

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Иркутский государственный университет»
(ФГБОУ ВО «ИГУ»)
Институт математики и информационных технологий
Кафедра алгебраических и информационных систем

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА МАГИСТРА
по направлению «02.04.02 Фундаментальная информатика и
информационные технологии»
профиль подготовки «Анализ данных научных исследований и
машинное обучение»

РАЗРАБОТКА ИНТЕРАКТИВНОГО ПРОГРАММНОГО ИНСТРУМЕНТА
ДЛЯ АНАЛИЗА И ПРОГНОЗИРОВАНИЯ ВРЕМЕННЫХ РЯДОВ

Студент 2 курса очного отделения
Группа 02271–ДМ
Танкович Артём Витальевич

Руководитель:
к.ф.-м.н., доцент
_____ Казимиров А.С.

Допущена к защите
Зав. кафедрой АиИС, д.ф.-м.н., доцент
_____ Пантелеев В.И.

Иркутск 2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	3
Глава 1. Теоретический обзор	6
1.1. Понятие временного ряда, свойства и компоненты	6
1.2. Задачи анализа временных рядов	9
1.3. Классические статистические методы прогнозирования	11
1.4. Машинное обучение и глубокое обучение	18
1.5. Гибридные модели (SARIMA-LSTM)	26
1.6. Обоснование выбора моделей для реализации в приложении	28
Глава 2. Требования и реализация приложения	30
2.1. Требования к приложению	30
2.2. Выбор технологического стека и его обоснование	35
2.3. Архитектура приложения	37
2.4. Разработка интерфейса пользователя	39
2.5. Модули обработки потока данных	43
2.6. Реализация оценки качества прогнозов	45
Глава 3. Результаты работы приложения	47
3.1. Данные для анализа	47
3.2. Исследование результатов моделей прогнозирования в приложении	48
3.3. Итоговые результаты и выбор модели	55
3.4. Выбор модели	56
Заключение	57
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	58
ПРИЛОЖЕНИЕ 1. Структурная схема программы	63

ВВЕДЕНИЕ

В современном мире наблюдается экспоненциальный рост объемов данных, среди которых временные ряды занимают значительное место в самых разных областях — от экономики и финансов до науки и техники. Эффективный анализ последовательностей данных, упорядоченных во времени, позволяет выявлять скрытые закономерности, строить обоснованные прогнозы и, как следствие, принимать более взвешенные управленческие и стратегические решения.

Анализ временных рядов стал неотъемлемой частью деятельности плановых, аналитических и маркетинговых подразделений. Однако практическое применение современных и сложных методов анализа временных рядов часто сопряжено с рядом трудностей. Оно требует от пользователя не только глубоких знаний в области статистики и машинного обучения, но и навыков программирования, а также значительных временных затрат на подготовку данных, выбор и настройку моделей. Существующие программные решения зачастую либо являются узкоспециализированными и дорогостоящими, либо представляют собой библиотеки, требующие написания кода, что создает существенный барьер для широкого круга специалистов, нуждающихся в инструментах прогнозирования.

Актуальность данной работы обусловлена необходимостью создания доступного, интерактивного и удобного инструмента, который автоматизирует рутинные этапы анализа и прогнозирования временных рядов. Такой инструмент должен предоставлять пользователю наглядные результаты и возможность гибкого применения различных моделей — от классических статистических до современных нейросетевых и гибридных подходов, без требования глубокого погружения в их математический аппарат или написания программного кода. Разработка такого приложения позволит снизить порог входа для проведения анализа временных рядов и повысить эффективность работы специалистов в различных предметных областях.

Целью работы является разработка и реализация прототипа настольного автоматизированного приложения с графическим пользовательским интерфейсом для анализа и прогнозирования временных рядов.

Для достижения поставленной цели были решены следующие задачи:

- Проведен анализ требований к функциональности и интерфейсу приложения для анализа временных рядов.
- Выбран и обоснован технологический стек для разработки (язык Python и библиотеки PyQt5, Pandas, NumPy, Plotly, Statsmodels, TensorFlow/Keras, Prophet, XGBoost, Scikit-learn).
- Реализован графический пользовательский интерфейс (GUI) с использованием фреймворка PyQt5, обеспечивающий интуитивное взаимодействие пользователя с приложением.
- Реализованы модули загрузки (включая Drag-and-Drop) и предварительной обработки данных из CSV-файлов, включая выбор релевантных столбцов и установку частоты ряда.
- Интегрированы средства визуализации данных на основе Plotly — библиотеки для построения интерактивных графиков временных рядов, результатов прогнозирования и разделения выборок.
- Реализовано разделение временного ряда на обучающую и тестовую выборки для оценки качества моделей.
- Интегрирован набор разнообразных моделей прогнозирования:
 - Количественные: экспоненциальное сглаживание и SARIMA.
 - Машинного обучения: LSTM, XGBoost, Prophet.
 - Гибридные: SARIMA-LSTM.
- Реализованы диалоговые окна, позволяющие пользователю настраивать параметры моделей в зависимости от задачи прогнозирования.

- Реализован расчет и отображение ключевых метрик качества прогноза.
- Проведено базовое тестирование и отладка реализованного приложения.

Объектом исследования являются методы и средства анализа и прогнозирования временных рядов.

Предметом исследования является процесс разработки настольного приложения для автоматизированного анализа и прогнозирования временных рядов с использованием количественных, нейросетевых и гибридных моделей.

Научная новизна работы заключается в разработке и реализации прототипа программного средства, которое интегрирует в рамках единого графического интерфейса пользователя широкий спектр современных методов прогнозирования временных рядов. Предоставление возможности настройки и сравнения различных моделей в интерактивном режиме без необходимости программирования является ключевым элементом.

Практическая значимость разработанного прототипа состоит в предоставлении специалистам из различных областей доступного и удобного инструмента для проведения анализа временных рядов и построения прогнозов. Приложение снижает требования к специальным знаниям в области программирования и статистики, автоматизирует рутинные операции и позволяет сравнивать эффективность различных подходов к прогнозированию на конкретных данных. Прототип также представляет собой расширяемую платформу для дальнейшего развития и добавления новых методов анализа и прогнозирования.

Глава 1. Теоретический обзор

Анализ временных рядов является важной областью статистики и эконометрики, занимающейся анализом последовательностей данных, упорядоченных во времени. Временные ряды возникают в самых разнообразных областях: экономика и финансы, например ВВП, курсы валют, цены на акции, метеорология: температура, осадки, медицина: ЭКГ, данные о пациентах, инженерия: данные с датчиков и многие другие [30; 43]. Понимание структуры и динамики временных рядов позволяет не только описывать прошлые события, но и прогнозировать будущее, а также управлять процессами.

1.1. Понятие временного ряда, свойства и компоненты

Определение временного ряда

Временной ряд представляет собой последовательность наблюдений одной или нескольких переменных, сделанных в последовательные моменты времени, обычно через равные промежутки [39]. Формально, временной ряд $\{Y_t\}$ — это набор значений $Y_{t_1}, Y_{t_2}, \dots, Y_{t_n}$, где t_i — момент времени и $t_1 < t_2 < \dots < t_n$. Одной из ключевых характеристик временного ряда является то, что порядок наблюдений играет важную роль, поскольку данные в таких рядах упорядочены по времени, и значения, расположенные рядом во временной шкале, как правило, обладают статистической зависимостью. Это означает, что текущее значение временного ряда часто зависит от предыдущих, что необходимо учитывать при анализе и построении моделей прогнозирования. [5].

Основные свойства временных рядов, которые необходимо обязательно учитывать при их анализе, представляют из себя следующие характеристики:

Стационарность

Стационарный временной ряд — это ряд, статистические свойства которого не изменяются с течением времени [1].

Автокорреляция

Это корреляция между значениями временного ряда и их лаговыми значениями. Функция автокорреляции (ACF) позволяет оценить наличие трендов, сезонности и определить порядок AR и MA в ARIMA [5].

Гетероскедастичность

Свойство, при котором дисперсия ошибки изменяется с течением времени и не остаётся постоянной. Оно характеризует нестабильность разброса значений вокруг среднего уровня и может существенно влиять на качество анализа и построения моделей временных рядов. Особенно часто это явление наблюдается в финансовых временных рядах, таких как котировки акций, валютных курсов или уровней рыночной волатильности, где периоды высокой нестабильности чередуются с относительно спокойными интервалами [2; 4].

Долгосрочная зависимость (долгая память)

Свойство ряда сохранять корреляцию даже между значительно удалёнными наблюдениями. Такое поведение характеризуется медленно затухающей автокорреляционной функцией [10].

Компоненты временного ряда

Традиционно временной ряд рассматривается как сумма или произведение нескольких компонент, отражающих закономерности в данных [39]:

- 1) Тренд (T_t) — долгосрочное изменение уровня ряда (восходящий, нисходящий или отсутствующий). Может быть линейным или нелинейным. Методы выделения: скользящее среднее, регрессия, сглаживание.
- 2) Сезонность (S_t) — периодические колебания фиксированной длительности (год, квартал, неделя). Часто наблюдаются в продажах, трафике и т.п. [29].
- 3) Цикличность (C_t) — долгосрочные нерегулярные волнообразные колебания, часто связанные с экономическими циклами.

- 4) Шум (E_t или I_t) — случайные, нерегулярные компоненты, не объяснённые трендом, сезонностью и цикличностью. Предполагается, что шум — это стационарный процесс с нулевым средним и постоянной дисперсией [35].

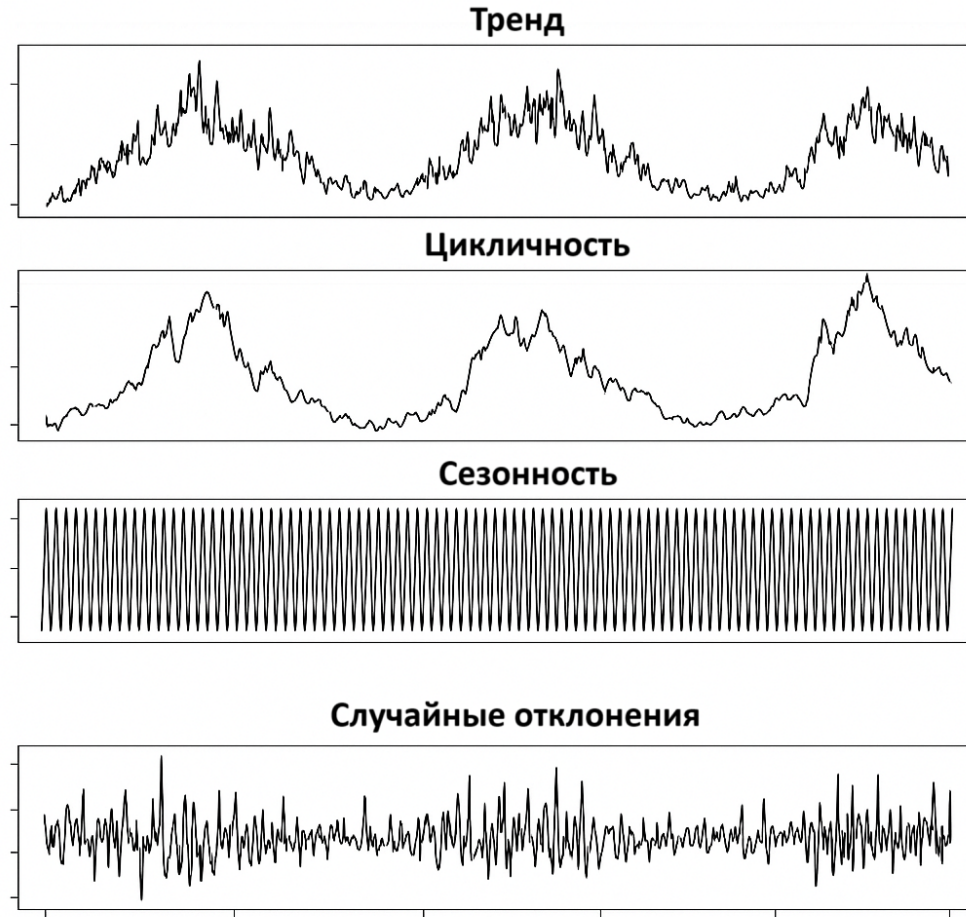


Рисунок 1. Компоненты временного ряда

Модели декомпозиции

- Аддитивная модель:

$$Y_t = T_t + S_t + C_t + E_t \quad (1)$$

В этой модели предполагается, что амплитуда сезонных и циклических колебаний, а также величина шума, не зависят от уровня тренда. Она подходит для рядов, где сезонные колебания остаются примерно постоянными по величине, независимо от уровня ряда [26].

- Мультипликативная модель:

$$Y_t = T_t \cdot S_t \cdot C_t \cdot E_t \quad (2)$$

Предполагается, что амплитуда сезонных и циклических колебаний, а также величина шума пропорциональны уровню тренда. Такая модель часто используется, когда сезонные колебания увеличиваются или уменьшаются вместе с ростом или падением тренда [19]. Мультипликативную модель часто можно преобразовать в аддитивную путём логарифмирования:

$$\log Y_t = \log T_t + \log S_t + \log C_t + \log E_t \quad (3)$$

Выбор между аддитивной и мультипликативной моделью зависит от характера данных. Декомпозиция помогает лучше понять структуру ряда и может использоваться как предварительный шаг для более сложного моделирования или прогнозирования [41].

1.2. Задачи анализа временных рядов

Анализ временных рядов преследует четыре ключевые цели [39; 44]:

- 1) Описание — направлено на выявление структуры временного ряда включающее следующие его характеристики:
 - визуализацию данных для выявления трендов, сезонности, циклическости, выбросов и структурных изменений;
 - расчёт описательных статистик, таких как среднее, дисперсия и др.;
 - анализ автокорреляции с использованием ACF и PACF;
 - декомпозицию ряда на основные компоненты: тренд, сезонность, циклическость и случайные остатки.

2) Моделирование — это процесс построения математической или статистической модели, которая наиболее точно описывает поведение временного ряда на основе имеющихся данных:

- выбор класса моделей (например, ARIMA, экспоненциальное сглаживание, нейросетевые модели);
- идентификация параметров (например, порядки p , d , q в ARIMA);
- оценка параметров по наблюдаемым данным;
- диагностика остатков модели для проверки её адекватности, остатки должны быть белым шумом [6; 12].

3) Прогнозирование — направлено на получение будущих значений ряда:

- построение точечных и интервальных прогнозов;
- оценка точности прогноза, обычно с использованием различных метрик: MAE, MSE, RMSE, MAPE.

4) Управление — используется в задачах, где необходимо воздействовать на будущие значения ряда через управляемые переменные. Применяется, например, в производственных процессах и системах автоматического регулирования, где необходимо отслеживать и предсказывать поведение ключевых показателей во времени для обеспечения стабильной работы. [9].

Понимание временного ряда, его ключевых свойств и компонент является основой для его успешного анализа.

В зависимости от целей исследования, анализ временных рядов позволяет описывать динамику процессов, объяснять их поведение с помощью математических моделей, прогнозировать будущие значения и, в некоторых случаях, управлять этими процессами [18]. Методы, такие как декомпозиция и ARIMA, остаются основой анализа, а современные подходы на базе машинного обучения дополняют их при работе со сложными и нелинейными временными рядами.

