МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Факультет прикладной математики и информатики
Кафедра математического моделирования и анализа данных

Курсовая работа

Разработка байесовских авторегрессионных моделей для прогнозирования стационарных временных рядов

Карповича Артёма Дмитриевича
студента 3 курса, специальность
«Прикладная математика»

Научный руководитель
Лобач Виктор Иванович
доцент кафедры ММАД
кандидат физ.-мат. наук

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет прикладной математики и информатики

Кафедра математического моделирования и анализа данных

ЗАДАНИЕ НА КУРСОВУЮ РАБОТУ

Студент Карпович Артём Дмитриевич

1. Тема Разработка байесовских авторегрессионных моделей для прогнозирования стационарных временных рядов

2. Срок представления курсового проекта к защите 17 мая 2024 г.

3. Исходные данные для научного исследования

3.1. Кувайскова, Ю. Е Статистические методы прогнозирования / Ю. Е. Кувайскова, В. Н. Клячкин/ – Минск: УГТУ, 2019.

3.2. Ивченко, Г. И. Математическая статистика: Учеб. Пособие для втузов / Ивченко Г. И., Медведев Ю. И./ – Москва: МВТУ им. Н. Э. Баумана, 1984.

3.3. Грэйнджер, Клайв У. Дж. Эконометрический анализ временных рядов / Клайв У. Дж. Грэйнджер.

3.4. Абрамова, А. Е. Метод Монте Карло по схеме марковской цепи для оценки вероятности редких событий в задачах биоинформатики / Абрамова А. Н./ – Санкт-Петербург: СПБГУ, 2017.

3.5. Четыркин, Е. М. Статистические методы прогнозирования / Четыркин Е. М. // издание «Статистика». – 1975.

3.6. Hamilton, J. D. Time Series Analysis / James D. Hamilton/ – Princeton, New Jersey: Princeton University, 1994.

4. Содержание курсовой работы

4.1. Подготовить обзор по теме «разработка байесовских авторегрессионных моделей для прогнозирования стационарных временных рядов».

4.2. Подготовить математическое описание авторегрессионных моделей прогнозирования стационарный временных рядов

4.3. Использование байесовского подхода для моделей ARIMA

4.4. Подготовить отчет по курсовому проекту.

Руководитель курсового проекта _____ Лобач В.И.

подпись, 19.09.2023 *фамилия, инициалы*

Задание принял к выполнению _____ Карпович А.Д.

подпись, 19.09.2023 *фамилия, инициалы*

Оглавление

Введение

В современном мире большое количество явлений можно представить в виде последовательности наблюдений, измеренных в разные моменты времени и упорядоченных в хронологическом порядке, такие последовательности называются временными рядами. Прогнозирование временных рядов играет важную роль во многих областях, включая финансы, экономику, климатологию, медицину и другие. Эффективный анализ временных рядов является ключевым инструментом для принятия обоснованных решений в различных сферах деятельности.

Анализ временных рядов включает в себя исследование статистических свойств ряда, выявление трендов, сезонности и других особенностей, а также построение моделей для прогнозирования будущих значений и выявления структуры временного ряда. Выявление структуры временного ряда необходимо для того, чтобы построить математическую модель того явления, которое является источником анализируемого временного ряда. Прогноз будущих значений временного ряда используется для эффективного принятия решений.

Временные ряды состоят из двух элементов:

- периода времени, за который или по состоянию на который приводятся числовые значения;
- числовых значений того или иного показателя, называемых уровнями ряда.

Нас будут интересовать стационарные временные ряды. Стационарные временной ряд – это ряд, чьи статистические свойства не меняются со временем. Формально, временной ряд считается стационарным, если выполнены следующие свойства:

- стационарность по среднему: среднее значение ряда не зависит от времени и остается постоянным на протяжении всего ряда;
- стационарность по дисперсии: дисперсия ряда не зависит от времени и остается постоянной на протяжении всего ряда;
- стационарность по автоковариации: автоковариация между значениями ряда на разных временных отрезках зависит только от длины этих отрезках, но не от их положения во времени.

Стационарность является важным свойством при анализе и моделировании, так как она позволяет применять статистические методы и модели, которые предполагают постоянство статистических свойств данных. В стационарном ряде проще выявить закономерности, построить модели и делать прогнозы на основе его статистических свойств, чем в нестационарном ряде, где эти свойства могут меняться со временем.

Основными моделями для работы со стационарными рядами можно упомянуть следующие виды: одномерные, линейные и гомоскедстичные (то есть модели, характеризующиеся постоянством дисперсий остатков).

ГЛАВА 1. Применение теоремы Байеса в машинном обучении.

1.1 Применение машинного обучения в прогнозировании.

Машинное обучение – это компьютерные методы выявления закономерностей в данных большого объема (Big Data).

Одной из задач, которую можно решить методами машинного обучения является задача *бинарной классификации*. Например, рассматривается технический объект, о котором известно, что при одном определенном наборе параметров функционирования объект был исправен, при другом – неисправен. Известны новые параметры функционирования. Спрогнозировать, будет ли объект исправен? Это и есть *задача бинарной классификации*, ориентированная на предсказание состояния объекта – прогнозирование.

Другой класс задач прогнозирования, который решается с помощью методов машинного обучения – *задачи регрессии*. Пусть, например, мы планируем открыть новый магазин. Имеющийся набор данных – это информация о характеристиках магазинов нужного профиля, работающих в рассматриваемом районе (место расположения, площадь, товарооборот и т.п.). Известна и прибыль магазинов за прошедший год. По этим данным можно построить модель для оценки предполагаемой прибыли в зависимости от характеристик магазина. Новый магазин можно открыть в нескольких местах с заданными характеристиками. Какое из них обеспечит максимальную прибыль?

Рассмотренные примеры относятся к классу задач машинного обучения с учителем, или по прецедентам: прецеденты – это известные данные о функционировании технических объектов, или о работе магазинов и т. п. В качестве исходных данных используется множество параметров - признаков объекта *X*:

$$X = \begin{pmatrix} x_{11} & \cdots & x_{1p} \\ \vdots & \ddots & \vdots \\ x_{l1} & \cdots & x_{lp} \end{pmatrix} \text{ или } X = \left(x_1, x_2, \dots, x_p\right),$$

где x_{ij} – результат *i*-го наблюдения по *j*-му параметру-признаку; *i* = 1, …, l, j = *1, ..., p*, *l* – количество строк – число наблюдений: сколько объектов исследовано, *p* – количество столбцов, или число признаков, например, параметров функционирования объекта; x_j – вектор значений *j*-го параметра-признака (значения признаков могут быть как количественными, так и бинарными, номинальными, порядковыми), и вектор-столбец ответов *Y*, состоящий для задач бинарной классификации из *1* (для тех опытов, в которых объект исправен) и *0* при неисправном объекте.

В задачах регрессии, в частности, в задаче об открытии нового магазина, вектор-столбец ответов *Y* состоит из значений прибыли каждого из

работающих магазинов.

Каждой строке x_i множества *X* соответствует определенное значение y_i вектора *Y*. Совокупность пар (x_i, y_i) образует выборку *исходных данных – прецедентов*.

Задача состоит в построении функции $a: X \to Y$, которая предскажет ответ *Y* для любого заданного *X*. Таким образом, функция *a* (ее называют еще и алгоритмом или моделью алгоритма) должна не только приближать искомый результат на выборке исходных данных, но и «работать» на всей генеральной совокупности, из которой получено множество X. При числовых значениях признаков x_j часто используют линейные модели с вектором параметров $w = (w_0, w_1, \dots, w_p)$:

$$a(x, w) = w_0 + w_1 * x_1 + \cdots + w_p * x_p. \quad (1.1.1)$$

При этом в задачах бинарной классификации обычно вместо нуля и единицы используют множество ответов *Y* = {–1; +1}. В этом случае модель алгоритма примет вид

$$a(x, w) = sign \sum_{j=0}^{p} w_j x_j \,, (x_0 = 1). \quad (1.1.2)$$

Параметры w_j подбираются по исходным данным. Процесс подбора оптимальных параметров называется *обучением алгоритма*. Найденные параметры должны обеспечить оптимальное значение функционала качества.

Вводится *функция потерь L(a, X)*, характеризующая ошибку алгоритма *a* на объекте *X*. При нулевом значении этой функции ответ считается корректным. В задачах бинарной классификации в качестве функции потерь можно использовать *индикатор ошибки*:

$$L(a, x) = [a(x_i) \neq y_i]. \quad (1.1.3)$$

Функционал качества алгоритма *a* на выборке объема *l* (среднее значение потерь) называют эмпирическим риском:

$$Q(a, X) = \frac{1}{l} \sum_{i=1}^{l} L(a, x_i) \,. \quad (1.1.4)$$

В задаче бинарной классификации минимизируется функционал ошибок:

$$Q(a,X) = \frac{1}{l}\sum_{i=1}^{l}(a(x_i) - y_i) \rightarrow \min. \qquad (1.1.5)$$

В задаче регрессии минимизируется среднеквадратичная ошибка:

$$Q(a,X) = \frac{1}{l}\sum_{i=1}^{l}(a(x_i) - y_i)^2 \rightarrow min. \qquad (1.1.6)$$

Алгоритм a, минимизирующий функционалы (1.1.5) или (1.1.6), может не обеспечивать хорошее приближение на произвольной выборке из генеральной совокупности. Ситуация, когда качество работы алгоритма на новых объектах значительно хуже, чем на исходной выборке, свидетельствует о переобучении: алгоритм слишком хорошо подогнан под обучающую выборку и не способен к обобщению на другие выборки. Таким образом, построенный алгоритм не сможет предсказывать состояние исследуемого объекта при новых параметрах функционирования.

Для оценки качества модели с точки зрения возможности прогнозирования исходную обучающую выборку из l опытов разбивают на два непересекающихся подмножества: собственно обучающую выборку объема l_0 (с помощью которой и решается задача обучения (1.1.5) – (1.1.6)) и контрольную (или тестовую) объема $l_k = l - l_0$, не используемую для обучения.

При использовании кросс-валидации выборка разбивается на N частей (на практике обычно принимают $N = 5$ или $N = 10$). ($N - 1$) часть используется для обучения, одна – для контроля. Последовательно перебираются все варианты, например, при разбиении на пять частей вначале в качестве обучающей выборки используются части 1 – 4, а часть 5 – тестирования.

