Федеральное государственное бюджетное образовательное учреждение высшего образования

**«ФИНАНСОВЫЙ УНИВЕРСИТЕТ ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ»**

**Кафедра анализа данных и машинного обучения**

Пояснительная записка к курсовой работе

по дисциплине “Технологии анализа данных и машинное обучение” на тему:

«Применение машинного обучения в прогнозировании стоимости ценных бумаг на фондовой бирже»

Выполнил:

студент группы ЗБ-ПИ20-1 факультета информационных технологий и анализа больших данных

Ерошкин К.В.

Наручный руководитель:

Старший преподаватель кафедры анализа данных и машинного обучения:

Добрина М.В.

2024 г

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc167459005)

[1. Теоретическая часть машинного обучения 6](#_Toc167459006)

[1.1 Понятие машинного обучения 6](#_Toc167459007)

[1.2 Способы машинного обучения 7](#_Toc167459008)

[1.3 Датафреймы в машинном обучении 10](#_Toc167459009)

[2. Машинное обучение для прогнозирования цен акций 13](#_Toc167459010)

[2.1 Подготовка среды разработки 13](#_Toc167459011)

[2.2 Создание графического интерфейса 14](#_Toc167459012)

[2.3 Подготовка датафрейма 16](#_Toc167459013)

[17](#_Toc167459014)

[2.4 Выбор метода для машинного обучения 18](#_Toc167459015)

[3. Оптимизация работы приложения 20](#_Toc167459016)

[3.1 Внедрение многопоточности 20](#_Toc167459017)

[3.1 Графики результатов 20](#_Toc167459018)

[3.2 Удобство отслеживания процесса обучения 21](#_Toc167459019)

[3.3 Практическое обоснование результатов исследования в предметной области 22](#_Toc167459020)

[ЗАКЛЮЧЕНИЕ 24](#_Toc167459021)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ И ИНТЕРНЕТ-РЕСУРСОВ 26](#_Toc167459022)

[ПРИЛОЖЕНИЕ 27](#_Toc167459023)

# ВВЕДЕНИЕ

Машинное обучение (ML) - это раздел искусственного интеллекта (ИИ), который предоставляет системам возможность автоматически обучаться и улучшаться на основе опыта без явного программирования. В последние годы машинное обучение стало неотъемлемой частью различных отраслей, от медицины и биоинформатики до финансов и торговли. Прогнозирование цен на акции с использованием методов машинного обучения представляет собой одну из наиболее интересных и практически значимых областей исследований и разработки. Это связано с огромным объемом данных, доступных в финансовом секторе, и потенциальной прибылью, которую можно извлечь из точного предсказания будущих цен на акции.

Применение машинного обучения для прогнозирования цен на акции сопряжено с рядом вызовов. Во-первых, финансовые данные часто шумные и непредсказуемые, что может затруднить обучение модели. Во-вторых, существует риск переобучения моделей, когда они хорошо справляются с историческими данными, но плохо предсказывают будущие события. В-третьих, необходимо учитывать правовые и этические аспекты использования алгоритмов в торговле.

Современные тенденции и инновации в прогнозировании цен на акции.

С развитием технологий и увеличением доступных данных, прогнозирование цен на акции с использованием машинного обучения стало не только более популярным, но и более точным. Современные тенденции и инновации в этой области включают:

1. Большие данные (Big Data): Современные финансовые рынки генерируют огромные объемы данных, они содержат в себе такие данные, как цена акций, объемы торгов, финансовые отчеты, а также и экономические индикаторы. Анализ таких данных в свою очередь дает моделям машинного обучения учитывать огромное колличество факторов, из-за которых могут меняться цены акций. Технологии обработки больших данных, такие как Hadoop и Spark, играют ключевую роль в управлении и анализе этих данных.

2. Интеграция данных из различных источников: Прогнозирование цен на акции требует интеграции данных из множества источников. Это включает не только традиционные финансовые данные, но и нетрадиционные источники, такие как социальные медиа, новости, климатические данные и даже данные о поведении пользователей. Такие интеграционные подходы помогают создавать более точные и надежные модели.

3. Автоматизация и алгоритмическая торговля: Алгоритмическая торговля, использующая машинное обучение, позволяет автоматизировать процессы принятия решений на финансовых рынках. Такие системы могут реагировать на изменения рыночных условий в реальном времени, что значительно повышает их эффективность. Алгоритмическая торговля также может использовать сложные стратегии, основанные на обучении с подкреплением, что позволяет системам адаптироваться к изменяющимся рыночным условиям.

4. Этичность и интерпретируемость моделей: С развитием машинного обучения возникает необходимость учитывать этические аспекты и обеспечивать интерпретируемость моделей. Важно, чтобы модели были прозрачными и понятными для пользователей, особенно в финансовом секторе, где неправильные прогнозы могут привести к значительным финансовым потерям. Методы интерпретируемого машинного обучения, такие как LIME и SHAP, помогают объяснять решения моделей и обеспечивать их прозрачность.

5. Квантовые вычисления: Квантовые вычисления представляют собой перспективное направление, которое может привести к революционным изменениям в прогнозировании цен на акции. Квантовые алгоритмы имеют потенциал значительно ускорить обработку данных и улучшить точность моделей машинного обучения. Несмотря на то, что квантовые вычисления находятся на ранней стадии развития, они уже привлекают внимание исследователей и финансовых аналитиков.

Будущее машинного обучения в финансовом секторе

Будущее машинного обучения в прогнозировании цен на акции обещает быть захватывающим. Ожидается, что развитие технологий, увеличение вычислительных мощностей и улучшение алгоритмов приведут к созданию еще более точных и эффективных моделей.

Целью выполнения курсовой работы является разработка и исследование моделей машинного обучения для повышения точности и эффективности прогнозирования цен акций.

Задача выполнения курсовой представляет собой, разработать программу с графическим интерфейсом, которая будет работать с базами данными в онлайн и офлайн режиме, позволяющая анализировать данные цен на акции за прошлый период времени, обучение машины для анализа данных и вывод предсказанных результатов.

# Теоретическая часть машинного обучения

# Понятие машинного обучения

Машинное обучение – это способ, который позволяет обучать компьютер для выполнения поставленной перед ним задачи. Машинное обучение позволяет нам обучить компьютер, в свою очередь это открывает для нас новые возможности в абсолютно разных направлениях деятельности. Данная технология позволяет составлять точные прогнозы основываясь на входных данных.

Машинное обучение позволяет получать выгоду для бизнеса различных направлений, оно подойдет как для малых предприятий, так и для огромных, уже сформировавшихся компаний. Машинное обучение может использоваться в любых странах и никак не зависит от местоположения, что в свою очередь дает удобство работы из любой точки мира.

Компоненты особой важности в машинном обучении:

1. Инновационность: возможности, которые представляет нам машинное обучение, практически безграничны.
2. Специфичность: машинное обучение используется для внедрения и создания новых продуктов, а также улучшение уже существующих исключительно людьми.
3. Простота: продукты, реализуемые с помощью машинного обучения, не требуют настолько высокого познания в данном направления, для того чтобы быть полезными.

В основе машинного обучения, лежат три важных компонента − данные, признаки, алгоритмы.