1.3. Классические статистические методы прогнозирования

Экспоненциальное сглаживание

Методы экспоненциального сглаживания являются одними из наиболее широко используемых и успешных подходов к прогнозированию временных рядов. Их популярность обусловлена простотой, хорошей точностью на многих типах рядов и интуитивной понятностью. Основная идея заключается в том, что при прогнозировании будущих значений временного ряда более значимый вес придается недавним наблюдениям, поскольку они обычно лучше отражают текущие тенденции и изменения, чем более старые данные. Веса убывают экспоненциально по мере "старения" данных [23].

Существует несколько вариантов экспоненциального сглаживания, предназначенных для различных характеристик временных рядов. Как отмечает автор [38], методы экспоненциального сглаживания могут рассматриваться как частные случаи моделей пространства состояний (State Space Models), что обеспечивает им прочную теоретическую основу и позволяет вычислять доверительные интервалы для прогнозов.

Простое экспоненциальное сглаживание

Метод простого экспоненциального сглаживания (Simple Exponential Smoothing, SES) предназначен для анализа временных рядов без выраженного тренда и сезонности, колеблющихся вокруг стабильного среднего уровня [8].

Прогноз на следующий период формируется как взвешенное среднее между последним наблюдением временного ряда и предыдущим сглаженным значением, позволяющее учитывать как актуальные данные, так и тенденции, выявленные на основе прошлых значений

- Сглаживание:

$$S_t = \alpha Y_t + (1 - \alpha)S_{t-1} \quad (4)$$

где:

- S_t — сглаженное значение (оценка уровня) в момент времени t ;
- Y_t — фактическое значение ряда в момент t ;
- α — параметр сглаживания, $0 \leq \alpha \leq 1$.

- Прогноз:

$$\hat{Y}_{t+h|t} = S_t \quad h > 1 \quad (5)$$

Таким образом, прогноз остаётся постоянным на любом горизонте.

Параметр сглаживания α :

- Определяет реакцию модели на новые наблюдения.
- При $\alpha \approx 1$ — модель быстро адаптируется, давая большую весовую нагрузку последним значениям.
- При $\alpha \approx 0$ — преобладает инерция прошлых сглаженных значений, прогнозы становятся более плавными.
- Значение α подбирается через минимизацию функции потерь (например, суммы квадратов ошибок).

Начальные условия:

Для расчёта S_t требуется начальное значение S_0 . Возможные варианты:

- $S_0 = Y_1$;
- среднее первых n наблюдений;
- оптимизация S_0 совместно с α .

Метод Хольта(Holt's Linear Trend Method)

Метод Хольта расширяет простое экспоненциальное сглаживание (SES), позволяя моделировать временные ряды с линейным трендом, но без сезонности.

Наряду со сглаживанием уровня ряда, метод включает сглаживание тренда. Прогноз строится как сумма текущей оценки уровня и произведения оценки тренда на горизонт прогноза [13].

- Сглаживание уровня:

$$L_t = \alpha Y_t + (1 - \alpha)(L_{t-1} + T_{t-1}) \quad (6)$$

- Сглаживание тренда:

$$T_t = \beta(L_t - L_{t-1}) + (1 - \beta)T_{t-1} \quad (7)$$

- Прогноз на h шагов вперёд:

$$\hat{Y}_{t+h|t} = L_t + h \cdot T_t \quad (8)$$

Параметры сглаживания:

- α — отвечает за сглаживание уровня;
- β — контролирует сглаживание тренда;
- Оба параметра выбираются через оптимизацию.

Для ограничения чрезмерного роста или падения тренда при долгосрочном прогнозировании используется модификация с демпфированием [29]:

$$T_t = \beta(L_t - L_{t-1}) + (1 - \beta)\phi T_{t-1} \quad (9)$$

$$\hat{Y}_{t+h|t} = L_t + (\phi + \phi^2 + \dots + \phi^h) T_t, \quad (10)$$

где $0 < \phi < 1$ — коэффициент демпфирования:

Начальные условия:

- $L_0 = Y_1$
- T_0 может быть задан как:
 - $Y_2 - Y_1$;
 - $(Y_N - Y_1)/(N - 1)$ — средний наклон по ряду;
 - значение, оптимизированное совместно с α и β .

Метод Хольта-Уинтерса (Holt-Winters' Seasonal Method)

Метод Хольта-Уинтерса расширяет метод Хольта для обработки временных рядов с как линейным трендом, так и сезонностью. Существует две основные версии: с аддитивной и мультипликативной сезонностью [16].

К компонентам уровня и тренда добавляется сглаживание сезонной составляющей. Выбор между аддитивной и мультипликативной моделью зависит от характера сезонности [40]:

- Аддитивная сезонность — при которой колебания имеют постоянную амплитуду независимо от уровня.

$$Y_t = L_t + T_t + S_t + I_t \quad (11)$$

- Мультипликативная сезонность — при которой амплитуда сезонных колебаний пропорциональна уровню.

$$Y_t = (L_t + T_t) \times S_t + I_t \quad или \quad Y_t = L_t \times T_t \times S_t \times I_t \quad (12)$$

Аддитивная модель

Пусть m — длина сезонности (например, $m = 12$ для месячных данных).

1) Сглаживание уровня:

$$L_t = \alpha(Y_t - S_{t-m}) + (1 - \alpha)(L_{t-1} + T_{t-1}) \quad (13)$$

2) Сглаживание тренда:

$$T_t = \beta(L_t - L_{t-1}) + (1 - \beta)T_{t-1} \quad (14)$$

3) Сглаживание сезонности:

$$S_t = \gamma(Y_t - L_t) + (1 - \gamma)S_{t-m} \quad (15)$$

4) Прогноз:

$$\hat{Y}_{t+h|t} = L_t + h \cdot T_t + S_{t-m+h_m}, \quad h_m = (h - 1) \bmod m + 1 \quad (16)$$

Мультипликативная модель

1) Сглаживание уровня:

$$L_t = \alpha \frac{Y_t}{S_{t-m}} + (1 - \alpha)(L_{t-1} + T_{t-1}) \quad (17)$$

2) Сглаживание тренда:

$$T_t = \beta(L_t - L_{t-1}) + (1 - \beta)T_{t-1} \quad (18)$$

3) Сглаживание сезонности:

$$S_t = \gamma \frac{Y_t}{L_t} + (1 - \gamma)S_{t-m} \quad (19)$$

4) Прогноз:

$$\hat{Y}_{t+h|t} = (L_t + h \cdot T_t) \cdot S_{t-m+h_m}, \quad h_m = (h - 1) \bmod m + 1 \quad (20)$$

Также возможно использование демпфирования тренда, аналогично модификации в методе Хольта, чтобы прогнозы не становились слишком оптимистичными на дальнем горизонте [14].

Параметры сглаживания

- α — сглаживание уровня;
- β — сглаживание тренда;
- γ — сглаживание сезонности;

Начальные условия

Для запуска метода требуется определить:

- L_0 — средний уровень первого сезона;
- T_0 — средний прирост (например, $(Y_m - Y_1)/(m - 1)$);
- Сезонные компоненты S_1, \dots, S_m — рассчитываются как средние отклонения или отношения от уровня.

Выбор модели экспоненциального сглаживания

Выбор между SES, Хольтом и Хольт–Уинтерсом зависит от наличия тренда и сезонности, а также от характера сезонности. Часто используют информационные критерии для выбора наилучшей модели [3].

Модель ARIMA

Основные компоненты и предпосылки

Центральным понятием для моделей ARIMA является стационарность. Временной ряд называется (слабо) стационарным, если его среднее, дисперсия и автоковариационная структура не изменяются со временем [15]:

- Среднее: $E[Y_t] = \mu$,
- Дисперсия: $Var(Y_t) = \sigma^2$,
- Автоковариация: $Cov(Y_t, Y_{t-k}) = \gamma_k$.

Большинство реальных временных рядов нестационарны, имеют тренд или меняющуюся дисперсию. Модели ARIMA включают механизм приведения ряда к стационарности через дифференцирование.

Компоненты ARIMA

Модель $ARIMA(p, d, q)$ для временного ряда Y_t (или для продифференцированного ряда $W_t = \nabla^d Y_t$) состоит из трёх компонентов [42]:

Авторегрессионная часть (AR(p)):

$$W_t = c + \phi_1 W_{t-1} + \dots + \phi_p W_{t-p} + \epsilon_t \quad (21)$$

где:

- c — константа,
- ϕ_1, \dots, ϕ_p — параметры авторегрессии,
- ϵ_t — белый шум: $E[\epsilon_t] = 0$, $Var(\epsilon_t) = \sigma^2$.

Интегрированная часть (I(d)):

Указывает на порядок дифференцирования d , необходимый для приведения Y_t к стационарному виду.

Если ряд нестационарен из-за тренда, его можно сделать стационарным путем взятия разностей:

- Первая разность: $\nabla Y_t = Y_t - Y_{t-1}$,
- Вторая разность: $\nabla^2 Y_t = (Y_t - Y_{t-1}) - (Y_{t-1} - Y_{t-2})$.

Порядок дифференцирования d в модели $ARIMA(p, d, q)$ указывает, сколько раз ряд был продифференцирован для достижения стационарности. Если $d = 0$, то ряд изначально стационарен.

Скользящее среднее (MA(q)):

$$W_t = c + \epsilon_t + \theta_1 \epsilon_{t-1} + \dots + \theta_q \epsilon_{t-q}, \quad (22)$$

где θ_j — параметры скользящего среднего.

Общая модель:

$$W_t = \phi_1 W_{t-1} + \dots + \phi_p W_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + \dots + \theta_q \epsilon_{t-q} \quad (23)$$

Лагооператор B , где $BY_t = Y_{t-1}$, позволяет записывать модель компактно:

$$\Phi(B)(1 - B)^d Y_t = \Theta(B)\epsilon_t \quad (24)$$

$$\Phi(B) = 1 - \phi_1 B - \dots - \phi_p B^p, \quad \Theta(B) = 1 + \theta_1 B + \dots + \theta_q B^q \quad (25)$$

Сезонная модель SARIMA

SARIMA обозначается как $\text{ARIMA}(p, d, q)(P, D, Q)_m$, где m — соответствует порядку для сезонной части модели.

Общая форма SARIMA:

$$\Phi_P(B^m)\Phi(B)(1 - B^m)^D(1 - B)^d Y_t = C + \Theta_Q(B^m)\Theta(B)\epsilon_t \quad (26)$$

$$\Phi_P(B^m) = 1 - \Phi_1 B^m - \dots - \Phi_P B^{Pm}, \quad \Theta_Q(B^m) = 1 + \Theta_1 B^m + \dots + \Theta_Q B^{Qm} \quad (27)$$

1.4. Машинное обучение и глубокое обучение

В последние годы методы машинного обучения (ML) показали впечатляющие результаты в задачах прогнозирования временных рядов, особенно для рядов со сложными нелинейными зависимостями и большим объёмом данных [3]

Машинное обучение для временных рядов

Методы машинного обучения, такие как деревья решений, случайные леса, градиентный бустинг и метод опорных векторов, могут быть адаптированы для прогнозирования временных рядов, хотя требуют преобразования задачи в формат обучения с учителем [22].

Основные этапы:

- Инжиниринг признаков — это процесс преобразования временного ряда в обучающий набор признаков, пригодный для использования в моделях машинного обучения из исходных данных [7].
 - Лаговые переменные: $Y_{t-1}, Y_{t-2}, \dots, Y_{t-k}$.
 - Скользящие статистики: среднее, дисперсия по окну.
 - Календарные признаки: день недели, месяц, индикаторы праздников.
 - Компоненты тренда и сезонности.
- Подходы к прогнозированию:
 - Одношаговое — прогноз следующего события на основе информации до наблюдаемого события.
 - Многошаговое:
 - Рекурсивное — каждый шаг использует результат предыдущего.
 - Прямое — отдельная модель для каждого горизонта H .
 - MIMO — одна модель прогнозирует сразу несколько значений.

Рекуррентные нейронные сети (RNN)

RNN (Recurrent Neural Networks) — это тип нейронных сетей, специально разработанный для обработки последовательных данных, таких как текст, речь или временные ряды, где порядок и контекст элементов имеют ключевое значение. В отличие от стандартных полносвязных нейросетей, RNN обладают способностью сохранять информацию о предыдущих состояниях с помощью внутренних скрытых состояний, передаваемых от шага к шагу. Это позволяет сети учитывать контекст предыдущих наблюдений при формировании отклика на текущий вход, что особенно важно при анализе временных зависимостей и паттернов, меняющихся во времени. Благодаря этой особенности RNN широко применяются в задачах прогнозирования временных рядов [21].

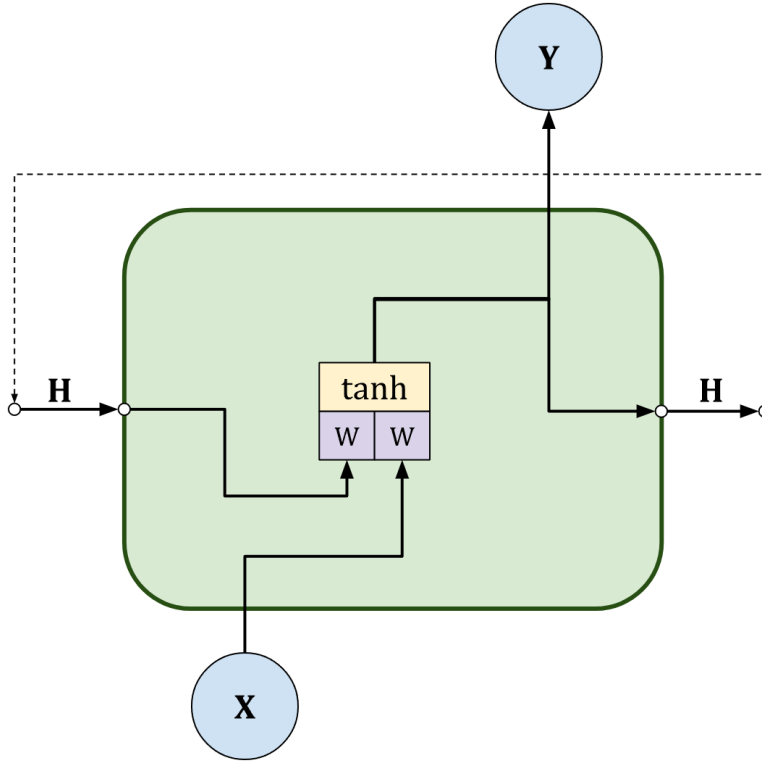


Рисунок 2. Схема архитектуры модели RNN

В каждый момент времени t скрытое состояние H_t зависит от текущего входа X_t и предыдущего состояния H_{t-1} [24].

$$H_t = f(W_h H_{t-1} + W_{xh} X_t + b_h) \quad (28)$$

$$y_t = W_{hy} H_t + b_y \quad (29)$$

где,

- X_t — вход на шаге t .
- H_t — скрытое состояние ("память").
- Y_t — выход сети.
- W_{xh}, W_{hh}, W_{hy} — весовые матрицы.
- b_h, b_y — смещения.
- f — функция активации.

При обучении RNN с помощью обратного распространения ошибки во времени градиенты могут либо экспоненциально уменьшаться включая исчезающий градиент, либо экспоненциально расти по мере распространения через множество временных шагов. Это затрудняет обучение RNN улавливать долгосрочные зависимости в данных [25].