На следующем шаге в качестве обучающей используются части 2 – 5, а часть 1 – для тестирования, и т.д. Для каждого разбиения решается задача обучения по выборке l_0 и вычисляется функция ошибок $Q(a, X)$ на контрольной выборке l_k. Среднее значение этой функции по всем вариантам разбиения и характеризует обобщающую способность алгоритма.

Для построения качественных моделей необходим предварительный анализ исходных данных: проверка значимости признаков, исключение выбросов, иногда необходима нормировка(стандартизация). При большом числе признаков целесообразно сократить размерность, используя, например, метод главных компонент.

Таким образом, основные этапы машинного обучения таковы:

1. Постановка задачи.

2. Выделение признаков (параметров функционирования), оказывающих влияние на состояние объекта.

3. Формирование выборки исходных данных и способа ее разбиения на обучающую и контрольную части.

4. Выбор функционала качества.

5. Предварительная обработка данных – отбор значимых признаков, обработка выбросов и пропусков, проведение стандартизации.

6. Построение модели по обучающей выборке.

7. Оценка качества модели по тестовой выборке.

8. Использование модели для прогнозирования состояния объекта по известным параметрам функционирования.

1.2 Метод MAP

Давайте обозначим наши данные — E, а наши гипотезы — H. Тогда нам надо найти вероятность гипотезы для наших данных $P(H|E)$, которая по теореме Байеса равна:

$$P(H|E) = \frac{P(E|H)P(H)}{P(E)} \tag{1.2.1}$$

Нас интересует только соотношение вероятностей, поэтому мы можем выкинуть из этого выражения $P(E)$ (E не зависит от H) и $P(H)$ (будем считать, что все гипотезы равновероятны; строго говоря, это не всегда так, мы ещё вернёмся к этому вопросу, но во многих случаях это справедливо).
Получается, что нам надо найти такую гипотезу H, для которой максимально $P(H|E)$. Говоря математическим языком, мы должны для каждой гипотезы вычислить её *апостериорную* вероятность и выбрать гипотезу, для которой эта вероятность максимальна. Этот подход называется *maximum a posteriori probability (MAP)*. Если строго следовать ему, то вы получите лучшую гипотезу. И это математически доказано.

К сожалению, полное следование MAP возможно далеко не всегда. Например, данных слишком много. Или решение надо дать быстро. Поэтому, в реальной жизни используются различные модификации MAP.

Однако MAP обладает некоторыми недостатками:

1. Честное Байесовское обучение предполагает, что мы вычисляем вероятности всех возможных гипотез для всех обучающих данных. То есть, если у нас 1000 гипотез и 1000 обучающих паттернов, то нам придётся рассчитать 1000000 апостериорых вероятностей. В реальной жизни, обычно, числа больше на порядки. Выполнить полный рассчёт просто невозможно;
2. Вторая важная проблема возникает при создании классификаторов на основе Байесовского обучения. Мы хотим решить: мужчина перед нами, или женщина. У нас есть 4 гипотезы. Их вероятности распределяются так:

$$P(\text{Ж}|H_1) = 0{,}2$$
$$P(\text{Ж}|H_2) = 0{,}2$$
$$P(\text{Ж}|H_3) = 0{,}2$$
$$P(\text{М}|H_4) = 0{,}4$$

Если мы выберем MAP-гипотезу, то это будет h_4. И она скажет нам, что перед нами мужчина. Однако, сумма остальных гипотез — 0.6, что больше, чем 0.4. То есть, следовало бы ответить, что перед нами *женщина*. Проблема не в том, что Байесовское обучение плохо. Просто в данном случае мы должны рассматривать не отдельные гипотезы, а все возможные сочетание гипотез. Тогда мы получим *идеальный Байесовский классификатор*. Основная его проблема даже не в том, как его получить, а уже в том, как его использовать. Ведь даже имея идеальный классификатор, чтобы им воспользоваться нам надо вычислить некую идеальную комбинацию *всех* гипотез.

1.3. MCMC

Рассмотрим вероятностное распределение $p(T)$. Методы Монте Карло (методы статистических испытаний) предполагают генерацию выборки из этого распределения:

$$T_1, \dots, T_N \sim p(T).$$

Данная выборка может быть использована для оценки вероятностных интегралов вида

$$E_T f(T) = \int f(T)p(T)dT \simeq \frac{1}{N}\sum_{n=1}^{N} f(T_n). \qquad (1.3.1)$$

К интегралам такого вида сводятся многие шаги при осуществлении байесовского вывода в вероятностных моделях. Например, такими интегралами являются функция обоснованности $p(t|X,\alpha)$ и прогнозное распределение $p(t_{new}|x_{new},t,X,\alpha)$ в модели RVM, а также функционал на M-шаге EM-алгоритма $E_q(T)\log p(X,T|\Theta)$. Здесь стоит отметить, что плотность $p(T)$ в подобных вероятностных интегралах часто известна с точностью до нормировочной константы

$$p(T) = \frac{1}{Z}\tilde{p}(\text{T}). \qquad (1.3.2)$$

Выборка $T_1, \dots, T_N$ может быть также использована для оценки моды

распределения $p(T)$:

$$\max_T p(T) \simeq \max_n p(T_n)\,, \tag{1.3.3}$$

т.к. появление точек выборки наиболее вероятно в областях больших значений плотности.

Основной вопрос, раскрываемый в дальнейшем, состоит в том, как эффективно сгенерировать выборку $T_1, \ldots, T_N$ из вероятностного распределения, заданного своей плотностью $p(T)$ или своей ненормированной плотностью $\tilde{p}(T)$.

Рассмотрим теперь вопрос генерации выборки из распределения $p(T)$ в многомерном пространстве с помощью методов Монте Карло по схеме марковской цепи (MCMC). В этих методах вводится некоторая марковская цепь с априорным распределением $p_0(T)$ и вероятностями перехода в момент времени n $q_n(T_{n+1}|T_n)$, а генерация выборки происходит следующим образом:

$$\begin{aligned} T_1 &\sim p_0(T), \\ T_2 &\sim q_1(T_2|T_1), \\ &\cdot \\ &\cdot \\ &\cdot \\ T_N &\sim q_{N-1}(T_N\,|T_{N-1}). \end{aligned} \tag{1.3.4}$$

Заметим, что при таком подходе генерируемая выборка не является набором независимых случайных величин. Однако, она подходит для оценки вероятностных интегралов вида (1.3.1) или оценки моды распределения. В том случае, если необходимо получить набор независимых величин, достаточно проредить полученный набор $T_1, \ldots, T_N$, взяв каждый m-ый отсчет, где m достаточно велико.

В дальнейшем рассматривается вопрос о том, как выбрать вероятности перехода $q_n(T_{n+1}|T_n)$ таким образом, чтобы выборка, генерируемая по схеме (3.2), была бы выборкой из интересующего нас распределения $p(T)$.

1.3.1. Схема Метрополиса-Хастинга

Рассмотрим схему генерации нужной выборки *Метрополиса-Хастингса*

Пусть необходимо сгенерировать выборку из распределения p(T), известного с точностью до нормировочной константы:

$$p(T) = \frac{1}{Z}\tilde{p}(T). \tag{1.3.5}$$

Рассмотрим шаг генерации по схеме Метрополиса-Хастингса. Пусть на шаге n сгенерирована конфигурация T_n. Тогда на шаге $n+1$ сначала генерируется конфигурация T_* из некоторого предложного распределения $r(T|T_n)$. Затем вычисляется величина

$$A(T_*, T_n) = min\left(1, \frac{\tilde{p}(T_*)r(T_n|T_*)}{\tilde{p}(T_n)r(T_*|T_n)}\right) \tag{1.3.6}$$

и точка T_* принимается в качестве следующей точки T_{n+1} с вероятностью $A(T_*, T_n)$. В противном случае, $T_{n+1} = T_n$. Таким образом, мы ввели марковскую цепь с вероятностью перехода

$$q(T_{n+1}|T_n) = \begin{cases} r(T_{n+1}|T_n)A(T_{n+1}, T_n), \text{если} T_{n+1} \neq T_n, \\ 1 - r(T_{n+1}|T_n)A(T_{n+1}, T_n), \text{если} T_{n+1} = T_n. \end{cases} \tag{1.3.7}$$

Покажем, что распределение p(T) является инвариантным относительно введенной марковской цепочки. Если $T_{n+1} = T_n$, то инвариантность сохраняется, т.к. значение T_n не изменяется. Для случая $T_{n+1} \neq T_n$ проверим выполнимость уравнения детального баланса:

$$\begin{aligned} p(T_n)q(T|T_n) &= min(p(T_n)r(T|T_n), p(T)r(T_n|T)) \\ &= min(p(T)r(T_n|T), p(T_n)r(T|T_n)) = p(T)q(T_n|T). \end{aligned} \tag{1.3.8}$$

Для эргодичности введенной марковской цепи достаточно потребовать выполнение $r(T|S) > 0, \forall T, S$.

В том случае, если предложное распределение является симметричным, т.е. $r(T|S) = r(S|T), \forall S, T$, то схема Метрополиса-Хастингса переходит в классическую схему Метрополиса. Согласно этой схеме, если значение плотности в новой точке T_* оказалось выше, чем значение плотности в предыдущей точке T_n, то эта точка гарантированно принимается в качестве следующей точки выборки. Если плотность в новой точке оказалась меньше, то такая точка тоже может быть принята, но с вероятностью, пропорциональной величине уменьшения плотности.

1.3.2. Схема Гиббса

Пусть необходимо сгенерировать выборку из многомерного распределения $p(T)$, где $T = \{t_1, \ldots, t_P\}$. Рассмотрим шаг генерации по схеме Гиббса. Пусть на шаге n сгенерирована конфигурация $T^n = \{t_1^p, \ldots, t_p^n\}$ Тогда генерация следующей точки выборки T^{n+1} происходит следующим образом:

$$t_1^{n+1} \sim p(t_1|t_2^n, t_3^n, \ldots t_P^n),$$

$$t_2^{n+1} \sim p(t_2|t_1^{n+1}, t_3^n, \dots, t_P^n),$$

$$t_3^{n+1} \sim p\left(t_3\middle|t_1^{n+1}, t^{(n+1)2}, t_4^n, \dots, t_P^n\right), \qquad (1.3.9)$$

$$\dots$$

$$t_P^{n+1} \sim p(t_P\ |t_1^{n+1}, t_2^{n+1}, \dots, t_{P-1}^{n+1}).$$

Здесь через $p(t_i|T_{\setminus i})$ обозначено маргинальное одномерное распределение значений i-ой компоненты при условии всех остальных. Таким образом, согласно схеме Гиббса генерация выборки из многомерного распределения заменяется на итерационную генерацию точек из одномерных распределений. По аналогии с методами одномерной оптимизации генерация выборки из одномерного распределения является существенно более простой задачей, чем генерация выборки из многомерного распределения.

