

Решение задачи прогнозирования динамики курса акций с помощью методов машинного обучения.

**от
Хена Агнон**

РЕФАТ

Тема исследования «Решение задачи прогнозирования динамики курса акций с помощью методов машинного обучения» в основном направлена на оценку нескольких моделей машинного обучения для прогнозирования курса акций трех компаний, а именно Amazon, Apple и Google. курса акций являются временными рядами, для прогнозирования временных рядов были реализованы многие методы машинного обучения. Прогнозирование временных рядов может применяться в различных областях, особенно в финансовой сфере. В этой диссертации изучается исследование прогнозирования временных рядов KERAS с помощью моделей LSTM (Long Short-Term Memory), ARIMA и PROPHET, эти модели были разработаны для прогнозирования скорректированной дневной цены закрытия выбранных акций с использованием данных за последние 10 лет, собранных из Yahoo Finance (с 1 апреля 2011 г. по 1 апреля 2021 г.). Результаты прогноза

показывают, что все модели трех временных рядов обладают высоким потенциалом в прогнозировании временных рядов. Keras с LSTM дает лучшие результаты по сравнению с двумя другими моделями прогнозирования временных рядов.

Глава 1 Введение

«Фондовый рынок - это средство для перевода денег от нетерпеливых к пациенту», - Уоррен Баффет (американский инвестор и генеральный директор Berkshire Hathaway)

«Четыре самых опасных слова в инвестировании: на этот раз все по-другому», - сэр Джон Темплтон (британский инвестор)

1.1. Термины и определение

Что такое курса акций и рынок акций ?

Рынок акций - это прогноз покупок и продаж компании в определенный рабочий день. Фондовая биржа похожа на аукцион, где инвесторы могут продавать и покупать акции акций, а инвесторы, владеющие акциями этих акций в компании или организации, называются акционерами . Фондовая биржа - это биржа, на которой трейдеры и биржевые маклеры покупают и продают акции. Фондовый рынок в основном имеет дело с ценными бумагами, и эти ценные бумаги задействованы таким образом, что они представляют собой ценные бумаги с фиксированной процентной ставкой, а торговля на фондовой бирже означает передачу ценной бумаги или акций от покупателя к продавцу [1].

Финансовая деятельность на фондовом рынке осуществляется через официальные официальные биржи или внебиржевые (внебиржевые) рынки, которые в основном будут работать в соответствии с обычным набором правил. Фондовый рынок и фондовая биржа, оба термина используются в бизнесе

взаимозаменяемо, но последний является подмножеством первого (пример: если кто-то торгует на рынке акций, это означает, что они могут покупать и продавать акции на одной из фондовых бирж, часть всего фондового рынка).

Ринок акций , который в основном используется для торговли акциями, также может быть полезен для других финансовых ценных бумаг, таких как корпоративные облигации, торгуемые на бирже фонды и некоторые другие основные производные инструменты, такие как товары, валюты и облигации, которые торгуются на фондовом рынке. В современном мире большинство людей могут совершать покупки в Интернете, но некоторые вещи можно купить только за пределами Интернета, для чего фондовый рынок играет решающую роль, в котором он обозначен аналогично рынку для различных торговых ценных бумаг в контролируемом , безопасная и управляющая среда. Основным фактором фондового рынка является то, что он объединяет сотни тысяч участников, желающих купить или продать акции, что делает их справедливыми и прозрачными в сделках [2].

Работа фондового рынка довольно проста, поскольку он позволяет широкой публике сделать выбор, инвестировать ли свои акции в конкретную организацию, позволяя им покупать или продавать акции организации населению в процессе первичного публичного размещения акций. . Эта деятельность поможет организации иметь дело со своими инвесторами. Чтобы инициировать эти процессы, организации нужна торговая площадка, на которой она может продавать свои акции, и место этой необходимой торговой площадки называется .

Что такое прогноз курса акций?

Будущая стоимость отдельной акции, рынка или всего рынка может быть нацелена на предсказание, это известно как прогнозирование фондового рынка. Обычно он использует технический анализ графиков или фундаментальный анализ компании или их комбинацию. Поскольку все инвесторы хотят предсказать стоимость акций, будет больше прогнозов фондового рынка, сделанных самопровозглашенными экспертами в СМИ и опубликованных инвестиционными консультантами и брокерами [3].

Короче говоря, прогноз фондового рынка - это определение будущей стоимости компании, с которой ее финансовый инструмент торговался на бирже. Успех биржевой цены будущего даст значительную прибыль в области фондового рынка. Существуют различные аспекты прогнозирования фондового рынка, а именно: фундаментальный анализ, технический анализ и машинное обучение.

Фундаментальный анализ в основном основывается на прошлых результатах деятельности компании, что дает надежность ее счетов, и в основном это делается с использованием коэффициента производительности, который в основном помогает при этом.

Технический анализ не рассматривается с фундаментальными принципами компании, здесь методы используются для прогнозирования будущей цены акций, и эти методы включают в себя экспоненциальную скользящую среднюю (ЕМА), осцилляторы, индикаторы объема и т. д. используется технический анализ в настоящее время.

Машинное обучение - это еще один способ прогнозирования фондового рынка, который использует множество алгоритмов, в основе которых лежат все эти алгоритмы на основе искусственных нейронных сетей (ИНС) и генетических алгоритмов. Основным аспектом искусственной нейронной сети является использование прогнозирования временных рядов, с помощью которого прогнозируются ожидаемые изменения тренда на многочисленных фондовых рынках и наборы данных временных рядов.

Прогностический анализ используется организациями или компаниями, отдельными лицами во всем мире путем извлечения исторических данных. Различные методы выполнения структуры прогноза, такие как машинное обучение, технический анализ, фундаментальный анализ и прогнозирование временных рядов. Продвигая инновации, искусственный интеллект включает в себя сознание, созданное руками человека, которое позволяет структуре подготовиться и, по сути, улучшиться [4].

Что такое прогнозирование временных рядов?

Самый важный аспект машинного обучения связан с прогнозированием временных рядов, и он важен, потому что здесь решается большинство проблем прогнозирования, включая временную составляющую. Прежде чем мы увидим время, прогнозирование рядов позволит нам больше узнать о временных рядах. Временные ряды моделируются с помощью стохастического процесса и в основном состоят из четырех компонентов: уровня, тренда, сезонности и шума.

Уровень- значение базовой линии в ряду, если это прямая линия.

Тренд - линейное возрастание или убывание ряда во времени.

Сезонность-представляет собой циклы поведения или повторяющиеся модели с течением времени.

Шум-это изменение в наблюдениях, которое невозможно объяснить для данной модели.

Прогнозирование временных рядов заключается в прогнозировании будущего, а процесс прогнозирования будущего называется экстраполяцией при статистической обработке временных рядов. Современная область в основном сосредоточена на теме и, следовательно, называет ее прогнозированием временных рядов. Прогнозирование в основном включает в себя сбор исторических данных, которые соответствуют модели, и их использование для прогнозирования будущих наблюдений, и важный аспект, который необходимо учитывать, если будущее не может произойти, и его можно предсказать только с оценкой уже произошедших событий.

Цель прогнозирования временных рядов состоит в основном из двух частей: понять или смоделировать стохастические механизмы, которые приводят к наблюдаемому ряду, и предсказать или спрогнозировать будущие значения ряда на основе уже известных исторических данных этого ряда. Прогнозирование временных рядов в основном используется в R и Python, и у него есть много типов моделей, а именно: наивное, экспоненциальное сглаживание, ARIMA, динамические линейные модели, FBProphet, LSTM. В этой статье в основном используются ARIMA, FBProphet и LSTM.

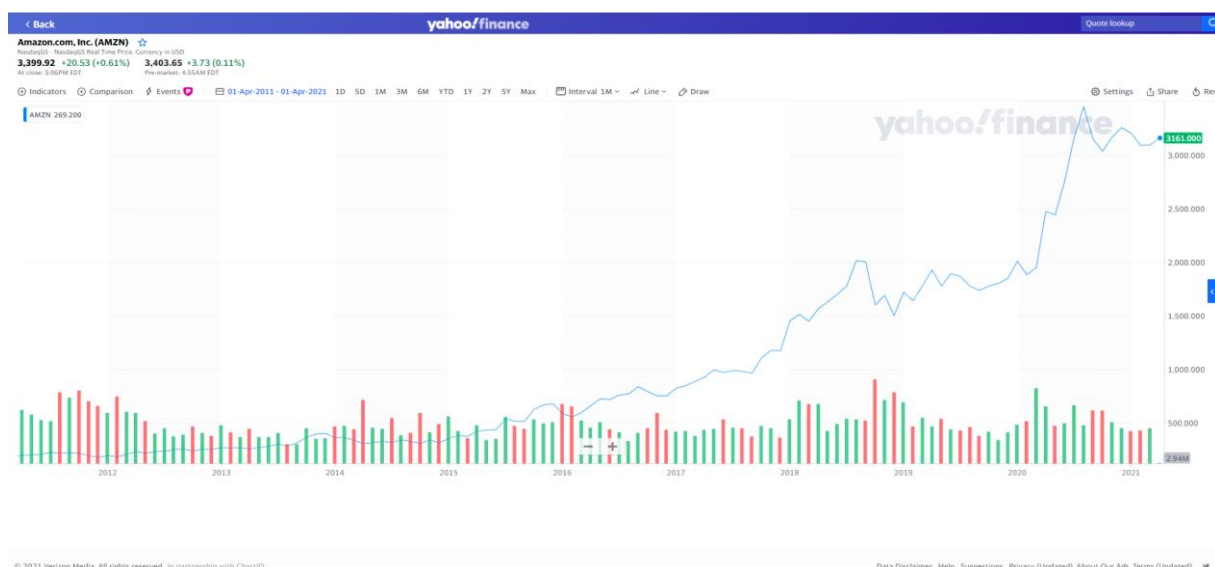


Рисунок 1: Представление цены акций компании Amazon в Yahoo Finance. [5]

На рисунке 1. показан пример представления данных временных рядов компании Amazon за 10 лет.

1.2. Обзор подхода

Этот тезис в первую очередь сосредоточен на прогнозировании цен на фондовом рынке трех всемирно известных компаний (Amazon, Apple и Google) с использованием моделей временных рядов. Yahoo Finance используется для сбора данных за 10 лет. Значение цены закрытия в наборе данных используется для прогнозирования будущей цены. За это KERAS с LSTM (Long Short Term-Memory), Пророк и ARIMA используются алгоритмы.

Для анализа временных рядов используется язык программирования Python версии 3. Jupyter Notebook используется для обработки и визуализации данных в библиотеках импорта первичного этапа и пакетах python для функционирования модели. Данные разделяются для целей обучения и тестирования. На

следующем этапе разрабатываются модели временных рядов. Данные прогнозируются и прогнозируются на финальном этапе. Посредством вычисления значения RMSE каждого алгоритма эффективность и точность каждого алгоритма определяется и визуализируется с помощью гистограммы.

Список технологий, использованных для реализации

- Язык программирования - Python версии 3.
- Библиотеки - Pandas, NumPy, Sklearn, TensorFlow, Keras.
- IDE - Блокнот Jupyter
- Пакет Python - Anaconda 3

1.3. Структура документа

Документ состоит из восьми глав. В первой главе содержится определение проблемы, общий вид проекта. Во второй главе рассказывается о мотивации проекта и исследовательской проблеме. В третьей главе описывается подробное изучение моделей временных рядов, анализ и прогнозирование в существующих исследовательских работах. В четвертой главе рассматривается подход CRISP DM и описывается используемый алгоритм. В пятой главе дается определение базовой структуры - аппаратного и программного обеспечения, диаграммы артефактов. В шестой главе он определяет сбор данных, предварительную обработку и т. Д. В седьмой главе описывается развертывание моделей прогнозирования - тестирование и оценка. В восьмой главе рассказывается о завершении и дальнейшей работе над проектом.

1.4. Дорожная карта диссертации



Рисунок 2: Дорожная карта диссертации

Глава 2. Мотивация и проблема исследования

Фондовый рынок или фондовый рынок имеет подавляющее влияние на сегодняшнюю экономику. Повышение или падение цены акций играет важную роль в определении прибыли инвестора. Прогнозирование фондового рынка всегда очень сложно. Никто не может видеть будущее - мир по своей природе неопределенен, и будут происходить удивительные вещи. Однако, даже если известно, что произойдет, вы можете не знать, как отреагируют рынки. Инвестиции в фондовый рынок считаются высокими рисками и высокой прибылью и поэтому привлекают большое количество инвесторов и экономистов. Однако информация о запасах обычно является неполной, сложной, неопределенной и расплывчатой, что затрудняет прогнозирование будущих экономических показателей. Люди инвестируют в фондовый рынок на основе некоторого анализа.

2.1. Исследовать вопрос

- Определение наиболее подходящей модели временных рядов для прогнозирования цен на фондовом рынке?
- Как прогнозирование цен на фондовом рынке помогает развитию экономики?

Ежедневный прогноз цен на фондовом рынке помогает инвесторам определить, какая компания больше подходит для вложения их денег. Прогнозируя прогноз цен на фондовом рынке, правительство также может определить, какие факторы влияют на рост и падение цен на акции?

2.2. Модели прогнозирования

1. KERAS с LSTM (долгосрочная краткосрочная память).
2. Модель Prophet.
3. Модель ARIMA (авторегрессионная интегрированная скользящая средняя).

Глава 3. Предварительное исследование

В этой главе обсуждались предпосылки исследования исследовательских работ, связанных с проектом.

3.1. Литературный обзор

3.1.1. Фондовая биржа: рождение

Акция - это слово, которое используется для обозначения собственности инвестора в компании, и те, кто владеет акциями, называются акционерами. Представление о современном фондовом мире зародилось в конце 16 века. Когда торговцы хотят начать крупный бизнес, у них нет ни одного крупного инвестора, поэтому они решили собрать сбережения от всех инвесторов, которые заинтересованы в том, чтобы отдать свою долю и внести свой вклад в крупный бизнес и, следовательно, стать деловыми партнерами. Все началось еще в 1602 году, когда голландская Ост-Индская компания выпустила первую бумагу акций, которая позволила акционерам удобно покупать, продавать и обменивать свои акции с другими акционерами [8].

3.1.2. Система прогнозирования цен на фондовом рынке - модульная нейронная сеть

Он основан на системе прогнозирования времени покупки и продажи акций на Токийской фондовой бирже и концентрируется в основном на модульных нейронных сетях. Ряд алгоритмов обучения и методов прогнозирования был разработан для системы

прогнозирования, которая также называется TOPIX (Индексы цен на Токийской фондовой бирже), и эта система прогнозирования дала точный прогноз, моделирование торговли акциями [9].

3.1.3. Прогнозирование цен на акции с использованием модели ARIMA (авторегрессивная интегрированная скользящая средняя)

С точки зрения экономики и финансов прогнозирование цен на акции играет решающую роль и дает больше интересов в улучшении разработки прогнозных моделей. Вот где модели авторегрессивной интегрированной скользящей средней (ARIMA) играют важную роль в прогнозировании временных рядов. Этот документ в основном сосредоточен на прогнозировании цен на акции с использованием модели ARIMA, и в нем в основном используются данные Нью-Йоркской фондовой биржи (NYSE) и Нигерийской фондовой биржи (NSE), используемые с моделями прогнозирования акций, и полученный результат.

в этой статье говорится, что модель ARIMA сильна для краткосрочного прогнозирования, а также поддерживает существующие методы прогнозирования цен на акции [10].

3.1.4. Прогнозирование цен на акции с использованием RNN, CNN и LSTM

Рост и падение рыночной цены акций решают судьбу прибыли инвестора. Существующие алгоритмы, которые являются линейными и нелинейными, ориентированы на прогнозирование движения фондовых индексов для отдельной компании, но, с другой стороны, предлагаемый метод является независимым от

модели подходом, и данные не соответствуют конкретной модели, а скорее определяют скрытую динамику в данные с использованием глубокого обучения. Он выполняет прогноз цен для компаний, зарегистрированных на NSE (Нигерийская фондовая биржа), сравнивает их результаты и применяет подход скользящего окна для прогнозирования будущих значений на краткосрочной основе [11].

3.1.5. Прогноз цен на акции с использованием данных временных рядов

Исследователям потребовалось некоторое время, чтобы спрогнозировать данные с помощью алгоритмов, достаточно быстрых и точных, чтобы делать прогнозы цен на акции. Инвесторы в основном хотят, чтобы цены на их акции прогнозировались с помощью более умных методов, и этот метод им хорошо сработал. В этой статье основное внимание уделяется трем основным алгоритмам прогнозирования временных рядов: LSTM (Long Short-Term Memory), Auto-regressive Integrated Moving Average (ARIMA), алгоритмам временных рядов Facebook Prophet [12].

3.1.6. Прогнозирование тренда фондового рынка

Анализ тенденций фондового рынка полезен для оценки изменений атрибутов фондового рынка с течением времени. Прогнозы могут быть сделаны с определенным уровнем точности, но цель прогнозов - снизить риск. Линейная регрессия, FBProphet, байесовская регрессия используются для прогнозирования и сравнения стоимости акций. Новая функция риска для долгосрочного прогнозирования фондового рынка разработана с использованием моделей прогнозирования [13]

3.1.7. Нейронные сети для прогнозирования курса акций

Это исследование сосредоточено в первую очередь на том, как предыдущая информация и нейронные сети могут быть использованы для улучшения предсказательной способности. Ежедневное ценообразование акций - сложная проблема в реальном мире, учитывая политические и мировые события, поэтому мы должны получать информацию, и поэтому осведомленность о событиях в основном определяется заголовками. В этом документе также используются многие экономические меры и нейронные сети для их ввода вместе с информацией о событиях. Этот метод также дает меньше ошибок прогноза, чем множественный регрессионный анализ [14].

