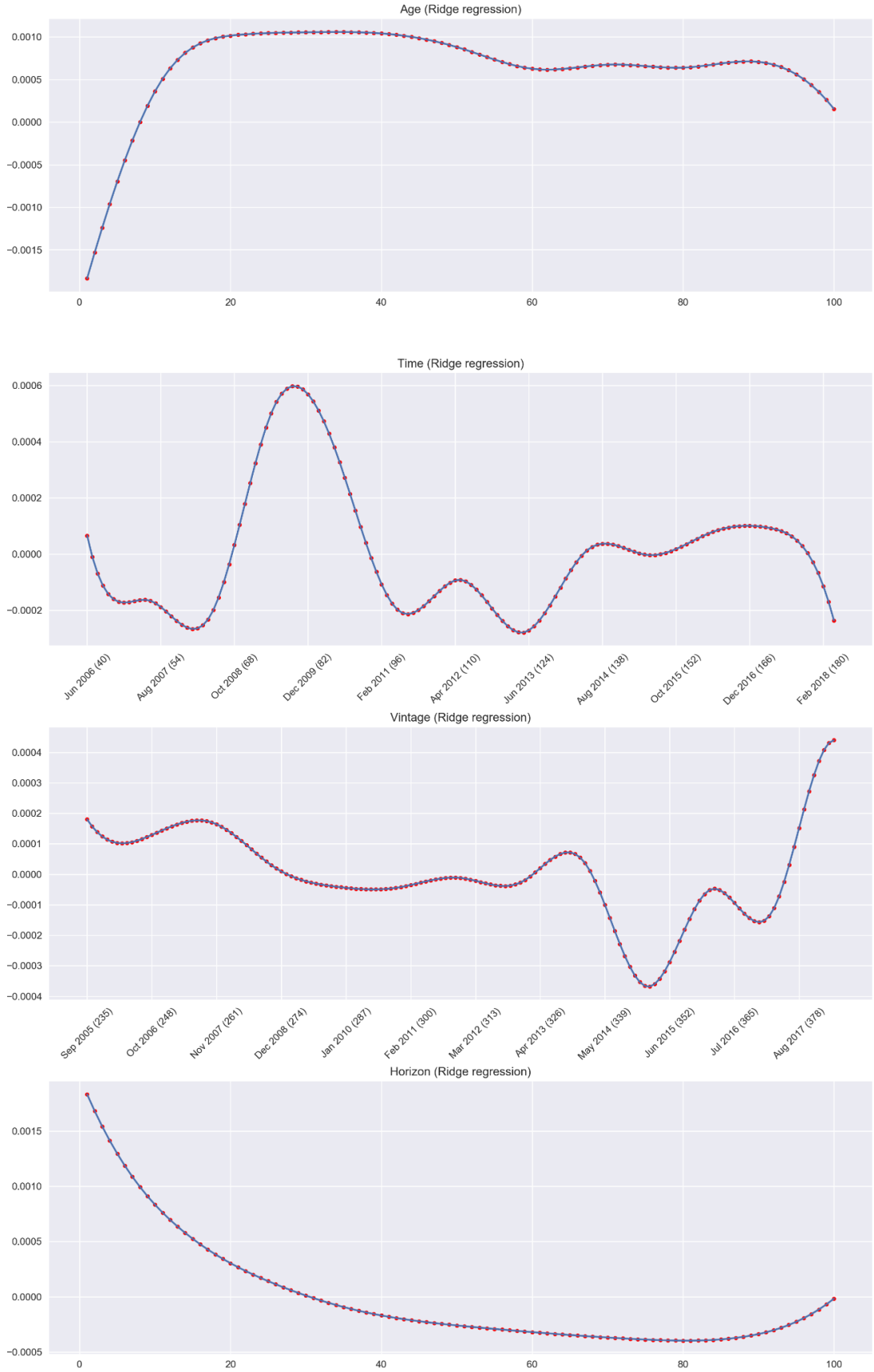

График 3.7 Графики функций F(a), H(t), G(v), K(h) для Ridge-регрессии

3.10 Построение прогнозов

При прогнозировании вероятности дефолта для заданных значений параметров t, v, h возможны два случая: 1) все параметры меньше максимального значения соответствующего параметра, используемого при построении AVTH модели, 2) значение одного или более параметров больше максимального значения соответствующего параметра, которое использовалось при построении модели, т.е. необходимо экстраполировать значение одной или нескольких функций AVTH модели.

С первым случаем нет никаких проблем: достаточно подставить конкретное значение параметра в соответствующие функции, и легко вычислить результат (поскольку функции $F(a), G(v), H(t), K(h)$ непрерывны на отрезках, задаваемых минимальным и максимальным значением соответствующих параметров).