Для машинного обучения, данные собираются любыми возможными способами. Для большей точности результатов, нужно собрать как можно больше исходных данных, так как в процессе обучение, именно количество и качество этих данных, будут основным фактором.

Признаки − существуют параметры, на которые будет опираться машинное обучение. Это индивидуальное измеримое свойство, а также характеристика наблюдаемого явления. Обычно это числовое значение, но и графы и строки также используются в синтаксическом распознавании образов.

В зависимости от выбранного метода обучения, будет также зависеть скорость, точность и размер готовой модели, важный момент заключается в том, что все это будет хорошо работать только при наличии хороших исходных данных.

# Способы машинного обучения

Для машинного обучения, стоит выделить три способа:

1. обучение с учителем;
2. обучение без учителя;
3. обучение с частичным привлечением учителя;
4. обучение с подкреплением.

Обучение с учителем подразумевает обучение компьютера, на основе подготовленных данных, которые содержат в себе правильный ответ. И задача этого алгоритма не отвечать на вопрос, который перед ней стоит, а понять, почему получаются именно эти данные. Результатом такого обучения является появление способности выстраивать корректные прогнозы и модели. В большинстве случаев этот метод используется для задач классификации и прогнозирования.

В таких задачах как классификации алгоритм может предсказывать дискретные значения, у каждого из которых есть свой номер класса, к котором принадлежит объект.

В задачах регрессии, которые в свою очередь связаны с непрерывными данным. Как пример линейной регрессии, можно рассмотреть такие: программа будет вычисляет значение Y, которое мы ожидаем получить от нее, учитывая при этом конкретные значения X, которые мы ей дали. В добавок можно прогнозировать покупательское поведение с учетом того набора продуктов, который уже был набран клиентов, а также выявлять финансовый риск на основе прошлой финансовой активности.

Обучение с учителем – данный метод в свою очередь будет больше всего подходить для задач, где у нас изначально имеется большое количество подготовленных данных, на основе которых будет порисоваться обучение алгоритма. Они должны быть размечены и подготовлены.

Идеально чистые и данные достать не так просто, особенно если эти данные не находятся в свободном доступе и для их создания требуется проанализировать огромное количество ресурсов. Поэтому часто прибегают к способу обучения без учителя. В способе без учителя машина самостоятельно исследует набор данных. После чего происходит этап выявления скрытыъ закономерностей между переменными.

Для данного типа обучения существует такое понятие как паттерн. Паттерн в свою очередь обрабатывает массивы данных. В следствии после чего данный алгоритм должен выявить закономерность. Важно, что сделать он должен это сам. После чего, на основе выявленных закономерности машина упорядочивает и подготавливает данные. Если мы хотим привести пример для применения обучения в котором не будет участвовать учитель, то нам подойдет алгоритм кластеризации. В свою очередь он будет использоваться для возможного соединения записей. Еще одним ярким примером с обучением в котором не будет учавстовать учитель, это использование его для анализа тональности высказываний.

В зависимости от задач, которые мы даем перед обучением, модель будет систематизировать данные по-разному.

1. Кластеризация: как пример можно привести разделение животных на определенные виды. В данном случае если рассматривать птиц, их можно распределить по группам, опираясь на цвет, размеры и формы клюва. Алгоритм работает похожие данные, находя общие признаки и группирует их.

2. Обнаружение аномалий: как пример можно привести банковскую систему. Например, банк может заметить подозрительные операции, если вы вдруг начнете тратить деньги необычным для вас способом.

3. Ассоциации: самым ярким примером данной систематизации можно назвать рекомендательные подборки в интернет магазинах.

4. Автоэнкодеры принимают данные, после чего их кодируют, а затем воссоздают исходные данные из полученного нами кода. Как пример можно привести возможность распознавания шумов из видео, фотографий, скриншотов, а также из медицинских сканов, для того чтобы повысить качество данных.

В обучении без учителя будет сложно понять точность обученного алгоритма. Это происходит из-за того, что алгоритм обучается самостоятельно, без указания ему правильных ответов, но данный способ дает возможность получить хорошие данные в случае отсутствия качественных данных.

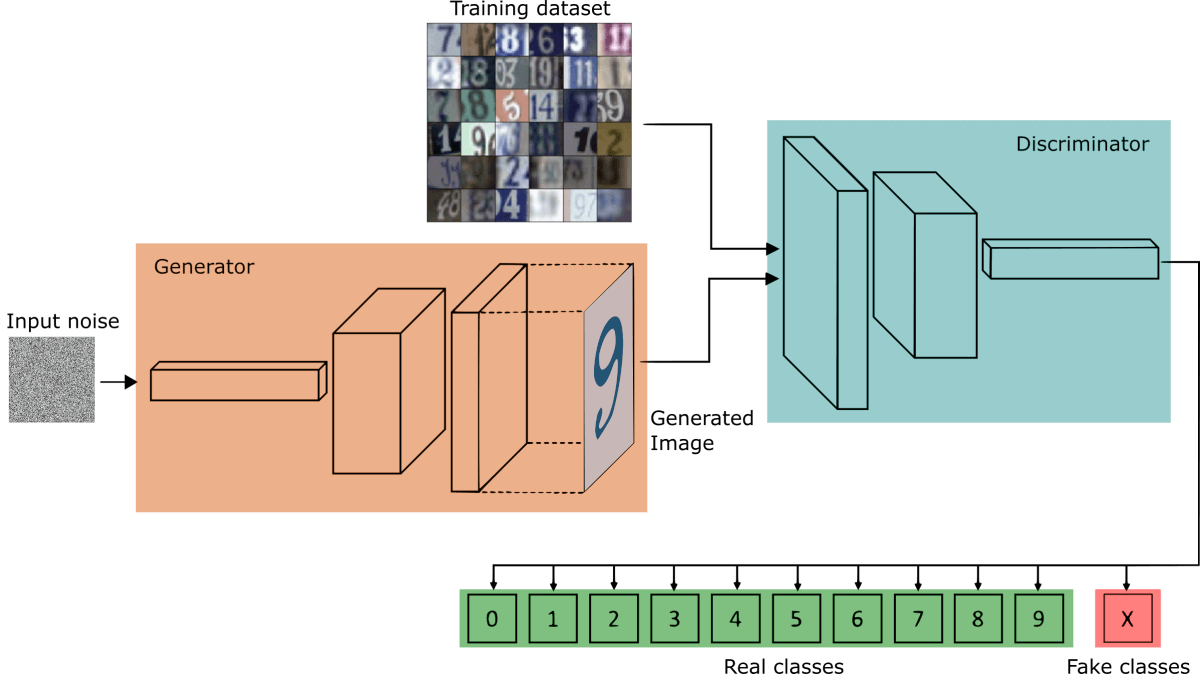
Метод обучения с частичным привлечением учителя в основном зарекомендовал себя в медицинском направлении. Он позволяет анализировать медицинские изображения. Как пример сканы МРТ и компьютерной томографии. Разметка таких сканов вручную при большом наборе данных может быть трудоемкой и дорогостоящей задачей. Нейронная сеть в данном случае может оказать помощь в извлечении информацию основываясь на небольшой доле размеченных данных, а также улучшить точность предсказаний, которая она даст.