- **Сравнение нейронных сетей и моделей ARIMA**

Это вычисляет эффективность прогнозирования ARIMA и модели искусственных нейронных сетей, опубликованной для Нью-Йоркской фондовой биржи. Известные методы, используемые здесь, делятся на две категории, в основном статистические и вычислительные. В котором ARIMA относится к методам статистических вычислений, а искусственные нейронные сети относятся к категории вычислительных методов и этой слишком мягкой вычислительной техники. Использование ARIMA в качестве прогнозирования временных рядов очень важно, но использование искусственной нейронной сети является наиболее точным, и эта статья в основном прогнозирует их использование и

сравнивает, что хорошо для прогнозирования фондового рынка [15].

3.1.8. Прогнозирование цен на акции и прогнозирование тенденций - нейронные сети

Это в основном заключается в анализе сети с прямой связью с использованием алгоритма обучения с обратным распространением с ранней остановкой, а радиальная базовая нейронная сеть используется для прогнозирования тренда цены акций. В этом исследовании фундаментальные данные или технические индикаторы не используются в качестве основной цели исследования для прогнозирования использования искусственных нейронных сетей для будущих цен на основе прошлых цен [16].

3.1.9. Сравнение архитектуры искусственных нейронных сетей на складе

Взлеты и падения фондового рынка предсказываются с помощью искусственных нейронных сетей в этой статье, и она обеспечивает численный анализ конкретных финансовых временных рядов. Эта концепция состоит из таких алгоритмов, как сверточные нейронные сети (CNN), долгосрочная краткосрочная память (LSTM) и рекуррентные нейронные методы многослойного персептрона (MLP). В этой статье в основном показано, что нейронные сети предназначены для прогнозирования движений финансовых временных рядов при обучении на простых данных временных рядов [17].

3.1.10. Прогноз курса акций с использованием LSTM

Прогнозирование цен на фондовом рынке является сложной задачей для многих исследователей, но, тем не менее, инвесторы очень заинтересованы в исследованиях прогнозирования цен на фондовом рынке, и даже многие инвесторы очень заинтересованы в будущей ситуации с ценами на фондовом рынке и с учетом всего этого. в уме, эта бумага в основном представляет собой долговременную краткосрочную память.

(LSTM) и рекуррентная нейронная сеть (RNN) для прогнозирования цен на фондовом рынке в будущем [18].

Глава 4. Методология

В этой главе обсуждались три темы

1. Методология CRISP-DM.
2. Инструменты и технологии, использованные в этом проекте.
3. Модели прогнозирования - KERAS с LSTM, PROPHET и ARIMA

4.1. Диаграмма CRISP DM

CRISP DM представляет собой междотраслевой процесс интеллектуального анализа данных. Это сильная и хорошо зарекомендовавшая себя система. Его адаптируемость и удобство при изучении сложных бизнес-задач. Это блестящая линия, через которую проходит почти каждое взаимодействие с клиентом.

Эта модель - прославленная череда времен. Многие сотрудники компаний могут действовать альтернативным образом, и важно отслеживать прошлые задачи и отслеживать определенные действия на регулярной основе. Модель не пытается пройти все мыслимые пути в процессе добычи информации. Это 6-фазная модель. [19].

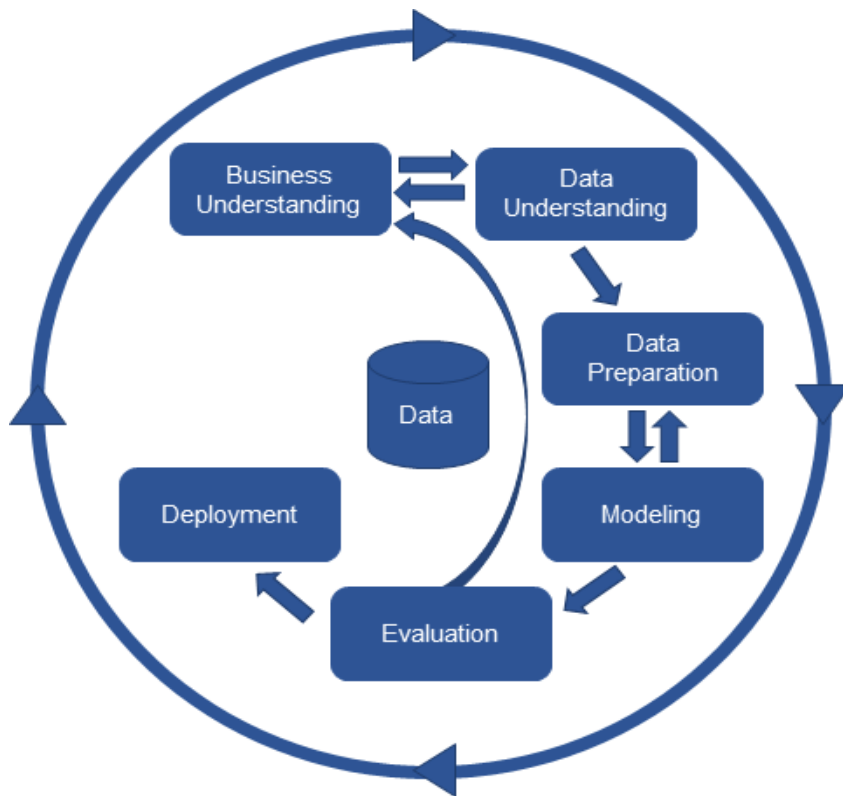


Рисунок 3: Диаграмма CRISP - DM. [20].

- Понимание бизнеса: определение бизнес-целей и задач интеллектуального анализа данных.
- Понимание данных: сбор данных из источников, изучение и проверка качества данных.
- Подготовка данных: этот этап требует много времени. На этом этапе происходит выбор, очистка, построение, интеграция данных.
- Моделирование: выбор методов моделирования, создание тестового проекта и доступ к моделям.
- Оценка: Оценка результатов.
- Развертывание: планирование развертывания и мониторинг.

Уильям Ворхис, один из создателей CRISP-DM, утверждает, что, поскольку все начинания в области информатики начинаются с

понимания бизнеса, имеют информацию, которую необходимо накапливать и очищать, и применяют вычисления в области информатики, «Fresh DM дает твердое направление даже самым исключительным настоящим упражнениям по науке о данных »[19].

4.2. Инструменты и технологии

4.2.1. Язык программирования Python

В этом проекте используется Python, поскольку Python - это интерпретируемый объектно-ориентированный язык программирования высокого уровня. Он был создан Гвидо Ван Россумом в 1990 году. Это кроссплатформенный язык программирования. Этот язык программирования можно свободно использовать в коммерческих целях. Python прост в использовании и очень мощный. Кроме того, этот язык универсален по своей природе. Такие компании, как Google, Facebook, Netflix и т. Д., Используют Python.

Преимущество Python:

Python - это объектно-ориентированный язык программирования, который мы можем использовать для научного и ненаучного программирования. Простой язык высокого уровня с множеством библиотек. Python - это платформенно-независимый язык программирования. Мы можем использовать python на серверах для создания веб-приложений. Он используется в системах баз данных, потому что может очень легко читать и изменять данные. Одним из основных преимуществ Python является то, что мы можем использовать его для обработки больших данных для выполнения сложной математики.

Python может работать на разных платформах. Python имеет простой синтаксис, очень похожий на английский язык. Прототипирование кода на Python выполняется очень быстро. Последняя версия python - это python версии 3 [21].

4.2.2. Панды

Пакет Pandas - один из самых быстрых и мощных пакетов на языке программирования Python. Он очень гибкий и простой в использовании для анализа данных с открытым исходным кодом. Он также используется для визуализации данных. Вот почему панды называют основой проектов в области науки о данных. Используя панды, мы можем очищать, преобразовывать и анализировать данные. Pandas может извлекать данные из файла csv во фреймы данных. Мы можем вычислять статистику, такую как среднее, среднее, максимальное и минимальное значение и т. Д. Визуализация данных с помощью matplotlib - еще одна особенность pandas. Мы можем построить линейный график, гистограмму, гистограмму и т.д. [22].

Основными важными структурами данных панд являются Panel. Панель - это структура данных, которая помогает управлять наборами данных и временными рядами.

Мы можем установить pandas с помощью команды pip и импортировать библиотеку panda с помощью функции импорта. Панды состоят из двух основных компонентов.

- Ряд
- DataFrame

Ряд:

Одномерный массив в пандах называется серией. Функция `pd.Series` используется для создания конструктора, который может включать множество аргументов. Обычно используемое имя аргумента называется данными, которое определяет все элементы в серии. Для ручного кастинга серии `panda` также использовалось ключевое слово `dtype`. Серии элементов вместе называются Индексом серии [23].

Параметры в серии `panda`: `data`, `index`, `dtype` и `copy`.

DataFrame:

`DataFrame` - это двумерный массив в пандах. Функция `pd.DataFrame` используется для создания `DataFrame`. `DataFrame` не скалярный. Индекс - это строки в `DataFrame`. Столбец - это еще один ярлык [23].

Параметры в `panda DataFrame` - это данные, индекс, столбцы, `dtype` и копия.

Преимущество панд

- Он может представлять данные в виде структур данных `Series` или `DataFrame`.
- Пакет `Panadas` содержит несколько методов фильтрации данных.
- Он очень эффективно обрабатывает данные.
- Он может создавать гибкие данные.

Недостатки панд

- У панд крутизна обучения очень крутая.
- У него сложный синтаксис.
- У него плохая документация.

- Панды нельзя использовать для трехмерных массивов [24].

4.2.3. NumPy

NumPy означает числовой питон. NumPy - это библиотека, используемая в программировании на Python. Это быстро и универсально. Он используется для обработки мощных N-мерных массивов. NumPy состоит из функций, которые могут работать в области линейной алгебры, преобразования Фурье и метрис. Структуры данных NumPy выполняются в

- Размер - требуется меньше места.
- Производительность - быстрее, чем списки
- Функциональность - SciPy и NumPy являются основными функциями линейной алгебры [25].

4.2.4. Sklearn

Scikit Learn, также известный как Sklearn, - это простой и эффективный инструмент, используемый для прогнозной аналитики. Это бесплатно доступно и многократно. Это также пакет библиотеки с открытым исходным кодом [26]. Основными функциями Sklearn являются предварительная обработка, выбор модели, регрессия, классификация, кластеризация.

4.2.5. TensorFlow

TensorFlow - это известная платформа, используемая в машинном обучении и глубоком обучении. Слово TensorFlow состоит из двух слов.

- Tensor - многомерный массив

- Flow - поток данных

Это бесплатная библиотека с открытым исходным кодом. Он выпущен 9 ноября 2015 года. Он разработан Google Brain Team. Для увеличения скорости TensorFlow полностью разработан на языке программирования Python. TensorFlow может обучать и запускать программы нейронных сетей. Он поддерживает такие платформы, как Windows, MacOS, Linux и Android [27].

Преимущества TensorFlow

- Графики: по сравнению с другими библиотеками имеет лучшую визуализацию графиков.
- Управление библиотекой: Google разработал TensorFlow. Он имеет преимущества быстрых обновлений и частых выпусков с новыми функциями и библиотеками.
- Отладка, масштабируемость и конвейерная обработка
- Он имеет отличную поддержку сообщества [27].

Недостаток TensorFlow

- Скорость вычислений очень высокая.
- Найти ошибки очень сложно из-за его уникальной структуры.
- Ему не нужна сверхнизкая материя [27].

4.2.6. KERAS

Keras - это библиотека нейронной сети с открытым исходным кодом на Python. Это структура высокого уровня. Он может легко построить модель нейронной сети за ограниченное время. Основными преимуществами Keras являются удобство в

использовании, быстрое развертывание, множественные серверные части, модульность [28].

Он работает как на процессоре, так и на графическом процессоре. Он поддерживает сверточные и рекуррентные нейронные сети. Он устанавливается просто с помощью команды `pip install keras`.

4.2.7. Блокнот Jupyter

Блокнот Jupyter - это интерактивное веб-приложение с открытым исходным кодом. Это позволяет разрабатывать коды и документы на одной странице. Мы можем использовать Jupyter Notebook для чистки, моделирования,

обучение и оценка данных. Отличительной особенностью Notebook является то, что мы можем визуализировать данные [29].

Основные особенности Jupyter Notebook заключаются в том, что мы можем выполнять код в браузере. Мы можем отображать результаты в нескольких форматах, таких как HTML, pdf и т. Д. Язык разметки Markdown помогает форматировать тексты в консоли. Документы записной книжки сохраняются внутри в формате `ipynb`. Мы можем установить Jupyter Notebook, используя команду `pip` в командной строке [30].

4.2.8. Anaconda 3

Anaconda - распространитель программ на R и Python с открытым исходным кодом. Он предоставляет инструменты для анализа данных и методов машинного обучения. Он включает в себя различные библиотеки Python и сотни научных пакетов. Навигатор

- это графический интерфейс Anaconda, в котором пользователь может открыть

различные инструменты и приложения. Еще одним важным фактором анаконды является облако анаконды, в котором мы можем сохранить наш код [33].

4.3. Модели прогнозирования

Прогнозирование идентификатора цены на фондовом рынке выполняется с помощью следующих алгоритмов. Временной ряд - это последовательность чисел, измеренных за определенный период. В зависимости от частоты временные ряды можно разделить на годовые, ежемесячные и ежедневные.

Прогнозирование одномерного временного ряда, в котором используются предыдущие значения временного ряда для оценки будущих значений. Тип прогнозирования многомерного временного ряда, в котором используются разные предикторы.

4.3.1. Модель ARIMA

ARIMA - это алгоритм прогнозирования, сокращенный от Auto Regressive Integrated Moving Average, основанный на том принципе, что значения прошлых временных рядов могут использоваться только для целей прогнозирования будущих значений. ARIMA - это класс моделей, которые описывают такие временные ряды на основе их собственных прошлых значений, то есть их собственных ограничений и задержанных ошибок прогнозирования. Модели ARIMA могут быть основаны на любых несезонных временных рядах, которые демонстрируют тенденции

и не являются белой естественной изменчивостью. В модели ARIMA есть 3 параметра p, d, q [34].

- p обозначает авторегрессивный.
- q обозначает скользящую среднюю.
- d обозначает количество разностей, необходимых для размещения временного ряда.

Когда временной ряд является сезонным, необходимо ввести сезонные модели и SARIMA - Seasonal ARIMA. Авторегрессия в ARIMA означает модель линейной регрессии, которая использует собственные предикторы лага. Линейные модели регрессии хорошо работают, если предикторы не взаимосвязаны или независимы. Различия - самый популярный подход, и в зависимости от характера последовательности может потребоваться различие. Следовательно, значение d - это минимальное количество разностей, необходимых для стабилизации ряда. И если временной ряд по-прежнему, то $d = 0$. « p » - это порядок «Регрессивного автоматического срока» (AR). Указывается количество недостатков Y , которые будут использоваться в качестве предикторов. « q » - это порядок термина «скользящая средняя». Это относится к количеству поздних ошибок прогнозирования, которые будет производить модель ARIMA [34].

Авторегрессивная модель (AR) - это модель, в которой Y_t полагается только на свои собственные недостатки. Y_t - это запаздывающий элемент y_t .

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \dots + \beta_p Y_{t-p} + \epsilon_1$$

Уравнение 1 - Уравнение AR [34].

Модель скользящего среднего (только МА) - это модель, в которой Y_t зависит только от запаздывающей ошибки прогнозирования.

$$Y_t = \alpha + \epsilon_t + \phi_1 \epsilon_{t-1} + \phi_2 \epsilon_{t-2} + \dots + \phi_q \epsilon_{t-q}$$

Уравнение 2 - Уравнение МА [34].

Модель ARIMA - это модель, которая отличает временной ряд, чтобы сделать его стационарным, по крайней мере, один раз, и сочетает в себе термины AR и МА. Итак, уравнение

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \dots + \beta_p Y_{t-p} + \epsilon_t + \phi_1 \epsilon_{t-1} + \phi_2 \epsilon_{t-2} + \dots + \phi_q \epsilon_{t-q}$$

Уравнение 3 - Уравнение ARIMA [34].

Теперь, уравнение ARIMA прописью

Прогнозируемое значение Y_t = константа + линейное запаздывание по оси Y (до p -запаздывания) + линейное запаздывание комбинации прогнозов (до q запаздываний) [34].

4.3.2. Модель FB Prophet

Prophet - это библиотека с открытым исходным кодом Facebook, ориентированная на разложимые модели (тренд + сезонность + праздники). Это дает нам возможность предсказывать временные ряды с точностью и поддерживает настраиваемую сезонность и праздники. Пакет Prophet предлагает легко настраиваемые, интуитивно понятные параметры. Те, у кого нет опыта в моделях прогнозирования, могут использовать это, чтобы делать точные

прогнозы по широкому кругу бизнес-проблем. Три основных компонента модели пророка - это тренд, сезонность и праздники [35].

$$y(t) = g(t) + s(t) + h(t) + \epsilon_t$$

Уравнение 4: компоненты пророка [35].

- $g(t)$: частично линейная кривая для моделирования неперiodического перехода временных рядов.
- $s(t)$: ежедневные изменения (сезонность еженедельно / ежегодно).
- $h(t)$: эффекты праздников нерегулярных сроков.
- ϵ_t : термин «ошибка» отвечает за любые неожиданные изменения, не соответствующие шаблону.