Долгосрочная краткосрочная память (LSTM)

LSTM-сети (Long Short-Term Memory), являются усовершенствованием RNN и предназначены для устранения проблемы исчезающего или взрывающегося градиента [28]. Они способны запоминать информацию на длительные промежутки времени и эффективно обучаться на длинных последовательностях и включают в себя следующую архитектуру LSTM-ячейки:

- Ключевым элементом является состояние ячейки S_t , которое играет роль долгосрочной памяти.
- Поток информации регулируется тремя вентилями:

1) Вентиль забывания:

- Определяет, какую часть информации из предыдущего состояния S_{t-1} нужно забыть.
- Формула:

$$f_t = \sigma(W_f \cdot [H_{t-1}, X_t] + b_f) \quad (30)$$

2) Входной вентиль и кандидатное состояние:

- Входной вентиль i_t определяет, какие значения нужно обновить:

$$i_t = \sigma(W_i \cdot [H_{t-1}, X_t] + b_i) \quad (31)$$

- Кандидатное состояние \tilde{S}_t создаёт новые значения:

$$\tilde{S}_t = \tanh(W_s \cdot [H_{t-1}, X_t] + b_C) \quad (32)$$

– Обновление состояния ячейки:

$$S_t = f_t \cdot S_{t-1} + i_t \cdot \tilde{S}_t \quad (33)$$

3) Выходной вентиль:

– Определяет, какая часть состояния S_t пойдет на выход:

$$o_t = \sigma(W_o \cdot [H_{t-1}, X_t] + b_o) \quad (34)$$

– Расчет скрытого состояния:

$$H_t = o_t \cdot \tanh(S_t) \quad (35)$$

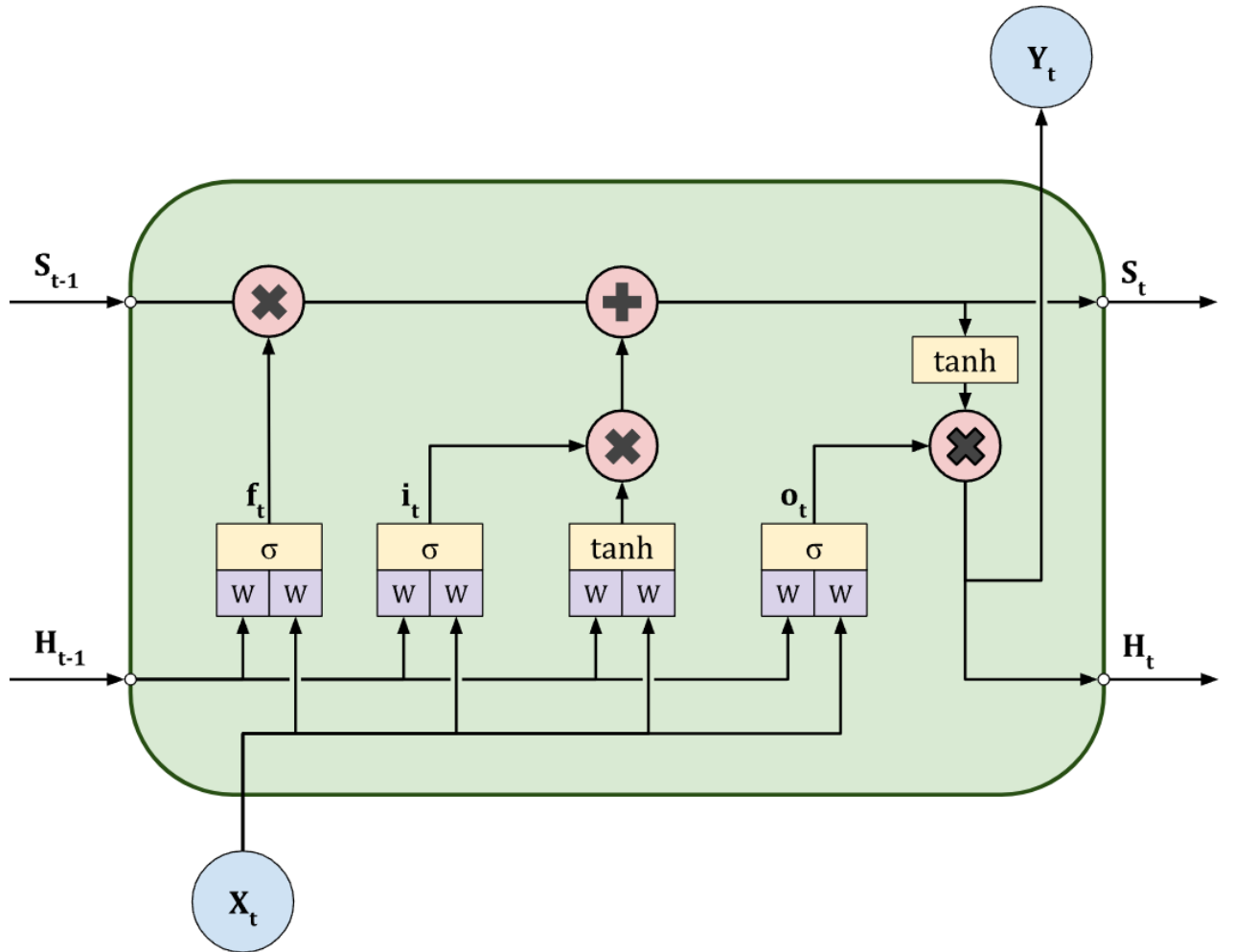


Рисунок 3. Схема ахитектуры модели LSTM

Применение LSTM к временным рядам:

- Временной ряд преобразуется в последовательности, например для предсказания Y_t используются Y_{t-k}, \dots, Y_{t-1} [27].
- Один или несколько слоев LSTM + полносвязный слой для предсказания.
- Минимизация ошибки прогноза и реальных значений.

Варианты LSTM:

- GRU (Gated Recurrent Unit) — упрощенная версия LSTM, имеет меньше параметров и обучается быстрее [27].
- Bidirectional LSTM — обрабатывает данные в обоих направлениях — вперед и назад. Полезен, когда важен контекст с обеих сторон — например в задачах анализа текста [31].
- Stacked LSTM — соединяет несколько слоёв LSTM последовательно друг с другом, что позволяет модели извлекать более сложные иерархические признаки из последовательных данных [42].

LSTM являются мощным и гибким инструментом для анализа временных рядов и различных типов последовательностей, особенно в тех случаях, когда важно учитывать долгосрочные зависимости между элементами данных. Благодаря своей архитектуре с механизмами забывания, запоминания и обновления информации, LSTM способны эффективно сохранять значимую информацию на протяжении больших промежутков времени, что делает их особенно полезными при работе с данными, где обычные RNN сталкиваются с проблемой исчезающего градиента. Тем не менее, несмотря на высокую эффективность, применение LSTM на практике требует значительных вычислительных ресурсов, особенно при работе с большими объемами данных или глубокими архитектурами [17].

Модель Prophet

Модель Prophet — это процедура прогнозирования временных рядов, разработанная командой Core Data Science компании Facebook [37]. Она предназначена для решения прикладных задач бизнес-прогнозирования и отличается простотой использования даже для пользователей без глубоких знаний в области анализа временных рядов.

Prophet основан на аддитивной модели, аналогичной обобщённой аддитивной модели. Временной ряд моделируется как сумма следующих компонент:

$$y(t) = g(t) + s(t) + h(t) + \varepsilon_t, \quad (36)$$

где:

- $g(t)$ — тренд, описывающий долгосрочные изменения;
- $s(t)$ — сезонность, отражающая периодические колебания;
- $h(t)$ — влияние праздников и особых событий;
- ε_t — случайная ошибка, предполагаемая как белый шум.

Компоненты модели Prophet

Prophet поддерживает два типа трендовых моделей:

- 1) Кусочно-линейный тренд:

$$g(t) = (k + a(t)^T \delta)t + (m + a(t)^T \gamma) \quad (37)$$

где:

- k — базовая скорость роста;
- $a(t)$ — вектор признаков, активных в момент t ;
- δ — вектор изменений наклона в точках изменений тренда s_j ;
- m — начальное смещение;
- $\gamma_j = -s_j \delta_j$ — для обеспечения непрерывности функции.

2) Логистический рост:

$$g(t) = \frac{C(t)}{1 + \exp(-(k + a(t)^T \delta)(t - (m + a(t)^T \gamma)))}, \quad (38)$$

где $C(t)$ — предельное значение, задаваемое пользователем.

Сезонные компоненты моделируются с помощью ряда Фурье:

$$s(t) = \sum_{n=1}^N \left(a_n \cos\left(\frac{2\pi nt}{P}\right) + b_n \sin\left(\frac{2\pi nt}{P}\right) \right), \quad (39)$$

где P — период, N — порядок ряда. .

Для праздников и других событий используется регрессионная модель:

$$h(t) = \sum_{i=1}^L \kappa_i \cdot I(t \in D_i), \quad (40)$$

где D_i — окно влияния события, κ_i — его эффект, I — индикаторная функция. Prophet поддерживает как предустановленные списки праздников по странам, так и пользовательские.

Градиентный бустинг (XGBoost)

Градиентный бустинг — это мощная техника машинного обучения из класса ансамблевых методов. Она строит ансамбль слабых моделей таким образом, что каждая последующая модель корректирует ошибки предыдущих. XGBoost (Extreme Gradient Boosting) — одна из самых популярных реализаций градиентного бустинга, известная высокой производительностью и скоростью.

Пусть y — целевая переменная, а X — множество признаков. Модель представляется в виде суммы M слабых моделей $H_m(X)$:

$$F_M(X) = \sum_{m=1}^M \alpha_m H_m(X) \quad (41)$$

Построение ансамбля происходит итеративно:

$$F_m(X) = F_{m-1}(X) + \alpha_m H_m(X) \quad (42)$$

Каждая новая модель $H_m(X)$ обучается на *псевдо-остатках*:

$$r_{im} = - \left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)} \right]_{F(x_i)=F_{m-1}(x_i)}, \quad (43)$$

где $L(y_i, F(x_i))$ — функция потерь .

Особенности XGBoost

- Включает L1 и L2 регуляризацию:

$$Obj(\Theta) = \sum_{i=1}^n l(y_i, \hat{y}_i) + \sum_{k=1}^K \Omega(f_k), \quad \Omega(f_k) = \gamma T + \frac{1}{2} \lambda \|w\|^2 \quad (44)$$

- Используется второе разложение Тейлора для функции потерь для более точного построения деревьев.
- XGBoost определяет, в какую ветвь направлять пропущенные значения.
- Эффективная реализация для многопроцессорных систем.
- Встроенная кросс-валидация, обрезка деревьев, ранняя остановка.

1.5. Гибридные модели (SARIMA-LSTM)

Идея гибридных моделей заключается в объединении сильных сторон различных подходов к прогнозированию для достижения более высокой точности и надежности. Разные модели хорошо справляются с различными типами паттернов во временных рядах, что делает их комбинацию полезной. Одной из популярных гибридных моделей является сочетание SARIMA и LSTM.

- SARIMA эффективно моделирует линейную структуру временного ряда, включая тренды и сезонность. Параметры SARIMA имеют четкую стати-

стическую интерпретацию. Модель хорошо подходит для данных с выраженной автокорреляционной структурой.

- LSTM Способна улавливать сложные нелинейные зависимости и долгосрочные паттерны, которые могут оставаться в остатках после моделирования линейной компоненты с помощью SARIMA.

Общая концепция [34]: Временной ряд Y_t можно представить как сумму линейной компоненты L_t и нелинейной компоненты N_t :

$$Y_t = L_t + N_t \quad (45)$$

Шаги построения гибридной модели SARIMA-LSTM

1) Построение модели SARIMA:

- Выполнить все шаги методологии Бокса-Дженкинса: идентификация параметров, их оценка, диагностика модели.
- Убедиться, что модель SARIMA адекватна, но ее остатки могут содержать структуру.

2) Рассчитать остатки $e_t = Y_t - \hat{L}_t$ на обучающей выборке.

3) Построение модели LSTM на остатках:

- Подготовить данные остатков для LSTM.
- Спроектировать и обучить LSTM для прогнозирования будущих значений остатков. Это может быть сложной задачей, так как остатки могут быть очень шумными.

4) Генерация прогнозов:

- Сделать прогноз \hat{L}_{t+h} с помощью модели SARIMA на h шагов вперед.
- Сделать прогноз \hat{N}_{t+h} с помощью модели LSTM на остатках.
- Итоговый прогноз: $\hat{Y}_{t+h} = \hat{L}_{t+h} + \hat{N}_{t+h}$.

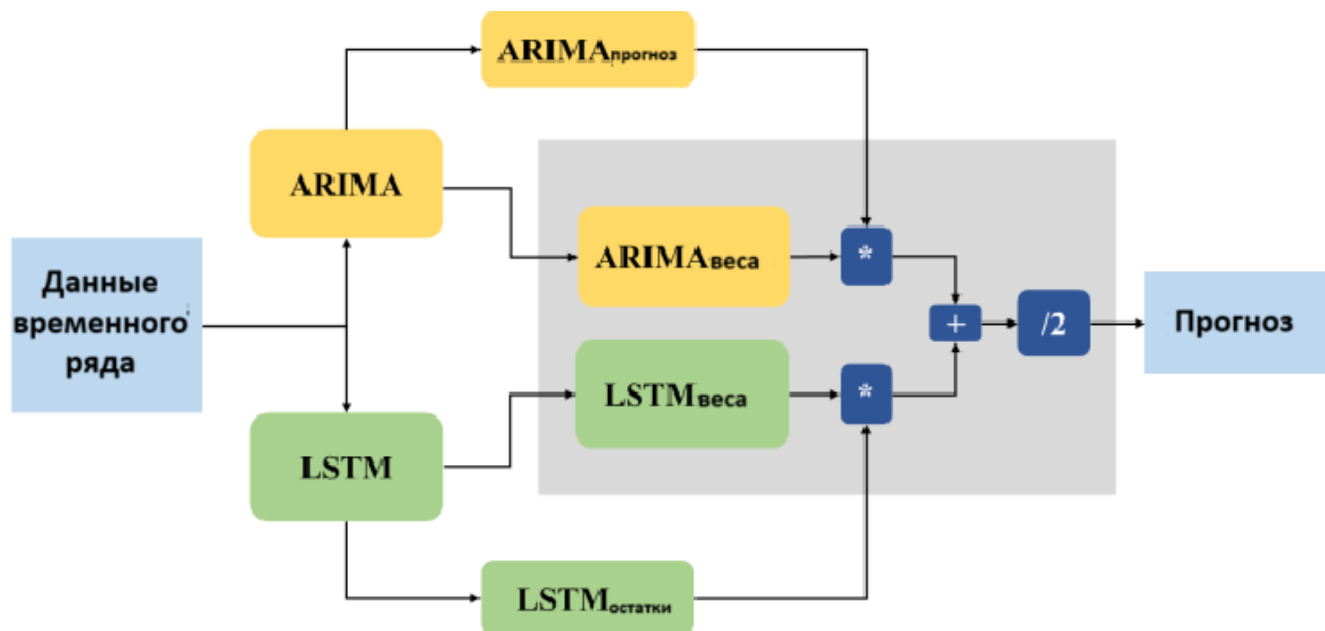


Рисунок 4. Схема построения модел SARIMA-LSTM

1.6. Обоснование выбора моделей для реализации в приложении

При выборе моделей для конкретного прикладного применения необходимо учитывать следующий ряд важных аспектов:

1) Цели приложения:

- Если требуется максимальная точность, и есть ресурсы, стоит рассматривать сложные модели.
- Модель Prophet и методы экспоненциального сглаживания выигрывают по скорости внедрения благодаря своей простоте, интерпретируемости и минимальным требованиям к обработке данных.
- Если пользователям важно понимать, почему прогноз именно такой, Prophet, ES и ARIMA предпочтительнее.
- Обучение и прогнозирование с помощью ML моделей может требовать значительных ресурсов.