Со вторым случаем все немного сложнее. На практике как правило необходимо экстраполировать функцию $H(t)$, т.к. прогнозирование вперед представляет наибольший интерес. Под прогнозированием вперед понимается то, что прогнозы строятся для времени t, которое больше максимального значения, использовавшегося при создании модели. Конкретно для экстраполирования функции $H(t)$ на практике можно использовать сценарии развития экономике, предоставляемые государством.

Еще один способ экстраполировать функции — рассматривать их как временные ряды, последующее значение которых необходимо спрогнозировать. Существует достаточное количество моделей, который позволяют это делать, в данной работе же будем использовать авторегрессионный подход. То есть для ряда $X(t)$ будем строить модель авторегрессии, которая описывается формулой (3.25):

$$X(t) = c + \sum_{i=1}^{p} a_i X(t-i), \tag{3.25}$$

где с — константа, a_i — коэффициенты авторегрессии, p — количество лагов, т.к. число наблюдений, которые используется для прогнозирования следующего значения. Параметры, которые необходимо найти — c и a_i . Задача по их нахождению легко приводится к матричному виду, и ее решение можно найти, например, методом наименьших квадратов.

3.11 Схема проверки построенных прогнозов

Для того, чтобы проверить, насколько хорошо модель теоретически может прогнозировать кредитные риски, проверим качество ее прогнозов на данных за

последний год: обучаем модель на всех записях, для которых время t не входит в последние 12 месяцев, а качество проверяем на оставшихся записях.

Важно отметить, что в данном случае кредитные риски оцениваются только для одного параметра t — как зачастую бывает на практике, поэтому необходимо определить, как мы будем определять вероятность дефолта при фиксированном значении времени t. Это можно делать по-разному, в данной же работе мы делаем следующее. Сперва определим вероятность дефолта при известных значениях всех параметров (временные параметры, количество дефолтов и активных аккаунтов) по формуле (3.26):

$$PD(t_k) = \frac{1}{n}\sum_{t=t_k}^{n}\frac{Def(t,v)}{Active(t-h,v)} \tag{3.26}$$

т.е. по сути мы усредняем значение вероятности дефолта для конкретного момента времени t_k.

Для предсказанных значений поступаем аналогичным образом, но здесь под знаком суммы стоит вероятность дефолта при конкретных значениях t, h, v. Имеем формулу (3.27):

$$PredictedPD(t_k) = \frac{1}{n}\sum_{t=t_k}^{n} Predicted(t,v,h) = \frac{1}{n}\sum_{t=t_k}^{n} PD(t,v,h) \tag{3.27}$$

3.12 Прогнозирование кредитных рисков

Приведем графики обученной модели на данных, из которых убраны записи за последний год. График (3.8) относится к линейной регрессии, а график (3.9) к Ridge-регрессии с параметром регуляризации равным 1,0.

На обоих графиках хорошо заметно то, что с увеличением времени t уменьшается количество доступных записей, и поэтому модель начинает сильно переобучаться: появляется очень резкий линейный тренд, который даже выбивается за границы финансового кризиса 2008-2009 гг. — из-за этого можно ожидать, что с увеличением времени, на которое строится прогноз, вероятность дефолта будет все сильнее завышаться, т.к. авторегрессия вынуждена учитывать этот тренд. И тем не менее есть надежда, что удастся построить достаточно хороший прогноз на ближайшую перспективу.