Тенденция:

Тренд формируется линейной отрезковой кривой, соответствующей шаблону или неперiodическим отрезкам. Метод линейной подгонки гарантирует, что выбросы / недостающие данные.

Сезонность:

Пророк адаптирует и предсказывает влияние сезонности на масштабируемую модель, используя четыре последовательности.

Следующий метод аппроксимирует сезонные эффекты $s(t)$:

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

Уравнение 5: Эффект сезонности [35].

Праздники и события:

Праздники и события способствуют предсказуемым потрясениям временного ряда. Prophet позволяет аналитикам индивидуально определять прошлые и будущие события. В определенные дни есть окно, и параметры используются для моделирования влияния праздников и событий [35].

4.3.3. Keras с LSTM

LSTM - это своего рода повторяющаяся нейронная сеть.

Рекуррентная нейронная сеть - это нейронная сеть, которая стремится сформировать поведение на основе времени или последовательности, например, язык, цены на акции,

потребность в энергии и т. д. Выход нейронной сети рассчитывается по входу того же сетевого уровня в момент времени $t + 1$ [36].

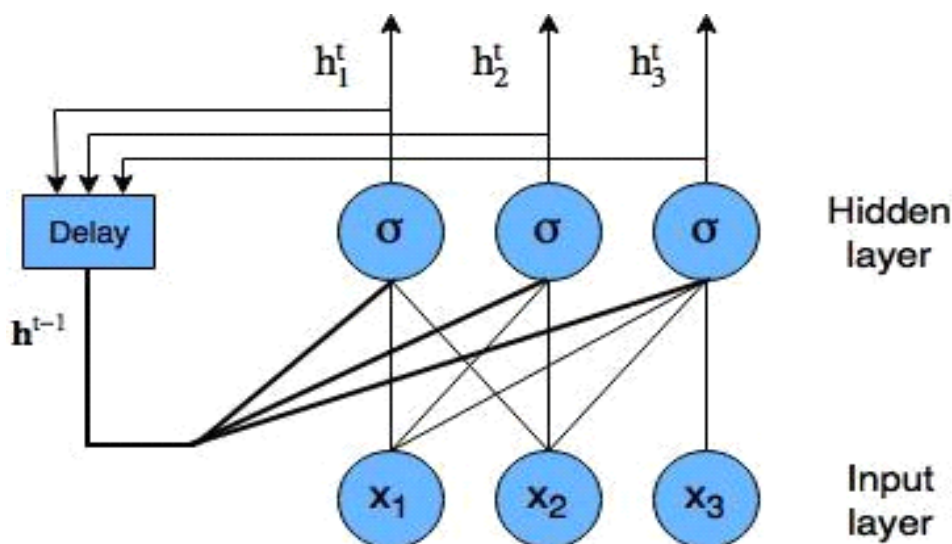


Рисунок 4: Рекуррентная нейронная сеть с узлами [36].

Во время обучения модели и предсказания рекуррентная нейронная сеть стала развернутой.

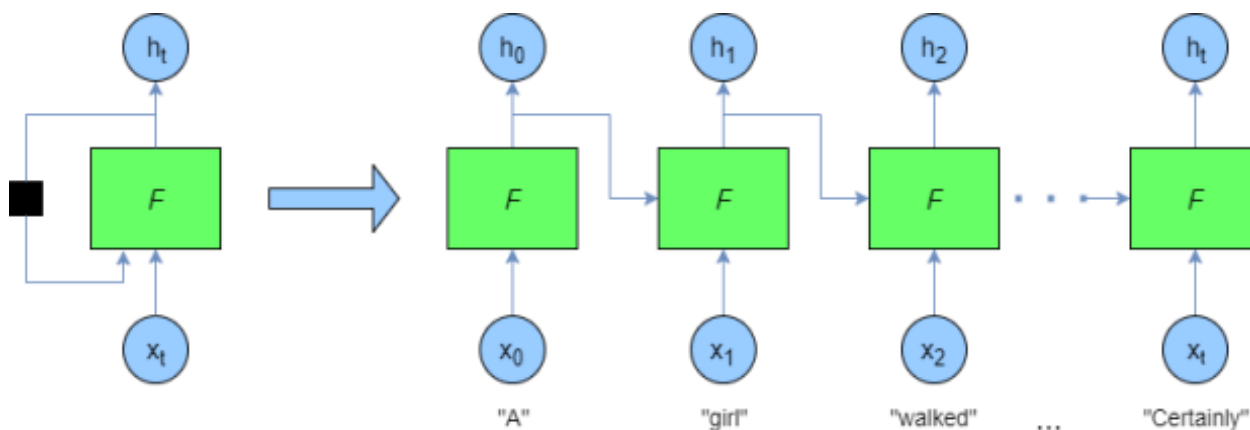


Рисунок 5: Развернутая рекуррентная нейронная сеть [36].

LSTM - это рекуррентная нейронная сеть, в которой блоки ячеек LSTM заменяют обычные слои нейронной сети. Различные компоненты этих ячеек называются входным вентилем, вентилем забывания и выходным вентилем [36].

- Уровень ввода: активность устройств данных относится к необработанным данным, которые могут поступать в систему.
- Скрытый слой: определите усилия каждого секретного подразделения. Упражнения введенных систем и нагрузки на связи между данными и компонентами. Также может присутствовать по крайней мере один скрытый слой.
- Уровень вывода: Прямой вывод зависит от действий навесных блоков и нагрузки между секретным и выходным устройствами.

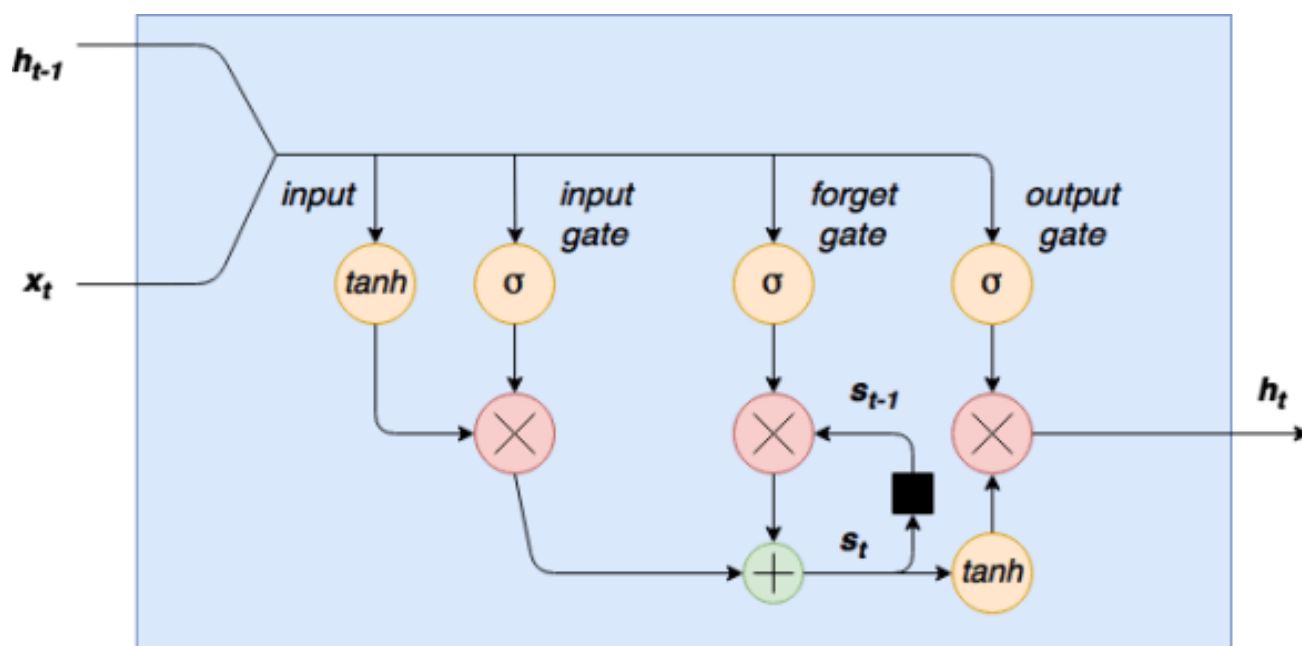


Рисунок 6: Ячеечная диаграмма LSTM [36].

LSTM специально разработаны для предотвращения проблемы долгосрочной зависимости. Долгосрочное сохранение знаний - это, по сути, их автоматическое поведение, а не то, что они пытаются понять. Все повторяющиеся нейронные сети имеют форму цепочки повторяющихся модулей нейронной сети. Этот повторяющийся модуль имеет структуру в стандартных RNN, как один слой \tanh [37].

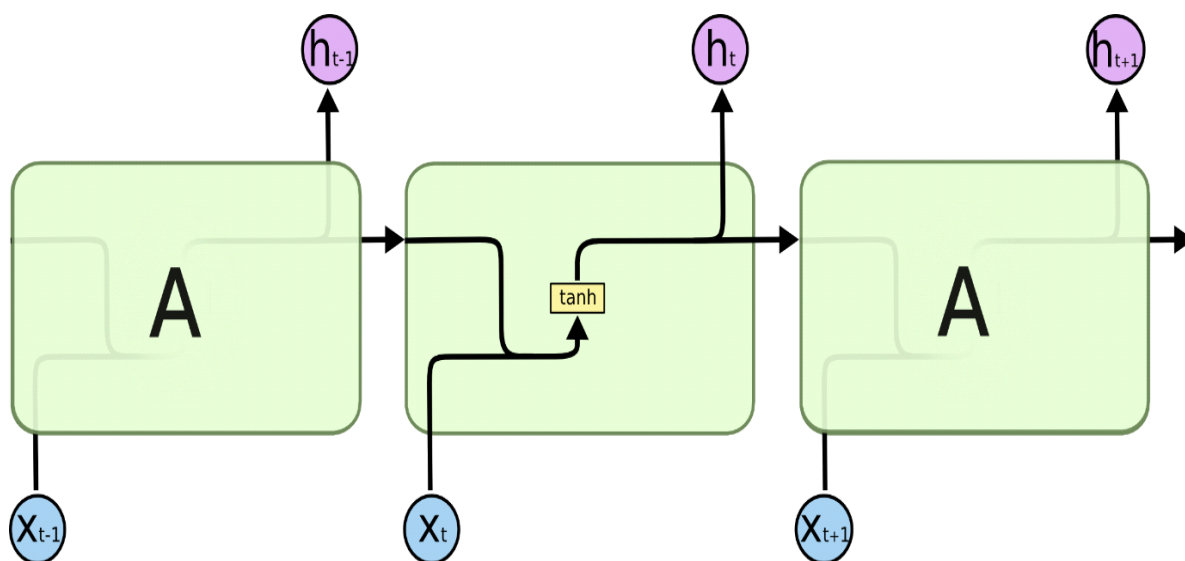


Рисунок 7: RNN содержит однослойный [37].

LSTM имеют структуру, подобную этой цепочке, но повторяющийся модуль имеет другую структуру. Их четыре, очень специфически взаимодействующих, вместо того, чтобы создавать один слой нейронной сети.

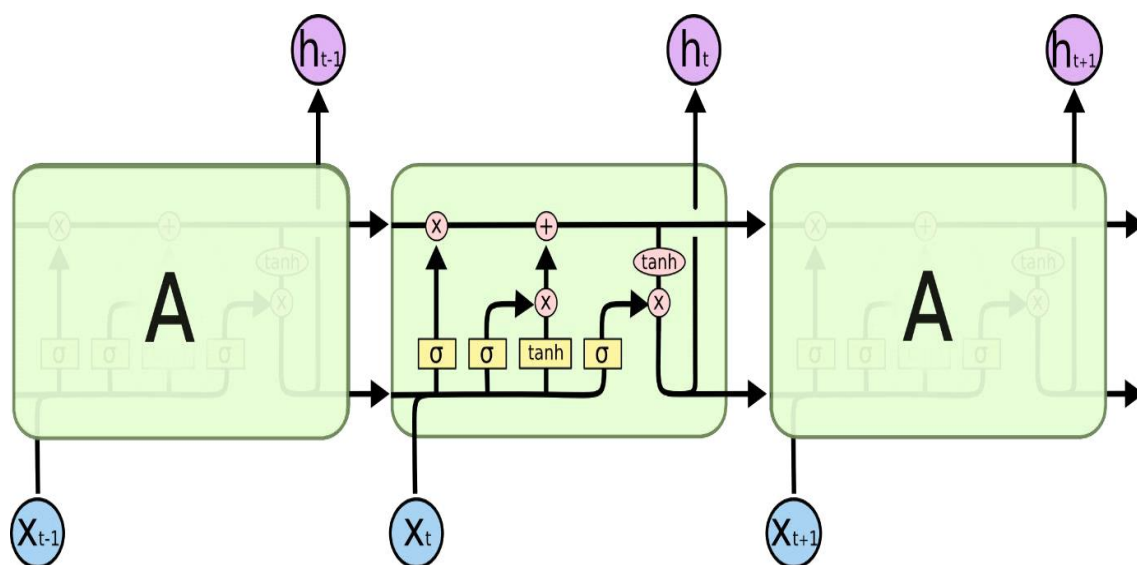


Рисунок 8: LSTM с интерактивными слоями [37].

Из-за своих зависимостей данных на большом и малом расстоянии LSTM могут помочь справиться с исчезающими и взрывающимися градиентами. LSTM во многих случаях работают немного лучше, чем ГРУ, но из-за большего размера скрытого состояния их обучение и эксплуатация обходятся дороже. LSTM являются прототипом нетривиальной модели скрытых авторегрессионных переменных с контролируемым состоянием [38].

Глава 5. АРХИТЕКТУРА ПРОЕКТА

5.1. Основная структура

В этом разделе мы обсуждаем различные модули, используемые в программе. Сначала из Yahoo Finance собираются исторические данные за 10 лет. Yahoo finance - это финансовый веб-сайт, который предоставляет данные о фондовых рынках, новости, обновления в реальном времени и т. Д. Эти данные обрабатываются и нормализуются для проведения анализа. Модели анализа временных рядов используются для определения производительности обработанных данных с помощью Jupyter Notebook. Jupyter Notebook - это IDE, которая поддерживает большинство функций библиотеки Python. Версия Python 3.8.2 используется не только для обработки данных, но также для прогнозирования данных фондового рынка.

5.1.1. Фаза анализа

На рисунке ниже показана архитектура фазы анализа. Фаза анализа подразделяется на подготовительную фазу и экспериментальную фазу.

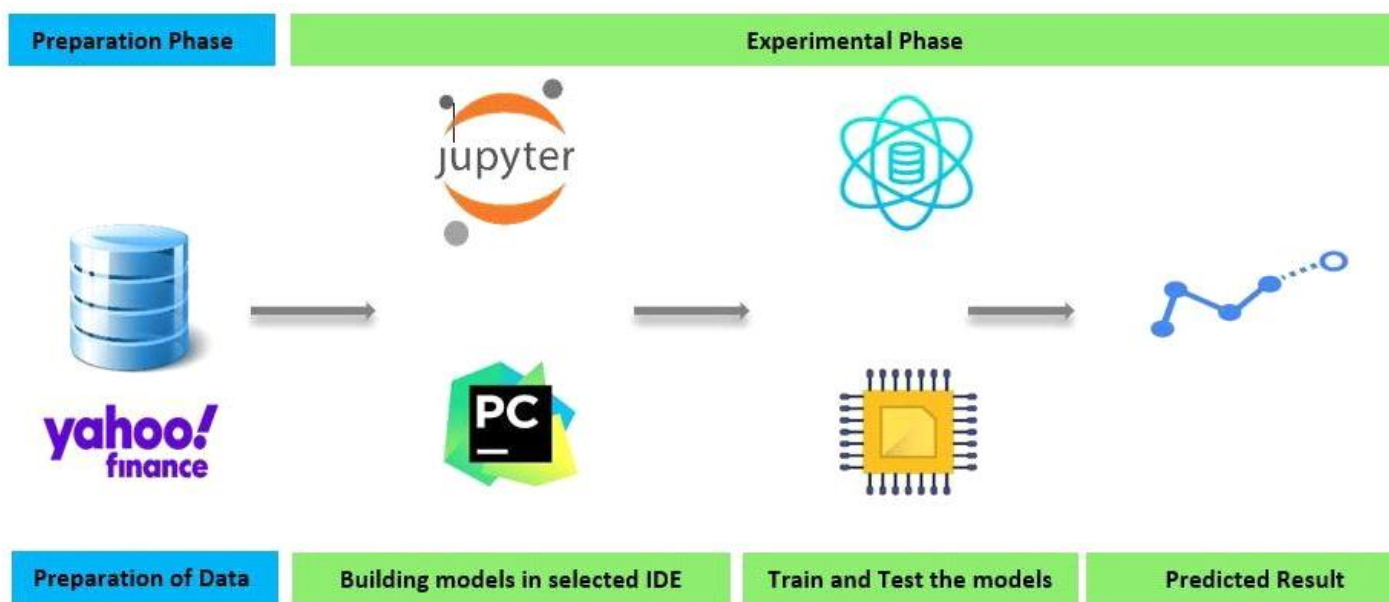


Рисунок 9: Архитектура фазы анализа.