Рис. 1 Пример обучения с частичным привлечением учителя

Обучение с подкреплением − это гибрид обучения с учителем и без учителя.

Видеоигры основаны на системе стимулов. Таким образом если вы проходите уровень, вы переходите на следующий или же вы совершаете ошибку и вас возвращает в начало. Таким образом система дает понять игроку, как лучше действовать в следующем раунде.

Учитель предоставляет небольшую часть данных и показывает машине, как следует кластеризовать остальную часть. Принцип этого обучения заключается в поиске оптимального действия. Алгоритм должен самостоятельно обнаружить связи в данных. Этот тип обучения особенно полезен в случаях, когда имеются частично размеченные наборы данных. Он пригодится, когда сложно выделить все важные признаки или размечать все объекты в данных.

# Датафреймы в машинном обучении

Статистика – это направление, которое имеет немало определений и выделяется своей много значимостью. Статистические данные позволяют увидеть полноту происходящего. Она помогает собирать и структурировать информацию, отвечать за корректность, измерять и анализировать данные. Без статистики просто не получится сделать корректный вывод и со всех сторон изучить вопрос.

Данные, которые будут использоваться для обучения машиной, должны быть наполнены достаточным количеством информации, так как от этого будет завесить насколько хорошо будет обучен алгоритм. Также стоит понимать различие наборов данных для контролируемого обучения и неконтролируемого обучения.

В контролируемом обучении люди помогают компьютеру учиться, показывая ему правильные ответы. Они дают компьютеру данные с метками, которые объясняют, что есть что. Например, если нужно научить компьютер распознавать спам, люди предоставляют ему много текстов, и каждый текст помечен как спам или не спам. Таким образом, компьютер понимает, на что обращать внимание, чтобы отличить спам от обычных сообщений.

В неконтролируемом обучении люди дают компьютеру неразмеченные данные, и он сам ищет в них закономерности. Например, можно дать компьютеру много уравнений, и он сам поймёт, какие из них сложные, а какие — нет, находя общие признаки.

Данные можно найти в интернете. Существует огромное количество сервисов по сбору данных. Для примера приведу “Федеральную службу государственной статистики”. Тут можно найти данные за последние года и месяца, по различным направлениям.

Что подразумевается под качественными данными для обучения. Они должны содержать в себе 4 характеристики:  
 1. Релевантность – набор данных должен содержать только те признаки, которые дадут модели значимую информацию. Стоит понимать, какие данные стоит учитывать для обучения, а какие стоит устранять.

2. Постоянство – схожие параметры должны иметь схожие метки, все данные должны быть однородным набором данных.

3. Однородность – значение всех атрибутов должны быть сравнимы для всех данных. Неравномерности могут плохо сказаться на обучаемости модели.

4. Полнота – набор данных должен быть наполнен достаточным количеством параметров и признаков.

Очистка данных является одной из важнейших задач при подготовке данных для обучения. Это процесс исправления и удаления неправильных, дублированных, а также поврежденных данных в пределах набора данных.

Рекомендации при очистке данных:

1. Проверять наличие дубликатов – так как хорошая модель для обучения собирается из огромного источника данных, не исключено присутствие одних и тех же данных. Их необходимо удалить, так как они могут плохо повлиять на обучение модели.

2. Устранение выбросов – некоторые части данных могут вести себя не так, как остальные. Это может случаться из-за SessionID. Поэтому слежение за выбросами, это одна из рекомендаций при устранении данных, неподходящих для обучения.

3. Исправление структурных данных – в некоторых случаях в наборе может присутствовать ошибочная разметка.

4. Проверка на отсутствующие элементы – в наборе данных могут быть пустые ячейки, такие данные, лучше не использовать для обучения. Так как неполнота действие плохо влияет на обучение.

Разметка данных – это процесс, во время него мы присваиваем данным значения в виде классов и меток. Нельзя использовать слишком много меток, так как потребуется анализа большего количества данных.

Таким образом разметка данных выделяет признаки (характеристики) данных, чтобы помочь модели анализировать данные и идентифицировать паттерны в исторических данных для точных предсказаний будущих.

# Машинное обучение для прогнозирования цен акций

# Подготовка среды разработки

В задаче машинного обучения используется 2 способа получения исходных данных, так как она может брать исходные данные напрямую с сайта finance.yahoo, а также может брать данные из файла формата \*.csv, в обоих случаях данные являются полностью заполненными базами данных. Сделано было 2 способа для удобства использования пользователем, так как при работе напрямую с finance.yahoo, требуется стабильное и хорошее подключение к интернету, в то время как при работе с файлом формата \*.csv интернет не требуется.

Перед началом работы стоял вопрос по выбору языка программирования, и он остановился на языке программирования Python.

Библиотеки, которые были использованные при создании программы:

1. Yfinance – данная библиотека позволяет получать исторические данные по акциям напрямую с сайта finance.yahoo [1];
2. Pandas – библиотека позволяющая работать с табличными данными,

используется для обработки и анализа данных, а также чтения CSV файлов [2];

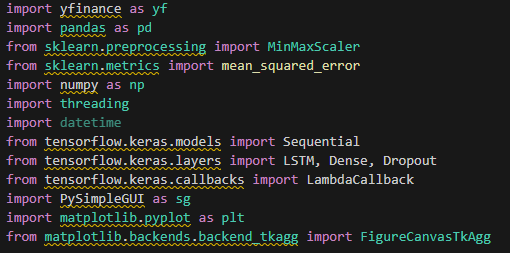
1. Sklearn – библиотека из которой будут использованы такие инструменты как – MinMaxScaler для масштабирования данных и mean\_squared\_error для оценки ошибки модели на тестовых данных [5].
2. Numpy – библиотека для работы с массивами, а так же числовыми вычислениями, используется для обработки данных и создания наборов данных [3].
3. Threading – библиотека для работы с потоками. Так как обучение модели может занимать длительное время, было принято решение об использовании отдельных потоков для интерфейса программы и работы с обучением модели [4].
4. Datetime – модуль для работы с датами и временем, использовался для обработки и манипулированием датами.
5. Tensorflow – из данной библиотеки были использованы сразу несколько важных аспектов, такие как Sequential, LSTM, Dense, Dropout, LambdaCallback.
6. PySimpleGUI – данная библиотека была использована для создания графического интерфейса программы.
7. Matplotlib.pyplot – библиотека которая была использована для визуализации данных, создания графиков [7].