2) Характеристики данных:

- Для коротких рядов лучше подходят простые модели: ES, простая ARIMA. LSTM и XGBoost требуют больше данных.
- ES, SARIMA, Prophet хорошо справляются с этим. Для XGBoost и LSTM это нужно учитывать при инжиниринге признаков.
- Если предполагаются сильные нелинейные зависимости, ML модели могут дать преимущество. Для преимущественно линейных рядов SARIMA часто бывает достаточно.
- Слишком сложные модели могут переобучиться на шумных данных.
- Если нужно прогнозировать тысячи или миллионы рядов, важна масштабируемость и скорость моделей.
- Наличие экзогенных переменных: Если есть важные внешние факторы, модель должна поддерживать их включение.

Разработка и внедрение сложных моделей машинного обучения требуют глубоких знаний, а также значительных временных затрат на проектирование, обучение и тонкую настройку. В отличие от простых алгоритмов, сложные модели часто зависят от большого количества гиперпараметров, архитектурных решений и методов регуляризации, что усложняет их оптимизацию.

Глава 2. Требования и реализация приложения

Данная глава посвящен анализу и описанию ключевых аспектов приложения, разработанного для автоматизации задач анализа и прогнозирования временных рядов. Приложение предоставляет удобный графический интерфейс для загрузки данных, их предварительной обработки, применения различных статистических моделей и моделей машинного обучения, а также визуализации результатов и оценки качества прогнозов. В рамках доклада будут подробно рассмотрены функциональные и не функциональные требования, которые легли в основу разработки, а также обоснован выбор технологического стека, использованного для реализации приложения.

2.1. Требования к приложению

Успешная разработка любого программного продукта начинается с четкого определения требований. Для приложения анализа временных рядов эти требования можно разделить на функциональные и нефункциональные.

Функциональные требования определяют основные возможности и операции, которые приложение предоставляет пользователю. В контексте данного приложения, анализ исходного кода позволяет выделить следующий набор функциональных возможностей:

Загрузка и импорт данных

- Приложение должно обеспечивать простой и интуитивно понятный механизм загрузки временных рядов. Поддерживаемый формат — CSV, являющийся стандартом для табличных данных.
- Загрузка осуществляется через диалоговое окно выбора файла или перетаскиванием файла в окно, что повышает удобство.
- После загрузки данные преобразуются в общий формат для последующей обработки.

Предварительная обработка и настройка данных

- Пользователь выбирает столбцы для временных меток и значений ряда с помощью выпадающих списков.
- Необходимо указать частоту ряда, что влияет на корректность анализа.
- При преобразовании частоты возможны пропуски. Реализована их обработка в виде линейной интерполяции.
- Применённые настройки создают готовый набор данных для анализа.

Разделение данных на выборки

- Для оценки точности и качества работы моделей временных рядов данные обычно разделяются на две части — обучающую и тестовую выборки.
- Пользователь указывает процент для тестовых данных.
- Разделение осуществляется по времени: ранние данные — обучающая выборка, поздние — выборка для тестирования.

Применение моделей прогнозирования

- Приложение поддерживает широкий спектр моделей:
 - Статистические: Holt-Winters, SARIMA.
 - Машинного обучения: XGBoost.
 - Глубокого обучения: LSTM.
 - Специализированные: Prophet.
 - Гибридные: SARIMA-LSTM.
- Выбор модели реализуется через боковое меню или кнопки.

Настройка параметров моделей

- Для моделей с гиперпараметрами предусмотрены окна настройки:
 - LSTM: число эпох, размер батча, длина входа.

- SARIMA: параметры p, d, q, P, D, Q, m .
- XGBoost: число деревьев, learning rate и др.

Генерация прогнозов

- После настройки модель обучается на данных и строит прогноз.
- Прогноз может быть выполнен на тестовых данных представляемых заданным горизонтом.

Визуализация результатов

- Используется библиотека Plotly для построения интерактивных графиков.
- На графике после разделения данных отображаются: обучающая выборка, тестовая выборка, подгонка модели и прогноз.
- При наличии — отображаются доверительные интервалы.
- Вертикальная линия показывает границу между данными выборками.

Табличное отображение данных

- Данные также выводятся в таблицу.
- Возможна сортировка по столбцам для удобства анализа.

Переключение представлений

- Пользователь может переключаться между графиком и таблицей через соответствующую кнопку в меню.

Оценка качества прогноза

- При наличии тестовых данных система автоматически выполняет расчет различных метрик точности модели [20]:
 - MAE (средняя абсолютная ошибка)

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (46)$$

- MSE (среднеквадратичная ошибка)

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (47)$$

- RMSE (среднеквадратическая ошибка)

$$RMSE = \sqrt{MSE} \quad (48)$$

- MAPE (средняя абсолютная процентная ошибка)

$$MAPE = \frac{100\%}{n} \sum_{i=1}^n \left| \frac{y_i - \hat{y}_i}{y_i} \right| \quad y_i \neq 0 \quad (49)$$

- Результаты расчета метрик выводятся в виде сообщений.

Нефункциональные требования определяют характеристики качества работы приложения, а не его конкретные функции.

Удобство использования

- Интерфейс должен быть интуитивно понятным даже для пользователей без опыта программирования. Использование библиотеки PyQt5 с боковой панелью для управления и основной областью для отображения данных способствует удобству оператора.
- Элементы управления логически сгруппированы по функционалу: настройки данных, разделение, выбор моделей.
- Визуальный стиль загружается из файла json-файла.

Производительность

- Приложение должно эффективно обрабатывать большие объемы данных и выполнять ресурсоемкие алгоритмы.
- Используются оптимизированные и достаточно мощные библиотеки: Pandas, NumPy, TensorFlow, XGBoost, Statsmodels.

- Для отображения хода долгих операций применяется диалогами прогресса, что предотвращает восприятие "зависания".

Надежность и устойчивость к ошибкам

- Приложение должно обрабатывать ошибки, возникающие при загрузке некорректных файлов, отсутствии нужных столбцов, сбоях моделей и пр.
- В случае ошибки приложение не должно завершаться аварийно, а должно оставаться работоспособным, уведомляя об соответствующей ошибке.

Поддерживаемость

- Код должен быть читаемым и структурированным для облегчения модификаций, сопровождения и логирования.
- Используется объектно-ориентированный подход.
- Логика разделена на отдельные методы.
- Стили хранятся в отдельном JSON-файле, что повышает гибкость и удобство настройки внешнего вида.

Интерактивность

- Использование библиотеки Plotly обеспечивает высокую интерактивность графиков: масштабирование, панорамирование, всплывающие подсказки.
- Приложение должно значительно повышает аналитические и функциональные возможности для пользователя.

Адаптивность интерфейса

- Реализована анимация сворачивания/разворачивания боковой панели, что повышает адаптивность интерфейса и улучшает пользовательский опыт.

2.2. Выбор технологического стека и его обоснование

Выбор технологического стека является критически важным решением, определяющим возможности, производительность и удобство разработки приложения. Для данного приложения был выбран следующий стек технологий, каждая из которых обоснована ее пригодностью для поставленных задач:

Python

Python является идеальным выбором для научных вычислений, анализа данных и машинного обучения благодаря своей богатой экосистеме и простоте использования [11]. Он позволяет быстро прототипировать и разрабатывать функциональность, что особенно важно для исследовательских приложений. Несмотря на наличие альтернатив, таких как C++ или Java, мощные возможности Python в аналитике делают его предпочтительным выбором.

PyQt5

PyQt5 предоставляет интерфейс к Qt — мощному кроссплатформенному GUI-фреймворку. Он позволяет создавать нативные приложения с продвинутыми интерфейсами. По сравнению с другими Python-фреймворками, PyQt5 предлагает более профессиональный внешний вид и широкую функциональность — критически важную для интерфейса, содержащего таблицы, графики и множество управляющих элементов [36].

Pandas

Pandas предоставляет структуры данных, идеально подходящие для временных рядов. Встроенные средства работы с датами, частотой, пропущенными значениями и агрегацией делают её незаменимой на этапе предобработки.

NumPy

NumPy — основа численных вычислений в Python. Данные и связанные с ними функции позволяют эффективно выполнять математические операции, необходимые в прогнозных моделях. Кроме того, большинство ML-библиотек используют NumPy-массивы как стандартный формат данных.

Statsmodels

Библиотека предоставляет классические статистические модели временных рядов — экспоненциальное сглаживание и SARIMA. Это упрощает использование подходов к прогнозированию без необходимости их ручной реализации.

TensorFlow / Keras

TensorFlow и его высокоуровневое API Keras являются одними из ведущих и наиболее популярных инструментов в области глубокого обучения благодаря своей гибкости, масштабируемости и широкому набору функций. Модели LSTM, реализуемые с помощью этих платформ, особенно эффективны для анализа временных рядов, так как обладают уникальной способностью запоминать и учитывать долгосрочные зависимости в данных [32].

XGBoost

XGBoost — мощная и быстрая реализация градиентного бустинга. Он показывает высокую точность при прогнозировании, особенно в задачах регрессии с лаговыми признаками, что делает его ценным инструментом в данном

Prophet

Prophet отлично подходит для бизнес-временных рядов с сезонностью и праздниками. Он устойчив к пропущенным значениям и выбросам, а также предоставляет интуитивные параметры для настройки модели.

Scikit-learn (sklearn)

Scikit-learn предоставляет обширный набор алгоритмов для машинного обучения, а также инструменты для предобработки данных и расчёта метрик [32]. Это делает её важной частью пайплайна подготовки и оценки модели.

Plotly

Plotly позволяет строить интерактивные графики высокого качества, включая линии, области, точки с подсказками. Встроенная поддержка масштабирования и навигации делает его идеальным инструментом для анализа результатов моделей временных рядов в интерфейсе.

Комбинированное использование этих технологий формирует мощную платформу для анализа и прогнозирования временных рядов: Python обеспечивает основу, PyQt5 — пользовательский интерфейс, Pandas и NumPy — обработку данных, модели реализуются через Statsmodels, TensorFlow/Keras, XGBoost и Prophet, Scikit-learn поддерживает предобработку и метрики, а Plotly отвечает за визуализацию. Такой стек позволяет реализовать все заявленные функциональные и нефункциональные требования.

2.3. Архитектура приложения

Архитектура приложения строится по классическому принципу "модель-представление-контроллер", где логика обработки данных и моделирования отделена от пользовательского интерфейса. В данном случае, основная логика сосредоточена в главном окне, которое выступает в роли главного контроллера и управляет представлением и взаимодействует с неявной "моделью" — объектами данных и функциями моделирования из различных библиотек. Основные модули и их взаимодействие:

- 1) Модуль Пользовательского Интерфейса

- Обеспечивает взаимодействие пользователя с приложением: загрузка данных, выбор опций, запуск обработки, просмотр результатов.
- Обрабатывает пользовательский ввод, инициирует соответствующие действия в других модулях и отображает результаты обработки.

2) Модуль Обработки Потока Данных

- Обеспечивает загрузку, очистку, преобразование и подготовку временных рядов к моделированию.
- Получает путь к файлу и настройки интерфейса, подготавливает данные и передает их модулю моделей.

3) Модуль Интеграции Моделей Прогнозирования

- Реализован вызов алгоритмов прогнозирования.
- Получает подготовленные данные и параметры, обучает модели, формирует прогнозы и передает их в модули визуализации и оценки.

4) Модуль Визуализации

- Реализована функция визуализации, которая позволяет одновременно отображать исходные данные временного ряда и построенные прогнозы на одном графике.
- Модуль получает данные от модулей обработки и моделей, строит графики и отображает их в интерфейсе.

5) Модуль Оценки Качества Прогнозов

- Обеспечивает расчет метрик точности прогнозов.
- Модуль получает данные от модуля моделей, рассчитывает метрики и отображает их в GUI.

2.4. Разработка интерфейса пользователя

Разработка пользовательского интерфейса является ключевым аспектом создания удобного и доступного приложения. Использование библиотеки PyQt5 позволило построить настольное полностью автоматизированное приложение с богатым набором интерактивных элементов.

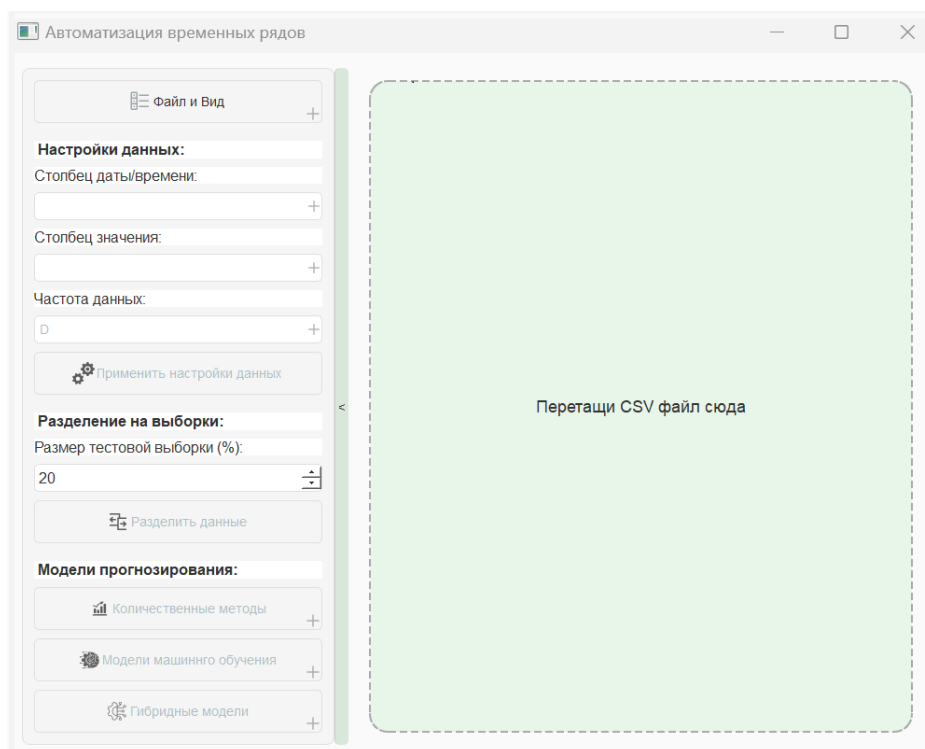


Рисунок 5. Начальный экран приложения

Основное окно

Класс основного окна является центральным и ключевым элементом графического интерфейса пользователя (GUI), поскольку именно он отвечает за организацию и управление всеми основными компонентами интерфейса. В его структуру входят различные элементы управления, панели и виджеты, необходимые для взаимодействия пользователя с приложением. Макет основного окна реализован с использованием горизонтального контейнера, который эффективно разделяет окно на две основные области:

1) Боковая панель:

- Содержит вертикальный макет для элементов управления.
- Элементы управления сгруппированы по назначению:
 - Меню файл и вид с прикреплёнными действиями: "Загрузить CSV" "Очистить всё" "Переключить график/таблицу".
 - Группа элементов выбора: столбцы даты и значений, частота, и кнопка "Применить настройки данных".
 - Элементы для ввода процента тестовой выборки и кнопка "Разделить данные".
 - Группы для выбора моделей: "Количественные методы": (Holt-Winters, SARIMA), "Машинное обучение": (LSTM, XGboost, Prophet), "Гибридные модели": (SARIMA-LSTM).
- Реализована анимация сворачивания/разворачивания панели.