Докажем, что распределение $p(T)$ является инвариантным относительно введенной марковской цепи. Рассмотрим один шаг генерации очередной компоненты $t_p \sim p(t_p|T_{\setminus p})$. По предположению индукции $T_{\setminus p} \sim (T_{\setminus p})$. Тогда совместная конфигурация $(t_p, T_{\setminus p}) \sim p(t_p|T_{\setminus p})p(T_{\setminus p}) = p(T)$. Отсюда, совместное распределение является инвариантным относительно одного шага процесса генерации (1.3.9). Следовательно, оно является инвариантным и относительно всего процесса (1.3.9).

При реализации схемы Гиббса на практике часто допускается следующая ошибка:
вместо шага

$$t_p^{n+1} \sim p\left(t_p\middle|t_1^{n+1}, \dots, t_{p-1}^{n+1}, t_{p+1}^n, \dots, t_P^n\right), \qquad (1.3.10)$$

делается шаг

$$t_p^{n+1} \sim p\left(t_p\middle|t_1^n, \dots, t_{p-1}^n, t_{p+1}^n, \dots, t_P^n\right), \qquad (1.3.11)$$

т.е. в условие подставляются значения компонент только с предыдущей итерации. При таком подходе вероятность перехода в марковской цепи определяется как

$$q(T|T^n) = \prod_{p=1}^{P} p\left(t_p\middle|T_{\setminus p}^n\right). \qquad (1.3.12)$$

Распределение $p(T)$ не является инвариантным относительно данной марковской цепи. Эту ситуацию легко исправить, если взять схему Метрополиса-Хастингса, где в качестве предложного распределения

фигурирует распределение (1.3.12). Заметим, что в отличие от схемы Гиббса, схема Метрополиса-Хастингса с предложным распределением (1.3.12) легко распараллеливается и на практике в некоторых ситуациях может работать быстрее, чем схема Гиббса.

ГЛАВА 2. Прогнозирование временного ряда на основе моделей авторегрессии

2.1. Прогнозирование по модели авторегресии

Случайный процесс называется *стационарным* (в широком смысле), если его математическое ожидание $m_X(t) = \mu$ и дисперсия $D_X(t) = \sigma^2$ постоянны (значение μ определяет постоянную, вокруг которой лежат значения временного ряда, а σ характеризует ширину полосы, в пределах которой рассеяны эти значения), а автоковариационная функция зависит лишь от разности аргументов:

$$K_X(t, t+\tau) = K_X(\tau). \quad (2.1.1)$$

Стационарность процесса может быть оценена визуально по графику временного ряда или с помощью специальных тестов.

В предположении о нормальности распределения можно разбить ряд на две части, проверить гипотезы о равенстве дисперсий в каждой части по критерию Фишера, затем – о равенстве средних по критерию Стьюдента.

Иногда целесообразно представить значение ряда в момент *t* в виде *авторегрессионной* зависимости: линейной функции от предыдущего наблюдения плюс случайный компонент.

Модель авторегрессии (AutoRegressive) *первого порядка AR(1)* имеет вид

$$x_t = a_0 + a_1 \cdot x_{t-1} + \varepsilon_t, \quad (2.1.2)$$

где a_0, a_1 – числовые коэффициенты ($|a_1| < 1$), ε_t – последовательность величин, образующих *белый шум* (так называется последовательность некоррелированных случайных величин с нулевым математическим ожиданием и постоянной дисперсией).

Свободный член a_0 часто приравнивается нулю, то есть рассматриваются центрированные процессы, средний уровень которых равен нулю – $(x_t - \mu)$.

Прогноз

$$\widetilde{x_{t+1}} = a_0 + a_1 \cdot x_t,$$

$$\widetilde{x_{t+2}} = a_0 + a_1 \cdot \widetilde{x_{t+1}} \text{ и т.д.}$$

Модель авторегрессии порядка *p* AR*(p)* определяется выражением:

$$x_t = a_0 + a_1 \cdot x_{t-1} + \cdots + a_p \cdot x_{t-p} + \varepsilon_t, \quad (2.1.3)$$

Прогноз по модели авторегрессии порядка *p*:

$$\tilde{x}_{t+1} = a_0 + a_1 \cdot x_t + \cdots + a_p \cdot x_{t-p+1},$$
$$\tilde{x}_{t+1} = a_0 + a_1 \cdot \tilde{x}_{t+1} + \cdots + a_p \cdot x_{t-p+1} \text{ и т. п.} \quad (3.1.4)$$

Коэффициенты модели авторегрессии $a_j, j=\overline{0,p}$ определяются методом наименьших квадратов. Значимость модели авторегрессии соответствующего порядка проверяется по критерию Стьюдента.

2.2. Прогнозирование по модели скользящего среднего

Случайную составляющую временного ряда иногда целесообразно представить и в виде взвешенной суммы настоящего и прошлых значений белого шума:

$$x_t = \varepsilon_t - b_1\varepsilon_{t-1} - b_2\varepsilon_{t-2} - \cdots - b_q\varepsilon_{t-q}. \quad (2.2.1)$$

Такой процесс называется *моделью скользящего среднего* (Moving Average) порядка *q* и обозначается *MA(q).*

Как и в случае авторегрессионных моделей, наибольшее распространение на практике имеют модели первого и второго порядка:

$$x_t = \varepsilon_t - b_1\varepsilon_{t-1}, \quad (2.2.2)$$

$$x_t = \varepsilon_t - b_1\varepsilon_{t-1} - b_2\varepsilon_{t-2}, \quad (2.2.3)$$

Часто полезно включить в модель и слагаемые, описывающие авторегрессию, и члены, соответствующие скользящим средним. *Модель авторегрессии – скользящего среднего ARMA(p,q)* имеет вид

$$x_t = a_0 + a_1 \cdot x_{t-1} + \cdots + a_p \cdot x_{t-p} + \varepsilon_t - b_1\varepsilon_{t-1} - \cdots - b_q\varepsilon_{t-q}. \quad (2.2.4)$$

В частности, *ARMA(1,1):*

$$x_t = a_0 + a_1 \cdot x_{t-1} + \varepsilon_t - b_1\varepsilon_{t-1}. \quad (2.2.5)$$

Прогноз выполняется так:

$$\tilde{x}_{t+1} = a_0 + a_1 \cdot x_t - b_1\varepsilon_t, \quad (2.2.6)$$

где

$$\varepsilon_t = x_t - a_0 - a_1 \cdot x_{t-1},$$

$$\tilde{x}_{t+2} = a_0 + a_1 \cdot \tilde{x}_{t+1}, \qquad (2.2.7)$$

$$\tilde{x}_{t+3} = a_0 + a_1 \cdot \tilde{x}_{t+2} \text{ и т.д.}$$

Модель *ARMA*(2,2):

$$x_t = a_0 + a_1 \cdot x_{t-1} + a_2 \cdot x_{t-2} + \varepsilon_t - b_1\varepsilon_{t-1} - b_2\varepsilon_{t-2}. \qquad (2.2.8)$$

Прогноз

$$\begin{gathered}\tilde{x}_{t+1} = a_0 + a_1 \cdot x_t + a_2 \cdot x_{t-1} - b_1\varepsilon_t - b_2\varepsilon_{t-1},\\ \tilde{x}_{t+2} = a_0 + a_1 \cdot \tilde{x}_{t+1} + a_2 \cdot x_t - b_2\varepsilon_t,\\ \tilde{x}_{t+3} = a_0 + a_1 \cdot \tilde{x}_{t+1} + a_2 \cdot \tilde{x}_{t+1},\\ \tilde{x}_{t+4} = a_0 + a_1 \cdot \tilde{x}_{t+3} + a_2 \cdot \tilde{x}_{t+2} \text{ и т.д.}\end{gathered} \qquad (2.2.9)$$

2.3. Модели авторегрессии – проинтегрированного скользящего среднего для нестационарных временных рядов

Мы рассмотрели некоторые модели стационарного временного ряда. Реальные временные ряды обычно *нестационарны*. Однако на практике часто встречаются такие временные ряды, которые стационарны после вычитания из уровней ряда x_t его неслучайной составляющей.

Назовем последовательной разностью первого порядка ряда x_t новый ряд:

$$\Delta x_t = x_t - x_{t-1}; t = 2, \dots, n, \qquad (2.3.1)$$

последовательной разностью второго порядка – разность последовательных разностей первого порядка:

$$\Delta^2 x_t = \Delta(\Delta x_t),$$

последовательной разностью d-го порядка –

$$\Delta^{(d)} x_t = \Delta\left(\Delta^{(d-1)} x_t\right).$$

Предположим, что нестационарную составляющую можно представить в виде полинома степени *(d – 1).* Можно доказать, что переход к

последовательным разностям *d*-го порядка исключает неслучайную составляющую. Будем предполагать, что оставшийся ряд, включающий только случайную составляющую, может быть представлен в виде модели *ARMA(p,q)*.

Тогда получим модель

$$\Delta^{(d)}x_t = a_0 + a_1\Delta^{(\mathrm{d})}\mathrm{x}_{\mathrm{t}-1} + \cdots + \mathrm{a}_{\mathrm{p}}\Delta^{(\mathrm{d})}\mathrm{x}_{\mathrm{t}-\mathrm{p}} + \varepsilon_t - \mathrm{b}_1\varepsilon_{t-1} - \cdots - \mathrm{b}_{\mathrm{q}}\varepsilon_{t-q}, \quad (2.3.2)$$

называемую моделью *авторегрессии – проинтегрированного скользящего среднего* (Auto Regressive Integrated Moving Average), или сокращенно, *ARIMA(p,d,q)*.

Свободный член a_0 в модели (2.3.2) часто приравнивается нулю.

В общем случае сначала подбирают параметр d (обычно $d \le 2$), затем после перехода к стационарной модели проводится идентификация модели *ARMA(p,q)*, при этом, как правило, и $p \le 2$, и $q \le 2$.