Подготовительный этап:

Подготовка данных - начальный этап проекта. Необработанные данные собираются из Yahoo Finance с использованием библиотеки Python fix Yahoo Finance. Собранные данные состоят из семи атрибутов. Атрибут даты относится к дате. Атрибут открытия указывает начальную стоимость акции. Атрибут Close относится к закрывающей стоимости акции. Высокий атрибут указывает на самую высокую стоимость акций в

день. Атрибут "Низкий" указывает на самую низкую стоимость акций за день. Атрибут Adj Close относится к цене закрытия после корректировок. Объем - это количество транзакций, совершенных за день. Для предварительной обработки данных, таких как очистка, форматирование и выбор данных, подходящих для алгоритмов. В этом проекте мы собрали данные за десять лет из разных банков.

Фаза эксперимента:

На этом этапе Jupyter Notebook используется для обучения и тестирования моделей временных рядов. Различные модели временных рядов разрабатываются с использованием ARIMA, PROPHET и KERAS с LSTM. Эти модели используются для прогнозирования стоимости акций.

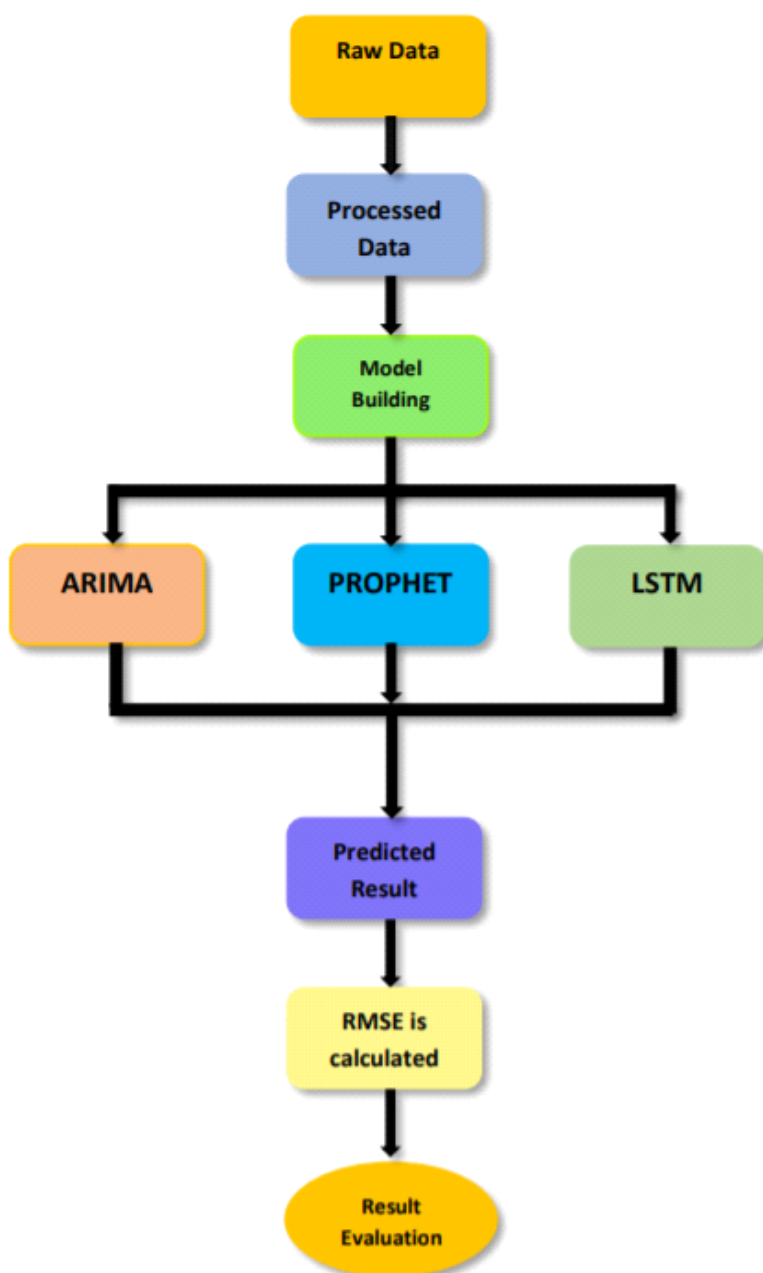


Рисунок 10: Пошаговая процедура фазы анализа

Глава 6. Реализация

В этой главе мы собираемся обсудить пошаговую процедуру фазы анализа.

6.1. Сбор информации

Исторические данные банков собираются из Yahoo Finance с использованием библиотеки Yfinance и сохраняются в каталоге в формате csv.

```
#importing of Libraries
import pandas
from pandas_datareader import data as pdr
import yfinance as yf |

import datetime
yf.pdr_override()

import warnings
warnings.filterwarnings('ignore')

stocks = ["AMZN"]
start = datetime.datetime(2011,4,1)
end = datetime.datetime(2021,4,1)

data = pdr.get_data_yahoo(stocks, start=start, end=end)

#file saved in the directory
data.to_csv('C://Users//HP//Desktop//Stock market prediction//AMAZON.csv')

data.head(5)
```

Рисунок 12: Сбор и сохранение данных из Yahoo Finance.

После выполнения вышеуказанного кода результат показан на рисунке 13.

[*****100%*****] 1 of 1 completed						
	Open	High	Low	Close	Adj Close	Volume
Date						
2011-03-31	179.309998	181.570007	178.500000	180.130005	180.130005	4826500
2011-04-01	181.580002	183.250000	178.589996	180.130005	180.130005	5684100
2011-04-04	180.889999	183.610001	180.690002	182.940002	182.940002	4186400
2011-04-05	182.100006	186.360001	181.800003	185.289993	185.289993	5569200
2011-04-06	186.149994	188.270004	181.119995	182.759995	182.759995	5430700

Рисунок 13: Собранные данные.

6.2. Внедрение модели ARIMA

Для реализации модели ARIMA сначала мы представляем пакеты и библиотеки python.

```
import pandas as pd
import numpy as np
import itertools

import warnings
warnings.filterwarnings("ignore")

import statsmodels.api as sm
import matplotlib
import matplotlib.pyplot as plt
plt.style.use('fivethirtyeight')

matplotlib.rcParams['axes.labelsize'] = 14
matplotlib.rcParams['xtick.labelsize'] = 12
matplotlib.rcParams['ytick.labelsize'] = 12
matplotlib.rcParams['text.color'] = 'k'
```

Рисунок 14: Импорт пакетов и библиотек.

На рисунке 14 выше представлен начальный этап импорта пакетов. Мы импортируем Pandas, NumPy и Itertools. Python Itertools - это модуль, который предлагает различные функции для создания сложных итераторов. Предупреждения импортируются, чтобы избежать предупреждений в будущем во время выполнения. Библиотека Matplotlib используется для построения переменных в графическом формате. Stats models - это модуль Python, содержащий классы и функции. Эти классы и функции используются для оценки широкого диапазона статистических моделей и тестирования.

```
AMAZON = pd.read_csv('C:/Users/HP/Desktop/Stock market prediction/AMAZON.csv')
AMAZON.head(5)
```

	Date	Open	High	Low	Close	Adj Close	Volume
0	2011-04-01	181.580002	183.250000	178.589996	180.130005	180.130005	5684100
1	2011-04-04	180.889999	183.610001	180.690002	182.940002	182.940002	4186400
2	2011-04-05	182.100006	186.360001	181.800003	185.289993	185.289993	5569200
3	2011-04-06	186.149994	188.270004	181.119995	182.759995	182.759995	5430700
4	2011-04-07	182.779999	185.169998	181.759995	184.910004	184.910004	4564000

Рисунок 15: Считывание данных о ценах на акции AMAZON из справочника и их вывод.

На рисунке 15 показано чтение набора данных в формате csv из каталога. Этот набор данных хранится в переменной AMAZON. После этого мы печатаем первые 5 строк фрейма данных с помощью head (). Мы также можем использовать tail для печати последних 5 строк фрейма данных.

```
AMAZON.Date = pd.to_datetime(AMAZON.Date, format='%Y%m%d', errors='ignore')
cols = ['High', 'Low', 'Open', 'Volume', 'Adj Close']
AMAZON.drop(cols, axis=1, inplace=True)
AMAZON = AMAZON.sort_values('Date')
AMAZON.isnull().sum()
```

```
Date      0
Close      0
dtype: int64
```

```
AMAZON['Date'].min()
```

```
'2011-04-01'
```

```
AMAZON['Date'].max()
```

```
'2021-03-31'
```

Рисунок 16: Пример предварительной обработки данных Amazon

На рисунке 16 показана проверка пропущенных значений, удаление столбцов и сортировка данных в формате даты. После этого мы проверяем минимальное и максимальное значение даты.


```
#creation data column as index
AMAZON = AMAZON.set_index('Date')
AMAZON.index
```

```
Index(['2011-04-01', '2011-04-04', '2011-04-05', '2011-04-06', '2011-04-07',
      '2011-04-08', '2011-04-11', '2011-04-12', '2011-04-13', '2011-04-14',
      ...,
      '2021-03-18', '2021-03-19', '2021-03-22', '2021-03-23', '2021-03-24',
      '2021-03-25', '2021-03-26', '2021-03-29', '2021-03-30', '2021-03-31'],
      dtype='object', name='Date', length=2516)
```

Рисунок 17: Индексирование с использованием данных временных рядов AMAZON.

```
#creating monthly mean from 2011
monthly_mean['2011']
```

```
Date
2011-04-30    184.576500
2011-05-31    198.166667
2011-06-30    191.667272
2011-07-31    215.195001
2011-08-31    199.458694
2011-09-30    223.242857
2011-10-31    226.838095
2011-11-30    205.480951
2011-12-31    183.933811
Freq: M, Name: Close, dtype: float64
```

Рисунок 18: Создание среднемесячного значения AMAZON.

Рисунки 17 и 18 показывают индексацию даты с данными временного ряда. С текущими данными даты и времени очень сложно работать. Итак, мы использовали среднее значение цены закрытия месяца и начало каждого месяца в качестве отметки времени.



Рисунок 19: Визуализация данных временных рядов Close.

```
# Visualising the Distinct components: trend, seasonality, and noise.
from pylab import rcParams
rcParams['figure.figsize'] = 18, 8
decomposition = sm.tsa.seasonal_decompose(monthly_mean, model='additive')
fig = decomposition.plot()
plt.savefig('amznstock_component.pdf')
plt.show()
```

Рисунок 20: Код для реализации метода декомпозиции временных рядов.

Фигура 20 описывает код для реализации метода декомпозиции временных рядов, который используется для разложения наших данных на три компонента: тренд, сезонность и шум. На рисунке 21 ниже показан график цен на наши акции AMAZON. График наглядно демонстрирует нестабильный характер цены закрытия.

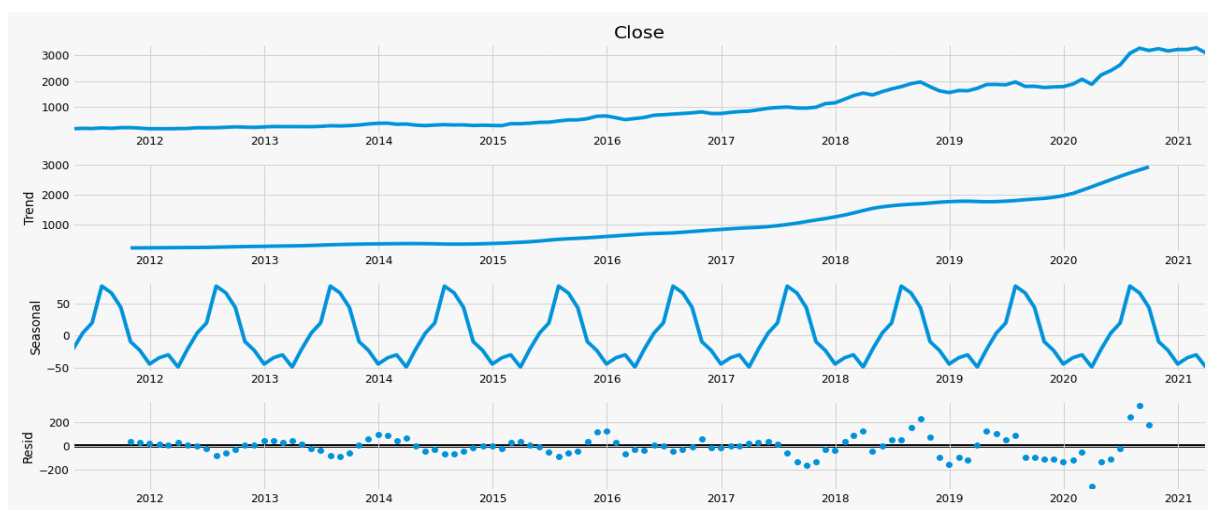


Рисунок 21: График разложения временных рядов.

Теперь мы собираемся спрогнозировать цену акций на момент закрытия, построив модель ARIMA. Вначале мы выбираем параметры p , d , q . Эти три параметра отражают сезонность, тенденцию и шум.

```
# Define the p, d and q parameters to take any value between 0 and 2
p = d = q = range(0, 2)
# Generate all different combinations of p, q and q triplets
pdq = list(itertools.product(p, d, q))
# Generate all different combinations of seasonal p, q and q triplets
seasonal_pdq = [(x[0], x[1], x[2], 12) for x in list(itertools.product(p, d, q))]
```

Рисунок 22: Код для генерации всех сезонных комбинаций ARIMA.

Здесь p - авторегрессионная часть модели. Это позволяет нам включить в нашу модель влияние прошлых ценностей. d - составная часть модели. Это означает, что количество прошлых временных точек вычитается из текущей стоимости. q - это скользящая средняя часть модели, которая позволяет нам определить модель ошибки. В сезонном ARIMA, определяемом как ARIMA (p, d, q) (P, D, Q) s , по отношению к сезонным эффектам. Буква s обозначает период временных рядов: 4 квартальных и 12 годовых. Теперь нам нужно найти значение ARIMA (p, d, q) (P, D, Q) s . Для этого мы использовали поиск по сетке, чтобы многократно исследовать все различные комбинации параметров. Для этого используется функция SARIMAX ().

```

import warnings
warnings.filterwarnings("ignore")

l_param = []
l_param_seasonal=[]
l_results_aic=[]
for param in pdq:
    for param_seasonal in seasonal_pdq:
        try:
            mod = sm.tsa.statespace.SARIMAX(monthly_mean,
                                             order=param,
                                             seasonal_order=param_seasonal,
                                             enforce_stationarity=False,
                                             enforce_invertibility=False)

            results = mod.fit()

            print('ARIMA{}x{}12 - AIC:{}'.format(param, param_seasonal, results.aic))

            l_param.append(param)
            l_param_seasonal.append(param_seasonal)
            l_results_aic.append(results.aic)
        except:
            continue

minimum=l_results_aic[0]
for i in l_results_aic[1:]:
    if i < minimum:
        minimum = i
i=l_results_aic.index(minimum)

mod = sm.tsa.statespace.SARIMAX(monthly_mean,
                                order=l_param[i],
                                seasonal_order=l_param_seasonal[i],
                                enforce_stationarity=False,
                                enforce_invertibility=False)

results = mod.fit()
print("\n\n")
print(results.summary().tables[0])

print(results.summary().tables[1])

print(results.summary().tables[2])

```

Рисунок 23: Сезонная модель ARIMA.

На рисунке 23 показана реализация функции SARIMAX для соответствия сезонной модели ARIMA.

Результаты SARIMAX предполагают, что SARIMAX (1,1,1) × (0,1,1,12) с наименьшим значением AIC

1250.205. Столбец coef описывает вес и влияние на временной ряд в каждой функции. Столбец P> | z | показывает значение каждого характеристического веса.

```

=====
SARIMAX Results
=====
Dep. Variable:          Close      No. Observations:      132
Model:                SARIMAX(1, 1, 1)x(0, 1, 1, 12)  Log Likelihood      -621.103
Date:                  Mon, 12 Apr 2021      AIC      1250.205
Time:                  09:02:19      BIC      1260.821
Sample:                04-30-2010      HQIC     1254.507
                    - 03-31-2021
Covariance Type:                opg
=====
=====
              coef      std err          z      P>|z|      [0.025      0.975]
-----
ar.L1          0.6428        0.213        3.012      0.003        0.224        1.061
ma.L1         -0.4625        0.247       -1.872      0.061       -0.947        0.022
ma.S.L12      -0.6977        0.103       -6.747      0.000       -0.900       -0.495
sigma2       7810.4782     676.913     11.538      0.000     6483.753     9137.203
=====
=====
Ljung-Box (L1) (Q):                0.29      Jarque-Bera (JB):                65.64
Prob(Q):                          0.59      Prob(JB):                  0.00
Heteroskedasticity (H):            31.91      Skew:                      0.11
Prob(H) (two-sided):              0.00      Kurtosis:                  6.87
=====

```

Рисунок 24: Результат SARIMAX для цен на акции AMAZON

```

#Plotting the diagnostics results
results.plot_diagnostics(figsize=(20, 10))
plt.savefig('applstock_diag.pdf')
plt.show()

```

Рисунок 25: Код для вывода результатов диагностики на график.

Теперь мы собираемся построить результаты диагностики с помощью диагностики графика. На этом рисунке показаны стандартизованный остаток, гистограмма плюс расчетная плотность, нормальный QQ и коррелограмма. На рисунке 26 гистограмма плюс оценочная плотность показывает, что красная линия KDE точно следует за $N(0,1)$, где in обозначает стандартное

обозначение со стандартным отклонением 1 и средним 0. На графике QQ показано, что синие точки следуют тенденции изменения стандартное нормальное распределение. Стандартный остаток не показывает явной сезонности, что подтверждается графиком коррелограммы. На графике коррелограммы остатки временных рядов имеют низкую корреляцию.