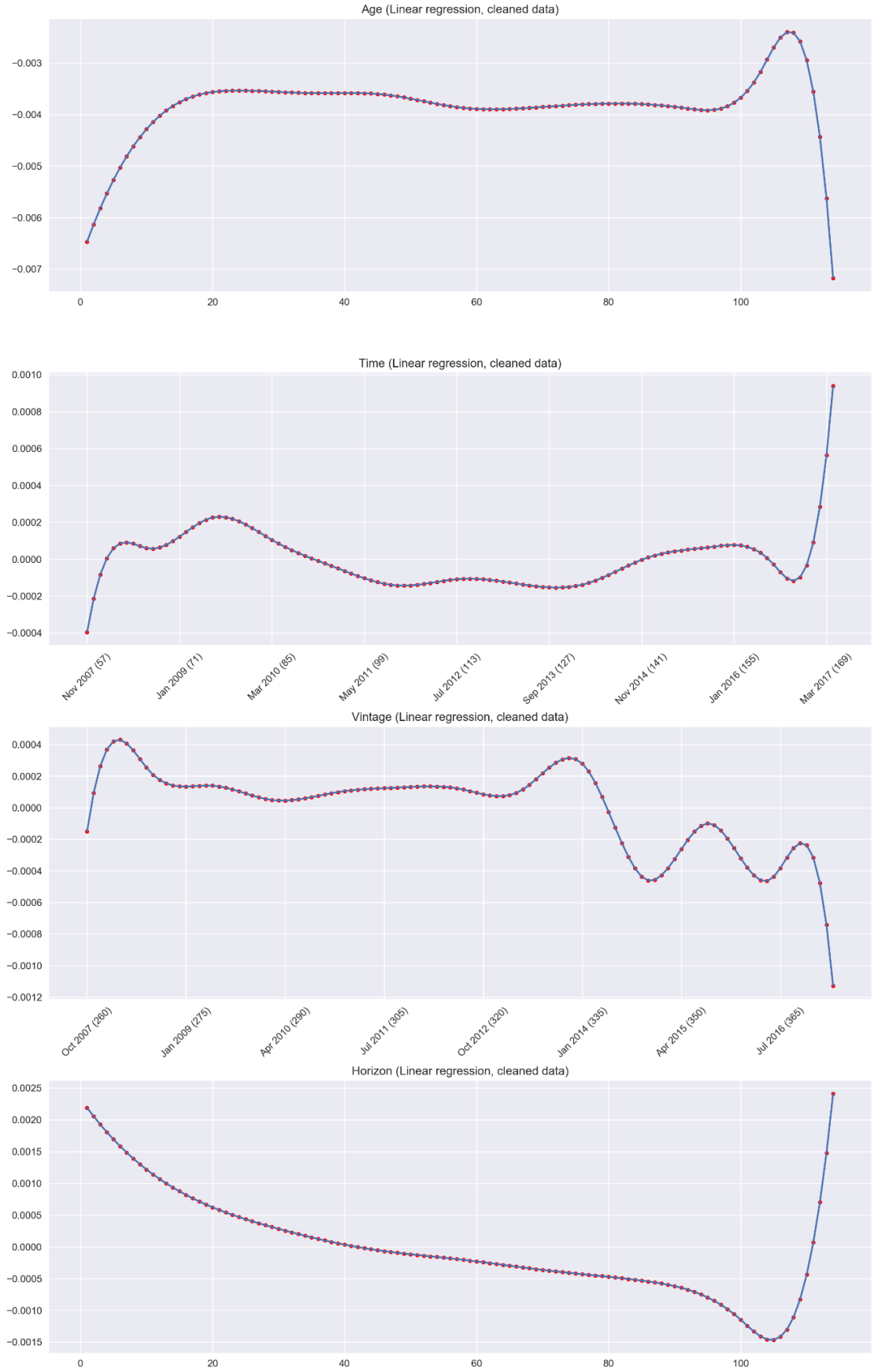

График 3.8 Графики функций F(a), H(t), G(v), K(h) для линейной регрессии при построении прогнозов