Прогноз, например, для модели *ARIMA*(1,1,1), можно найти так:

$$\begin{gathered}\Delta^{(1)}x_t = a_0 + a_1\Delta^{(1)}x_{t-1} + \varepsilon_t - b_1\varepsilon_{t-1},\\ x_t - x_{t-1} = a_0 + a_1(x_{t-1} - x_{t-2}) + \varepsilon_t - b_1\varepsilon_{t-1},\\ x_t = x_{t-1} + a_0 + a_1(x_{t-1} - x_{t-1}) + \varepsilon_t - b_1\varepsilon_{t-1},\end{gathered} \quad (2.3.3)$$

тогда

$$\begin{gathered}\tilde{x}_{t+1} = x_t + a_0 + a_1(x_t - x_{t-1}) - b_1\varepsilon_t,\\ \tilde{x}_{t+2} = \tilde{x}_{t+1} + a_0 + a_1(\tilde{x}_{t+1} - x_t) \text{ и т.д.}\end{gathered} \quad (2.3.4)$$

Для модели *ARIMA*(2, 2, 1) по аналогии:

$$\Delta^{(2)}x_t = a_0 + a_1\Delta^{(2)}x_{t-1} + a_2\Delta^{(2)}x_{t-2} + \varepsilon_t - b_1\varepsilon_{t-1},$$

$$\begin{aligned}&\Delta^{(1)}x_t - \Delta^{(1)}x_{t-1}\\ &\quad = a_0 + a_1\left(\Delta^{(1)}x_{t-1} - \Delta^{(1)}x_{t-2}\right) + a_2\left(\Delta^{(1)}x_{t-2} - \Delta^{(1)}x_{t-3}\right) + \varepsilon_t\\ &\quad - b_1\varepsilon_{t-1},\end{aligned} \quad (2.3.5)$$

$$\begin{aligned}&x_t - x_{t-1} - (x_{t-1} - x_{t-2})\\ &\quad = a_0 + a_1\left(x_{t-1} - x_{t-2} - (x_{t-2} - x_{t-3})\right)\\ &\quad + a_2\left(x_{t-2} - x_{t-3} - (x_{t-3} - x_{t-4})\right) + \varepsilon_t - b_1\varepsilon_{t-1},\end{aligned}$$

$$x_t = 2x_{t-1} + x_{t-2} + a_0 + a_1(x_{t-1} - 2x_{t-2} - x_{t-3}) + a_2(x_{t-2} - 2x_{t-3} - x_{t-4}) + \varepsilon_t - b_1\varepsilon_{t-1},$$

тогда

$$\tilde{x}_{t+1} = 2x_t + x_{t-1} + a_0 + a_1(x_t - x_{t-2}) + a_2(x_{t-1} - x_{t-3}) - b_1\varepsilon_t, \quad (2.3.6)$$

$$\tilde{x}_{t+1} = 2\tilde{x}_{t+1} + x_t + a_0 + a_1(\tilde{x}_{t+1} - x_{t-1}) + a_2(x_t - x_{t-2}) \text{ и т.п.}$$

ГЛАВА 3. Прогнозирование временного ряда

3.1. Подготовка данных

Проведем анализ данных по объемам продаж сети бензоколонок в США в промежутке между январём 1967 года и январь 2001 года. Первым делом нам необходимо понять, с каким рядом мы работаем, наиболее простым методом является построение графика

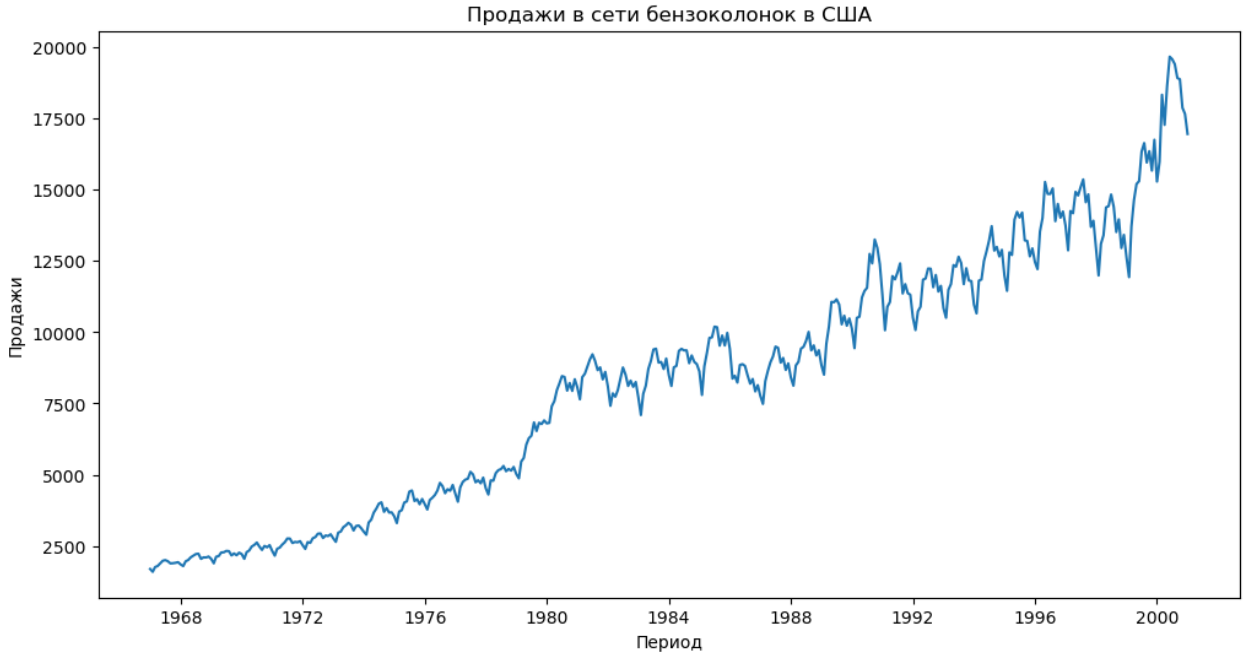

Рисунок 3.1. Временной ряд

На рисунке 3.1 мы отчетливо можем видеть, что у ряда присутствует сезонность, что не соответствует определению стационарного ряда. Для закрепления нашего предположения выполним расширенный тест Дики-Фуллера, который на основе регрессионной модели (3.1.1) выполняет проверку значимости коэффициентов этой модели

$$y_t = a + by_{t-1} + xt + v_1 y_{t-1} + \cdots + v_{p-1} y_{t-p+1} + \varepsilon_t, \qquad (3.1.1)$$

где y – первая разница временного ряда; a – постоянный член; b – коэффициент запаздывающего уровня временного ряда; x – коэффициент временного ряда; v – коэффициент запаздывающих первых разностей; ε – остаточный член.

На основании следующей t-статистики

$$t_\alpha = \frac{\hat{\alpha}}{se(\hat{\alpha})}, \qquad (3.1.2)$$

где $\hat{\alpha}$ – оценка параметра α; $se(\hat{\alpha})$ – стандартная ошибка оценки.

Выполняется проверка гипотез

$H_0: b = 0$, – ряд не стационарен;
$H_1: b < 0$, – ряд стационарен.

Статистика ADF рассчитывается на основе оценочного значения b и его стандартной ошибки. Его сравнивают с критическими значениями распределения Дики-Фуллера. Если статистика ADF более отрицательная, чем критическое значение на определенном уровне значимости, нулевая гипотеза единичного корня отклоняется. Это означает, что ряд стационарен. Рассмотрим ситуацию нашего ряда

Таблица 1

Тест Дики-Фуллера

ADF-статистика	p-value	Критические значения		
		1%	5%	10%
0.715323	0.990138	-3.447229	-2.868979	- 2.570733

Как мы можем заметить в таблице 1, p-value у нас больше 0.05, что позволяет нам отвергнуть гипотезу о стационарности нашего ряда.

Нам необходимо привести ряд к стационарному, для этого можем постараться побороть изменчивую дисперсию путем логарифмирования нашего ряда с использованием, например, преобразования Бокса-Кокса:

$$y = \begin{cases} \frac{y^{\lambda}-1}{\lambda}, \ \lambda = 0 \\ ln\, y, \ \lambda \neq 0 \end{cases}, \qquad (3.1.3)$$

Подбор λ зачастую осуществляется по методу максимального правдоподобия, однако мы осуществим это программно и углубляться в это не будем.

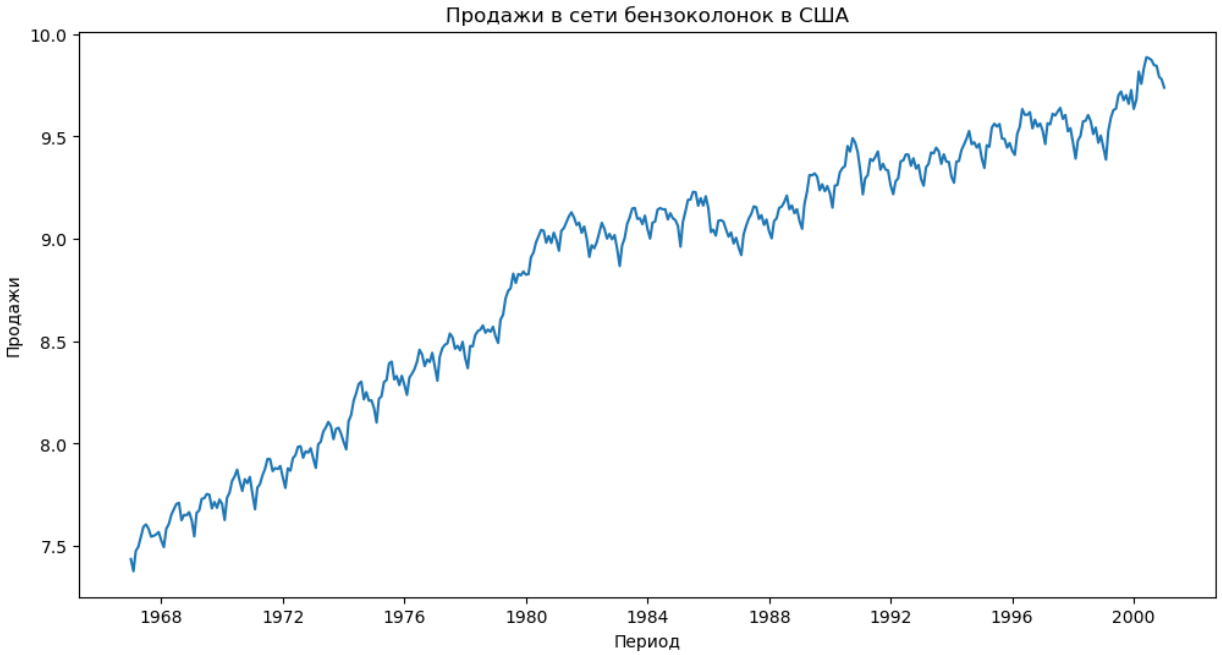

Рисунок 3.2. Прологарифмированный временной ряд

Как можем заметить, логарифмирование улучшило дисперсию, однако сезонность осталась, попробуем избавиться от нее, применив к нему операцию конечной разности, то есть просто вычтем из ряда этот же ряд, смещенный на длину периода, которая в нашем случае равна 12