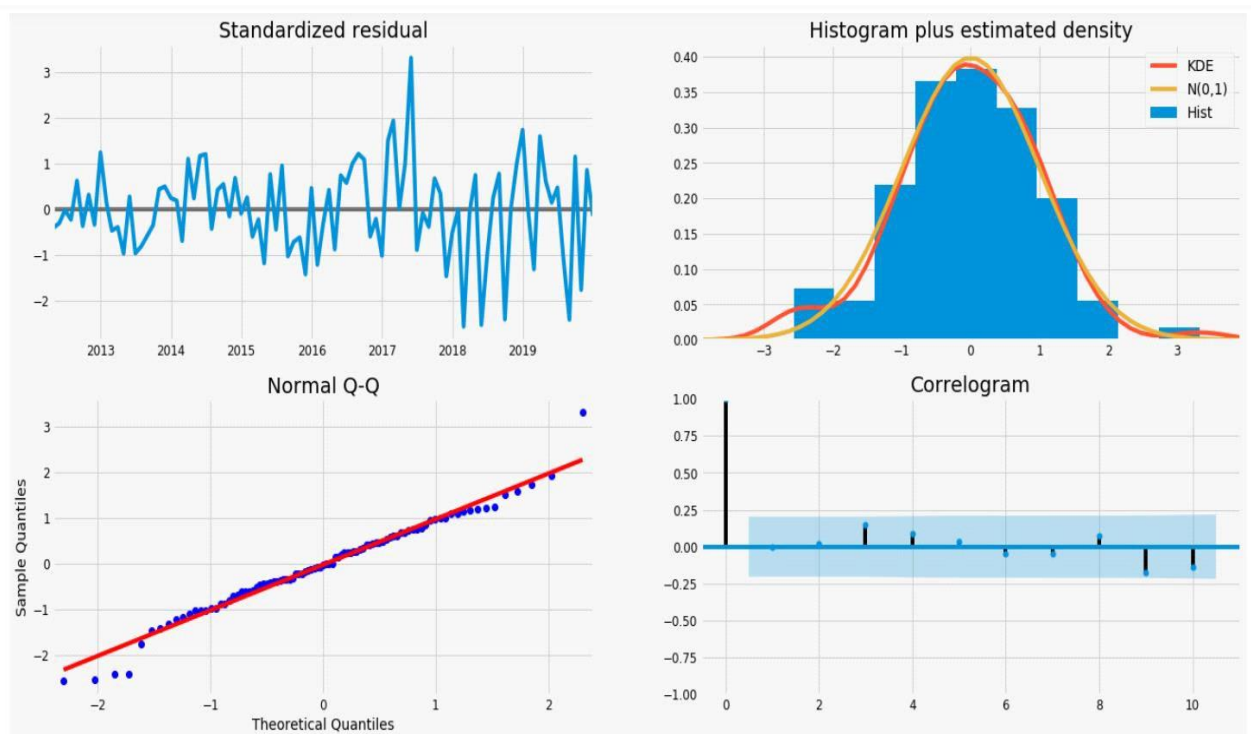


Рисунок 26: Диаграмма результатов диагностики.

Приведенный выше рисунок помогает нам сделать вывод о том, что наша модель достаточно хороша для прогнозирования будущей стоимости акций.

```

pred = results.get_prediction(start=pd.to_datetime('2011-12-31'), dynamic=False)
pred_uc = results.get_forecast(steps=24)
pred_ci = pred.conf_int()

ax = monthly_mean['2012:'].plot(label='observed')
pred.predicted_mean.plot(ax=ax, label='Predicted', alpha=.7, figsize=(12, 7))
pred_uc.predicted_mean.plot(ax=ax, label='Forecast')

ax.fill_between(pred_ci.index,
                pred_ci.iloc[:, 0],
                pred_ci.iloc[:, 1], color='k', alpha=.2)

plt.title('AMAZON STOCK PRICES')
ax.set_xlabel('Date')
ax.set_ylabel('close price')
plt.legend()
plt.savefig('amznstock_pred.pdf')

plt.show()

```

Рисунок 27: Код для построения графика наблюдаемых, прогнозируемых и прогнозируемых результатов.

Для проверки прогноза мы начинаем со сравнения значений прогноза со значением реального временного ряда. Это помогает нам понять точность прогнозов. Атрибут `get_prediction()` и `coef_int()`, который помогает получить значения и доверительный интервал для прогнозирования временных рядов. Динамичный `= False` аргумент, который означает, что прогноз создается в каждой точке с использованием всей истории.

На рисунке 27 поясняется код для предсказания и прогнозирования цены закрытия на графике. Мы взяли дату начала 31-12-2011 для прогноза и 24 месяца для прогнозирования. Теперь сначала мы наносим на график наблюдаемое значение. Затем мы собираемся построить прогнозируемое значение и прогнозируемое значение. После всего этого мы определили заголовок, y-метку и x-метку графа.

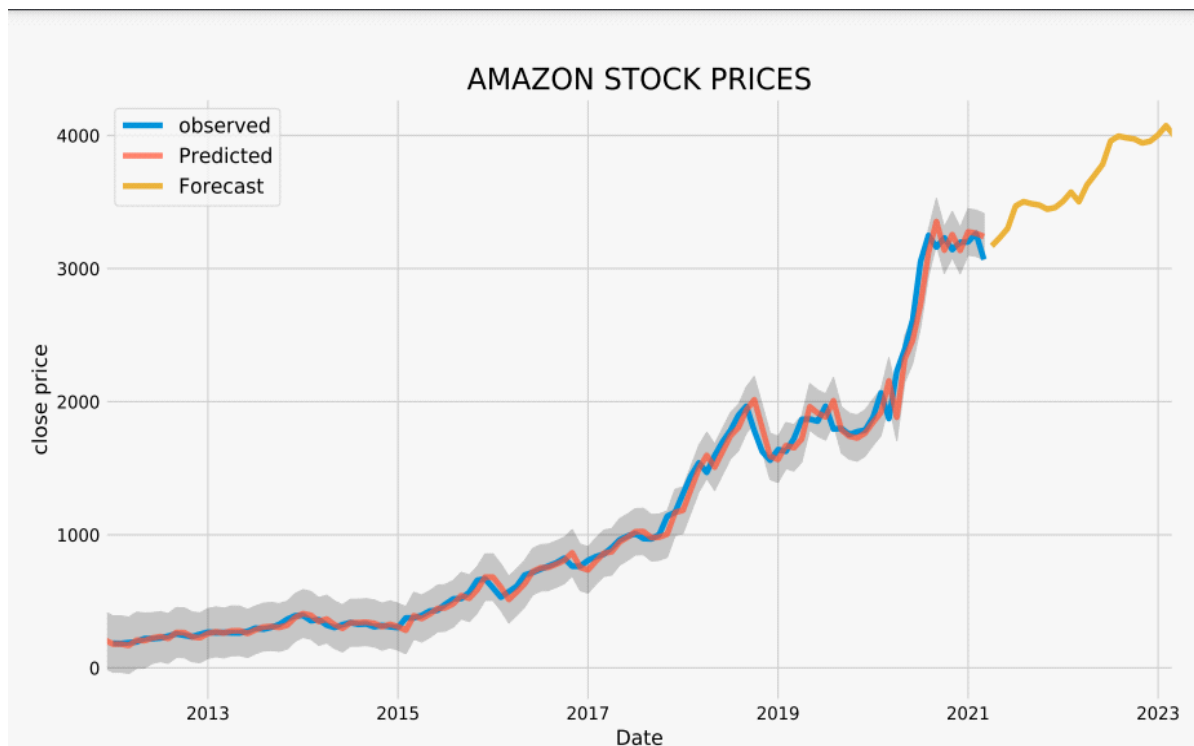


Рисунок 28: Визуализация прогнозируемого и прогнозируемого графика цены закрытия.

6.3. Реализация модели Пророка

Для реализации модели Prophet сначала мы представляем пакеты и библиотеки python.

```
#imports
import pandas as pd
import numpy as np
from fbprophet import Prophet
import matplotlib.pyplot as plt
from functools import reduce

%matplotlib inline
import warnings
warnings.filterwarnings('ignore')

import logging, sys
logging.disable(sys.maxsize)

pd.options.display.float_format = "{:,.2f}".format
```

Рисунок 29: Импорт необходимых библиотек.

На рисунке 29 показан импорт необходимых библиотек для построения, прогнозирования и прогнозирования модели Prophet. Мы импортируем пророка из FBProphet. Предупреждения импортируются, чтобы избежать предупреждений в будущем во время выполнения. Библиотека Matplotlib используется для построения переменных в графическом формате.

```
AMAZON = pd.read_csv('C:/Users/HP/Desktop/Stock market prediction/AMAZON.csv')
```

```
AMAZON.describe()
```

	Open	High	Low	Close	Adj Close	Volume
count	2,516.00	2,516.00	2,516.00	2,516.00	2,516.00	2,516.00
mean	1,004.71	1,015.30	992.65	1,004.35	1,004.35	4,252,377.31
std	883.72	893.85	871.76	882.81	882.81	2,347,787.13
min	169.62	174.55	166.97	173.10	173.10	881,300.00
25%	299.96	303.59	296.27	299.70	299.70	2,752,825.00
50%	665.58	674.39	659.29	664.65	664.65	3,649,200.00
75%	1,680.00	1,699.93	1,661.03	1,675.07	1,675.07	4,962,775.00
max	3,547.00	3,552.25	3,486.69	3,531.45	3,531.45	24,134,200.00

Рисунок 30: Импорт и описание данных.

На рисунке 30 показано чтение данных из каталога и их сохранение в переменной. Функция `Describe` используется для описания количества, среднего, стандартного, минимального, максимального и т. Д. Каждого атрибута в наборе данных. Анализ даты означает создание столбца даты как индекса.

```
AMAZON = AMAZON[['Date', 'Close']]  
  
AMAZON.columns = ['ds', 'y']  
AMAZON.head(10)
```

	ds	y
0	2011-04-01	180.13
1	2011-04-04	182.94
2	2011-04-05	185.29
3	2011-04-06	182.76
4	2011-04-07	184.91
5	2011-04-08	184.71
6	2011-04-11	184.04
7	2011-04-12	180.48
8	2011-04-13	182.29
9	2011-04-14	181.82

Рисунок 31: Подготовка данных.

На рисунке 31 показана подготовка данных. Для работы `prophet` мы изменили имена атрибутов `Date` и `Close` на `ds` и `y`. `y` - это атрибут, который мы используем для предсказания будущего.

```
#prophet  
AMAZON.set_index('ds').y.plot(figsize=(15,8), title = 'AMAZON STOCK PRICES')
```

Рисунок 32: Код для визуализации данных (пример данных Amazon).

Рисунок 32 описывает код для построения графика. Мы устанавливаем атрибут `ds`, который содержит дату в качестве индекса и определяем размер графика. Библиотека `Matplotlib`

используется для построения графика. Вывод кода показан на рисунке 33.



Рисунок 33: Визуализация данных.

После визуализации данных мы вызвали `prophet()` для запуска модели `prophet` и присвоили ей переменную под названием `model`.

Используя метод подбора, мы подбираем цену нашей акции.

```
model = Prophet()  
model.fit(AMAZON)
```

```
<fbprophet.forecaster.Prophet at 0x25186f0c9c8>
```

Рисунок 34: Код для реализации модели.

```

future = model.make_future_dataframe(365, freq='d')

future_boolean = future['ds'].map(lambda x : True if x.weekday() in range(0, 5) else False)
future = future[future_boolean]

future.tail()

```

	ds
3127	2022-03-25
3130	2022-03-28
3131	2022-03-29
3132	2022-03-30
3133	2022-03-31

Рисунок 35: Предсказание и прогноз будущей цены акций.

Фигура 36 показывает предсказание и прогнозирование будущих переменных. Для создания будущих дат мы использовали вспомогательную функцию в prophet под названием `make_future_dataframe` с периодом прогнозирования 365 дней. Акции торгуются только в будние дни. Итак, нам нужно удалить выходные, потому что мы использовали логическое выражение. Логическое выражение, указывающее, что понедельник равен 0, а суббота - 5, и если день не равен 0-4, то возвращается как false.

```

In [10]: forecast = model.predict(future)

In [11]: forecast.tail()

```

Out[11]:

	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	additive_terms	additive_terms_lower	additive_terms_upper	weekly	weekly_lower	weekly_upper
2474	2020-01-20	83.30	75.39	90.57	83.30	83.30	-0.41	-0.41	-0.41	0.41	0.41	0.41
2475	2020-01-21	83.28	74.86	90.07	83.28	83.28	-0.48	-0.48	-0.48	0.50	0.50	0.50
2476	2020-01-22	83.26	75.33	89.70	83.26	83.26	-0.79	-0.79	-0.79	0.37	0.37	0.37
2477	2020-01-23	83.24	75.17	90.16	83.24	83.24	-0.81	-0.81	-0.81	0.52	0.52	0.52
2478	2020-01-24	83.22	74.09	89.40	83.22	83.22	-1.10	-1.10	-1.10	0.41	0.41	0.41

Рисунок 36: Прогноз цены акций.

На рисунке 37 показано прогнозируемое значение цены акций. Для прогнозирования значения цены мы создали прогноз и вызываем прогноз из модели и передаем прогноз в будущий фрейм данных. После выполнения программы мы получаем результат с множеством атрибутов. `Yhat` - это атрибут, обозначающий прогнозируемое значение.

```
In [13]: plt.figure(figsize=(16,6))  
         model.plot(forecast);
```

<Figure size 1152x432 with 0 Axes>

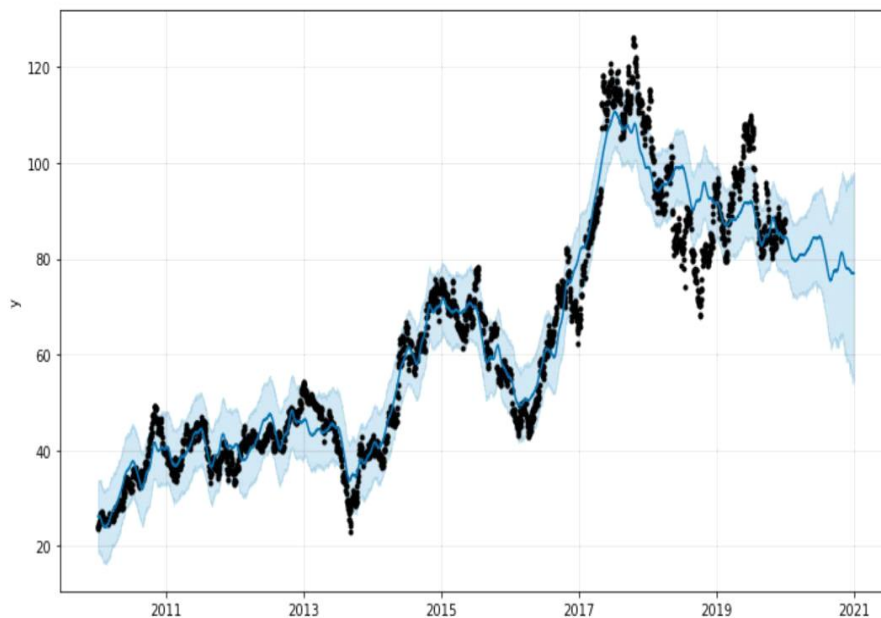


Рисунок 37: Визуализация прогнозируемой цены акций с использованием функции построения графика.

После визуализации прогнозируемого графика. Мы визуализировали компоненты графика, используя компоненты графика.

```
model.plot_components(forecast);
```

Рисунок 38: Код для отображения компонентов.

Приведенный выше код используется для визуализации компонентов прогнозируемого графика.

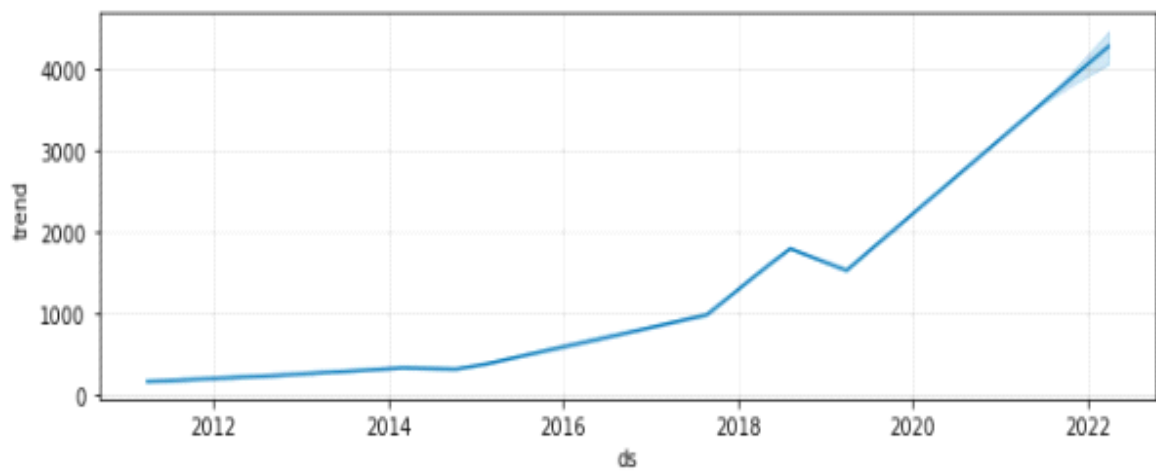


Рисунок 39: Трендовый компонент графика.