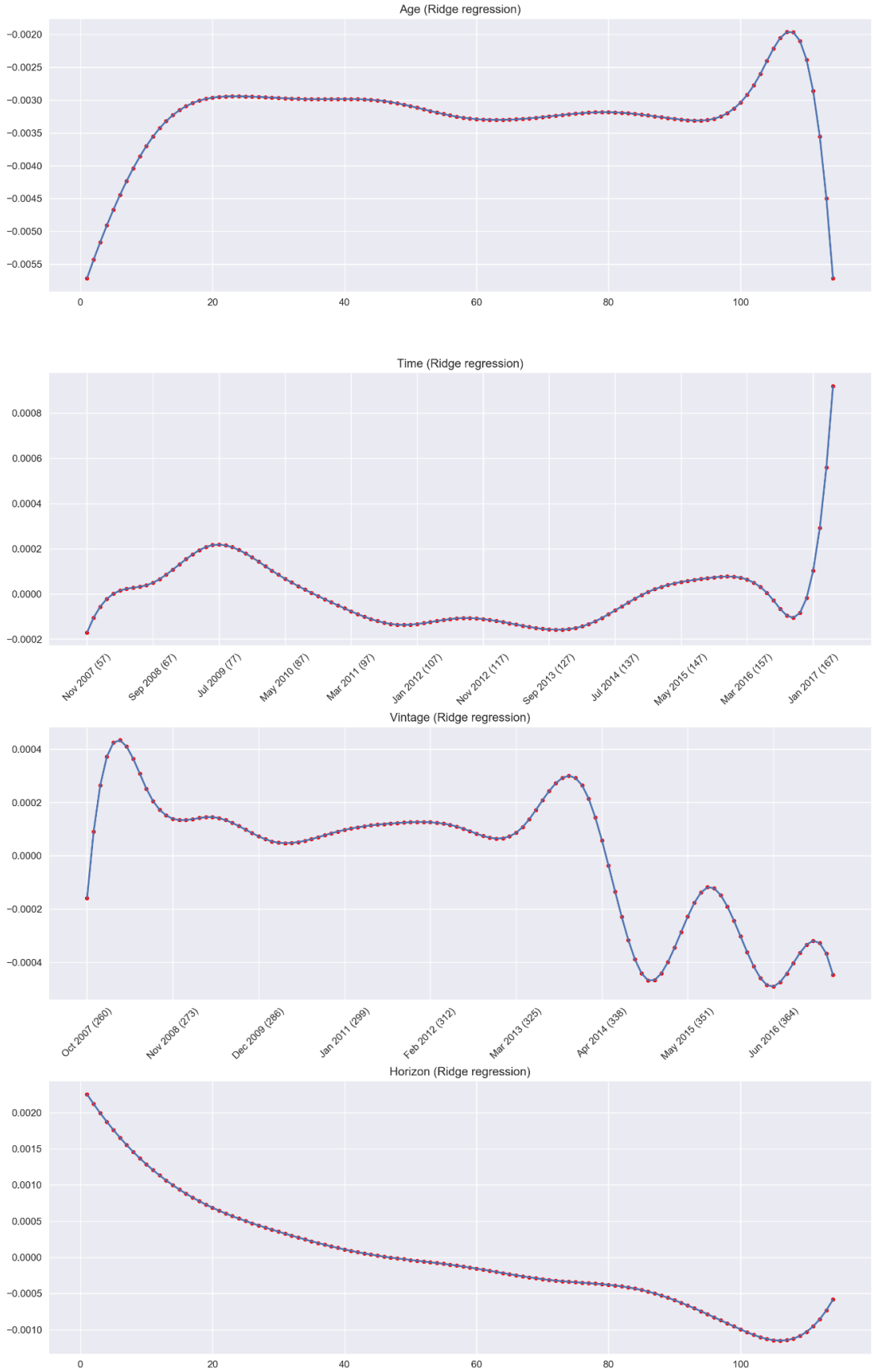

График 3.9 Графики функций F(a), H(t), G(v), K(h) для Ridge-регрессии при построении прогнозов

Построенные прогнозы представлены на графике (3.9). Как и ожидалась, с течением времени модель начинала все сильнее завышать вероятности дефолта из-за сильно увеличивающегося тренда на конце у функции $H(t)$. Однако на ближайшие 3-4 месяца прогнозы оказываются достаточно неплохими. Из двух алгоритмов регрессии лучше себя показывает Ridge-регрессия.

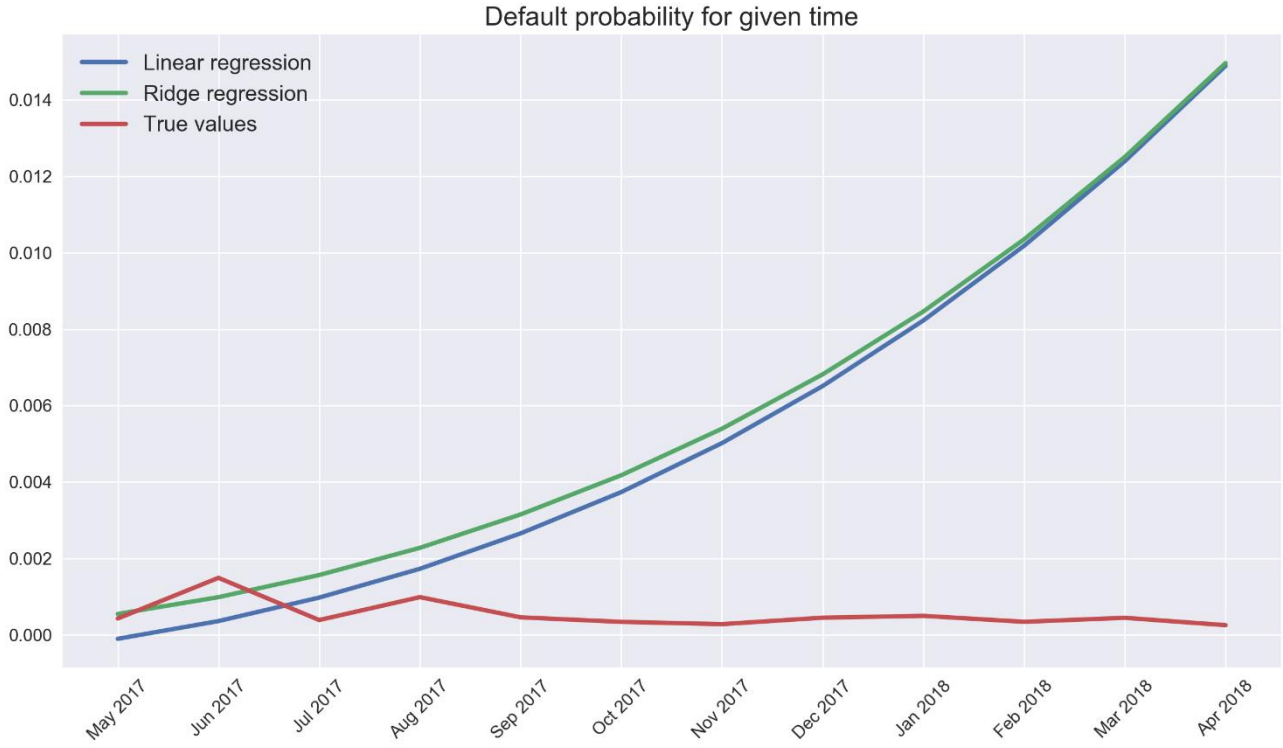

График 3.10 Построенные прогнозы

3.13 Выводы по главе

Построенная AVTH модель хорошо согласуется с историческими данными и естественными предположениями о виде некоторых функций.