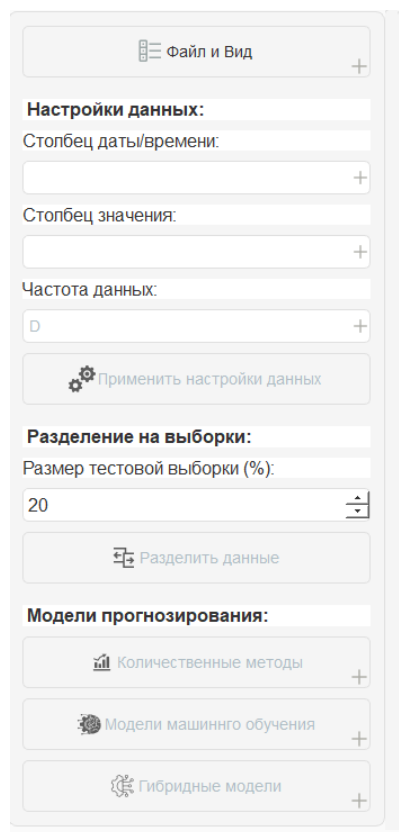
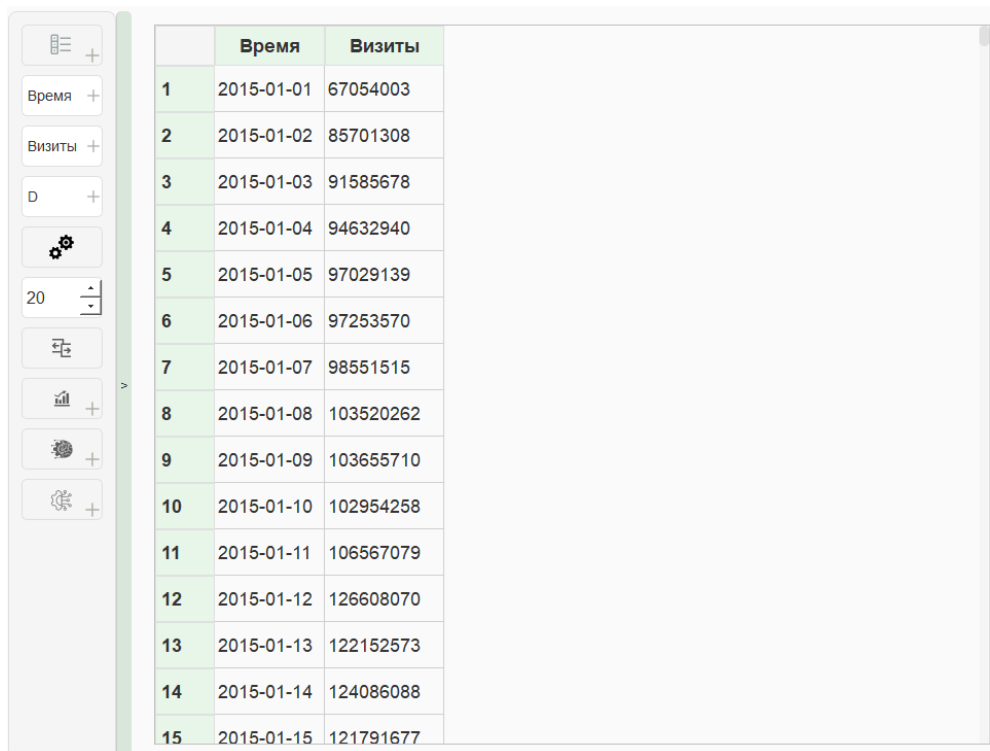


Рисунок 6. Боковая панель

2) Основная область:

- Содержит виджет для загрузки файлов перетаскиванием. После загрузки скрывается из отображения.
- Содержит виджет , переключающийся между графиком и таблицей. Изначально скрыт, отображается после загрузки данных.



	Время	Визиты
1	2015-01-01	67054003
2	2015-01-02	85701308
3	2015-01-03	91585678
4	2015-01-04	94632940
5	2015-01-05	97029139
6	2015-01-06	97253570
7	2015-01-07	98551515
8	2015-01-08	103520262
9	2015-01-09	103655710
10	2015-01-10	102954258
11	2015-01-11	106567079
12	2015-01-12	126608070
13	2015-01-13	122152573
14	2015-01-14	124086088
15	2015-01-15	121791677

Рисунок 7. Основная область. Представление — таблица

Виджет Drag-and-Drop

Созданный виджет, предназначен для загрузки файлов перетаскиванием:

- Включает поддержку drag-and-drop.
- Проверяет, содержит ли событие URL и принимает его.
- Извлекает URL, проверяет, что файл имеет расширение .csv, и вызывает callback-функцию для передачи, непосредственно самого файла.

Табличное представление

Для табличного отображения данных используется стандартные таблицы.

- Данные заполняют таблицу.
- Заголовки берутся из названий столбцов.
- Включена сортировка .

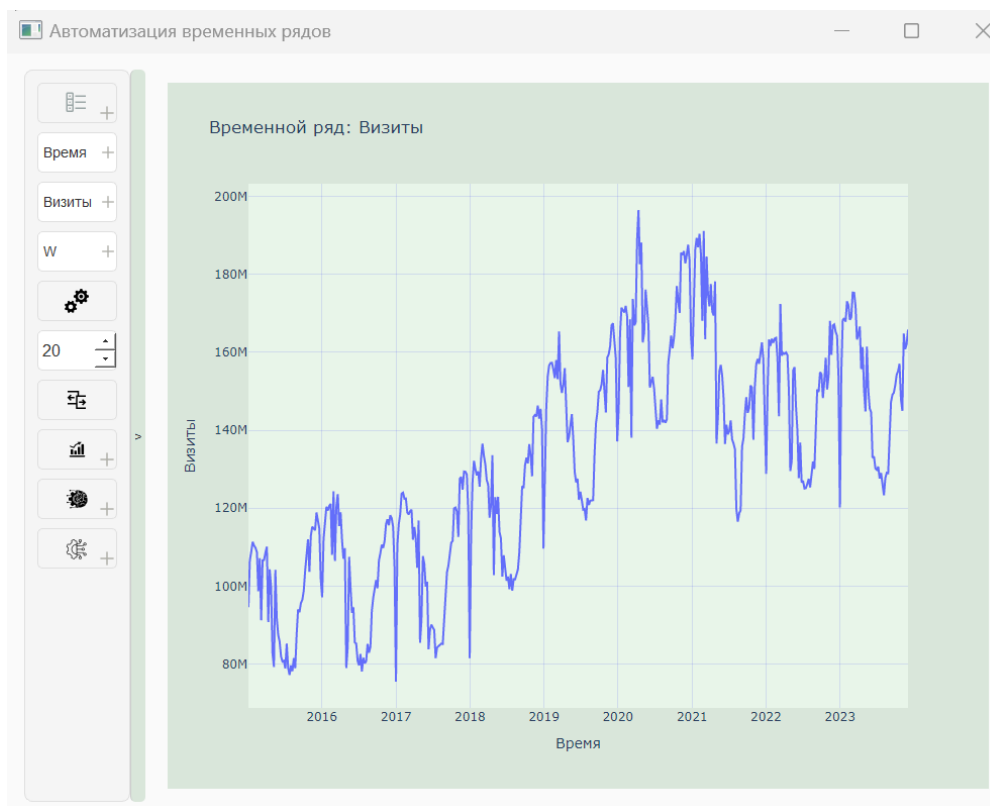


Рисунок 8. Основная область. Представление — график

Стилизация

Внешний вид приложения задаётся с помощью стилей Qt, загружаемых из json-файла

- Стили аналогичны CSS и задаются для виджетов.
- Позволяют изменять оформление без изменения логики приложения.

2.5. Модули обработки потока данных

Обработка потока данных — это последовательность шагов, подготавливающих исходные данные для анализа и моделирования. Эти шаги реализованы как отдельные методы.

Загрузка CSV

- Метод принимает путь к файлу.
- Считывает данные.
- Проверяет файл на пустоту.
- Очищает предыдущее состояние приложения.
- Сохраняет исходный набор данных.
- Заполняет списки выбора столбцов данными из загруженного файла.
- Отображает загруженные данные в таблице.
- Скрывает виджет, отображает таблицу.
- Активирует элементы управления для настройки данных.

Выбор столбцов и установка частоты

- Метод считывает выбранные столбцы даты и значения и частоту из поля.
- Проверяет корректность выбора.
- Создает рабочую копию с выбранными столбцами.
- Преобразует столбец даты.
- Устанавливает столбец даты как индекс.

Обработка пропущенных значений

- Метод проверяет наличие пропусков.
- Предлагается заполнение интерполяцией.

- Далее применяется заполнение по краям.
- Если остаются пропуски или пользователь нажимает на кнопку остановки (cancel) — операция отменяется.
- Обработанные данные сохраняются в памяти.
- Обновляется сезонность.
- Обновляются график и таблица.
- Активируются элементы управления для анализа.

Разделение на выборки

- Метод запускается после настройки данных.
- Проверяется наличие данных в памяти.
- Считывается процент тестовой выборки.
- Определяется индекс разделения.
- Данные делятся на тренировочные и тестовые.
- Пользователю показывается информация о размерах.
- На графике отображается вертикальная линия разделения.

Масштабирование (для LSTM)

- Выполняется внутри методов моделей.
- Применяется к тренировочной выборке.
- То же преобразование используется для тестовой или всей выборки.
- Прогнозы масштабируются обратно.

Интеграция моделей прогнозирования

Приложение поддерживает разные модели прогнозирования, каждая из которых вызывается отдельным методом в общем классе .

2.6. Реализация оценки качества прогнозов

Оценка качества прогнозов является неотъемлемой частью процесса моделирования. Она позволяет количественно измерить, насколько хорошо модель предсказывает неизвестные данные. Этот функционал реализован в методе.

- Метод принимает три аргумента:
 - серию или массив истинных значений тестовой выборки,
 - серию или массив предсказанных значений,
 - название модели.
- Выполняются базовые проверки входных данных на их наличие и соответствие требуемому формату.
- Ключевым шагом является выравнивание истинных и предсказанных значений по их временному индексу. Это гарантирует, что метрики рассчитываются только для тех точек, для которых есть и истинное, и предсказанное значение, исключая пропущенные значения.
- Для расчета стандартных метрик используются функции из библиотеки `sklearn.metrics`:
 - MAE (Mean Absolute Error) — измеряет среднюю абсолютную ошибку. Менее чувствителен к выбросам, чем MSE/RMSE.
 - MSE (Mean Squared Error) — средняя квадратичная ошибка, сильнее наказывает большие отклонения.
 - RMSE (Root Mean Squared Error) — квадратный корень из MSE, имеет ту же размерность, что и исходные данные.
 - MAPE (Mean Absolute Percentage Error) — Выражается в процентах, удобен для сравнения моделей на разных временных рядах. Реализация учитывает случай, когда истинные значения равны нулю.

- Все рассчитанные метрики форматируются в текстовую строку.
- Результаты отображаются пользователю в информационном диалоговом окне.
- Предусмотрена обработка исключений на этапе расчета метрик.

Заключение к главе 2

Детальное рассмотрение архитектуры приложения, процесса разработки пользовательского интерфейса, реализации модулей обработки данных, интеграции разнообразных моделей прогнозирования и механизма оценки качества прогнозов демонстрирует комплексный подход к решению задачи анализа и прогнозирования временных рядов.

Четкое разделение на логические модули, использование специализированных библиотек Python для каждого этапа рабочего процесса, а также продуманная реализация графического интерфейса с учетом удобства пользователя делают это приложение эффективным и гибким инструментом. Способность обрабатывать данные, применять различные статистические и машинные методы, визуализировать результаты и предоставлять количественную оценку точности прогнозов охватывает полный цикл анализа временных рядов, делая приложение ценным ресурсом для исследователей, аналитиков и специалистов, работающих с временными данными. Дальнейшее развитие может включать добавление новых моделей, более сложные методы обработки пропусков и выбросов, учет экзогенных переменных факторов.

Глава 3. Результаты работы приложения

3.1. Данные для анализа

Данные охватывают период с января 2014 года по апрель 2023 года. Это позволяет провести анализ долгосрочных тенденций и сезонных колебаний в использовании поисковых систем.

Информация была получена с платформы Kaggle [33] и предоставлена сервисом StatCounter Global Stats. Kaggle является популярной платформой для обмена наборами данных, а StatCounter Global Stats – авторитетным источником статистики по использованию веб-технологий.

Данные представлены в виде временного ряда, где каждая отдельная запись в наборе содержит информацию о значении:

1. Даты: конкретный день наблюдения.
2. Количестве посетителей: число пользователей, использовавших поисковые системы в указанную дату.

Для дальнейшего удобства вычисления и представления данные будут агрегированы с ежедневных до еженедельных.

StatCounter признан лидером в области веб-аналитики. Сервис известен предоставлением детализированной, точной и актуальной информации о посещаемости веб-сайтов и онлайн-платформ. Это обеспечивает высокую степень достоверности используемых данных для анализа.

Данные представлены в формате CSV-файла, который содержит ежедневные данные о количестве посетителей, использующих поисковые системы. Каждая строка файла соответствует одному дню и содержит дату в формате ГГГГ-ММ-ДД и соответствующее ей число посетителей, разделённые запятой.

Из визуального анализа представленных данных о количестве посетителей, использующих поисковые системы, можно сделать следующие наблюдения:

общее число пользователей демонстрирует тенденцию к росту с течением времени. Кроме того, прослеживаются периодические колебания, которые могут указывать на сезонные факторы и характерные недельные и месячные циклы активности. Исходя из этого, можно предположить, что в последующие периоды сохранится как общая тенденция роста, так и подобные циклические закономерности в использовании поисковых систем.

3.2. Исследование результатов моделей прогнозирования в приложении

Точность моделирования временных рядов будут оценены при помощи указанных выше метрик. Если средняя абсолютная ошибка в процентах превышает 15%, применение такой модели считается недопустимым.

Оптимальное значение параметров модели определяется исследователем, который использует приложение для настройки и анализа.

Выбор коэффициентов осуществляется методом наименьших квадратов для моделей: скользящие средней, авторегрессии и нейронных сетей. Использование параметров модели, основывается на минимизации суммы квадратов остатков (разницы между наблюдаемыми значениями и значениями, предсказанными моделью). Также выбор может быть выполнен методом наибольшего правдоподобия на примере моделей Хольта – Уинтерсона и авторегрессионной условной гетероскедастичности, в которых параметры модели выбираются таким образом, чтобы максимизировать вероятность или правдоподобие получения наблюдаемых данных при заданных параметрах модели. Оценки параметров модели выбираются таким образом, чтобы максимизировать функцию правдоподобия.