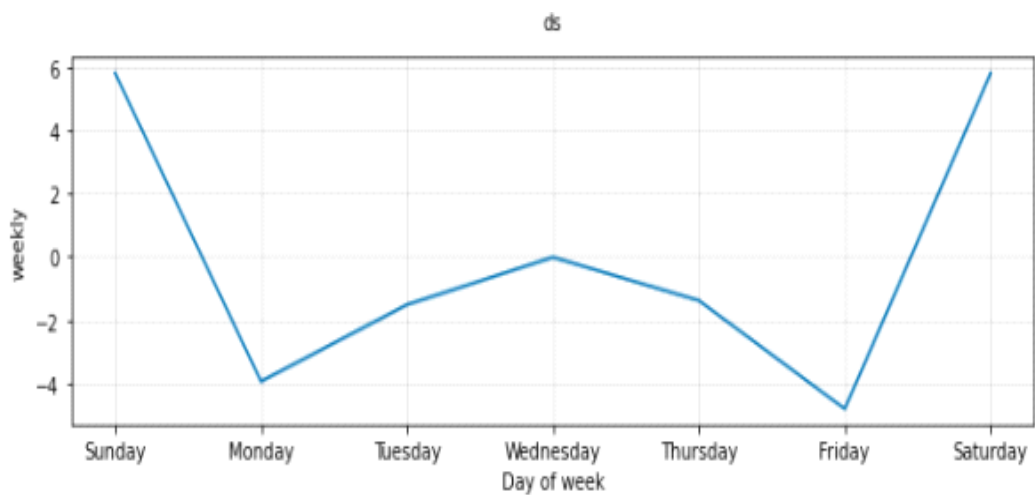


Рисунок 40: Недельный компонент графика.

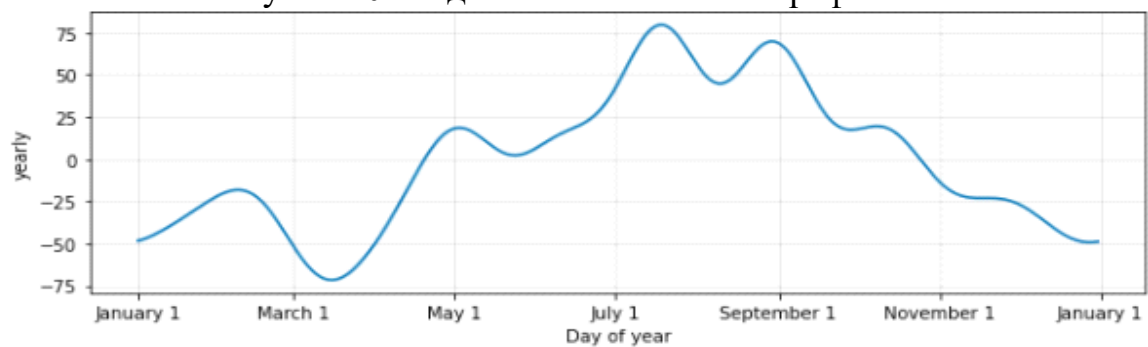


Рисунок 41: Годовой компонент графика.

Теперь мы собираемся визуализировать прогнозируемый график.

На рисунке 42 показан код для визуализации прогнозируемого графика. На графике мы визуализируем `ds`, `yhat`, `yhat_lower` и `yhat_upper`.

```
AMAZON_forecast = forecast[['ds', 'yhat', 'yhat_lower', 'yhat_upper']]
df = pd.merge(AMAZON, AMAZON_forecast, on='ds', how='right')
df.set_index('ds').plot(figsize=(16,8), title='AMAZON STOCK PRICES', color=['#F29F0E', '#000000', '#0EB6F2', '#0EB6F2'], gri
```

Рисунок 42: Код для визуализации прогнозируемого графика.

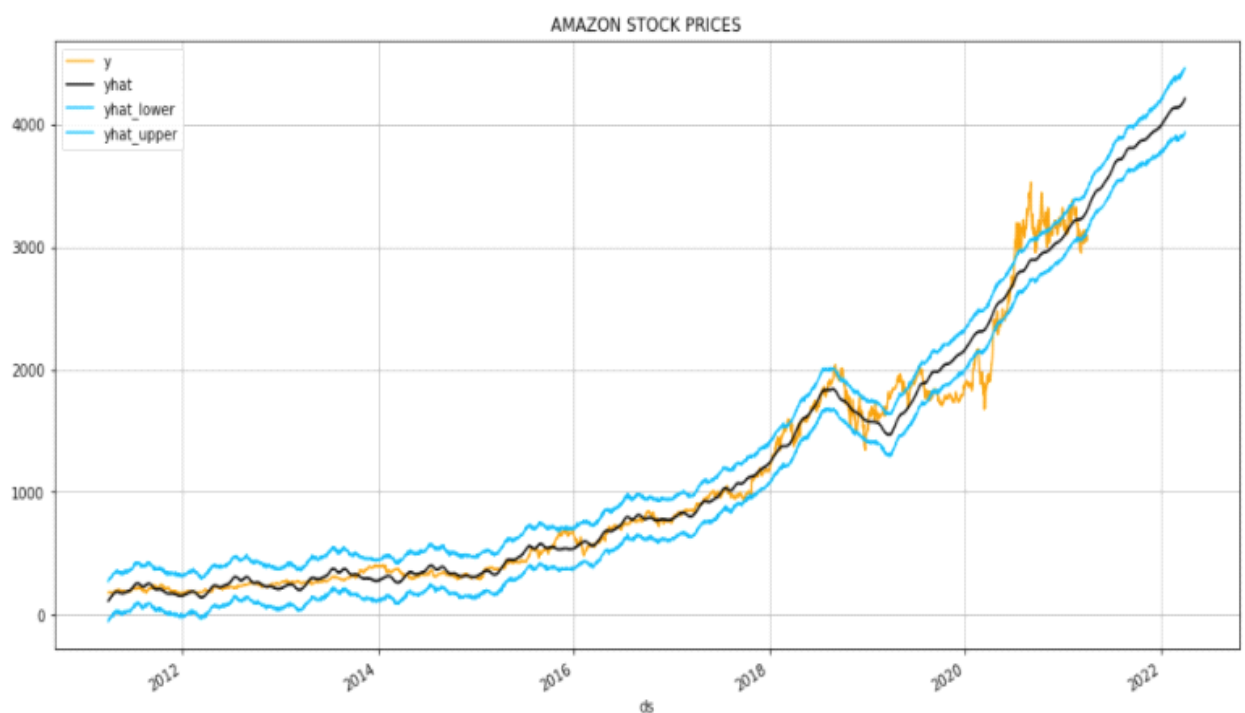


Рисунок 43: Прогнозируемый график.

`Y` - Фактическая сток-цена.

`Yhat` - Прогнозируемая цена акций.

`Yhat_lower` и `Yhat_upper` - интервал неопределенности цены акции.

6.4. Реализация модели KERAS - LSTM

Для реализации модели LSTM нам нужно изначально импортировать библиотеки. Основные пакеты и библиотеки, такие как Pandas, NumPy, Matplotlib, Sklearn и keras. Слои. Библиотека Matplotlib используется для построения переменных в графическом формате.

```
#importing libraries

import pandas as pd
import numpy as np

import matplotlib
import matplotlib.pyplot as plt
import matplotlib.dates as mdates

%matplotlib inline

from sklearn import linear_model
from keras.layers import LSTM,Dense,Dropout
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.model_selection import TimeSeriesSplit
```

Рисунок 44: Импорт пакетов и библиотек.


```
AMAZON = pd.read_csv('C:/Users/HP/Downloads/AMZN (1).csv', na_values=['null'], index_col='Date', parse_dates=True, infer_datetime_format=True)
AMAZON.head(5)
```

	Open	High	Low	Close	Adj Close	Volume
Date						
2010-04-01	135.80	136.51	131.18	131.81	131.81	8785800
2010-04-05	132.85	133.74	130.78	131.49	131.49	5816500
2010-04-06	131.23	136.00	131.18	135.56	135.56	7950300
2010-04-07	135.96	136.08	133.86	134.87	134.87	5945400
2010-04-08	134.71	141.25	134.71	140.96	140.96	12689100

Рисунок 45: Чтение данных из каталога.

На рисунке 45 показано чтение набора данных в формате csv из каталога, присвоенного переменной. Дата анализа используется для создания столбца даты в качестве индекса. Вывод печатается с использованием `head()`. На рисунке мы ясно видим, что дата перенесена как столбец индекса.

На рисунке 46 ниже показаны этапы предварительной обработки. Сначала мы проверяем форму даты, которую мы импортируем для анализа. После этого мы описываем данные с помощью функции описания. Наконец, мы проверяем недостающие значения в наборе данных.

```
AMAZON.shape
```

```
(2769, 6)
```

```
AMAZON.describe()
```

	Open	High	Low	Close	Adj Close	Volume
count	2,769.00	2,769.00	2,769.00	2,769.00	2,769.00	2,769.00
mean	926.74	936.55	915.60	926.44	926.44	4,429,499.89
std	877.56	887.52	865.80	876.68	876.68	2,530,122.21
min	105.93	111.29	105.80	108.61	108.61	881,300.00
25%	257.30	259.74	254.57	257.73	257.73	2,833,300.00
50%	533.74	539.20	526.57	533.16	533.16	3,801,900.00
75%	1,604.00	1,622.72	1,590.72	1,602.91	1,602.91	5,257,500.00
max	3,547.00	3,552.25	3,486.69	3,531.45	3,531.45	42,421,100.00

```
AMAZON.isnull().values.any()
```

Рисунок 46: Предварительная обработка набора данных



Рисунок 47: Визуализация графика с ценой закрытия.

На рисунке 47 показан график цены акций закрытия. Библиотека Matplotlib используется для построения графика. Дата действует как значения x, а цена акций на момент закрытия - как значение y на графике.

```
#Correlation Analysis
X=AMAZON.drop(['Close'],axis=1)
X=X.drop(['Adj Close'],axis=1)

X.corrwith(AMAZON['Close']).plot.bar(
    figsize = (16, 6), title = "Correlation with Close", fontsize = 20,
    rot = 90, grid = True)
```

Рисунок 48: Корреляционный анализ.

На рисунке 48 показан корреляционный анализ данных. Для проведения корреляционного анализа сначала мы отбрасываем цену закрытия от df_final. После этого строится график корреляции со всеми остальными атрибутами. Название графика - «Корреляция с закрытием». Размер шрифта 20.

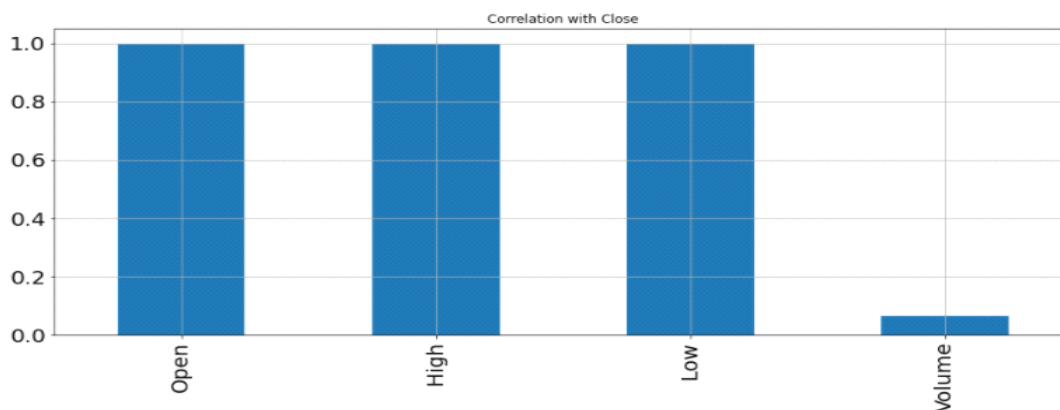


Рисунок 49: График корреляции.



Рисунок 50: Нормализация данных.

На рисунке 50 показан процесс нормализации данных. Для нормализации данных сначала мы импортируем MinMaxScaler из Sklearn. Предварительная обработка. После этого определить скаляр будет равен MinMaxScaler.

На рисунке 51 ниже показан код для отображения формы столбца функций и целевого столбца. После печати фигур мы сдвигаем целевой массив, потому что хотим предсказать значение $n + 1$ th day. Затем мы взяли последние 180 строк в качестве набора для проверки. Наконец, мы печатаем форму и целевую цену закрытия.

```

display(feature_minmax_transform.head())
print('Shape of features : ', feature_minmax_transform.shape)
print('Shape of target : ', target_adj_close.shape)

# Shift target array because we want to predict the n + 1 day value

target_adj_close = target_adj_close.shift(-1)
validation_y = target_adj_close[-180:-1]
target_adj_close = target_adj_close[:-180]

# Taking last 90 rows of data to be validation set
validation_X = feature_minmax_transform[-180:-1]
feature_minmax_transform = feature_minmax_transform[:-180]
display(validation_X.tail())
display(validation_y.tail())

print("\n -----After process----- \n")
print('Shape of features : ', feature_minmax_transform.shape)
print('Shape of target : ', target_adj_close.shape)
display(target_adj_close.tail())

```

-----After process-----

Shape of features : (2589, 4)
Shape of target : (2589, 1)

	Close
Date	
2020-07-08	3,182.63
2020-07-09	3,200.00
2020-07-10	3,104.00
2020-07-13	3,084.00
2020-07-14	3,008.87

Рисунок 51: Отображение набора данных после нормализации.

```

ts_split= TimeSeriesSplit(n_splits=10)
for train_index, test_index in ts_split.split(feature_minmax_transform):
    X_train, X_test = feature_minmax_transform[:len(train_index)], feature_minmax_transform[len(train_index): (len(train_index)+len(test_index))]
    y_train, y_test = target_adj_close[:len(train_index)].values.ravel(), target_adj_close[len(train_index): (len(train_index)+len(test_index))].values.ravel()

```

X_train.shape

(2124, 4)

y_train.shape

(2124,)

X_test.shape

(212, 4)

y_test.shape

(212,)

Рисунок 52: Обучающее и тестовое разделение данных.

На рисунке 52 показано разделение данных обучения и тестирования с использованием разделения таймсерий. Затем мы распечатали формы xtrain, ytrain, xtest, ytest. После этого мы приступили к обработке данных для построения модели LSTM. Для создания моделей мы импортируем Sequential из keras.models, Dense из keras.layers,

```
from keras.models import Sequential
from keras.layers import Dense
import keras.backend as K
from keras.callbacks import EarlyStopping
from keras.optimizers import Adam
from keras.models import load_model
from keras.layers import LSTM
K.clear_session()
model_lstm = Sequential()
model_lstm.add(LSTM(16, input_shape=(1, X_train.shape[1]), activation='relu', return_sequences=False))
model_lstm.add(Dense(1))
model_lstm.compile(loss='mean_squared_error', optimizer='adam')
early_stop = EarlyStopping(monitor='loss', patience=5, verbose=1)
history_model_lstm = model_lstm.fit(X_train, y_train, epochs=200, batch_size=8, verbose=1, shuffle=False, callbacks=[early_stop])
```

Рисунок 53: Построение модели LSTM.

Получив эталонную модель, мы приступили к обработке данных для построения модели LSTM. Для создания моделей мы импортируем Sequential из keras.models, Dense из keras.layers, Keras.backend, Ранняя остановка из keras.callbacks, Adam из keras.optimizer, Loadmodel из keras.models и LSTM из keras.layers.

```
#Evaluation of model
y_pred_test_lstm = model_lstm.predict(X_test)
y_train_pred_lstm = model_lstm.predict(X_train)
print("The R2 score on the Train set is:\t{:0.3f}".format(r2_score(y_train, y_train_pred_lstm)))
r2_train = r2_score(y_train, y_train_pred_lstm)

print("The R2 score on the Test set is:\t{:0.3f}".format(r2_score(y_test, y_pred_test_lstm)))
r2_test = r2_score(y_test, y_pred_test_lstm)

The R2 score on the Train set is:      0.999
The R2 score on the Test set is:      0.956
```

Рисунок 54: Оценка модели с использованием R2 Score.