Рис. 2 Подключенные библиотеки для корректной работы программы

# Создание графического интерфейса

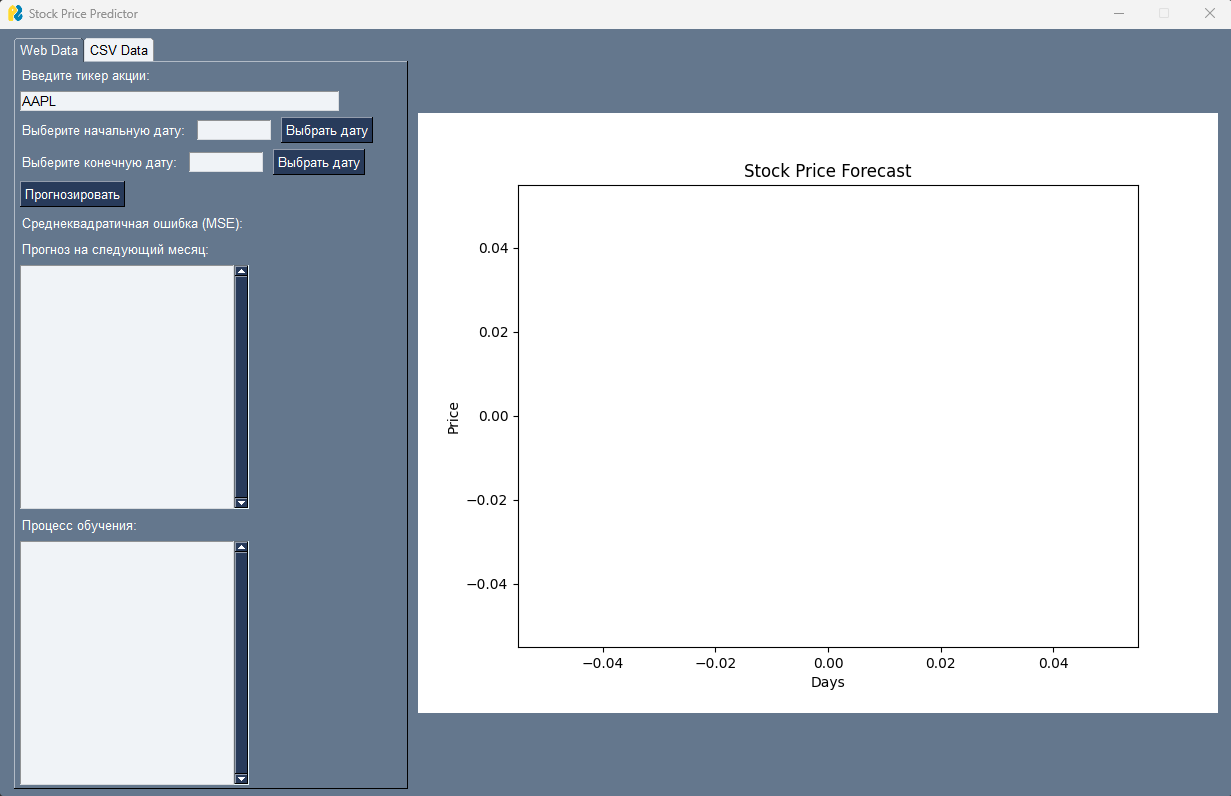
При запуске программы нас встречает графический интерфейс, который был создан при помощи PySimpleGUI.

Рис. 3 Внешний вид программы вкладки Web Data

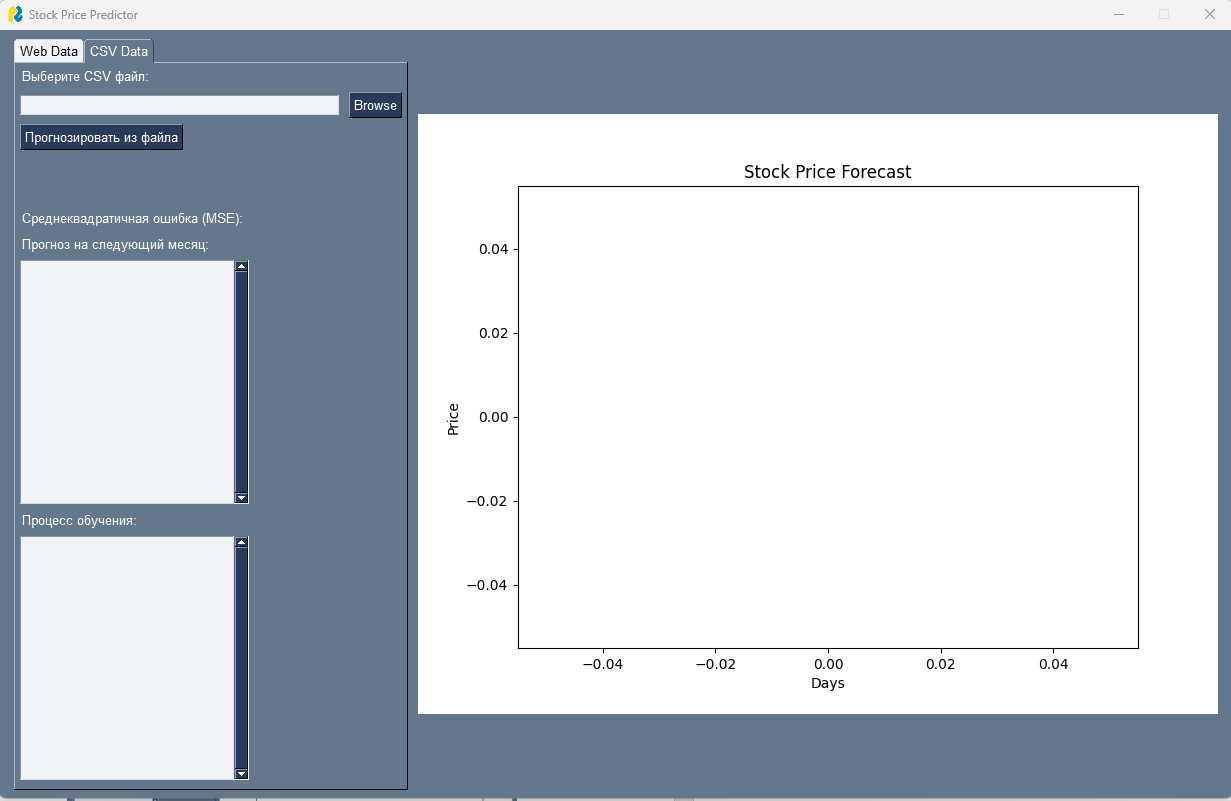
 У нас есть 2 вкладки и одно общее пространство, первая вкладка отвечает за работу с данными через интернет, а именно для работы с сайтом finance.yahoo через библиотеку Yfinance, вторая вкладка отвечает за работу с локальными файлами формата \*.csv.

Рис. 3 Внешний вид программы вкладки CSV Data

Для обоих вкладок было реализовано прогнозирование на следующий месяц в числовом формате, отображение процесса обучения в режиме реального времени, а так же графическое отображение результатов.