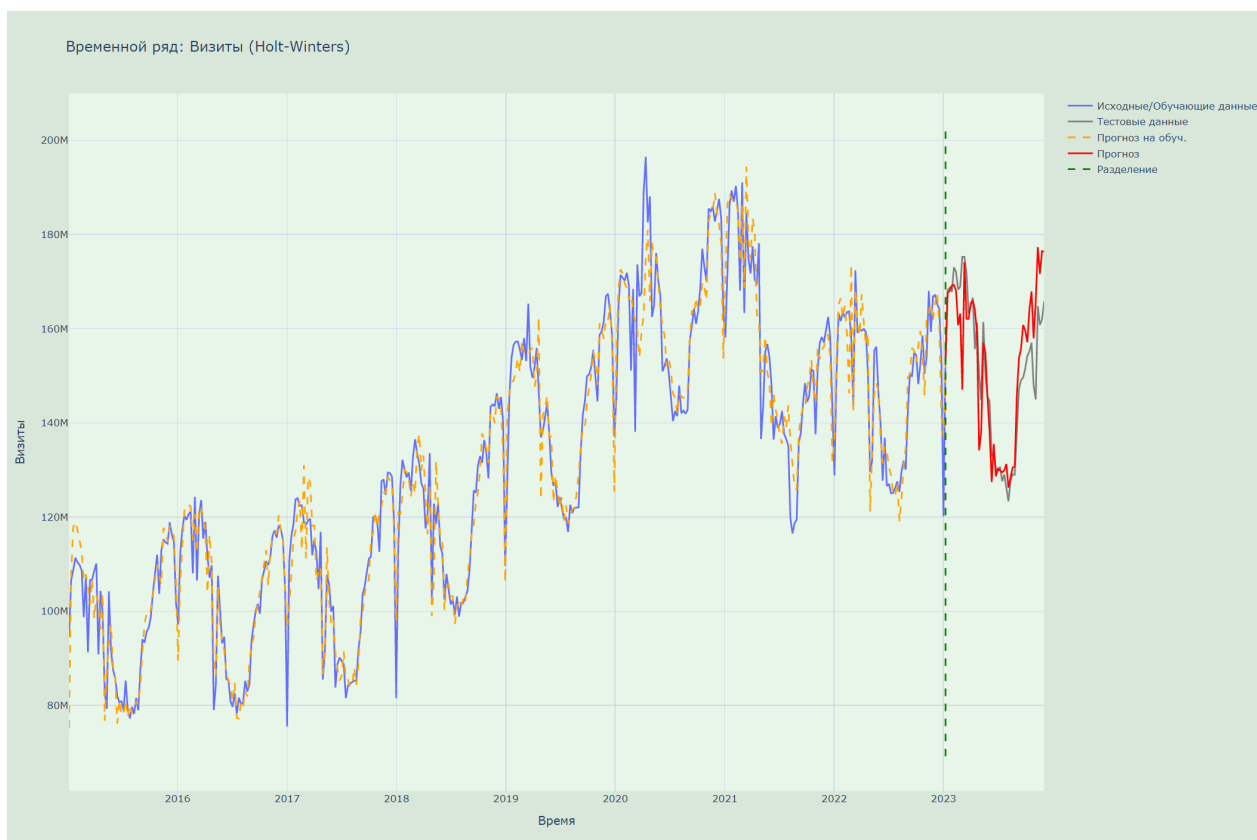


Рисунок 9. Модель Хольта-Уинтерса

Экспоненциальное сглаживание (Хольта-Уинтерса)

Метрики точности модели для 10% тестовой выборки, параметры модели α, β, γ вычисляются автоматически.

Таблица 1

Метрики точности модели Хольта-Уинтерса

MAE	5995319.4255
MSE	68971174248049.6484
RMSE	8304888.5753
MAPE	3.8869%

SARIMA

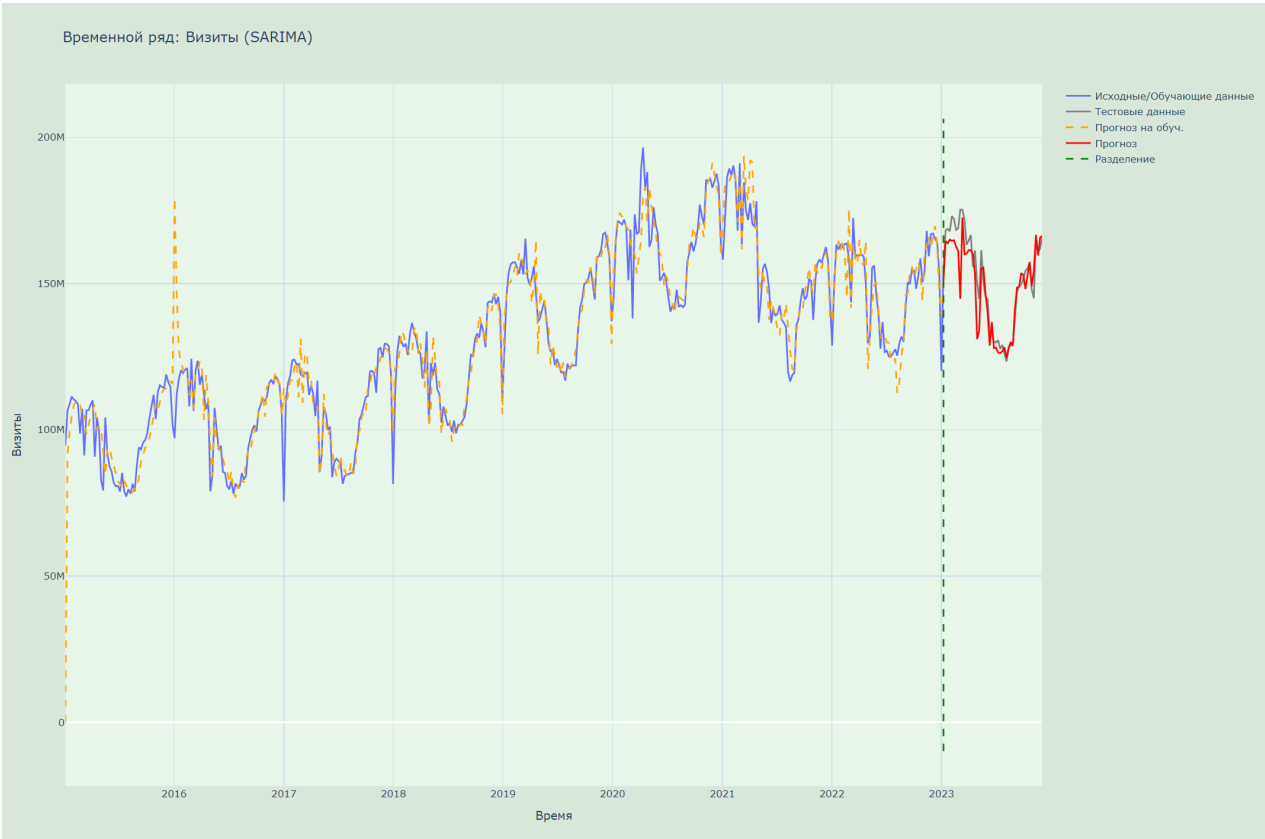


Рисунок 10. Модель SARIMA

Метрики точности модели для 10% тестовой выборки, с заданными параметрами: $(1,1,1)$, $(1,1,1,52)$

Таблица 2

Метрики точности модели SARIMA

MAE	4647283.1492
MSE	49036667107997.6719
RMSE	7002618.5894
MAPE	2.9672%

LSTM

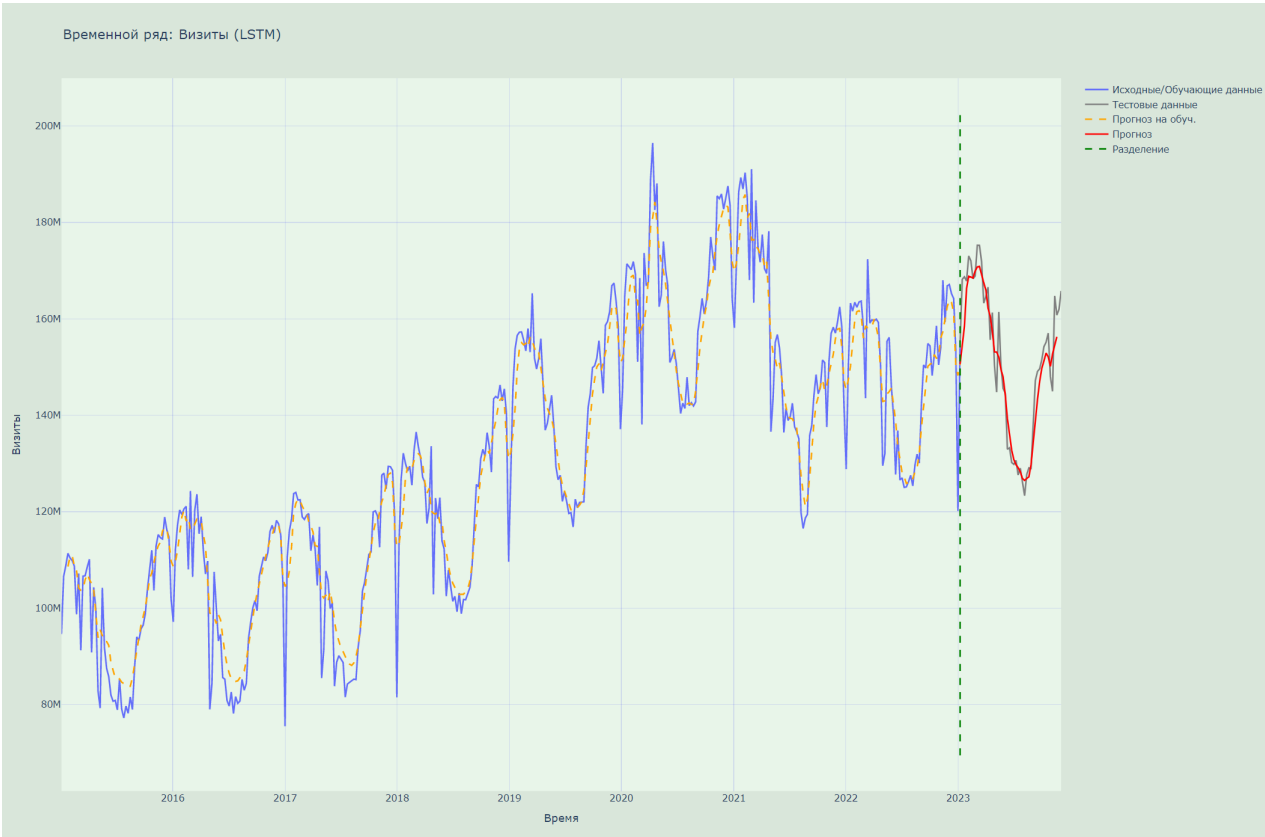


Рисунок 11. Модель LSTM

Метрики точности модели для 10% тестовой выборки, с заданными параметрами: (длина входной последовательности: 52, количество LSTM-юнитов: 50, количество эпох: 50, размер батча: 32)

Таблица 3

Метрики точности модели LSTM

MAE	6795499.5000
MSE	71367760281600.0000
RMSE	8447944.1453
MAPE	4.5165%

XGBoost

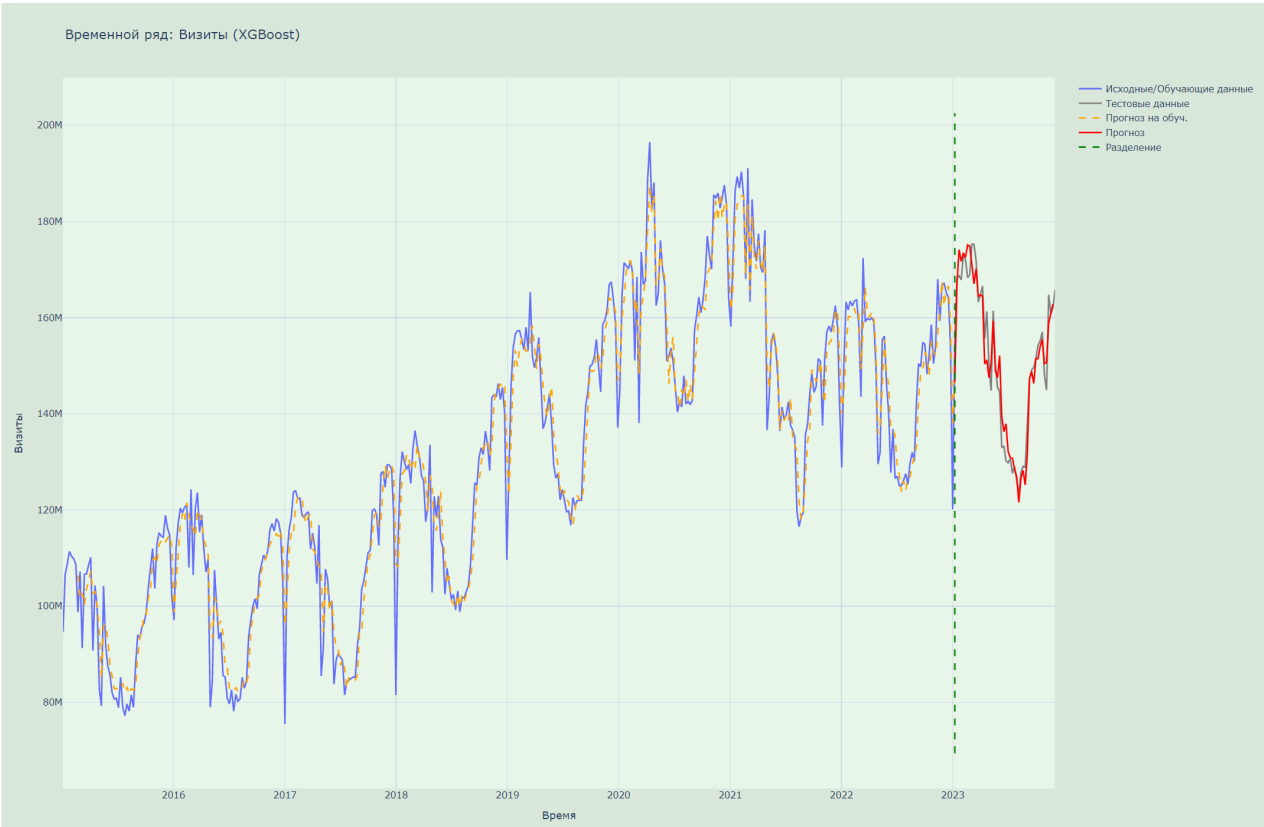


Рисунок 12. Модель XGBoost

Метрики точности модели для 10% тестовой выборки, с параметрами: (Количество деревьев: 100, Макс. глубина дерева: 3, Скорость обучения: 0.1, Доля подвыборки: 0.8, Доля признаков для дерева: 0.8, Количество лагов: 7)

Таблица 4

Метрики точности модели XGBoost

MAE	3052860.5000
MSE	16031549489152.0000
RMSE	4003941.7440
MAPE	2.0260%

Prophet

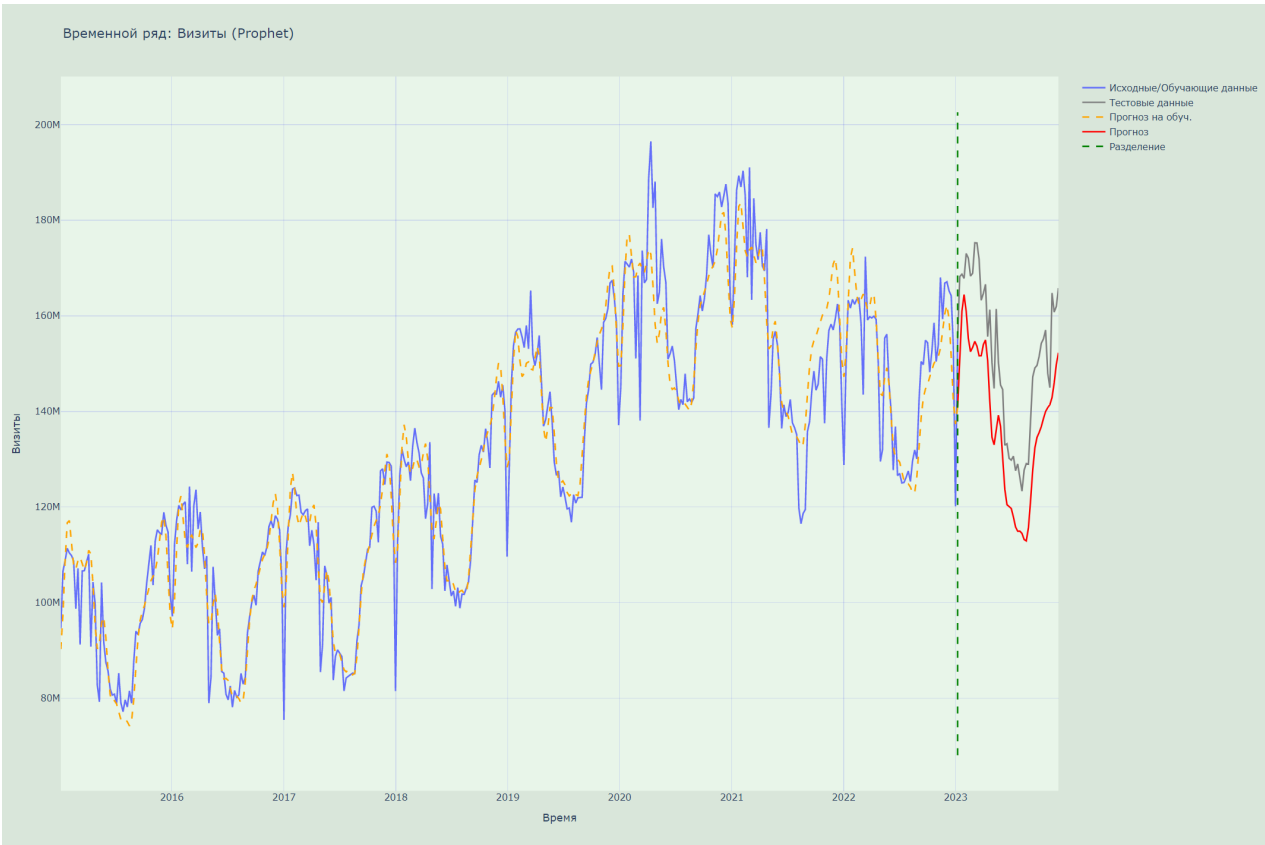


Рисунок 13. Модель Prophet

Метрики точности модели для 10% тестовой выборки, параметры модели вычисляются автоматически.