Оценка R^2 - это функция оценки регрессии. Это статистическая мера, которая используется для описания доли дисперсии зависимой переменной, которая объясняется в регрессионной модели. Он может быть отрицательным или положительным. Здесь оценка R^2 для набора поездов составляет 0,999, а оценка R^2 для набора тестов составляет 0,956.

```
y_pred_test_LSTM = model_lstm.predict(X_tst_t)
```

```
plt.figure(figsize=(12,6))
plt.plot(y_test, label='True')
plt.plot(y_pred_test_LSTM, label='LSTM')
plt.title("LSTM's_Prediction - AMAZON STOCK PRICES")
plt.xlabel('Observation')
plt.ylabel('INR_Scaled')
plt.legend()
plt.show()
```

Рисунок 55: Код для визуализации прогноза LSTM

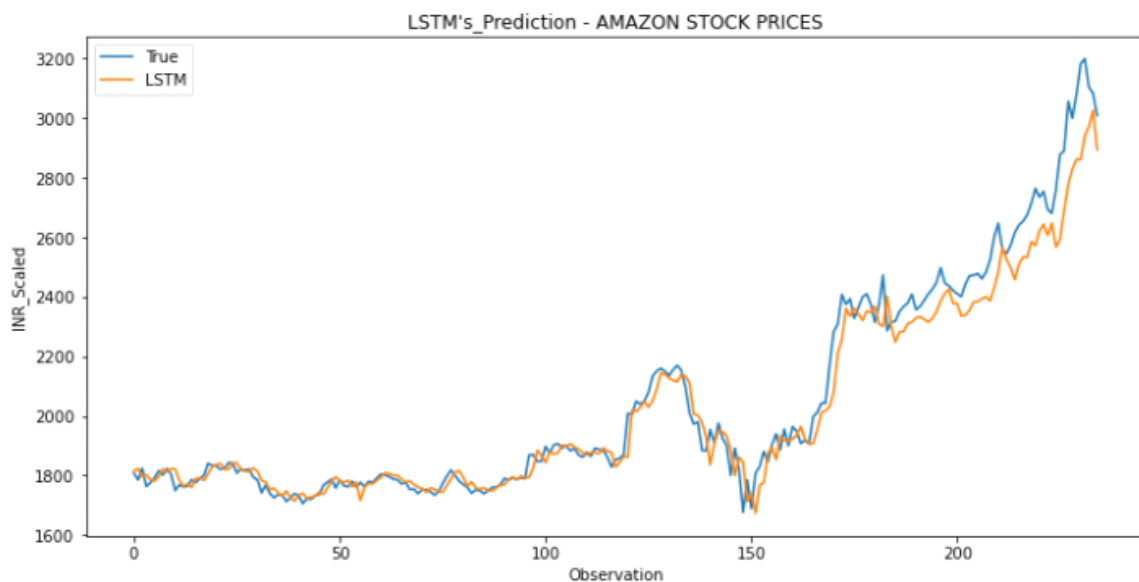


Рисунок 56: Прогнозируемый график.

6.5. Прогнозируемые графики всех компаний - модель ARIMA

Sl.No	Название банка	Прогнозируемые графики
1	АМАЗОНКА	 <p>The chart, titled 'AMAZON STOCK PRICES', displays the stock price history and forecast for Amazon from 2013 to 2023. The y-axis represents the 'close price' ranging from 0 to 4000, and the x-axis represents the 'Date' with markers for 2013, 2015, 2017, 2019, 2021, and 2023. The legend indicates three data series: 'observed' (blue line), 'Predicted' (red line), and 'Forecast' (yellow line). The observed data shows a steady upward trend with some volatility, reaching approximately 3200 in early 2021. The predicted data follows the observed data closely until early 2021, after which the forecast (yellow line) continues the upward trend, reaching approximately 4000 by 2023. A grey shaded area around the lines represents the confidence interval of the model.</p>

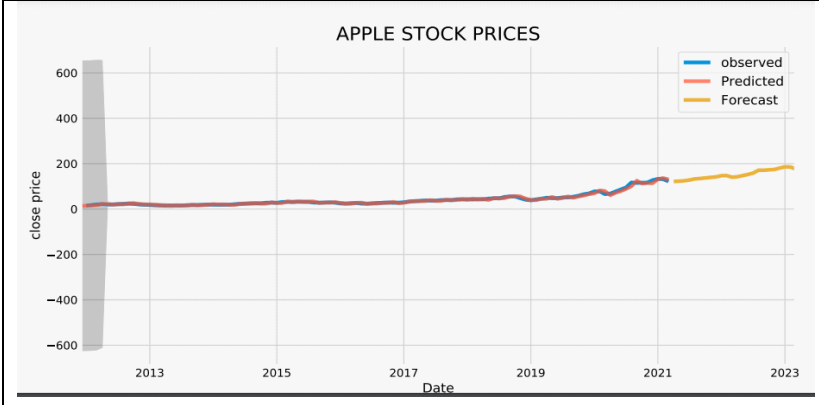
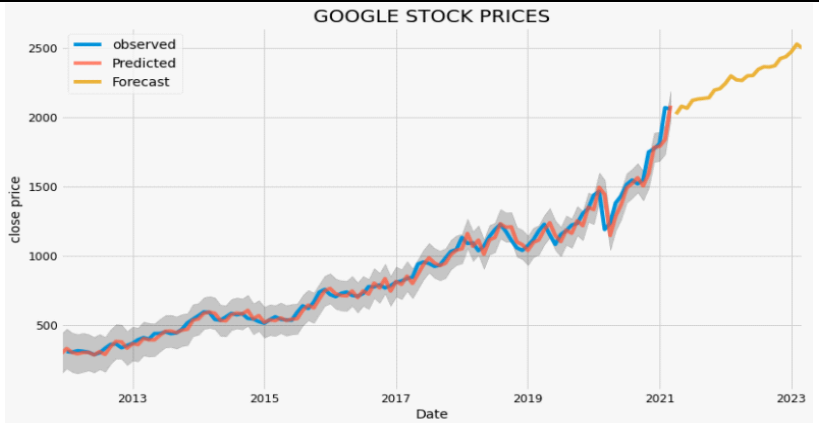
2	ЯБЛОКО	
3	GOOGLE	

Таблица 1: Прогнозируемые графики - ARIMA.

6.6. Прогнозируемые графики компаний - Prophet

Sl.No	Название банка	Прогнозируемые графики
1	АМАЗОНКА	

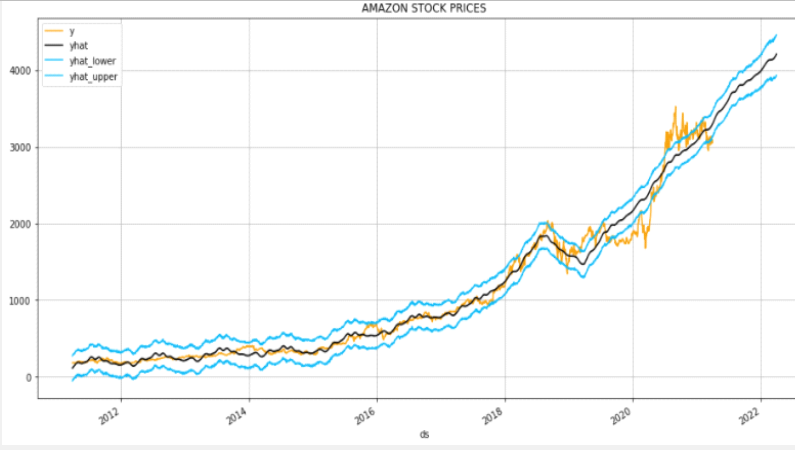
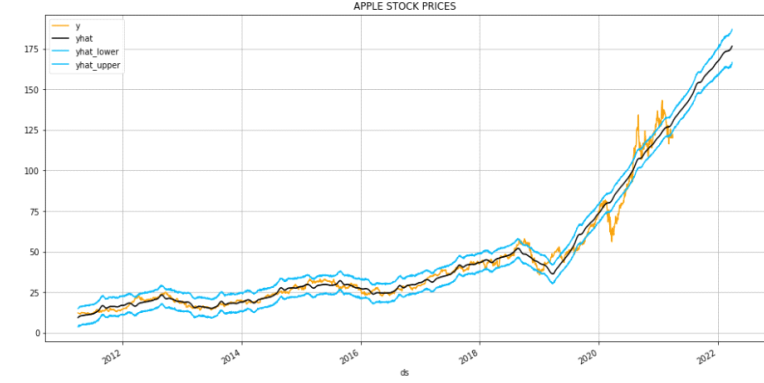
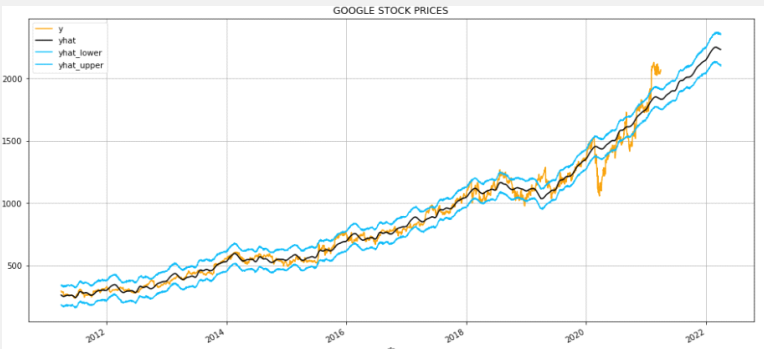
		
2	ЯБЛОКО	
3	GOOGLE	

Таблица 2: Прогнозируемые графики - Пророк.

- **Прогнозируемые графики всех банков - модель Keras-LSTM**

Sl.No	Название банка	Прогнозируемые графики
-------	----------------	------------------------


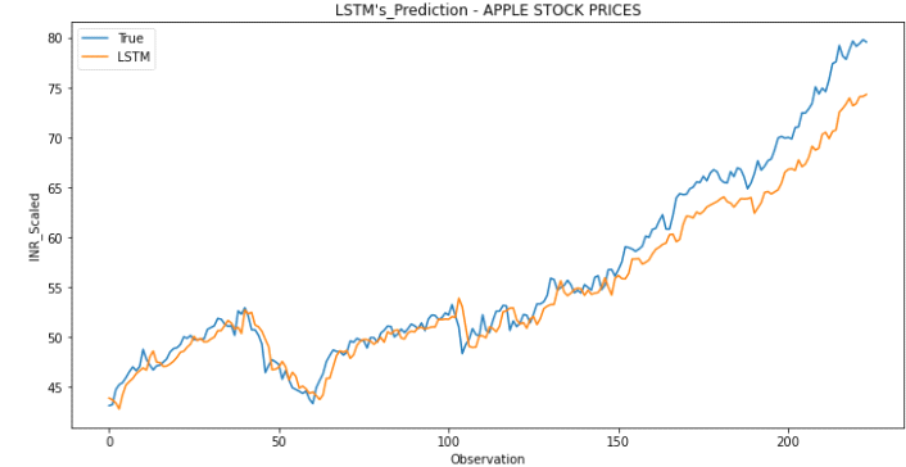
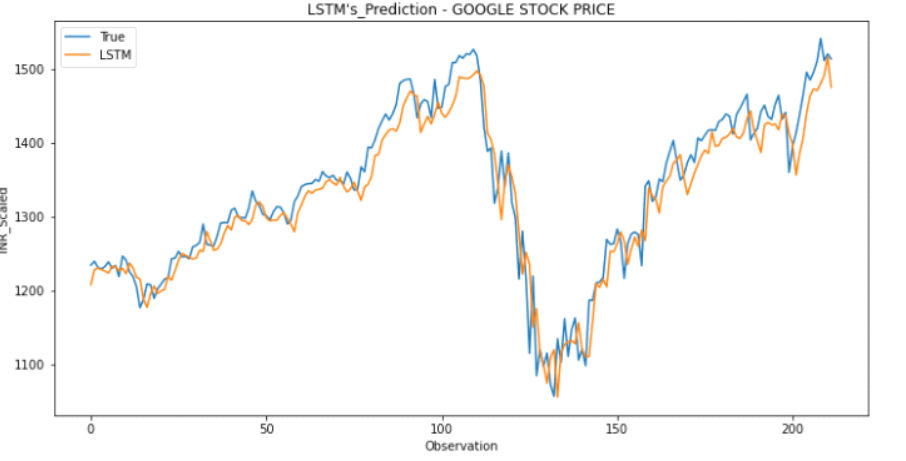
1	АМАЗОНКА	
2	ЯБЛОКО	
3	GOOGLE	

Таблица 3: Прогнозируемые графики - Keras с LSTM.

Глава 7. Тестирование и оценка

7.1. RMSE - среднеквадратическое значение ошибки

Значение RMSE - это квадратный корень из среднеквадратичной ошибки. RMSE измеряет разницу между прогнозируемым значением и исходным значением. Мы знаем, что более низкое значение RMSE дает лучшую производительность в моделях временных рядов. Значение RMSE рассчитывается для трех моделей временных рядов и сравнивается между каждой из них. Заключительный этап проекта - выполнение оцененного результата.

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (P_i - O_i)^2}{n}}$$

Уравнение 6: Формулы для RMSE.

Уравнение 6 - это формулы для расчета значения RMSE. Здесь n означает положительное целое число, P_i - идеальное расстояние, а O_i - производительность.

7.2. Значение RMSE - модель ARIMA

7.2.1. AMAZON

```
y_forecasted = pred.predicted_mean
y_truth = monthly_mean['2011-12-31:']

from sklearn import metrics
print('Mean Absolute Error:', metrics.mean_absolute_error(y_truth, y_forecasted))
print('Mean Squared Error:', metrics.mean_squared_error(y_truth, y_forecasted))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_truth, y_forecasted)))
```

Mean Absolute Error: 54.62281905757571
Mean Squared Error: 7358.8388235904
Root Mean Squared Error: 85.78367457500524

Рисунок 57: Отчет о проверке стоимости акций AMAZON.

На рисунке 57 показан отчет о проверке стоимости акций AMAZON. Здесь мы импортируем метрики из Sklearn для расчета ошибок между предсказанной моделью и фактической моделью. Мы также рассчитали RMSE, MAE и MSE.

RMSE AMAZON = 85,78

7.2.2. APPLE

```
y_forecasted = pred.predicted_mean
y_truth = monthly_mean['2011-12-31:']

from sklearn import metrics
print('Mean Absolute Error:', metrics.mean_absolute_error(y_truth, y_forecasted))
print('Mean Squared Error:', metrics.mean_squared_error(y_truth, y_forecasted))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_truth, y_forecasted)))
```

Mean Absolute Error: 2.4626403305743603
Mean Squared Error: 13.477614122293293
Root Mean Squared Error: 3.6711870181581996

Рисунок 58: Отчет о проверке стоимости акций APPLE.

На рисунке 58 показан отчет о проверке стоимости акций APPLE. Здесь мы импортируем метрики из Sklearn для расчета ошибок между прогнозируемой моделью и фактической моделью. Мы также рассчитали RMSE, MAE и MSE.

RMSE APPLE 3.67.

7.2.3. GOOGLE

```
y_forecasted = pred.predicted_mean
y_truth = monthly_mean['2011-12-31:']

from sklearn import metrics
print('Mean Absolute Error:', metrics.mean_absolute_error(y_truth, y_forecasted))
print('Mean Squared Error:', metrics.mean_squared_error(y_truth, y_forecasted))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_truth, y_forecasted)))
```

Mean Absolute Error: 35.488297471602756
Mean Squared Error: 2782.8208529234244
Root Mean Squared Error: 52.752448786036695

Рисунок 59: Отчет о проверке стоимости акций GOOGLE.

На рисунке 59 показан отчет о проверке стоимости акций GOOGLE. Здесь мы импортируем метрики из Sklearn для расчета ошибок между прогнозируемой моделью и фактической моделью. Мы также рассчитали RMSE, MAE и MSE.

RMSE GOOGLE = 52,75.

7.3. Значение RMSE - модель PROPHET

7.3.1. AMAZON

```
from sklearn.metrics import mean_squared_error, mean_absolute_error
print('Mean Absolute Error:', mean_absolute_error(metric_AMAZON.y, metric_AMAZON.yhat))
print('Mean Squared Error:', mean_squared_error(metric_AMAZON.y, metric_AMAZON.yhat))
print('Root Mean Squared Error:', np.sqrt(mean_squared_error(metric_AMAZON.y, metric_AMAZON.yhat)))
```

Mean Absolute Error: 77.65379620247134
Mean Squared Error: 16629.598921057743
Root Mean Squared Error: 128.95580220004737

Рисунок 60: Результат проверки цены акций AMAZON.

На рисунке 60 показан отчет о проверке стоимости акций AMAZON. Здесь мы импортируем среднеквадратичную ошибку и среднюю абсолютную ошибку из Sklearn для вычисления ошибок между прогнозируемой моделью и фактической моделью. Мы также рассчитали RMSE, MAE и MSE.

RMSE AMAZON = 128,95.

7.3.2. APPLE

```
from sklearn.metrics import mean_squared_error, mean_absolute_error
print('Mean Absolute Error:', mean_absolute_error(metric_APPLE.y, metric_APPLE.yhat))
print('Mean Squared Error:', mean_squared_error(metric_APPLE.y, metric_APPLE.yhat))
print('Root Mean Squared Error:', np.sqrt(mean_squared_error(metric_APPLE.y, metric_APPLE.yhat)))
```

Mean Absolute Error: 2.7754684944520216
Mean Squared Error: 19.73795036145253
Root Mean Squared Error: 4.4427413115612

Рисунок 61: Отчет о проверке стоимости акций APPLE.

На рисунке 61 показан отчет о проверке стоимости акций APPLE. Здесь мы импортируем среднеквадратичную ошибку и среднюю абсолютную ошибку из Sklearn для расчета ошибок между прогнозируемой моделью и фактической моделью. Мы также рассчитали RMSE, MAE и MSE.

RMSE APPLE = 4,44.

7.3.3. GOOGLE

```
from sklearn.metrics import mean_squared_error, mean_absolute_error
print('Mean Absolute Error:', mean_absolute_error(metric_GOOGLE.y, metric_GOOGLE.yhat))
print('Mean Squared Error:', mean_squared_error(metric_GOOGLE.y, metric_GOOGLE.yhat))
print('Root Mean Squared Error:', np.sqrt(mean_squared_error(metric_GOOGLE.y, metric_GOOGLE.yhat)))
```

Mean Absolute Error: 36.74722554240493
Mean Squared Error: 3538.3798347479387
Root Mean Squared Error: 59.48428224958202

Рисунок 62: Отчет о проверке GOOGLE.