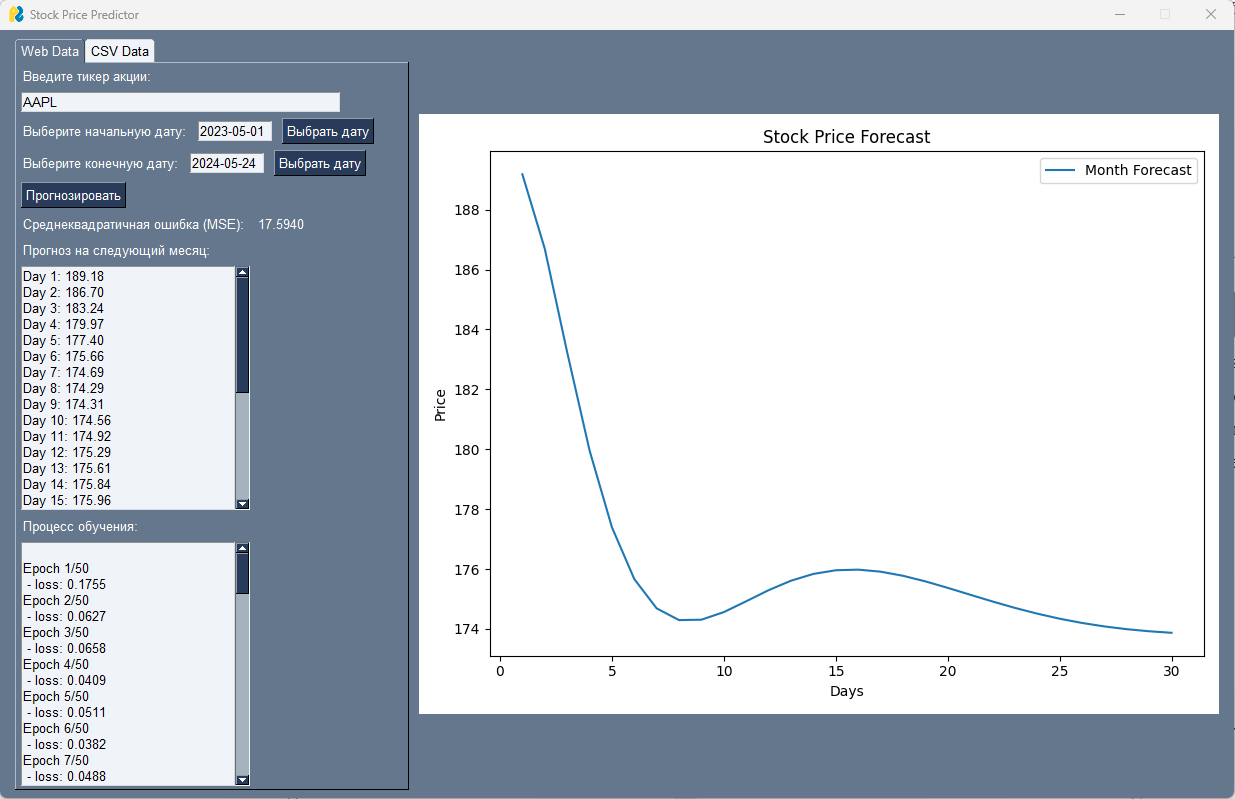


Рис. 4 Демонстрация работы программы через графический интерфейс

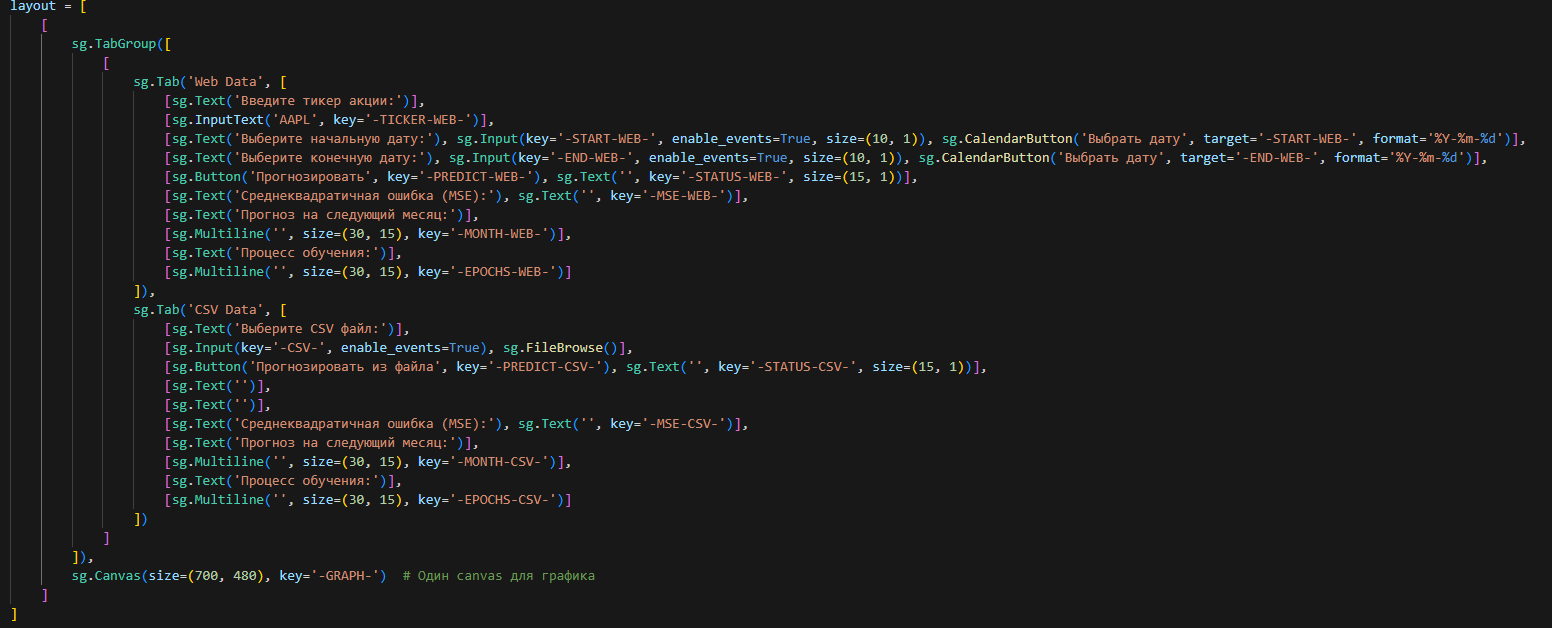
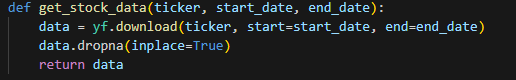
Для реализации данного графического интерфейса был написан вот такой код для расположения всех элементов управления.

Рис. 5 Код для реализации графического интерфейса

# Подготовка датафрейма

В данной работе используются данные с finance.yahoo [1], наборы данных на данном ресурсе находятся в полном обьеме и значений NaN в исходных данных отсутствуют, в свою очередь это позволяет использовать качественный набор данных.

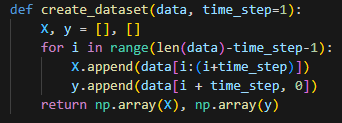
Рис. 6 Функция для получения данных акций

Рис. 7 Функция для создания набора данных

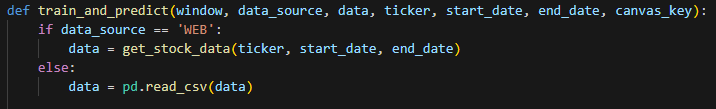
В моем коде функция get\_stock\_data отвечает за получение данных акций, здесь используется библиотека yfinance для загрузки данных через интернет.  
 В случае выбора опции CVS data данные будут загружаться через pd.read\_csv

Рис. 8 Чтение данных через pd.read\_csv

Теперь перейдем к предобработке данных, для начала нам нужно определить какие признаки мы будем использовать для обучения нашей модели, я использую Close, Open, High, Low, Volume. Такое количество необходимо для более лучших результатов предсказаний цен на акции.

# 

Рис. 9 Предобработка данных

# Выбор метода для машинного обучения

Я остановил свой выбор на LSTM так как он отлично подходит для работы с последовательными данными, например для работы с временными рядами, что в свою очередь является типичной задачей при прогнозировании цен акций.