$$\Delta x_t = x_t - x_{t-12}$$

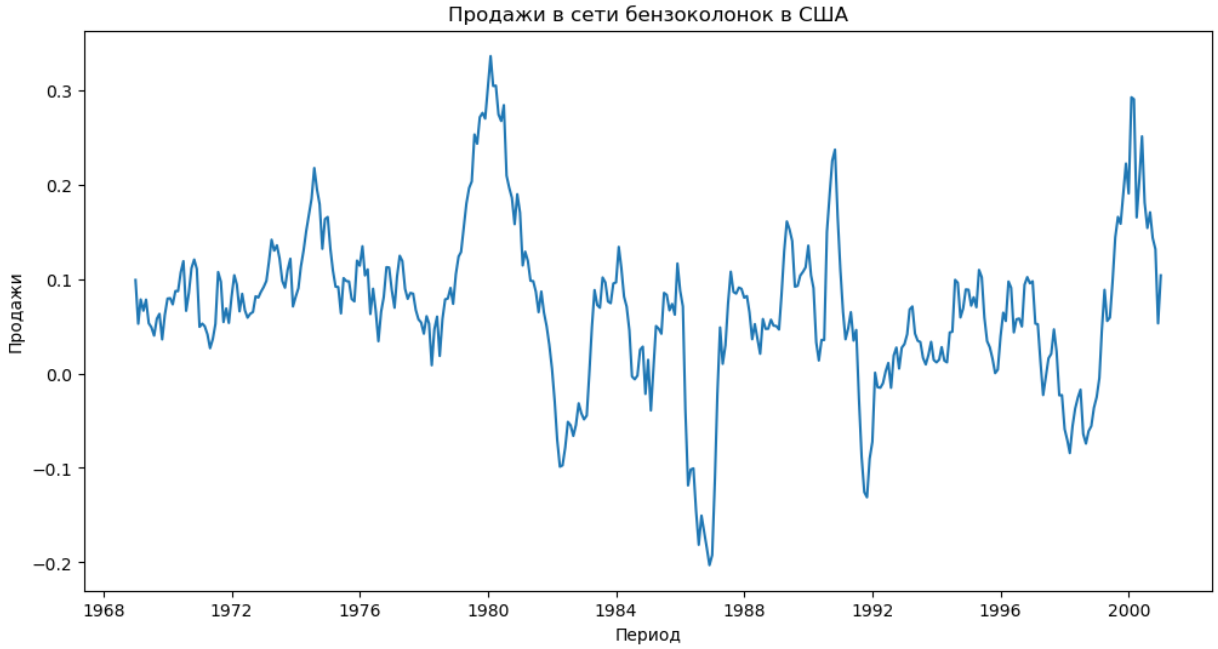

Рисунок 3.3. Продифференцированный временной ряд

Дифференцирование приблизило нас к стационарному ряду, для подтверждения гипотезы о стационарности, выпо.лним тест Дики-Фуллера.

Таблица 2

Тест Дики-Фуллера

ADF-статистика	p-value	Критические значения		
		1%	5%	10%
-3.106534	0.026073	-3.447494	-2.869096	-2.570795

В этот раз тест Дики-Фуллера позволяет нам сделать вывод о стационарности рассматриваемого временного ряда, поскольку $p < 0.05$.

Построим график автокорреляции нашего временного ряда

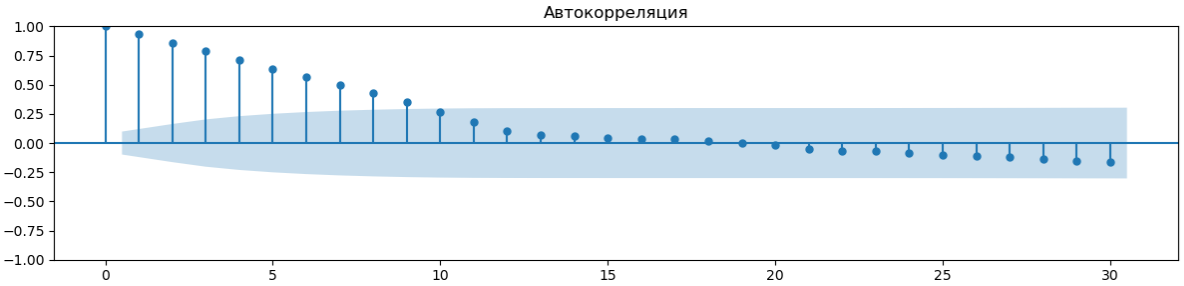

Рисунок 3.4. График автокорреляции

Как можем заметить на рисунке 3.4, наш ряд с течением времени затухает, что указывает на отсутствие у него сезонности, то есть для автоковариационной функции выполняется условие (2.1.1).

Таким образом, мы смогли привести наш временной ряд к стационарному путем одной операции логарифмирования и дифференцирования. Следующим этапом станет прогнозирование этого ряда, которое будем осуществлять с помощью модели (2.3.2) – модели авторегрессии проинтегрированного скользящего среднего.

Помимо этого, определим, по какому закону распределены данные в нашем временно ряде, выполним это наиболее простым методом – построим гистограмму распределения

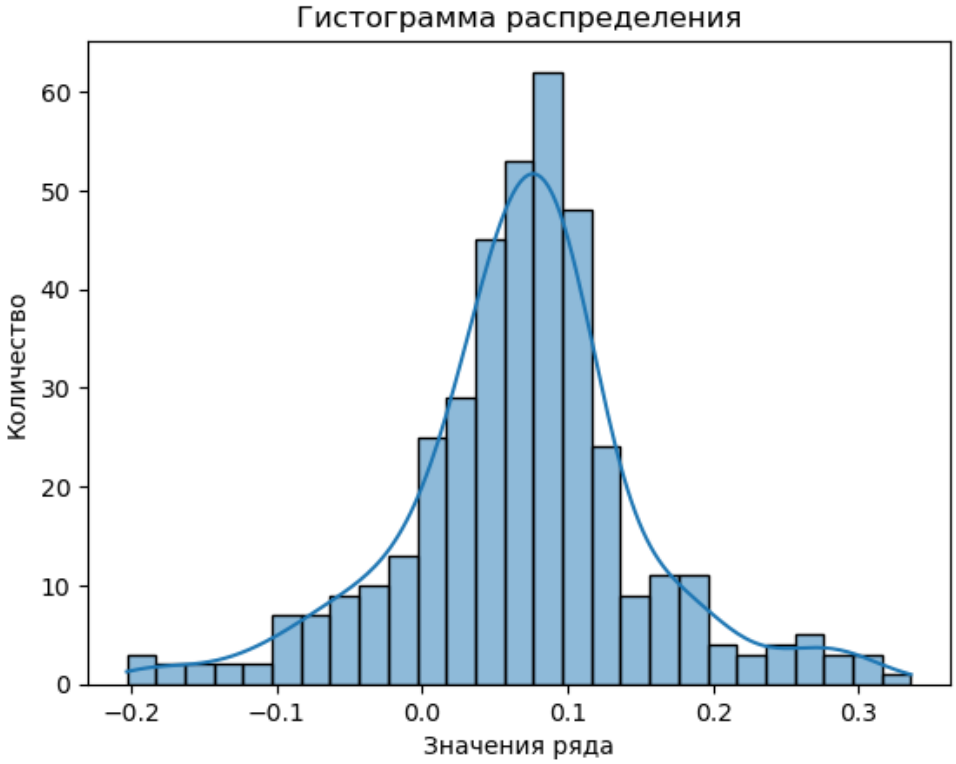

Рисунок 3.5. Гистограмма распределения

Как видно на рисунке 3.5 распределение нашего ряда напоминает нормальное распределение, соответственно, в дальнейшем и будем пользоваться этим распределением.

3.2. Прогнозирование с помощью ARIMA

Первой проблемой, с которой мы столкнемся при прогнозировании при использовании *ARIMA(p, d, q)*, станет нахождение трех целочисленных параметров:

- p – порядок авторегрессии;
- d – порядок дифференцирования исходного временного ряда;
- q – порядок скользящего среднего.

Подбор этих параметров не является тривиальной задачей. Самым простым методом будет банальный перебор различных наборов параметров, так называемый поиск по решетке. Такой перебор должен сопровождаться некоторой оценкой производительности, в нашем случае будем обращаться к информационным критериям, которые оценивают то, на сколько хорошо модель подходит под данные. Будем использовать два критерия

$$AIC = 2k - 2l, \tag{3.2.1}$$

$$BIC = k\,ln\,n - 2l. \tag{3.2.2}$$

Формула (3.2.1) называется информационным критерием Акаика (AIC), в нем

l – значение логарифмической функции правдоподобия построенной модели, k – количество использованных параметров. Формула (3.2.2) является Байесовским информационным критерием (BIC), который является модификацией AIC, разработанный исходя из байесовского подхода, в нем l – значение логарифмической функции правдоподобия построенной модели, k – количество использованных параметров, n – объем выборки.

Будем рассматривать критерий BIC и делать перебор для следующих диапазонов параметров

$$p = \overline{0,9}, q = \overline{0,2}, d = \overline{0,2}.$$

Выполним это программно и получим следующие показатели

$$\mathrm{p} = 8, q = 1, d = 2$$

Значение нашего байесовского информационного критерия следующее:

$$BIC \approx 5795.49$$

Для проверки оптимальности подобранных параметров выведем таблицу с результатами построенной модели.