Использование базисных сплайнов позволило сгладить функции F(a), G(v), H(t), K(h), что упростило их анализ и добавило наглядности их графикам.

Построенные прогнозы остаются актуальными на ближайшие 3-4 месяца. Дальше построенная авторегрессия для функции H(t) начинает очень сильно завышать ее значения при экстраполировании — это происходит из-за того, что ближе к максимальным значениям параметра t данных становится все меньше, поэтому линейная регрессия сильно переобучается (в данном случае возникает сильный возрастающий тренд), как результат — с увеличением времени, на которое строится прогноз, вероятности дефолта сильно завышаются относительно реальных значений. Данную проблему могло бы решить большее количество данных, однако на практике всегда оказывается так, что чем новее данные, тем их

меньше, поэтому нужны другие способы борьбы с недостатком данных. В качестве решения можно предложить использовать более продвинутые методики анализа временных рядов, или же строить продолжение функции в виде разумно выбранной константы (исходя из графика функции).

ЗАКЛЮЧЕНИЕ

В ходе выполнения дипломной работы были изучены понятия кредитного риска, а также рассмотрено влияние временных переменных на вероятность ухода счетов в дефолт. Были изучены различная литература, посвященная AVT/APC моделям. Рассмотрена различные виды регрессии, и некоторые из методов прогнозирования временных рядов.

В дипломной работе использовалась AVTH модель, которая представляет собой стандартную AVT модель, применяемую в банковском деле, с добавлением нового параметра — h, горизонта, который является дискретным параметром, показывающий, как далеко строится прогноз (на сколько горизонтов вперед), и измеряемый в тех же единицах, что и возраст: в данной работе в месяцах. Кроме того функции $F(a), G(v), H(t), K(h)$ сглаживались при помощи B-сплайнов.

Введение нового параметра не принесло ожидаемого эффекта: судя по графикам функции $K(h)$, можно сказать, что модель с течением времени все сильнее уменьшаем вероятность дефолта (последние возрастающие значения на некоторых графиках вполне могут быть получены из-за того, что для них было недостаточно данных), т.е. чем ближе наш прогноз к последнему известному значению, чем выше вероятность дефолта, и чем дальше, тем модель увереннее, что дефолта не случиться.

Использование B-сплайнов же оказалось чрезвычайно полезным. Во-первых функции $F(a), G(v), H(t), K(h)$ прекрасно сгладились, и их намного проще анализировать, чем функции, которые получаются при использовании стандартной модели. Более того, функции согласуются с историческими данными и с естественными предположениями об их поведении. Во-вторых время вычисления неизвестны параметров модели значительно сократилось, поскольку теперь стало намного меньше.

Использование AVTH модели не дало ожидаемых результатов: относительно успешное прогнозирование возможно лишь на ближайшую перспективу (около 4 месяцев). При попытках строить прогноз на год вперед для дальних значений наблюдалось сильное завышение вероятности дефолтов — это было связано с тем, что базовая модель сильно переобучилась для тех значений параметров, где было мало данных: появился сильный восходящий тренд для функции $H(t)$ — именно ее значения чаще всего прогнозируются временными рядами.

Стоит также отметить, что лучше себя показала Ridge-регрессия, поскольку она обучается «с некоторым запасом»: она как правило не достигает лучшего качества на обучающей выборке, но зато она также и не слишком сильно под нее подстраивается (особенно там, где мало данных), и поэтому для новых данных ее предсказания оказываются лучше.