Преимущества, основываясь на которые был сделан выбор:

1. Умение улавливать долгосрочные зависимости. Так как в данном способе присутствует скруктуре работы с воротами, это позволяет модели учитывать информацию о прошлых временных шагах.
2. Работа с последовательными данными, так же является особо важной при работе с ценами на акции, так как при работе c ценами на акции учитываются все временные промежутки.
3. Устойчивость к затуханию градиентов. В LSTM есть механизмы, которые позволяют более эффективно обучаться на длинных последовательных данных и решает проблему с затуханием градиентов.
4. Гибкость в управлении памятью, в LSTM есть такие механизмы как “Забывание” и “Запоминание”, которые позволяют лучше выбирать какую информацию стоит хранить в памяти, а какие факторы будут нам не нужны и которые можно не запоминать.

Подводя итоги выбора метода обучения, я решил однозначно остановиться на модели LSTM, так как это мощный инструмент для анализа временных рядов и прогнозирования.

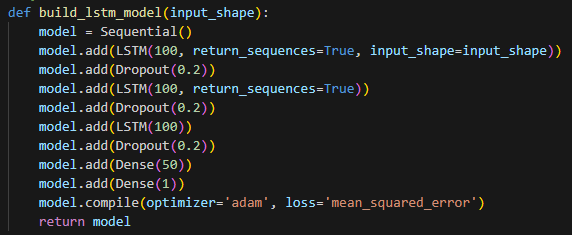
Рис. 10 Функция создания модели LSTM



Рис. 11 Создание тестового набора данных

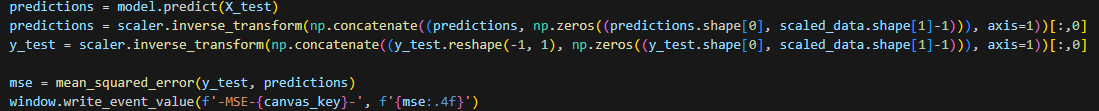


Рис. 12 Прогнозирование и оценка модели

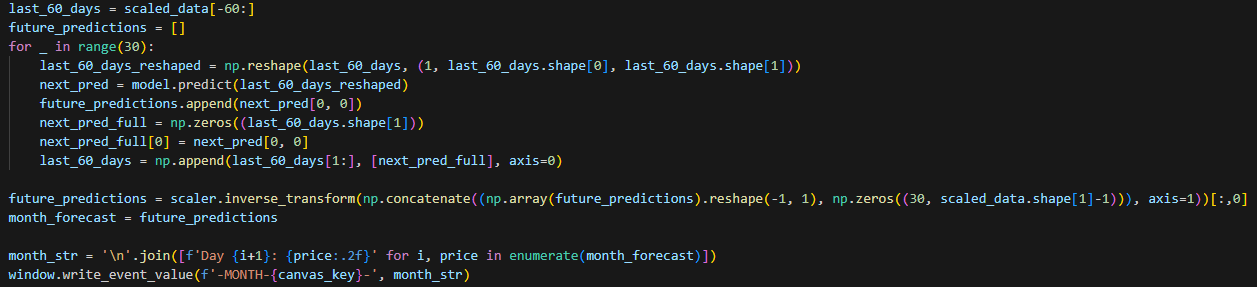


Рис. 13 Прогнозирование цен на месяц

# Оптимизация работы приложения

# 3.1 Внедрение многопоточности

Что такое многопоточность и зачем она нужна? Многопоточность – это один из инструментов управления одновременным выполнением нескольких задач в программе.

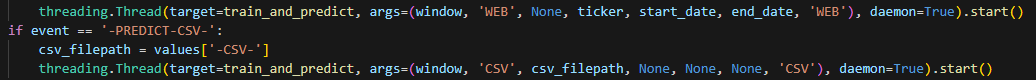
Для ускорения работы программы, было принято решение о внедрении технологии использования многопоточности, в данной курсовой работе данная технология используется в двух местах, первое она позволяет ускорить процесс обучения тестовой модели, второе она позволяет использовать интерфейс программы вне зависимости от обучения машины.

Рис 14. События с использованием многопоточности

# Графики результатов

Для работы с графиками и использовании их в графическом интерфейсе своей программы, была реализована отрисовка пустого графика в самом начале, для того чтобы PySimpleGUI выделил определенное место для графика. Этот способ позволяет работать с уже заранее отрисованным пустым графиком и обновлять его в зависимости от данных, которые мы получаем при обучении машины. Также у данного способа есть большой плюс, так как график является универсальным, и нам не приходится отрисовывать его каждый раз с нуля, мы просто обновляем уже существующий.

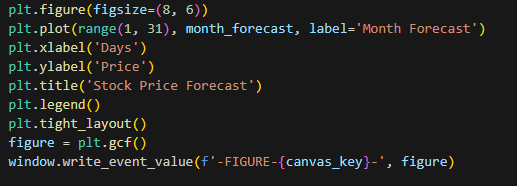
Для работы с графиками была выбрана библиотека Matplotlib. Данная библиотека — это пакет для визуализации данных. Данная библиотека позволяет строить графики любых сложностей, но в нашем случае мне нужно было построить график, который будет отображать динамику цен. 

Рис. 15 Визуализация прогнозов

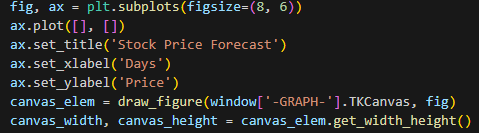
Рис. 16 Использования пространства в layout

Рис. 17 Создание пустого графика при запуске приложения

# Удобство отслеживания процесса обучения

Для удобства отслеживания процесса обучения машины, я добавил на графический интерфейс окно, которое отображает количество операций обучение а так же loss для каждой из операций.

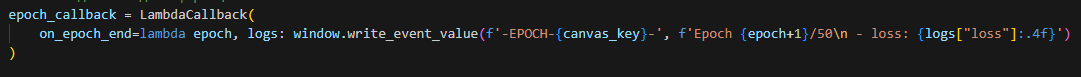


Рис. 18 Использование LabdaCallBack для отслеживания



Рис. 19 Использование Epoch CSV для вывода в layout



Рис. 20 Использование Epoch WEB для вывода в layout

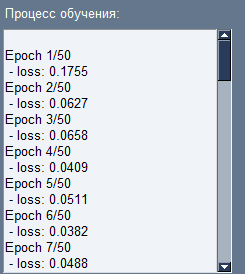


Рис. 21 Отображение процесса обучение в текстовом формате

# Практическое обоснование результатов исследования в предметной области

На практике данные результаты могут быть использованы инвесторами для подбора наиболее выгодных для вложений акций, а также для более подробного анализа рынка на предмет недооцененных акций и переоценённых акций.

Графа с текстовыми значениями цен акций, позволят получить быстрый доступ к данным, которые мы смогли получить в результате обучения нашей машины, а график позволит отслеживать динамику цен наглядным способом.