SARIMAX Results

Dep. Variable:	sales	No. Observations:	397
Model:	ARIMA(8, 1, 2)	Log Likelihood	-2864.851
Date:	Sun, 12 May 2024	AIC	5751.701
Time:	23:04:46	BIC	5795.497
Sample:	0	HQIC	5769.052
	- 397		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	-1.1364	0.060	-19.000	0.000	-1.254	-1.019
ar.L2	-0.6329	0.077	-8.238	0.000	-0.783	-0.482
ar.L3	0.0773	0.075	1.035	0.301	-0.069	0.224
ar.L4	-0.1740	0.079	-2.207	0.027	-0.328	-0.020
ar.L5	-0.1552	0.092	-1.680	0.093	-0.336	0.026
ar.L6	-0.5406	0.102	-5.303	0.000	-0.740	-0.341
ar.L7	-0.4327	0.087	-4.973	0.000	-0.603	-0.262
ar.L8	-0.4078	0.056	-7.280	0.000	-0.518	-0.298
ma.L1	1.5064	0.060	25.217	0.000	1.389	1.623
ma.L2	0.8349	0.050	16.563	0.000	0.736	0.934
sigma2	1.385e+05	9057.338	15.289	0.000	1.21e+05	1.56e+05

Ljung-Box (L1) (Q):	0.25	Jarque-Bera (JB):	176.89
Prob(Q):	0.62	Prob(JB):	0.00
Heteroskedasticity (H):	14.77	Skew:	0.32
Prob(H) (two-sided):	0.00	Kurtosis:	6.21

Рисунок 3.5. Результаты *ARIMA(9,2,2)*

Больше всего нас интересует таблица коэффициентов. Столбец *coef* показывает влияние каждого параметра на временной ряд, а $P > |z|$ – значимость. Чем ближе значение $P > |z|$ к нулю, тем выше значимость.

Для достоверности проверим гипотезу о нормальности распределения остатков для построенной модели, для этого построим несколько графиков.

Начнем с построения гистограммы распределения остатков.

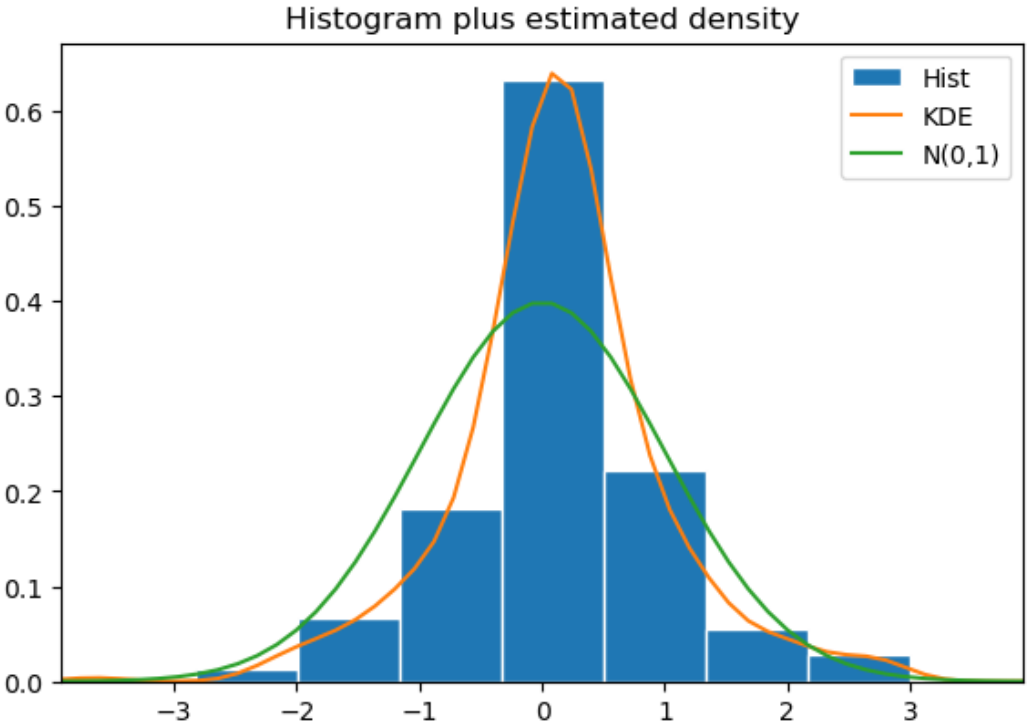

Рисунок 3.6. Гистограмма распределения остатков

На рисунке 3.6 не наблюдается никаких аномалий и можно сказать, что гистограмма напоминает нормальное распределение, соответственно, следующим построим график квантиль-квантиль

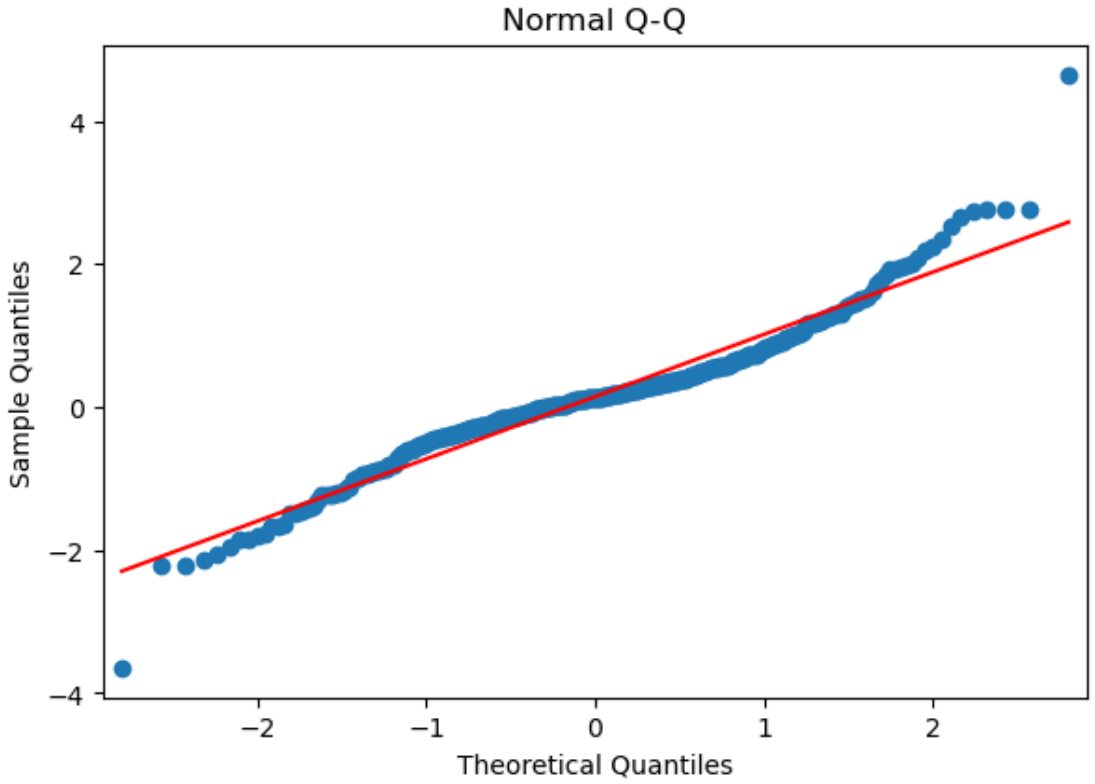

Рисунок 3.7. График квантиль-квантиль для остатков.

Можем заметить, что значения на концах отходят от нормального распределения, что можно обусловить некоторыми нюансами в самих данных.

Перейдем к построению прогноза нашего временного ряда

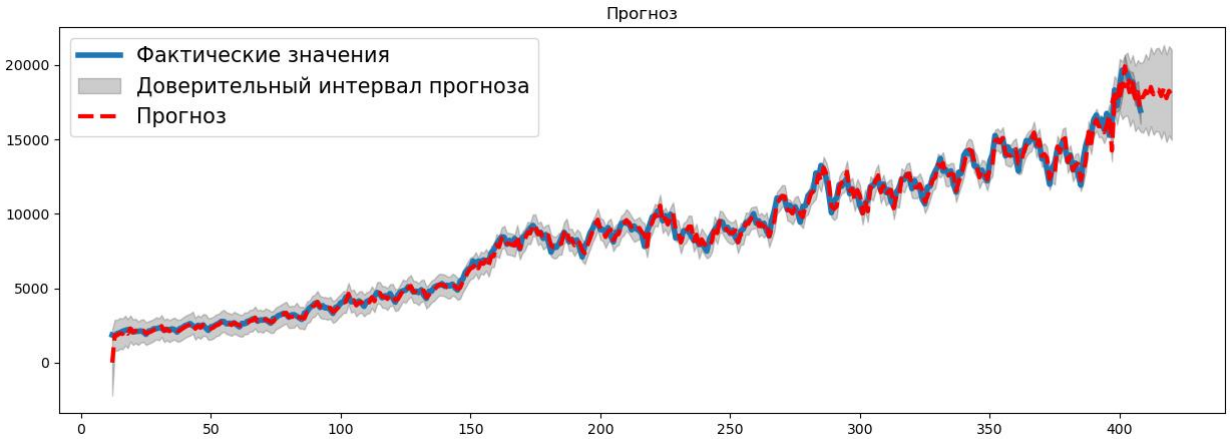

Рисунок 3.8. Прогноз с помощью *ARIMA(8, 1, 2)*

На рисунке 3.8 мы видим, что реальные значения ряда в большинстве случаев попадают в пределы доверительного интервала прогноза ARIMA. У дальнейшего прогноза точность будет уменьшаться с увеличением дальности. Согласно нашему прогнозу, в течение следующих двух лет тенденция с увеличением количества продаж закончится и в этом периоде ожидается “плато”.

Оценим построенную модель с помощью такой метрики, как средний

квадрат ошибки *(MSE)*

$$MSE = \frac{1}{n}\sum_{i=0}^{n}(y_i - \hat{y}_i)^2\,, \tag{3.2.3}$$

где n – количество наблюдений, y_i – фактическое значение временного ряда, $\hat{y}_i$ – предсказанные значения моделью ARIMA(8, 1, 2). Рассматривать будем саму изначальную выборку, для которой данная метрика будет следующей

$$\mathrm{M}SE \approx 116377.67$$

Мы получили не самые выдающиеся результаты, улучшением которых займемся в следующем разделе.

3.3. Байесовский подход

Рассмотрим оценивание параметров нашей модели *ARIMA* с точки зрения байесовского подхода. В отличие от рассмотренного в предыдущем пункте поиска по сетке, данный метод для подбора оптимальных параметров отслеживает результаты прошлых оценок, которые используются для подбора гиперпараметров модели.