СПИСОК ИПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. CS229 Lecture notes, Andrew Ng – Режим доступа: http://cs229.stanford.edu/notes/cs229-notes1.pdf – Дата доступа: 29.05.2018
2. B-Spline – Режим доступа: http://mathworld.wolfram.com/B-Spline.html – Дата доступа: 29.05.2018
3. Документация scipy.interpolate.BSpline – Режим доступа: https://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.BSpline.html#id2 – Дата доступа: 29.05.2018
4. Parametric Models of Function – Режим доступа: http://www.brnt.eu/phd/node11.html#SECTION00632200000000000000 – Дата доступа: 29.05.2018
5. De Boor's Algorithm – Режим доступа: http://pages.mtu.edu/~shene/COURSES/cs3621/NOTES/spline/de-Boor.html – Дата доступа: 29.05.2018
6. Python Plotting With Matplotlib (Guide) – Режим доступа: https://realpython.com/python-matplotlib-guide/ – Дата доступа: 29.05.2018
7. Age–Period–Cohort Analysis, Theodore R. Holford
8. Kupper, L.L., Janis, J.M., Karmous, A. & Greenberg, B.G. (1985). Statistical age–period–cohort analysis: a review and critique, Journal of Chronic Diseases 38, 811–830.
9. Kupper, L.L., Janis, J.M. Salama, I.A. Yoshizawa, C.N. & Greenberg, B.G. (1983). Age–period–cohort analysis: an illustration of the problems in assessing interaction in one observation per cell data, Communications in Statistics – Theory and Methods 12, 2779–2807.
10. Holford, T.R. (1991). Understanding the effects of age, period, and cohort on incidence and mortality rates, Annual Reviews of Public Health 12, 425–457.
11. Holford, T.R. (1983). The estimation of age, period and cohort effects for vital rates, Biometrics 39, 311–324.
12. Rogers, W.L. (1982). Estimable functions of age, period, and cohort effects, American Sociology Review 47, 774–796.
13. Barrett, J.C. (1973). Age, time, and cohort factors in mortality from cancer of the cervix, Journal of Hygiene (Cambridge) 71, 253–259.
14. Barrett, J.C. (1978). The redundancy factor method and bladder cancer mortality, Journal of Epidemiology and Community Health 32, 314–316.
15. Fienberg, S.E. & Mason, W.M. (1978). Identification and estimation of age–period–cohort models in the analysis of discrete archival data, in Sociological Methodology 1979, K.F. Schuessler, ed. Jossey-Bass, San Francisco, pp. 1–67.

16. Osmond, C. & Gardner, M.J. (1982). Age, period and cohort models applied to cancer mortality rates, Statistics in Medicine 1, 245–259
17. Roush, G.C., Schymura, M.J., Holford, T.R., White, C. & Flannery, J.T. (1985). Time period compared to birth cohort in Connecticut incidence rates for twenty-five malignant neoplasms, Journal of the National Cancer Institute 74, 779–788.
18. Roush, G.C., Holford, T.R., Schymura, M.J. & White, C. (1987). Cancer Risk and Incidence Trends, The Connecticut Perspective. Hemisphere, New York.
19. Wickramaratne, P.J., Weissman, M.M. Leaf, P.J. & Holford, T.R. (1989). Age, period and cohort effects on the risk of major depression: results from five United States communities, Journal of Clinical Epidemiology 42, 333–343.
20. Berzuini, C. & Clayton, D. (1994). Bayesian analysis of survival on multiple time scales, Statistics in Medicine 13, 823–838.
21. Moolgavkar, S.H., Stevens, R.G. and Lee, J.A.H. (1979). Effect of age on incidence of breast cancer in females, Journal of the National Cancer Institute 62, 493–501.
22. Holford, T.R., Zhang, Z. & McKay, L.A. (1994). Estimating age, period and cohort effects using the multistage model for cancer, Statistics in Medicine 13, 23–41.
23. Armitage, P. & Doll, R. (1954). The age distribution of cancer and a multi-stage theory of carcinogenesis, British Journal of Cancer 8, 1–12.
24. Brown, C.C. & Kessler, L.G. (1988). Projections of lung cancer mortality in the United States: 1985–2025, Journal of the National Cancer Institute 80, 43–51.
25. Fienberg, S.E. & Mason, W.M. (1985). Specification and implementation of age, period and cohort models, in Cohort Analysis in Social Research, W.M. Mason & S.E. Fienberg, eds. Springer-Verlag, New York, pp. 45–88.
26. Holford, T.R., Roush, G.C. & McKay, L.A. (1991). Trends in female breast cancer in Connecticut and the United States, Journal of Clinical Epidemiology 44, 29–39.

ПРИЛОЖЕНИЕ А

Использованное программное обеспечение

- Python 3, Jupyter — среда программирования
- Scipy — B-сплайны, разряженные матрицы
- SKLearn — различные виды регрессии
- Statsmodels — временные ряды
- Matplotlib, Seaborn — построение графиков
- Pandas — манипулирование данными (загрузка, группировка…)
- Numpy — стандартные матрицы
- Joblib — параллельное применение функций

ПРИЛОЖЕНИЕ Б

КОД ПРОГРАММЫ

Загрузка и подготовка данных

```
data = pd.read_csv("Data/fully_aggregated_data.csv")
p = data["Defaults"] / (data["Accounts"])

age_threshold = 10
time_threshold = 10
vintage_threshold = 10

few_age = data.groupby(by=["Age"]).apply(lambda x: x.shape[0]).sort_values()
few_time = data.groupby(by=["Time"]).apply(lambda x: x.shape[0]).sort_values()
few_vintage = data.groupby(by=["Vintage"]).apply(lambda x: x.shape[0]).sort_values()

threshold_idxs = list(set(list(few_age[few_age < age_threshold].index) \
+ list(few_time[few_time < time_threshold].index) \
+ list(few_vintage[few_vintage < vintage_threshold].index)))
print("These idxs should be removed (without vintage < 120):", threshold_idxs)

extra_idxs = list(set(
    list(data[data["Vintage"] < 235].index) +
    list(data[data["Age"] > 100].index) +
    #list(data[data["Age"] < 6].index) +
    list(data[data["Time"] < 40].index) +
    list(data[data["Horizon"] > 115].index) +
    threshold_idxs
))

new_idxs = list(set(data.index).difference(set(extra_idxs)))
new_data = data.loc[new_idxs]
new_p = p[new_idxs]

print("In the end we should remove these number of rows: ", len(extra_idxs))
```