На рисунке 62 показан отчет о проверке стоимости акций GOOGLE. Здесь мы импортируем среднеквадратичную ошибку и среднюю абсолютную ошибку из Sklearn для вычисления ошибок между прогнозируемой моделью и фактической моделью. Мы также рассчитали RMSE, MAE и MSE.

RMSE GOOGLE = 59,48.

7.4. Значение RMSE - KERAS с моделью LSTM

7.4.1. AMAZON

```

from sklearn import metrics
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred_test_LSTM))
print('Mean Squared Error:', metrics.mean_squared_error(y_test,y_pred_test_LSTM ))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test,y_pred_test_LSTM )))

```

Mean Absolute Error: 48.74962753706782
Mean Squared Error: 5610.731840834851
Root Mean Squared Error: 74.90481854216624

Рисунок 63: Отчет о проверке стоимости акций AMAZON.

На рисунке 63 показан отчет о проверке стоимости акций AMAZON. Здесь мы импортируем метрики из Sklearn для расчета ошибок между предсказанной моделью и фактической моделью. Мы также рассчитали RMSE, MAE и MSE. Ведь печатаем все значения.

RMSE акции AMAZON = 74,90.

7.4.2. APPLE

```

from sklearn import metrics
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred_test_LSTM))
print('Mean Squared Error:', metrics.mean_squared_error(y_test,y_pred_test_LSTM ))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test,y_pred_test_LSTM )))

```

Mean Absolute Error: 1.7893405207563127
Mean Squared Error: 5.830101872065758
Root Mean Squared Error: 2.4145603889871463

Рисунок 64: Отчет о проверке стоимости акций APPLE.

На рисунке 64 показан отчет о проверке стоимости акций APPLE. Здесь мы импортируем метрики из Sklearn для расчета ошибок между предсказанной моделью и фактической моделью. Мы также рассчитали RMSE, MAE и MSE. Ведь печатаем все значения.

RMSE цены акций APPLE = 2,41.

7.4.3. GOOGLE

```

from sklearn import metrics
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred_test_LSTM))
print('Mean Squared Error:', metrics.mean_squared_error(y_test,y_pred_test_LSTM ))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test,y_pred_test_LSTM )))

Mean Absolute Error: 22.13527312480053
Mean Squared Error: 883.420178773488
Root Mean Squared Error: 29.72238514610643

```

Рисунок 65: Отчет о проверке стоимости акций GOOGLE.

На рисунке 65 показан отчет о проверке стоимости акций GOOGLE. Здесь мы импортируем среднеквадратичную ошибку и среднюю абсолютную ошибку из Sklearn для вычисления ошибок между прогнозируемой моделью и фактической моделью. Мы также рассчитали RMSE, MAE и MSE. Ведь печатаем все значения.

RMSE цены акции GOOGLE = 29,72.

Приведенные выше рисунки ясно показывают тестирование и оценку моделей временных рядов различных банков. Теперь сравниваем модели на гистограммах.

7.5. Сравнение моделей прогнозирования

Мы оформили данные в виде гистограммы в Jupyter Notebook. Для сравнения значения RMSE мы импортируем библиотеки NumPy и Matplotlib. NumPy - это библиотека Python, которая работает с массивами. Matplotlib - это библиотека визуализации на Python.

Comparison of Forecasting Models Using RMSE value

```

import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

```

Рисунок 66: Импорт библиотек.


```
Models=['ARIMA','PROPHET','KERAS with LSTM']
RMSE=[85.78,128.95,74.9]
graph = np.arange(len(Models))
plt.style.use('fivethirtyeight')
plt.figure(figsize=(10,6))
plt.bar(graph,RMSE, label="RMSE")
plt.xticks(graph,Models)
plt.ylabel("Value of RMSE")
plt.title('Comparison Report Graph - Federal Bank')
plt.legend()
```

Рисунок 67: Код для разработки гистограммы.

На рисунке 67 показан код для создания столбчатой диаграммы сравнения. Сначала мы инициализируем имена моделей в переменной Models. Затем мы сохраняем значения RMSE в переменной массива с именем RMSE. Мы используем стиль сюжета как пять тридцать восемь. Размер рисунка графика 10,6. Название графика дается с использованием plt.title.

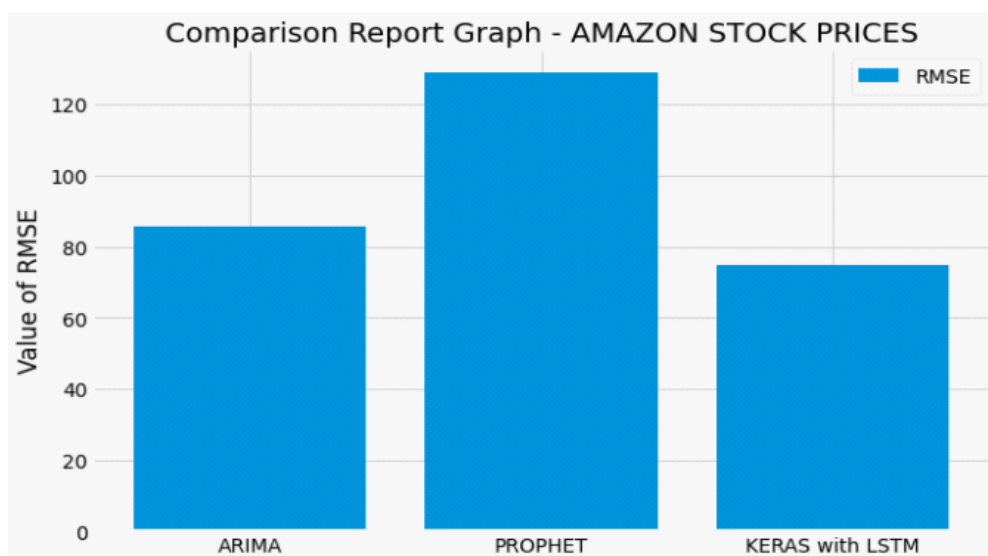


Рисунок 68: График сравнительного отчета - цена акций AMAZON.

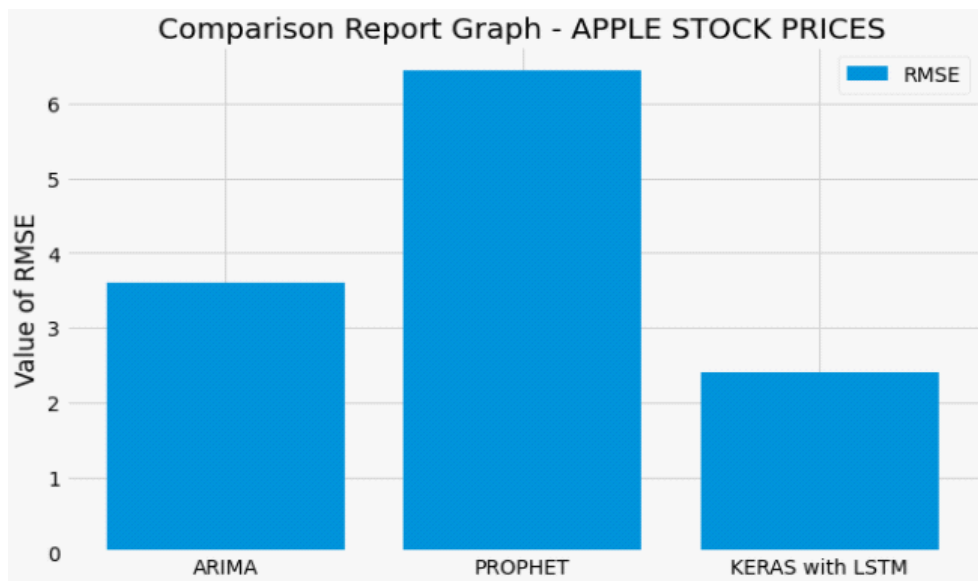


Рисунок 69: График сравнительного отчета - цена акций APPLE.

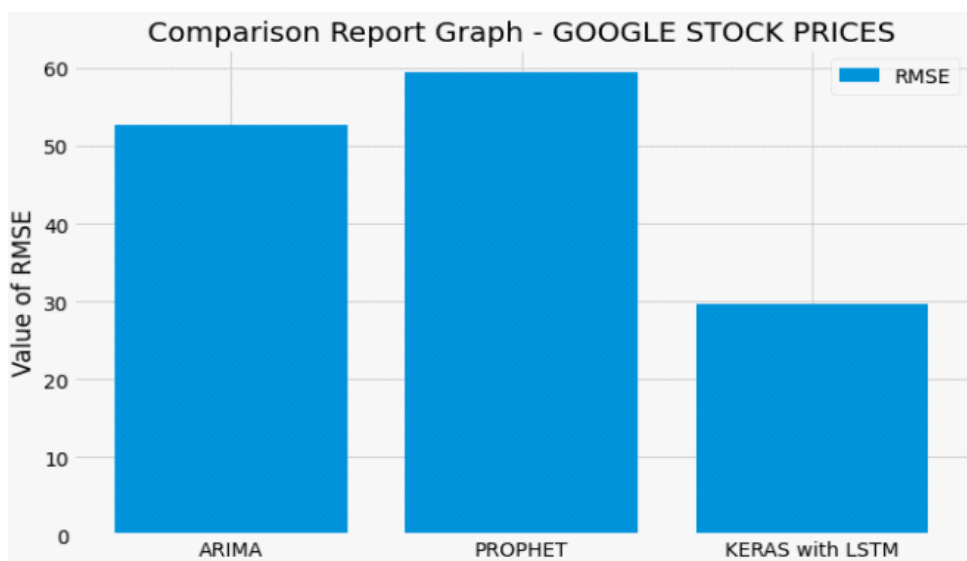


Рисунок 70: График сравнительного отчета - курс акций GOOGLE.

На всех приведенных выше графиках четко видно, что KERAS с моделью временных рядов LSTM имеет наименьшее значение RMSE. Более низкое значение RMSE дает лучшую модель производительности.

Глава 8. Заключение и дальнейшая работа

8.1. Заключение

В этом исследовании речь шла о прогнозировании будущей стоимости акций с использованием различных моделей прогнозирования машинного обучения. Прогноз курса акций прошел успешно, основная цель работы выполнена. Был проведен сравнительный анализ трех моделей ARIMA, PROPHET и LSTM, примененных к историческим данным за 10-летний период (с апреля 2011 года по апрель 2021 года) трех крупных компаний, таких как AMAZON, APPLE и GOOGLE, данные из известного Yahoo! Финансы. После применения каждого алгоритма к нашим различным наборам данных результаты прогнозов были удовлетворительными для наших 3 моделей, но сравнение различных значений RMSE, полученных для каждой модели KERAS с LSTM, дает лучшую модель временных рядов для прогнозирования цены акций. ARIMA и PROPHET также создали надежный прогнозируемый график с ценой акций.

8.2. Будущая работа

Будущие масштабы проекта указаны ниже.

- Создание веб-сайта позволяет прогнозировать цены на акции
- Точность прогноза акций зависит от рынка, процентных ставок, дивидендов, экономических колебаний, политического климата. Так что нужно сосредоточиться и на внешних факторах.
- Внедрить больше алгоритмов для прогнозирования курса акций.

Рекомендации

[1]	J. Chen, "Investopedia," 28 February 2020. [Online]. Available: https://www.investopedia.com/terms/s/stockmarket.asp .
[2]	K. Amadeo, "The Balance," 15 May 2020. [Online]. Available: https://www.thebalance.com/how-does-the-stock-market-work-3306244].
[3]	17 March 2020. [Online]. Available: https://capital.com/stock-market-prediction-definition .
[4]	"Valamis," [Online]. Available: https://www.valamis.com/hub/predictive-analytics .
[5]	"Yahoofinance," 02 June 2020. [Online]. Available: https://finance.yahoo.com/chart/FEDERALBNK.BO#eyJpbmRlcnZhbCI6Im1vbml0IjpuZDVsLCJjYW5kbGVXaWR0aCI6OC40NzQ1NzYyNzExODY0NDZhc2hhaXliOnRydWUslmNoYXJ0V
[6]	R. Das, "PhysicsPI," 13 February 2019. [Online]. Available: https://thephysicspi.blogspot.com/2019/02/advantages-and-disadvantages-of-time.html .
[7]	"advisorymandi," 17 November 2018. [Online]. Available: http://www.advisorymandi.com/blog/advantages-and-disadvantages-of-investing-in-banking-sector/ .
[8]	L. Bramble. [Online]. Available: https://smallbusiness.chron.com/stock-market-started-whom-14745.html .
[9]	T. Kimoto, K. Asakawa and M. Yoda, 15 October 2012. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/5726498 .
[10]	A. A. Ariyo, A. O. Adewumi and C. K. Ayo, "IEEE Xplore," 23 February 2015. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7046047 .
[11]	S. Selvin, R. Vinayakumar, E. A. Gopalakrishnan, V. K. Menon and K. P. Soman, 04 December 2017. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8126078 .
[12]	M. Mazed, "Bracu," 2019. [Online]. Available: http://dspace.bracu.ac.bd/xmlui/handle/10361/12818 .

[14]	K. K. T. I. Y. F. Y. NAKAMURA, "Wiley Online Library," 4 December 1998. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/(SICI)1099-1174(199703)6:1%3C11::AID-ISA115%3E3.0.CO;2-3 .
[15]	M. M. Ali, "Hindawi," 05 March 2015. [Online]. Available: https://www.hindawi.com/journals/iam/2014/614342/ .
[16]	P. R. Charkha, 29 July 2008. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/4579969 .
[17]	L. D. Persio, "IRIS.it," [Online]. Available: https://iris.univr.it/retrieve/handle/11562/955101/60620/Artificial%20Neural%20Networks%20architectures%20for%20stock%20price%20prediction%20comparisons%20and%20applications.pdf .
[18]	M. Roondiwai, "Semantic Scholar," 02 June 2020. [Online]. Available: https://pdfs.semanticscholar.org/3f5a/cb5ce4ad79f08024979149767da6d35992ba.pdf .
[19]	"Datascience," 2012. [Online]. Available: http://www.datascience-pm.com/crisp-dm-2/ .
[20]	V. Dsouza, "Research gate," July 2018. [Online]. Available: https://www.researchgate.net/figure/CRISP-DM-Model-Taylor-2017_fig1_326235288 .
[21]	adishesha. [Online]. Available: https://www.slideshare.net/adishesha12/python-programming-168124234 .
[22]	G. McIntire. [Online]. Available: https://www.learndatasci.com/tutorials/python-pandas-tutorial-complete-introduction-for-beginners/ .
[23]	"edpresso," [Online]. Available: https://www.educative.io/edpresso/what-is-pandas-in-python .
[24]	DataFlair, 31 May 2019. [Online]. Available: https://data-flair.training/blogs/pandas-HYPERLINKhttps://data-flair.training/blogs/pandas-%20dataframe/ HYPERLINK "https://data-flair.training/blogs/pandas-%20dataframe/"dataframe/
[25]	"W3 Schools and UCF," [Online]. Available: https://webcourses.ucf.edu/courses/1249560/pages/python-lists-vs-numpy-arrays-what-is-the-difference , https://www.w3schools.com/python/numpy_intro.asp .
[26]	"ScikitLearn," [Online]. Available: https://scikit-learn.org/stable/ .
[27]	J. T. Point. [Online]. Available: https://www.javatpoint.com/advantage-and-disadvantage-of-tensorflow .
[28]	DATA FLAIR TEAM, "dataflair," 9 May 2020. [Online]. Available: https://data-flair.training/blogs/python-keras-advantages-and-limitations/ .

[29]	P. Dar, "Analytics Vidhya," 24 May 2018. [Online]. Available: https://www.analyticsvidhya.com/blog/2018/05/starters-guide-jupyter-notebook/ .
[30]	"Jupyter Notebook," [Online]. Available: https://jupyter-notebook.readthedocs.io/en/stable/notebook.html .
[31]	S. Arora, "Hackr," 13 April 2020. [Online]. Available: https://hackr.io/blog/what-is-pycharm .
[32]	"Codingdojo," [Online]. Available: https://www.codingdojo.com/blog/choosing-python-web-frameworks .
[33]	Anaconda Documentation, "Anaconda," [Online]. Available: https://docs.anaconda.com/anaconda/user-guide/getting-started/ .
[34]	S. Prabhakaran, "Machine learning plus," [Online]. Available: https://www.machinelearningplus.com/time-series/arima-model-time-series-forecasting-python/ .
[35]	A. Choudhary, "Analytics Vidhya," 10 May 2018. [Online]. Available: https://www.analyticsvidhya.com/blog/2018/05/generate-accurate-forecasts-facebook-prophet-python-r/ .
[36]	adventuresinmachinelearning, "Adventures in machine learning," [Online]. Available: https://adventuresinmachinelearning.com/keras-lstm-tutorial/ .
[37]	colah's blog, "colah's blog," 27 August 2015. [Online]. Available: https://colah.github.io/posts/2015-08-Understanding-LSTMs/ .
[38]	"Classic.d2l.ai," [Online]. Available: https://classic.d2l.ai/chapter_recurrent_neural-networks/lstm.html .
[39]	A. Singh, "Analytics Vidhya," 25 October 2018. [Online]. Available: https://www.analyticsvidhya.com/blog/2018/10/predicting-stock-price-machine-learningnd-deep-learning-techniques-python/ .
[40]	M. Rouse, "TechTarget," [Online]. Available: https://whatis.techtarget.com/definition/time-series-forecasting .
[41]	J. Brownlee, "Machine Learning," 2 December 2016. [Online]. Available: https://machinelearningmastery.com/time-series-forecasting/ .