Байесовская оптимизация может использоваться для оптимизации каждого отдельного параметра (p, d, q) модели *ARIMA*. Для каждого параметра, процесс байесовской оптимизации может быть следующим:

1. *Определение априорной модели.* Вначале необходимо определить априорную вероятностную модель для каждого параметра. Априорная модель может быть задана в виде распределения параметра. Например, можно предположить, что параметр p следует равномерному распределению на заданном диапазоне значений, то есть его априорное распределение

$$p(\theta_p) = \begin{cases} 0, x \notin [a, b] \\ \dfrac{1}{b-a}, x \in [a, b] \end{cases}, \tag{3.3.1}$$

   где $[a, b]$ – диапазон параметра, θ_p – случайная величина, представляющая параметр p.
2. *Определение целевой функции*. Далее, необходимо определить целевую функцию, которую мы хотим оптимизировать. В случае оптимизации параметра p, целевая функция может основываться на точности прогнозирования модели ARIMA с заданным значением p. Будем рассматривать средний квадрат ошибки

$$f(\theta_p) = \frac{1}{n}\sum_{i=0}^{n}(y_i - \hat{y}_i)^2, \quad (3.3.2)$$

где n – количество наблюдений, y_i – фактическое значение временного ряда, $\hat{y}_i$ – предсказанные значения моделью ARIMA с параметром p равным θ_p.

3. *Апостериорная модель*. Апостериорная модель вычисляется с использованием байесовского правила и обновляет вероятностные представления параметра p на основе имеющихся данных. По формуле Байеса:

$$p(\theta_p|Y) = \frac{p(Y|\theta_p)\cdot p(\theta_p)}{p(Y)} = \frac{p(Y|\theta_p)\cdot p(\theta_p)}{\int p(Y|\theta)p(\theta)d\theta}, \quad (3.3.3)$$

где $p(\theta_p|Y)$ – апостериорное распределение параметра p после учета данных Y, $p(Y|\theta_p)$ – функция правдоподобия, $p(Y)$ – нормализующий множитель (вероятность данных Y).
Функция правдоподобия $p(Y|\theta_p)$ вычисляется следующим образом

$$p(Y|\theta_p) = \prod_{i=1}^{n}\frac{1}{2\pi\sigma^2}e^{-\frac{(y_i-\theta_p)^2}{2\sigma^2}}$$

где y_i – i-й элемент временного ряда, σ^2 – дисперсия временного ряда, который в случае стационарности имеет нормальное распределение.
Подставим в формулу (4.3.3)

$$p(\theta_p|Y) = \frac{\prod_{i=1}^{n}\frac{1}{2\pi\sigma^2}e^{-\frac{(y_i-\theta_p)^2}{2\sigma^2}}\cdot\frac{1}{b-a}}{\prod_{i=1}^{n}\frac{1}{2\pi\sigma^2}e^{-\frac{(y_i-\mu)^2}{2\sigma^2}}} = \frac{\prod_{i=1}^{n}e^{-\frac{(y_i-\theta_p)^2}{2\sigma^2}}}{(b-a)\prod_{i=1}^{n}e^{-\frac{(y_i-\mu)^2}{2\sigma^2}}} = \quad (3.3.4)$$

$$= \frac{1}{b-a}\prod_{i=1}^{n}e^{\frac{(y_i-\mu)^2-(y_i-\theta_p)^2}{2\sigma^2}}.$$

Математическое ожидание и дисперсию посчитаем программно

$$\mu \approx 0.0689, \sigma^2 \approx 0.0068.$$

Подставим эти значения в (4.3.4)

$$p(\theta_p|Y) = \frac{1}{b-a}\prod_{i=1}^{n} e^{\frac{(y_i-0.0689)^2-(y_i-\theta_p)^2}{2\cdot 0.0068}}. \quad (3.3.5)$$

Дальнейшую подстановку выполним после определения промежутка для оценки параметров.

4. *Выбор следующего значения параметра.* Выбор следующего значения параметра p может быть выполнен с использованием различных методов выбора, таких как выбор значения с наибольшей апостериорной вероятностью или методы, основанные на оценке риска и исследовании пространства параметров. Например, можно использовать метод максимальной вероятности (*MAP*), который выбирает значение параметра p, соответствующее наибольшей апостериорной вероятности:

$$\theta_p^* = argmax[p(\theta_p|Y)], \quad (3.3.6)$$

где θ_p^* обозначает оптимальное значение параметра p.

5. *Обновление и повторение.* Выбранное значение параметра p используется для обновления модели *ARIMA* и вычисления новой апостериорной модели. Затем процесс повторяется для следующего параметра (например, d и q), чтобы оптимизировать их значения.

Весь процесс байесовской оптимизации выполняется итеративно, где на каждой итерации обновляются апостериорные модели и выбираются новые значения параметров, пока не будет достигнуто условие остановки или найдено оптимальное значение параметров p, d, q, соответствующее оптимальной модели *ARIMA*.

Как уже было сказано, главным преимуществом байесовской оптимизации над условным поиском по сетке является скорость работы. Однако это действительно ощущается случаев, когда мы работаем с большим объемом количества параметров, так, если, например, предположить, что параметры нашей модели находятся в следующих промежутках

$$\mathrm{p} = \overline{0{,}20}, d = \overline{0{,}20}, q = \overline{0{,}20},$$

то алгоритм поиска по сетке должен перебрать $20 \cdot 20 \cdot 20 = 8000$ комбинаций, что, если считать, что на одну комбинацию программа потратит 1 секунду, займет больше 2-ух часов.

В случае Байесовской оптимизации вернемся к формуле (3.3.5) и выполним подстановку значений

$$p(\theta_p|Y) = \frac{1}{20}\prod_{i=1}^{n} e^{\frac{(y_i-0.0689)^2-(y_i-\theta_p)^2}{2\cdot 0.0068}}.$$

Дальнейший этап по оценке параметров методом MAP выполним программно и получим

$$p = 18, d = 1, q = 15$$

При этом лучшее значение целевой функции было

$$\mathrm{M}SE \approx 65275.27$$

Как мы видим, результат в сравнении с моделью ARIMA(8, 1, 2) стал лучше практически в два раза, однако он не является идеальным, что можно объяснить недостаточным интервалом для параметров.

При этом график уменьшения целевой функции имеет следующий вид

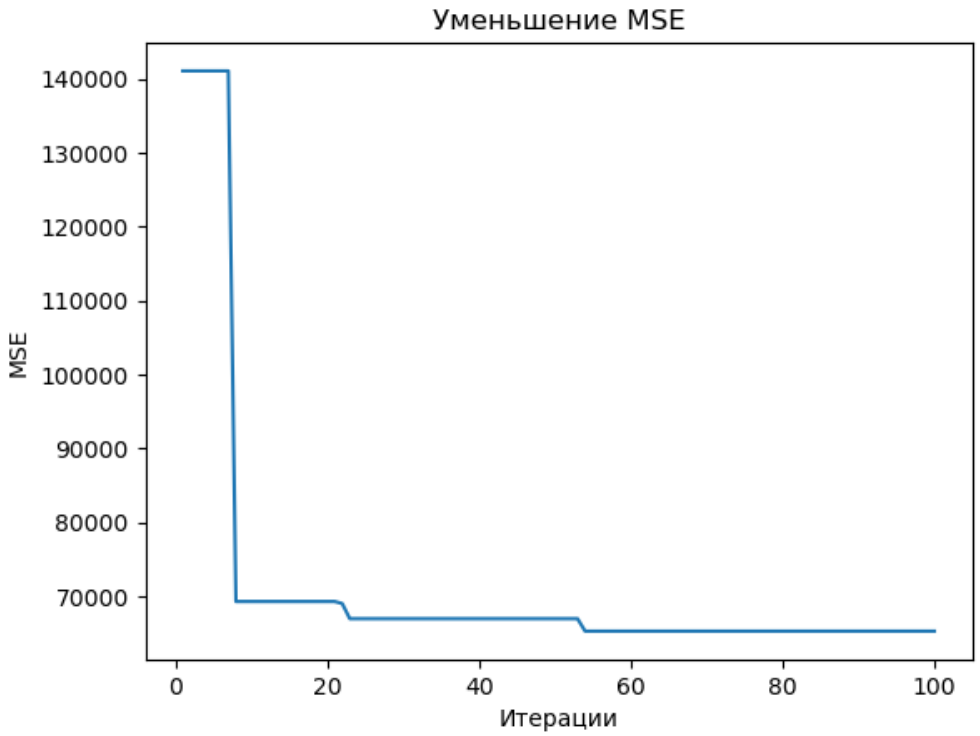

Рисунок 3.9. График уменьшения MSE

Попробуем построить прогноз с помощью модели *ARIMA(18, 1, 15).*

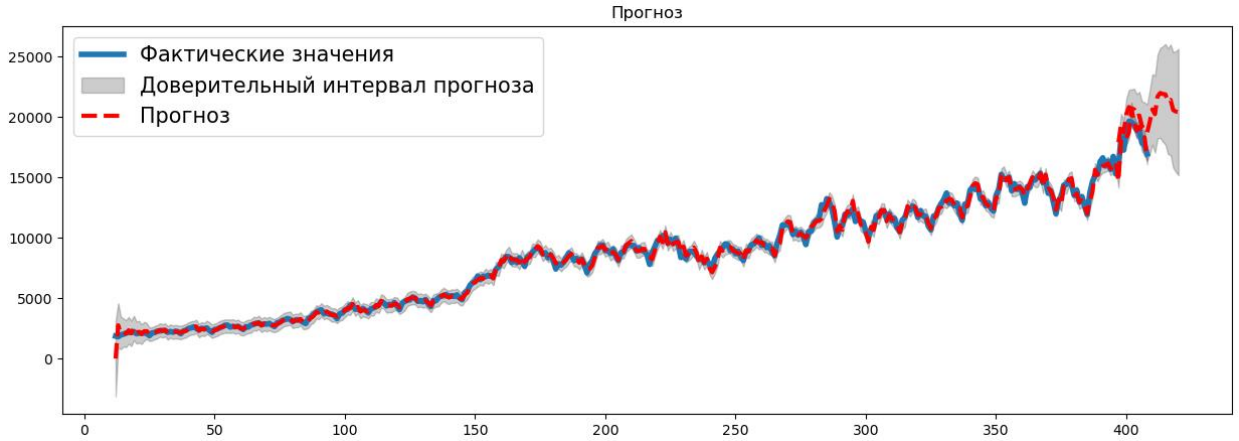

Рисунок 3.10. Прогноз с помощью *ARIMA(18, 1, 15)*

Если мы вспомни рисунок 3.8 и попробуем сравнить полученный результат с данным, то можем увидеть, что итоговый прогноз на последующие два года у нас изменился, в данном случае прогноз указывает на то, что количество продаж увеличится.