Построение модели

```
n_age_splines_1 = 10
age_splines_degree_1 = 3
age_size_1 = n_age_splines_1 + age_splines_degree_1
```

```
age_bsplines_1 = evaluate_bsplines(new_data["Age"], n_splines=n_age_splines_1,
degree=age_splines_degree_1)

## Аналогично сплайны для времени, винтажа и горизонта.
## Только другое количество при необходимости
…

design_matrix_1 = sp.hstack([age_bsplines_1, time_bsplines_1, vintage_bsplines_1,
horizon_bsplines_1])

lin_regr = LinearRegression(fit_intercept=True)
design_matrix_1 = design_matrix_1.tocsr()
lin_regr.fit(design_matrix_1, new_p)

a_coeffs_1, t_coeffs_1, v_coeffs_1, h_coeffs_1 = split_coeffs(
   lin_regr.coef_, [age_size_1, time_size_1, vintage_size_1, horizon_size_1]
)
age_spline_1 = get_bspline(a_coeffs_1,
                     data=new_data["Age"],
                     n_splines=n_age_splines_1,
                     degree=age_splines_degree_1
                    )
## Аналогично сплайны для винтажа, времени и горизонта

trend_regr_1 = LinearRegression(fit_intercept=True)

trend_regr_1_time = np.arange(new_data["Time"].min(), new_data["Time"].max() + 1.,
1., dtype=np.int16)
trend_regr_1_x = np.array(trend_regr_1_time).reshape(-1,1)
trend_regr_1_y = time_spline_1(trend_regr_1_x)
trend_regr_1.fit(trend_regr_1_x, trend_regr_1_y)

a1 = trend_regr_1.intercept_[0]
b1 = trend_regr_1.coef_.flatten()[0]
# Create x_points for age, time, vintage, horizon and
# calculate mean for time, vintage, horizon and define coef before constant for age
full_step_1 = 1.

age_full_x1 = np.arange(new_data["Age"].min(), new_data["Age"].max() + 1.,
full_step_1)
age_cp_x1 = np.sort(new_data["Age"].unique())
```

```
p1 = lin_regr.intercept_

time_full_x1 = np.arange(new_data["Time"].min(), new_data["Time"].max() + 1.,
full_step_1, dtype=np.int16)
time_cp_x1 = np.sort(new_data["Time"].unique())
t1 = np.mean(np.array([y - (a1 + b1*t) for t, y in zip(time_full_x1,
time_spline_1(time_full_x1))]))

vintage_full_x1 = np.arange(new_data["Vintage"].min(), new_data["Vintage"].max() +
1., full_step_1, dtype=np.int16)
vintage_cp_x1 = np.sort(new_data["Vintage"].unique())
v1 = np.mean(np.array([y + (a1 + b1*v) for v, y in zip(vintage_full_x1,
vintage_spline_1(vintage_full_x1))]))

horizon_full_x1 = np.arange(new_data["Horizon"].min(), new_data["Horizon"].max() +
1., full_step_1)
horizon_cp_x1 = np.sort(new_data["Horizon"].unique())
h1 =  np.mean(horizon_spline_1(horizon_full_x1))

age_full_y1 = [p1 + a1 + age*b1 + y + t1 + v1 + h1 for age, y in zip(age_full_x1,
age_spline_1(age_full_x1))]
age_cp_y1 = [p1 + a1 + age*b1 + y + t1 + v1 + h1 for age, y in zip(age_cp_x1,
age_spline_1(age_cp_x1))]
age_1 = {
   "full": dict(zip(age_full_x1, age_full_y1)),
   "cp": dict(zip(age_cp_x1, age_cp_y1))
}

time_full_y1 = [y - (a1 + b1*t) - t1 for t, y in zip(time_full_x1,
time_spline_1(time_full_x1))]
## Аналогично возрасту
…
vintage_full_y1 = [y + (a1 + v*b1) - v1 for v, y in zip(vintage_full_x1,
vintage_spline_1(vintage_full_x1))]
## Аналогично возрасту
…
horizon_full_y1 =[y - h1 for h, y in  zip(horizon_full_x1,
horizon_spline_1(horizon_full_x1))]
## Аналогично возрасту
…
plot_solution(
```

```
    age=age_1, time=time_1, vintage=vintage_1, horizon=horizon_1,
    model_used = "Linear regression", plot_mode=22, vintage_freq_xticks=13
)
```