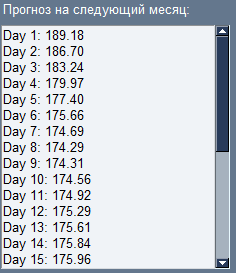


Рис. 22 Результаты работы в текстовом формате

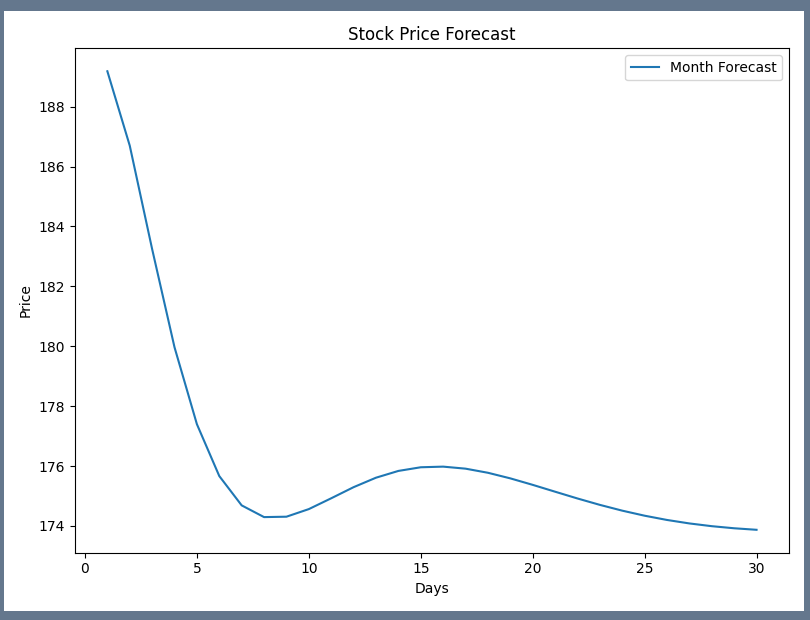


Рис. 23 Результаты работы в графическом формате

# ЗАКЛЮЧЕНИЕ

В данной работе был продемонстрирован принцип работы машинного обучения для предсказания цен на стоимость акций на месяц вперед, основываясь на данных предыдущих готов, использование графического интерфейса, позволяющий красиво, удобно и наглядно продемонстрировать работу машинного обучения, а так же использование многопоточности для ускорения работы машинного обучения и использований программы параллельно с обучением модели.

Чтобы получить хорошие результаты, был выбран наиболее подходящий метод обучения для этой задачи, учитывая его особенности. Для выполнения этой задачи использовали следующие методы, которые позволяют в режиме реального времени отслеживать колебание цен в пределах одного дня, мы так же использовали преобразование данных, обучение машины, получение результатов, а также построение графиков.

Необходимость работы с машинным обучением обусловлена огромным количеством данных, которые собирались за длительный промежуток времени, и продолжают собираться по сей день.

В данной работе были изучены основы машинного обучение, а именно:

1. виды машинного обучения;

2. задачи машинного обучения;

3. сферы применения;

4. используемые алгоритмы.

Был проведен анализ данных цен на акции различных компаний, на основе данных finance.yahoo.

Результатом работы является программа с графическим интерфейсом позволяющая работать с ценными бумаги, на основе машинного обучения при помощи LSTM модели. Данная программа позволяет инвесторам более подробно анализировать и делать выводы о том, какие акции стоит продавать, а какие акции стоит покупать. Помимо того, что данная программа будет полезна инвесторам при выборе ценных бумаг для покупки и продажи, данная программа будет также полезна и для самих компаний, чьи акции будут анализироваться, так как это даст общее понятие о том, в каком направлении им стоит развиваться, а в каком нет, так как все эти данные влияют на цену акций.

Почему же машинное обучение так важно в наше время. Ручным перебором данных за такой короткий промежуток времени нецелесообразен, когда существуют настолько быстрые способы более качественно проанализировать большие объёмы данных.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ И ИНТЕРНЕТ-РЕСУРСОВ

1. Сайт для отслеживания цен на акции за длительный промежуток времени. URL:<https://finance.yahoo.com/>
2. Документация по работе с библиотекой Pandas. URL: <https://pandas.pydata.org/>
3. Документация по работе с библиотекой NumPy. URL: <https://numpy.org/>
4. Документация по работе с библиотекой threading. URL:<https://docs-python.ru/standart-library/modul-threading-python/>
5. Рашка С., Мирджали В. Python и машинное обучение: машинное и глубокое обучение с использованием Python, scikit-learn и TensorFlow 2, 3-е изд.: Пер. с англ. – СПб. :ООО “Диалектика”, 2020.
6. Подготовка данных для алгоритмов машинного обучения. URL: <http://blog.datalytica.ru/2018/04/blog-post.html>
7. Документация по работе с библиотекой matplotlib.pyplot. URL: <https://matplotlib.org/3.5.3/api/_as_gen/matplotlib.pyplot.html>

# ПРИЛОЖЕНИЕ

Технические характеристики компьютера, который был использован для обучения машины:

1. Процессор: AMD Ryzen 7 5700x
2. Объём оперативной памяти: 32gb

Код программы:   
import yfinance as yf

import pandas as pd

from sklearn.preprocessing import MinMaxScaler

from sklearn.metrics import mean\_squared\_error

import numpy as np

import threading

import datetime

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import LSTM, Dense, Dropout

from tensorflow.keras.callbacks import LambdaCallback

import PySimpleGUI as sg

import matplotlib.pyplot as plt

from matplotlib.backends.backend\_tkagg import FigureCanvasTkAgg

# Функция для получения данных акции

def get\_stock\_data(ticker, start\_date, end\_date):

data = yf.download(ticker, start=start\_date, end=end\_date)

data.dropna(inplace=True)

return data

# Функция для создания набора данных

def create\_dataset(data, time\_step=1):

X, y = [], []

for i in range(len(data)-time\_step-1):

X.append(data[i:(i+time\_step)])

y.append(data[i + time\_step, 0])

return np.array(X), np.array(y)

# Функция для создания модели LSTM

def build\_lstm\_model(input\_shape):

model = Sequential()

model.add(LSTM(100, return\_sequences=True, input\_shape=input\_shape))

model.add(Dropout(0.2))

model.add(LSTM(100, return\_sequences=True))

model.add(Dropout(0.2))

model.add(LSTM(100))

model.add(Dropout(0.2))

model.add(Dense(50))

model.add(Dense(1))

model.compile(optimizer='adam', loss='mean\_squared\_error')

return model

# Функция для отображения графика на интерфейсе

def draw\_figure(canvas, figure):

figure\_canvas\_agg = FigureCanvasTkAgg(figure, canvas)

figure\_canvas\_agg.draw()

figure\_canvas\_agg.get\_tk\_widget().pack(side='top', fill='both', expand=1)

return figure\_canvas\_agg

def train\_and\_predict(window, data\_source, data, ticker, start\_date, end\_date, canvas\_key):

if data\_source == 'WEB':

data = get\_stock\_data(ticker, start\_date, end\_date)

else:

data = pd.read\_csv(data)

if not data.empty:

# Предобработка данных

data = data[['Close', 'Open', 'High', 'Low', 'Volume']]

scaler = MinMaxScaler(feature\_range=(0, 1))

scaled\_data = scaler.fit\_transform(data)

training\_data\_len = int(np.ceil(len(scaled\_data) \* .8))

train\_data = scaled\_data[0:int(training\_data\_len), :]