3.4. Вывод

По итогам проведенной работы можно сказать, что байесовский оптимизатор позволил нам достичь лучших результатов прогнозирования в равнении с подбором параметров по методу поиска по сетке.

В целом, байесовский подход открывает нам больше возможностей для прогнозирования, поскольку значительно ускоряет процесс подбора параметров, что позволяет рассматривать более широкие промежутки для подбора параметров.

Однако, сколько бы времени не было потрачено, нужно помнить, что прогнозирование стационарных временных рядов всё еще остается весьма сложной задачей, ведь зачастую реальные данные зависят от многих факторов, в то время как рассматриваемый набор данных содержит лишь прогнозируемую величину и даты.

ЗАКЛЮЧЕНИЕ

В ходе работы были получены следующие результаты

- Дан обзор методов машинного обучения
- Исследован байесовский подход в машинном обучении
- Рассмотрены авторегрессионные модели прогнозирования
- Байесовский подход для авторегрессионных моделей
- Разработано программное приложение на языке Python, реализующее анализ временного ряда, прогнозирование его с помощью модели ARIMA и байесовский подход к модели ARIMA.

СПИСОК ИСПОЛЬЗОВАНЫХ ИСТОЧНИКОВ

1. Кувайскова, Ю. Е Статистические методы прогнозирования / Ю. Е. Кувайскова, В. Н. Клячкин/ – Минск: УГТУ, 2019.
2. Ивченко, Г. И. Математическая статистика: Учеб. Пособие для втузов / Ивченко Г. И., Медведев Ю. И./ – Москва: МВТУ им. Н. Э. Баумана, 1984.
3. Грэйнджер, Клайв У. Дж. Эконометрический анализ временных рядов / Клайв У. Дж. Грэйнджер.
4. Абрамова, А. Е. Метод Монте Карло по схеме марковской цепи для оценки вероятности редких событий в задачах биоинформатики / Абрамова А. Н./ – Санкт-Петербург: СПБГУ, 2017.
5. Временной ряд – Википедия: [Электронный ресурс]. URL: https://ru.wikipedia.org/ (Дата обращения: 26.11.2023).
6. Градиентный спуск: всё, что нужно знать: [Электронный ресурс]. URL: https://neurohive.io/ (Дата обращения: 10.12.2023).
7. Теорема Байеса [3Blue1Brown] – YouTube: [Электронный ресурс]. URL: https://www.youtube.com/ (Дата обращения: 02.12.2023)
8. Hamilton, J. D. Time Series Analysis / James D. Hamilton/ – Princeton, New Jersey: Princeton University, 1994.
9. A Conceptual Explanation of Bayesian Hyperparameter Optimization for Machine Learning: [Электронный ресурс]. URL: https://towardsdatascience.com/a-conceptual-explanation-of-bayesian-model-based-hyperparameter-optimization-for-machine-learning-b8172278050f (Дата обращения: 08.05.2024).
10. Tuning ARIMA for Forecasting: An Easy Approach in Python: [Электронный ресурс]. URL: https://medium.com/@sandha.iitr/tuning-arima-for-forecasting-an-easy-approach-in-python-5f40d55184c4 (Дата обращения: 08.05.2024).
11. Оптимизация гиперпараметров — Википедия: [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki (Дата обращения: 08.05.2024).
12. Подбор гиперпараметров: [Электронный ресурс]. URL: https://education.yandex.ru/handbook/ml/article/podbor-giperparametrov (Дата обращения: 09.05.2024).

Приложение

Листинг кода приложения на языке Python:

```
import pandas as pd
import numpy as np
import seaborn as sns
import time
import itertools
import matplotlib.pyplot as plt
from scipy import stats
from scipy.special import inv_boxcox
from statsmodels.tsa.stattools import adfuller
from statsmodels.tsa.arima.model import ARIMA
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
from hyperopt import fmin, tpe, hp, Trials
ds = pd.read_excel(r'RETAIL.xlsx', decimal=',')
df = pd.DataFrame(ds, columns=['date', 'sales'])
month_dict = {'Янв': 1, 'Фев': 2, 'Мар': 3, 'Апр': 4, 'Май': 5, 'Июн': 6, 'Июл': 7, 'Авг': 8, 'Сен': 9, 'Окт': 10, 'Ноя':
11, 'Дек': 12}
df['date'] = df['date'].apply(lambda x: pd.to_datetime(f"{month_dict[x.split('-')[0]]}-{x.split('-')[1]}"))
df.to_excel('data.xlsx')
plt.figure(figsize=(12, 6))
plt.plot(df['date'],df['sales'])
plt.title('Продажи в сети бензоколонок в США')
plt.xlabel('Период')
plt.ylabel('Продажи')
plt.show()
result = adfuller(df['sales'])
print('ADF статистика:', result[0])
print('p-значение:', result[1])
print('Критические значения:')
for key, value in result[4].items():
    print(f'  {key}: {value}')
df['log_sales'] = np.log(df['sales'])
plt.figure(figsize=(12, 6))
plt.plot(df['date'], df['log_sales'])
plt.title('Продажи в сети бензоколонок в США')
plt.xlabel('Период')
plt.ylabel('Продажи')
plt.show()
result = adfuller(df['log_sales'])
print('ADF статистика:', result[0])
print('p-значение:', result[1])
print('Критические значения:')
for key, value in result[4].items():
    print(f'  {key}: {value}')
df['time_series_diff_1'] = df['log_sales'] - df['log_sales'].shift(12)

df.dropna(inplace=True)
plt.figure(figsize=(12, 6))
plt.plot(df['date'], df['time_series_diff_1'])
plt.title('Продажи в сети бензоколонок в США')
plt.xlabel('Период')
plt.ylabel('Продажи')
plt.show()
result = adfuller(df['time_series_diff_1'])
print('ADF статистика:', result[0])
print('p-значение:', result[1])
print('Критические значения:')
```

```
for key, value in result[4].items():
    print(f'  {key}: {value}')
plt.figure(figsize=(12, 6))
plt.subplot(211)
plot_acf(df['time_series_diff_1'], lags=30, ax=plt.gca())
plt.title('Автокорреляция')
plt.tight_layout()
plt.show()
sns.histplot(df['time_series_diff_1'], kde='norm')
plt.xlabel('Значения ряда')
plt.ylabel('Количество')
plt.title('Гистограмма распределения')
plt.show()
print('Математическое ожидание:',df['time_series_diff_1'].mean())
print('Дисперсия временного ряда:',variance(df['time_series_diff_1']))
%%time
p = range(0,10)
d = range(0,3)
q = range(0,3)
pdq = list(itertools.product(p, d, q))
best_pdq = (0,0,0)
best_bic = np.inf
for params in pdq:
    model_test = ARIMA(df['sales'], order = params)
    result_test = model_test.fit()
    if result_test.bic < best_bic:
        best_pdq = params
        best_bic = result_test.bic
p, d, q = best_pdq
print(f'Порядок авторегресии: {p}\nПорядок дифференцирования: {d}\nПорядок скользящего среднего: {q}')
print(f'Наилучшее значение BIC: {best_bic}')
model = ARIMA(df['sales'], order=(p, d, q))
model_fit = model.fit()
print(model_fit.summary())
model = ARIMA(df['sales'], order=(p, d, q))
model_fit = model.fit()
print(model_fit.summary())
forecast = model_fit.get_prediction()
forecast_ci = forecast.conf_int()
plt.figure(figsize=(15,5))
plt.plot(df['sales'], label="Фактические значения", linewidth=4)
plt.fill_between(forecast_ci.index,
forecast_ci.iloc[:, 0],
forecast_ci.iloc[:, 1], color='k', alpha=.2, label='Доверительный интервал прогноза')
plt.plot(forecast.predicted_mean, color='red', label="Прогноз", linewidth=3)
forecast = model_fit.get_forecast(steps=24)
plt.plot(forecast.predicted_mean, color='red', linestyle='dashed', linewidth=3)
forecast_ci = forecast.conf_int()
plt.fill_between(forecast_ci.index,
forecast_ci.iloc[:, 0],
forecast_ci.iloc[:, 1], color='k', alpha=.2)
plt.legend(fontsize=15)
plt.title('Прогноз')
plt.show()
forecast = model_fit.predict()
mse = mean_squared_error(df['sales'], forecast)
print(f'MSE: {mse}')
%%time
def arima_objective_function(params):
    global data_values
    p, d, q = params
```

```
    try:
        model = ARIMA(data_values, order=(p, d, q))
        predictions = model.fit()
        y_pred = predictions.predict()
        mse = mean_squared_error(data_values, y_pred)
    except:
        mse = float('inf')
    return mse
param_space = [hp.choice('p', range(0, 30)), hp.choice('d', range(0, 30)), hp.choice('q', range(0, 30))]
data_values = df['sales']
trials = Trials()
best = fmin(fn=arima_objective_function, space=param_space, algo=tpe.suggest, max_evals=100, trials=trials)
print(best)
best_loss = trials.best_trial['result']['loss']
print('best loss:', best_loss)
p, d, q = best['p'], best['d'], best['q']
model = ARIMA(data_values, order=(p, d, q))
forecast = model_fit.get_prediction()
forecast_ci = forecast.conf_int()
plt.figure(figsize=(15,5))
plt.plot(df['sales'], label="Фактические значения", linewidth=4)
plt.fill_between(forecast_ci.index,
forecast_ci.iloc[:, 0],
forecast_ci.iloc[:, 1], color='k', alpha=.2, label='Доверительный интервал прогноза')
plt.plot(forecast.predicted_mean, color='red', label="Прогноз", linestyle='dashed', linewidth=3)
forecast = model_fit.get_forecast(steps=24)
plt.plot(forecast.predicted_mean, color='red', linestyle='dashed', linewidth=3)
forecast_ci = forecast.conf_int()
plt.fill_between(forecast_ci.index,
forecast_ci.iloc[:, 0],
forecast_ci.iloc[:, 1], color='k', alpha=.2)
plt.legend(fontsize=15)
plt.title('Прогноз')
plt.show()
losses = [trial['result']['loss'] for trial in trials.trials]
best_losses = [min(losses[:i+1]) for i in range(len(losses))]
plt.plot(range(1, len(losses)+1), best_losses)
plt.xlabel('Итерации')
plt.ylabel('MSE')
plt.title('Уменьшение MSE')
plt.show()
```