Построение прогноза

```
forecast_time = np.sort(new_data["Time"].unique())[-12:]
train_data = new_data[new_data["Time"] < min(forecast_time)]
train_y = p.iloc[train_data.index]
valid_data = new_data[new_data["Time"] >= min(forecast_time)]

max_age = train_data["Age"].max()
…
valid_data = valid_data.assign(p=valid_data["Defaults"] / valid_data["Accounts"])
valid_proba = dict(valid_data.groupby("Time").apply(lambda x: x["p"].sum() /
x.shape[0]))

def get_ar_model(data_dict, maxlag=10):
    time_series = [(k, data_dict[k]) for k in data_dict.keys()]
    time_series.sort(key=lambda x: x[0])
    time_values = [x[1] for x in time_series]
    model = AR(time_values)
    fitted_model = model.fit(maxlag=maxlag)
    return fitted_model

default_maxlag = 5
class Scoring_predictor:
    def __init__(self, age, vintage, time, horizon, maxlag=default_maxlag):
        self.age_dict = age
        self.vintage_dict = vintage
        self.time_dict = time
        self.horizon_dict = horizon

        self.max_age = max(age.keys())
        self.max_time = max(time.keys())
        self.max_vintage = max(vintage.keys())
        self.max_horizon = max(horizon.keys())

        self.min_age = min(age.keys())
        self.min_time = min(time.keys())
        self.min_vintage = min(vintage.keys())
        self.min_horizon = min(horizon.keys())
```

```
    # Define autoregressive models
    self.age_ar = get_ar_model(age, maxlag=maxlag)
    self.time_ar = get_ar_model(time, maxlag=maxlag)
    self.vintage_ar = get_ar_model(vintage, maxlag=maxlag)
    self.horizon_ar = get_ar_model(horizon, maxlag=maxlag)

def predict_age(self, age):
    if age <= self.max_age:
        return self.age_dict[age]
    else:
        age_size = len(self.age_dict.keys())
        start = age_size
        end = int(age - self.min_age)
        prediction = self.age_ar.predict(start=start, end=end)
        return prediction[-1]

def predict_time(self, time):
    if time <= self.max_time:
        return self.time_dict[time]
    else:
        time_size = len(self.time_dict.keys())
        start = time_size
        end = int(time - self.min_time)
        prediction = self.time_ar.predict(start=start, end=end)
        return prediction[-1]

def predict_vintage(self, vintage):
    if vintage <= self.max_vintage:
        return self.vintage_dict[vintage]
    else:
        vintage_size = len(self.vintage_dict.keys())
        start = vintage_size
        end = int(vintage - self.min_vintage)
        prediction = self.vintage_ar.predict(start=start, end=end)
        return prediction[-1]

def predict_horizon(self, time):
    # For how many horizons we predict value
    # We don't make too long forecasts, so horizon dict should have all necessary keys
    horizon = time - self.max_time
    return self.horizon_dict[horizon]
```

```
    def predict(self, data_row):
        a = data_row["Age"]
        t = data_row["Time"]
        v = data_row["Vintage"]
        return self.predict_age(a) + self.predict_time(t) + self.predict_vintage(v) +
self.predict_horizon(t)

lin_regr_predictor = Scoring_predictor(
    age=age_1["full"],
    time=time_1["full"],
    vintage=vintage_1["full"],
    horizon=horizon_1["full"],
    maxlag=10
)

def create_df_predictions(scoring_predictor):
    predictions = valid_data.apply(lambda row: scoring_predictor.predict(row), axis=1)
    df = pd.DataFrame(
        {
            "P": predictions.values,
            "Time": valid_data["Time"]
        },
        index=predictions.index
    )
    return df

def calculate_forecasts(df):
    return dict(df.groupby("Time").apply(lambda group: group["P"].sum() /
group.shape[0]))

lin_regr_predictions_df = create_df_predictions(lin_regr_predictor)
lin_regr_forecasts = calculate_forecasts(lin_regr_predictions_df)
freq_xticks = 1
xticks_locs = list(range(min(valid_proba.keys()), max(valid_proba.keys()) +
1))[::freq_xticks]
xticks_values = [convert_to_str(add_months(time_min_date, d)) for d in xticks_locs]

linewidth=3
plt.figure(figsize=(15,8), dpi=200)
```

```
plt.plot(list(lin_regr_forecasts.keys()), list(lin_regr_forecasts.values()), label="Linear
regression", linewidth=linewidth)
plt.plot(list(ridge_regr_forecasts.keys()), list(ridge_regr_forecasts.values()),
label="Ridge regression", linewidth=linewidth)
plt.plot(list(valid_proba.keys()), list(valid_proba.values()), label="True values",
linewidth=linewidth)
plt.legend(prop={'size': 15})
plt.xticks(xticks_locs, xticks_values, rotation=45, fontsize=12)
plt.yticks(fontsize=12)
plt.title("Default probability for given time", fontsize=18)
plt.show()
```