X\_train, y\_train = create\_dataset(train\_data, time\_step=60)

X\_train = np.reshape(X\_train, (X\_train.shape[0], X\_train.shape[1], X\_train.shape[2]))

# Создание и обучение модели

model = build\_lstm\_model((X\_train.shape[1], X\_train.shape[2]))

# Колбэк для вывода информации об эпохах

epoch\_callback = LambdaCallback(

on\_epoch\_end=lambda epoch, logs: window.write\_event\_value(f'-EPOCH-{canvas\_key}-', f'Epoch {epoch+1}/50\n - loss: {logs["loss"]:.4f}')

)

model.fit(X\_train, y\_train, batch\_size=64, epochs=50, verbose=0, callbacks=[epoch\_callback])

# Создание тестового набора данных

test\_data = scaled\_data[training\_data\_len - 60:, :]

X\_test, y\_test = create\_dataset(test\_data, time\_step=60)

X\_test = np.reshape(X\_test, (X\_test.shape[0], X\_test.shape[1], X\_test.shape[2]))

# Прогнозирование и оценка модели

predictions = model.predict(X\_test)

predictions = scaler.inverse\_transform(np.concatenate((predictions, np.zeros((predictions.shape[0], scaled\_data.shape[1]-1))), axis=1))[:,0]

y\_test = scaler.inverse\_transform(np.concatenate((y\_test.reshape(-1, 1), np.zeros((y\_test.shape[0], scaled\_data.shape[1]-1))), axis=1))[:,0]

mse = mean\_squared\_error(y\_test, predictions)

window.write\_event\_value(f'-MSE-{canvas\_key}-', f'{mse:.4f}')

# Прогноз на месяц

last\_60\_days = scaled\_data[-60:]

future\_predictions = []

for \_ in range(30):

last\_60\_days\_reshaped = np.reshape(last\_60\_days, (1, last\_60\_days.shape[0], last\_60\_days.shape[1]))

next\_pred = model.predict(last\_60\_days\_reshaped)

future\_predictions.append(next\_pred[0, 0])

next\_pred\_full = np.zeros((last\_60\_days.shape[1]))

next\_pred\_full[0] = next\_pred[0, 0]

last\_60\_days = np.append(last\_60\_days[1:], [next\_pred\_full], axis=0)

future\_predictions = scaler.inverse\_transform(np.concatenate((np.array(future\_predictions).reshape(-1, 1), np.zeros((30, scaled\_data.shape[1]-1))), axis=1))[:,0]

month\_forecast = future\_predictions

month\_str = '\n'.join([f'Day {i+1}: {price:.2f}' for i, price in enumerate(month\_forecast)])

window.write\_event\_value(f'-MONTH-{canvas\_key}-', month\_str)

# Визуализация прогнозов

plt.figure(figsize=(8, 6))

plt.plot(range(1, 31), month\_forecast, label='Month Forecast')

plt.xlabel('Days')

plt.ylabel('Price')

plt.title('Stock Price Forecast')

plt.legend()

plt.tight\_layout()

figure = plt.gcf()

window.write\_event\_value(f'-FIGURE-{canvas\_key}-', figure)

# Определение макета интерфейса

layout = [

[

sg.TabGroup([

[

sg.Tab('Web Data', [

[sg.Text('Введите тикер акции:')],

[sg.InputText('AAPL', key='-TICKER-WEB-')],

[sg.Text('Выберите начальную дату:'), sg.Input(key='-START-WEB-', enable\_events=True, size=(10, 1)), sg.CalendarButton('Выбрать дату', target='-START-WEB-', format='%Y-%m-%d')],

[sg.Text('Выберите конечную дату:'), sg.Input(key='-END-WEB-', enable\_events=True, size=(10, 1)), sg.CalendarButton('Выбрать дату', target='-END-WEB-', format='%Y-%m-%d')],

[sg.Button('Прогнозировать', key='-PREDICT-WEB-'), sg.Text('', key='-STATUS-WEB-', size=(15, 1))],

[sg.Text('Среднеквадратичная ошибка (MSE):'), sg.Text('', key='-MSE-WEB-')],

[sg.Text('Прогноз на следующий месяц:')],

[sg.Multiline('', size=(30, 15), key='-MONTH-WEB-')],

[sg.Text('Процесс обучения:')],

[sg.Multiline('', size=(30, 15), key='-EPOCHS-WEB-')]

]),

sg.Tab('CSV Data', [

[sg.Text('Выберите CSV файл:')],

[sg.Input(key='-CSV-', enable\_events=True), sg.FileBrowse()],

[sg.Button('Прогнозировать из файла', key='-PREDICT-CSV-'), sg.Text('', key='-STATUS-CSV-', size=(15, 1))],

[sg.Text('')],

[sg.Text('')],

[sg.Text('Среднеквадратичная ошибка (MSE):'), sg.Text('', key='-MSE-CSV-')],

[sg.Text('Прогноз на следующий месяц:')],

[sg.Multiline('', size=(30, 15), key='-MONTH-CSV-')],

[sg.Text('Процесс обучения:')],

[sg.Multiline('', size=(30, 15), key='-EPOCHS-CSV-')]

])

]

]),

sg.Canvas(size=(700, 480), key='-GRAPH-') # Один canvas для графика

]

]

window = sg.Window('Stock Price Predictor', layout, finalize=True)

# Создаем пустой график при запуске

fig, ax = plt.subplots(figsize=(8, 6))

ax.plot([], [])

ax.set\_title('Stock Price Forecast')

ax.set\_xlabel('Days')

ax.set\_ylabel('Price')

canvas\_elem = draw\_figure(window['-GRAPH-'].TKCanvas, fig)

canvas\_width, canvas\_height = canvas\_elem.get\_width\_height()

while True:

event, values = window.read()

if event == sg.WINDOW\_CLOSED:

break

if event == '-PREDICT-WEB-':

ticker = values['-TICKER-WEB-']

start\_date = values['-START-WEB-']

end\_date = values['-END-WEB-']

threading.Thread(target=train\_and\_predict, args=(window, 'WEB', None, ticker, start\_date, end\_date, 'WEB'), daemon=True).start()

if event == '-PREDICT-CSV-':

csv\_filepath = values['-CSV-']

threading.Thread(target=train\_and\_predict, args=(window, 'CSV', csv\_filepath, None, None, None, 'CSV'), daemon=True).start()

if event.startswith('-EPOCH-'):

canvas\_key = event.split('-')[2]

current\_text = values[f'-EPOCHS-{canvas\_key}-']

window[f'-EPOCHS-{canvas\_key}-'].update(current\_text + '\n' + values[event] + '\n')

if event.startswith('-MSE-'):

canvas\_key = event.split('-')[2]

window[f'-MSE-{canvas\_key}-'].update(values[event])

if event.startswith('-MONTH-'):

canvas\_key = event.split('-')[2]

window[f'-MONTH-{canvas\_key}-'].update(values[event])

if event.startswith('-FIGURE-'):

figure = values[event]

for child in window['-GRAPH-'].TKCanvas.winfo\_children():

child.destroy()

canvas\_elem = draw\_figure(window['-GRAPH-'].TKCanvas, figure)

window.close()