Таблица 5

Метрики точности модели Prophet

MAE	13792478.3741
MSE	211772932375996.9062
RMSE	14552420.1553
MAPE	9.1156%

SARIMA-LSTM

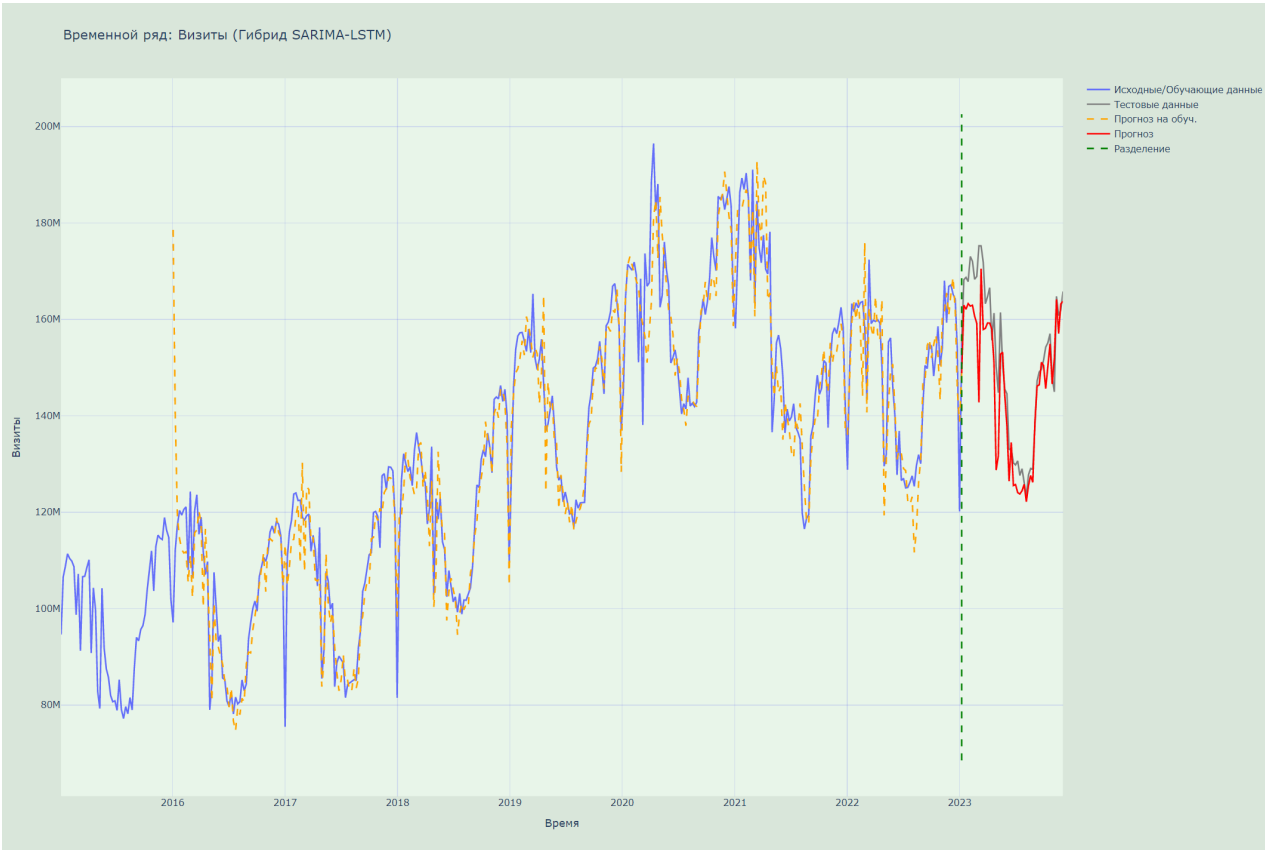


Рисунок 14. Модель SARIMA-LSTM

Метрики точности модели для 10% тестовой выборки, с заданными параметрами: (1,1,1), (1,1,1,52), (длина входной последовательности: 52, количество LSTM-юнитов: 50, количество эпох: 50, размер батча: 32)

Таблица 6

Метрики точности модели SARIMA-LSTM

MAE	5697735.0073
MSE	65081003402011.3594
RMSE	8067279.8019
MAPE	3.6524%

3.3. Итоговые результаты и выбор модели

В рамках данного исследования было проведено всестороннее сравнение шести различных моделей прогнозирования временных рядов с целью определения их эффективности при работе с реальными данными. В качестве исследуемого объекта использовался набор данных, содержащий информацию о количестве пользователей поисковых систем за период с января 2014 года по апрель 2023 года. Для объективной оценки качества построенных моделей был выделен тестовый сегмент, составляющий 10% от общего объёма данных, на котором и производилась финальная проверка точности прогнозов. Оценка производительности моделей осуществлялась с применением стандартных метрик качества прогнозирования, таких как MAE, MSE, RMSE и MAPE.

Таблица 7

Результаты тестирования моделей

Метрика	Хольт-Уинтерс	SARIMA	LSTM	XGBoost	Prophet	SARIMA-LSTM
MAE	6.00×10^6	4.65×10^6	6.80×10^6	3.05×10^6	1.38×10^7	5.70×10^6
MSE	6.90×10^{13}	4.90×10^{13}	7.14×10^{13}	1.60×10^{13}	2.12×10^{14}	6.51×10^{13}
RMSE	8.30×10^6	7.00×10^6	8.45×10^6	4.00×10^6	1.46×10^7	8.07×10^6
MAPE	3.8869%	2.9672%	4.5165%	2.0260%	9.1156%	3.6524%

Все протестированные модели продемонстрировали значение средней абсолютной процентной ошибки, значительно ниже установленного порогового значения в 15%, что свидетельствует о высокой точности полученных прогнозов. Такие результаты подтверждают применимость выбранных моделей для анализа и прогнозирования данного временного ряда, а также указывают на то, что выявленные зависимости и структура данных были успешно учтены в процессе моделирования. Это позволяет использовать разработанные модели в практических задачах.

3.4. Выбор модели

При сравнении метрик наилучшие результаты по всем показателям продемонстрировала модель XGBoost, показав самый низкий MAPE в 2.0260%, а также минимальные MAE, MSE и RMSE. Это говорит о высокой точности и надежности данной модели для прогнозирования количества пользователей поисковых систем на исследуемых данных.

Модель SARIMA также показала высокую точность с $\text{MAPE} = 2.9672\%$, что делает ее вторым предпочтительным вариантом. Модель LSTM показала значение $\text{MAPE} = 4.5165\%$. Гибридная модель SARIMA-LSTM и модель Хольта-Уинтерса продемонстрировали схожие результаты с $\text{MAPE} = 3.6524\%$ и 3.8869% соответственно. Модель Prophet, несмотря на свою простоту использования и интерпретируемость, показала наименьшую точность среди всех рассмотренных моделей с $\text{MAPE} = 9.1156$

Таким образом, для данного набора данных и задачи прогнозирования XGBoost является наиболее предпочтительной моделью благодаря наивысшей точности прогноза.

Заключение

В результате выполнения выпускной квалификационной работы был разработан и реализован прототип интерактивного программного инструмента для анализа и прогнозирования временных рядов. Для достижения поставленной цели были решены следующие ключевые задачи:

- 1) Проведен детальный анализ требований к функциональности и интерфейсу приложения, ориентированного на анализ временных рядов.
- 2) Выбран и обоснован технологический стек, включающий Python и специализированные библиотеки для обеспечения функциональных возможностей и эффективности работы.
- 3) Реализован графический интерфейс, обеспечивающий понятное взаимодействие пользователя с приложением для выполнения анализа
- 4) Разработаны модули для загрузки, предварительной обработки данных из CSV-файлов, включая выбор столбцов и установку частоты ряда .
- 5) Интегрирован широкий набор моделей прогнозирования, включая статистические методы, модели машинного обучения и гибридный подход
- 6) Реализован механизм оценки качества прогнозов с помощью метрик.
- 7) В ходе исследования работы приложения на реальных данных о количестве пользователей поисковых систем было установлено, что модель XGBoost продемонстрировала наилучшие способности, показав наименьшее значение средней абсолютной процентной ошибки (MAPE) в 2.0260%.

Разработанный прототип представляет собой масштабируемую платформу. Дальнейшее развитие приложения может включать добавление новых моделей прогнозирования, реализацию более сложных методов обработки пропусков и выбросов, интеграцию возможности учёта экзогенных переменных, а также расширение функционала автоматизированной генерации отчётов.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Айвазян С. А., Мхитарян В. С. Прикладная статистика и основы эконометрики : учебник для вузов. — М. : ЮНИТИ, 1998. — С. 1022.
2. Андерсон Т. Статистический анализ временных рядов. — М. : Мир, 1976. — С. 760. — Пер. с англ.
3. Архипова А. А. Применение нейронных сетей в задаче прогнозирования финансовых временных рядов // Economy and Business. — 2023. — № 6. — С. 18—22. — DOI: 10.24412/2411-0450-2023-6-1-18-22.
4. Афанасьев В. Н., Юзбашев М. М. Анализ временных рядов и прогнозирование. — М. : Финансы и статистика, 2010. — С. 320.
5. Бокс Д., Дженкинс Г. Анализ временных рядов, прогноз и управление. Вып. 1. — М. : Мир, 1974. — С. 406. — Пер. с англ. под ред. В. Ф. Писаренко.
6. Бондарь А. В. Алгоритмы машинного обучения для прогнозирования спроса на товары и услуги // Наука. Бизнес. Информатика. — 2024. — Т. 2, № 4. — DOI: 10.15688/NBIT.jvolsu.2024.2.4.
7. Воронцов К. В. Математические методы обучения по прецедентам (теория обучения машин). Курс лекций. — б.г. — URL: machinelearning.ru (дата обр. 12.5.2025). Электронный ресурс.
8. Гаврикова С. Обзор баз данных временных рядов // International Journal of Open Information Technologies. — 2023. — Т. 11, № 11. — С. 83—101.
9. Гудвин Г., Пейн К. Проектирование систем управления. — М. : Мир, 1978. — Пер. с англ.
10. Дайитбегов Д. М., Горелик В. А., Трегуб В. Г. Анализ и прогнозирование временных рядов (эконометрический подход) : Учебное пособие. — М. : Вузовский учебник, 2009. — С. 138.

11. Джоши П. Искусственный интеллект с примерами на Python / под ред. В. Р. Гинзбург ; пер. А. Г. Гузикевич. — Санкт-Петербург : Диалектика, 2019. — С. 448.
12. Дьяченко С. С., Мохаммад А. А. Обнаружение аномалий в видеопотоке при помощи анализа временных рядов // Международный научный журнал «Вестник науки». — 2024. — Т. 2, 6 (75). — С. 1497—1500. — DOI: 10.24412/2071-6168-2024-6-75-2.
13. Жерон О. Прикладное машинное обучение с помощью Scikit-Learn, Keras и TensorFlow. Концепции, инструменты и техники для создания интеллектуальных систем. — 2-е изд. — М. : Диалектика, 2020. — Пер. с англ.
14. Задачи анализа и моделирования тенденций временных рядов. — URL: <https://core.ac.uk/download/pdf/326324283.pdf> (дата обр. 12.5.2025).
15. Зелезецкий Д. В. Глубокое обучение в задаче прогнозирования финансовых временных рядов // Труды МФТИ. — 2024. — Т. 16, № 3. — С. 35.
16. Зиненко А. В. Прогнозирование финансовых временных рядов с использованием сингулярного спектрального анализа // Бизнес-информатика. — 2023. — Т. 17, № 3. — С. 88—100. — DOI: 10.17323/2587-814X.2023.3.87.100.
17. Исследование LSTM-нейросетевого подхода при моделировании временных рядов / Г. Г. Рапаков [и др.] // Вестник Череповецкого государственного университета. — 2023. — 3 (114). — С. 47—54. — DOI: 10.23859/1994-0637-2023-3-114-4.
18. Кендалл М. Д., Стьюарт А. Многомерный статистический анализ и временные ряды. — М. : Наука, 1976. — С. 736. — Пер. с англ.
19. Круглов В. В., Борисов В. В. Искусственные нейронные сети. Теория и практика. — М. : Горячая линия – Телеком, 2001. — С. 382.

20. Лоскутов А. Ю. Анализ временных рядов: курс лекций / Физический факультет МГУ. — URL: https://chaos.phys.msu.ru/loskutov/PDF/Lectures_time_series_analysis.pdf (дата обр. 12.5.2025).
21. Лукашин Ю. П. Адаптивные методы краткосрочного прогнозирования временных рядов : Учебное пособие. — М. : Финансы и статистика, 2003. — С. 416.
22. Лукьяница А. А., Шишкин А. Д. Цифровая обработка сигналов. — М. : Горячая линия – Телеком, 2019.
23. Магнус Я. Р., Катышев П. К., Пересецкий А. А. Эконометрика. Начальный курс. — М. : Дело, 2007. — С. 504.
24. Мандельброт Б. Фрактальная геометрия природы. — М. : Институт компьютерных исследований, 2002.
25. Меликов П. И. Изучаем основы Python. Практический курс для дата-аналитиков. — Москва, Алматы : Ай Пи Ар Медиа, EDP Hub (Идипи Хаб), 2023. — С. 480.
26. Многомерный статистический анализ в экономике : Учеб. пособие для вузов / Л. А. Сошникова [и др.] ; под ред. В. Н. Тамашевич. — М. : ЮНИТИ-ДАНА, 1999.
27. Натроби́на О. В., Рожкова А. Н. Анализ временных рядов в экономике: методы и приложения // Калужский государственный университет им. К. Э. Циолковского. — 2024. — 9 (1). — С. 127—130. — DOI: 10.24412/2411-0450-2024-9-1-127-130.
28. Николенко С., Каду́рин А., Архангельская Е. Глубокое обучение. Погружение в мир нейронных сетей. — СПб. : Питер, 2018. — С. 480.
29. Носко В. П. Эконометрика. Введение в регрессионный анализ временных рядов. — М. : Экономический факультет МГУ, ТЕИС, 2002.

30. Осипов В. С., Цыпин А. П. Эволюция научной парадигмы построения и статистического изучения исторических временных рядов // Государственное управление. Электронный вестник. — 2023. — № 101. — С. 72—84. — DOI: 10.24412/2070-1381-2023-101-72-84.
31. Осовский С. Нейронные сети для обработки информации / пер. И. Д. Рудинский. — М. : Финансы и статистика, 2002. — С. 344. — Пер. с польск.
32. Рашка С., Мирйалили В. Python и машинное обучение. Машинное и глубокое обучение с использованием Python, scikit-learn и TensorFlow. — М. : ДМК Пресс, 2017. — Пер. с англ.
33. Сегментация активности по сессиям во временных данных / StatCounter. — URL: <https://www.kaggle.com/code/sykuramisomo/Session-detection-in-time-series-data-ru> (дата обр. 12.5.2025) ; Электронный ресурс.
34. Сизых Н. В., Оршанская Е. С., Сизых Д. С. Прогностическая способность гибридных методов прогнозирования котировок акций на примере ARIMA/LSTM // Machine Learning and Smart Data. — 2022. — DOI: 10.25728/mlsd.2022.0.
35. Смородинский С. С. Анализ и прогнозирование временных рядов в экономике и финансах. — Минск : БГЭУ, 2011.
36. Тарланов А. Т. Базы данных и дополнительные компоненты библиотеки PyQT : учебно-методическое пособие. — 2021. — URL: <https://e.lanbook.com/book/176526> (дата обр. 17.4.2025) ; Текст : электронный // Лань : электронно-библиотеческая система.
37. Тейлор Ш. Дж. и Летам Б. Прогнозирование в масштабе (Forecasting at Scale) // The American Statistician. — 2018. — Т. 72, № 1. — С. 37—45. — DOI: 10.1080/00031305.2017.1380080.

38. Хайкин С. Нейронные сети: полный курс. — 2-е изд. — М. : Вильямс, 2006. — С. 1104. — Пер. с англ.
39. Хайндман Р. Прогнозирование: принципы и практика. — 3-е изд. — М. : Логунов А., 2023. — С. 458. — Пер. с англ.
40. Червяков А. А., Никульчев Е. В. Робастное интервальное прогнозирование временных рядов // International Journal of Open Information Technologies. — 2023. — Т. 11, № 4. — С. 122—128.
41. Чубукова И. А. Data Mining : учебное пособие. — М. : Интернет-Университет Информационных Технологий (ИНТУИТ), Бином. Лаборатория знаний, 2008.
42. Шолле Ф. Глубокое обучение на Python. — СПб. : Питер, 2018. — Пер. с англ.
43. Эконометрика : учебник / И. И. Елисеева [и др.] ; под ред. И. И. Елисеева. — 2-е изд., перераб. и доп. — Москва : Финансы и статистика, 2007. — С. 576.
44. Эконометрика. Практикум : Учебно-практическое пособие / под ред. И. А. Кацко. — Москва : КНОРУС, 2019. — С. 218.

ПРИЛОЖЕНИЕ 1.

Структурная схема программы

```
1  # --- Импорты (основные библиотеки и UI компоненты) ---
2  import sys
3  import pandas as pd
4  import numpy as np
5  # Импорты для моделей (statsmodels, tensorflow, prophet, xgboost)
6  from PyQt5.QtWidgets import (
7      QApplication, QWidget, QVBoxLayout, QPushButton, QHBoxLayout,
8      QFrame, QLabel, QComboBox, QMenu, QSizePolicy,
9      QTableWidget, QStackedWidget, QSpinBox, QDoubleSpinBox,
10     QMessageBox, QDialog, QFormLayout, QDialogButtonBox, QProgressDialog
11 )
12 from PyQt5.QtWebEngineWidgets import QWebEngineView
13 from PyQt5.QtCore import Qt, QUrl, QPropertyAnimation, QEasingCurve, QSize
14 from PyQt5.QtGui import QIcon
15 # Импорты метрик и утилит (sklearn, tempfile, traceback, logging)
16
17 # --- Вспомогательные функции (кратко) ---
18 # def load_styles_from_json(file_path): ...
19
20 # --- Пользовательские виджеты и диалоги (кратко) ---
21 # class DragDropWidget(QLabel): ...
22 # class LSTMConfigDialog(QDialog): ...
23 # class XGBoostConfigDialog(QDialog): ...
24 # class SARIMAConfigDialog(QDialog): ...
25 # class HybridConfigDialog(QDialog): ...
26
27 class MainWindow(QWidget):
28     """
29     Основной класс приложения, управляющий интерфейсом и логикой
30     анализа временных рядов.
```

```

31     """
32     def __init__(self):
33         super().__init__()
34         self.setWindowTitle("Автоматизация временных рядов")
35         self.resize(1000, 700) # Установка начального размера
36
37         # --- Атрибуты для хранения состояния и данных ---
38         self.raw_df = None # Исходный DataFrame
39         self.current_df = None # Обработанный DataFrame (после настроек)
40         self.train_df = None # Обучающая выборка
41         self.test_df = None # Тестовая выборка...
42
43         # --- Основная компоновка интерфейса ---
44         self.layout = QHBoxLayout(self) # Горизонтальный менеджер компоновки
45         self.layout.setContentsMargins(0, 0, 0, 0) # Убираем внешние отступы
46         self.layout.setSpacing(0) # Убираем промежутки между виджетами
47
48         # Боковая панель (фрейм)
49         self.sidebar = self.create_sidebar() # создается отдельным методом
50         self.layout.addWidget(self.sidebar)
51
52         # Основная рабочая область (контейнер с вертикальной компоновкой)
53         main_content_container = QWidget()
54         # Вертикальный лейаут
55         self.main_area_layout = QVBoxLayout(main_content_container)
56         self.main_area_layout.setContentsMargins(5, 5, 5, 5)
57
58         # Виджет для перетаскивания CSV файла
59         self.dragdrop_widget = DragDropWidget(self.load_csv) # Кастомный QLabel
60         self.main_area_layout.addWidget(self.dragdrop_widget)
61
62         # Стек для отображения графика или таблицы
63         self.stack_widget = QStackedWidget()

```



```

64     self.plot_canvas = QWebEngineView() # Для отображения Plotly графиков
65     self.table_widget = QTableWidgetItem() # Для отображения данных в таблице
66     self.stack_widget.addWidget(self.plot_canvas) # Индекс 0
67     self.stack_widget.addWidget(self.table_widget) # Индекс 1
68     self.stack_widget.setCurrentIndex(0) # По умолчанию - график
69     self.stack_widget.hide() # Изначально скрыт
70
71     self.main_area_layout.addWidget(self.stack_widget)
72     self.layout.addWidget(main_content_container)
73
74     # --- Инициализация (загрузка стилей, отключение контролов) ---
75     # try: self.setStyleSheet(load_styles_from_json("styles.json"))
76     # except FileNotFoundError: print("styles.json not found")
77     self._disable_controls() # Отключение большинства контролов при запуске
78
79     def create_sidebar(self):
80         """ Создает и возвращает боковую панель (QFrame) с элементами управления
81         . """
82
83         sidebar_frame = QFrame()
84
85         # sidebar_frame.setMaximumWidth(...)
86
87         sidebar_frame.setObjectName("SidebarFrame") # Для стилей
88
89         # Вертикальная компоновка для содержимого панели
90         sidebar_content_layout = QVBoxLayout()
91         sidebar_content_layout.setAlignment(Qt.AlignTop) # Выравнивание элементов
92         по верху
93
94         # --- 1. Главное меню (Файл и Вид) ---
95         main_menu = QMenu(self)
96         load_action = main_menu.addAction("Загрузить CSV")
97         clear_action = main_menu.addAction("Очистить все")
98         toggle_view_action = main_menu.addAction("Переключить график/таблицу")
99         load_action.triggered.connect(self.load_csv_from_dialog)

```

```

95     clear_action.triggered.connect(self.clear_all)
96     toggle_view_action.triggered.connect(self.toggle_view)
97     self.main_menu_btn = QPushButton("Файл и Вид")
98     self.main_menu_btn.setMenu(main_menu)
99     # self.main_menu_btn.setIcon(...) # Установка иконки
100     self.original_icon_sizes[self.main_menu_btn] =
self.main_menu_btn.iconSize()
101     sidebar_content_layout.addWidget(self.main_menu_btn)
102
103     # --- 2. Настройки данных ---
104     # sidebar_content_layout.addWidget(QLabel("Настройки данных:"))
105     self.date_col_combo = QComboBox() # Выбор столбца даты/времени
106     self.value_col_combo = QComboBox() # Выбор столбца значения
107     self.freq_combo = QComboBox()      # Выбор частоты данных
108     self.freq_combo.addItem('D', 'B', 'W', 'M', 'Q', 'Y'])
109     self.apply_settings_btn = QPushButton("Применить настройки данных")
110     self.apply_settings_btn.clicked.connect(self.apply_data_settings)
111     # ... добавление QLabel и QComboBox/QPushButton в sidebar_content_layout
...
112
113     # --- 3. Разделение на выборки ---
114     # sidebar_content_layout.addWidget(QLabel("Разделение на выборки:"))
115     self.test_size_spinbox = QSpinBox() # Размер тестовой выборки (%)
116     self.test_size_spinbox.setRange(5, 50)
117     self.test_size_spinbox.setValue(20)
118     self.split_data_btn = QPushButton("Разделить данные")
119     self.split_data_btn.clicked.connect(self.split_data)
120     # ... добавление QLabel и QSpinBox/QPushButton в sidebar_content_layout
...
121
122     # --- 4. Модели прогнозирования (кнопки с меню) ---
123     # sidebar_content_layout.addWidget(QLabel("Модели прогнозирования:"))
124     # Количественные методы (Holt-Winters, SARIMA)

```

```

125         quant_menu = QMenu(self);
quant_menu.addAction("Holt-Winters").triggered.connect(self.holt_winters_prediction)
126         sarima_action = quant_menu.addAction("SARIMA");
sarima_action.triggered.connect(self.show_sarima_config_dialog)
127         self.quant_menu_btn = QPushButton("Количественные методы");
self.quant_menu_btn.setMenu(quant_menu)
128         # ML модели (LSTM, XGBoost, Prophet)
129         nn_menu = QMenu(self);
nn_menu.addAction("LSTM").triggered.connect(self.show_lstm_config_dialog)
130
nn_menu.addAction("XGBoost").triggered.connect(self.show_xgboost_config_dialog)
131         nn_menu.addAction("Prophet").triggered.connect(self.prophet_prediction)
132         self.nn_menu_btn = QPushButton("Модели машинного обучения");
self.nn_menu_btn.setMenu(nn_menu)
133         # Гибридные модели (SARIMA-LSTM)
134         hybrid_menu = QMenu(self);
hybrid_menu.addAction("SARIMA-LSTM").triggered.connect(self.show_hybrid_config_dialog)
135         self.hybrid_menu_btn = QPushButton("Гибридные модели");
self.hybrid_menu_btn.setMenu(hybrid_menu)
136         # ... добавление кнопок моделей в sidebar_content_layout ...
137
138         # Растягивающийся элемент и кнопка сворачивания
139         sidebar_content_layout.addStretch(1)
140         sidebar_content_widget = QWidget();
sidebar_content_widget.setLayout(sidebar_content_layout)
141         sidebar_content_widget.setObjectName("SidebarContent")
142         horizontal_layout = QHBoxLayout(sidebar_frame);
horizontal_layout.setContentsMargins(0,0,0,0);
horizontal_layout.setSpacing(0)
143         horizontal_layout.addWidget(sidebar_content_widget);
144         self.toggle_button = QPushButton("<");
self.toggle_button.setFixedWidth(15);
self.toggle_button.clicked.connect(self.toggle_sidebar)

```

```

145         horizontal_layout.addWidget(self.toggle_button)
146
147     return sidebar_frame
148
149     # --- Методы управления UI (видимость, анимация) ---
150     def toggle_sidebar(self): pass # Анимация сворачивания/разворачивания
151     def _toggle_sidebar_elements_visibility(self, visible): pass # Скрытие/показ
элементов
152     def _update_button_text_positions(self, expanded): pass # Обновление текста
/размера элементов
153     def _disable_controls(self, disable_all=True): pass # Отключение контролов
154     def _enable_data_settings_controls(self): pass # Включение контролов данных
155     def _enable_analysis_controls(self): pass # Включение контролов анализа
156
157     # --- Обработчики событий и основная логика данных ---
158     def clear_all(self): pass # Сброс данных и UI
159     def load_csv(self, file_path): pass # Загрузка и первичная обработка CSV
160     def load_csv_from_dialog(self): pass # Вызов диалога выбора файла
161     def _populate_column_selectors(self, columns): pass # Заполнение комбо
-боксов столбцов
162     def apply_data_settings(self): pass # Применение выбранных столбцов и частоты
163     def _update_default_seasonality(self): pass # Определение сезонности по часто
те
164     def split_data(self): pass # Разделение данных на train/test
165
166     # --- Методы конфигурации и вызова моделей прогнозирования ---
167     def show_lstm_config_dialog(self): pass # Открытие диалога LSTM
168     def lstm_prediction(self, params): pass # Прогнозирование LSTM
169     def holt_winters_prediction(self): pass # Прогнозирование Holt-Winters
170     def show_sarima_config_dialog(self): pass # Открытие диалога SARIMA
171     def sarima_prediction(self, params): pass # Прогнозирование SARIMA
172     def show_hybrid_config_dialog(self): pass # Открытие диалога SARIMA-LSTM
Hybrid

```

```

173     def sarima_lstm_prediction(self, params): pass # Прогнозирование SARIMA-LSTM
Hybrid
174     def show_xgboost_config_dialog(self): pass # Открытие диалога XGBoost
175     def create_xgboost_features(self, df, target_col, n_lags=1): pass # Создание
признаков для XGBoost
176     def xgboost_prediction(self, params): pass # Прогнозирование XGBoost
177     def prophet_prediction(self): pass # Прогнозирование Prophet
178
179     # --- Методы отображения и метрик ---
180     def plot_data(self, df_to_plot, forecast_df=None, title_suffix="",
split_point=None): pass # Построение и отображение графика
181     def display_table(self, df): pass # Отображение данных в таблице
182     def toggle_view(self): pass # Переключение между графиком и таблицей
183     def calculate_and_display_metrics(self, y_true, y_pred, model_name): pass # Р
асчет и вывод метрик
184
185 # --- Точка входа в приложение ---
186 # if __name__ == '__main__':
187 #     app = QApplication(sys.argv)
188 #     app.setStyle("Fusion") # Применение стиля
189 #     win = MainWindow()
190 #     win.show()
191 #     sys.exit(app.exec())

```

Листинг 1. Структурная схема программы