1.1.

ФИО разработчиков:

Голышев Пётр Владимирович, Салищева Руслана Сергеевна.

1.2.

Д-Э342

1.3. Тема проекта:

"Анализ финансовых временных рядов и прогнозирование на Python"

1.4. Цель проекта

Исследовать и продемонстрировать возможности и ограничения инструментов и методов языка программирования Python для анализа и моделирования исторических открытых данных цен акций, а также оценить, какие подходы обеспечивают более надёжные прогнозы и анализ временного ряда.

1.5. Задачи проекта:

Выполнить загрузку и предобработку исходных данных: очистка пропусков и нулей, коррекция типов, возможное устранение аномалий.

Провести первичный исследовательский анализ: визуализировать временные ряды, распределения, тренды; построить корреляционную матрицу для оценки взаимосвязей признаков.

Разработать и оценить модели регрессии (линейную и множественную), чтобы проверить, насколько хорошо можно “восстановить” цену по доступным признакам и оценить силу линейных зависимостей.

Построить модели классификации (например, дерево решений, KNN), чтобы попытаться предсказать направление движения цены (рост/падение) и оценить их качество.

Провести кластерный анализ (K-Means +, при необходимости, проекция (PCA) / визуализация), чтобы сгруппировать дни по схожим характеристикам и попытаться выявить “типы рыночных состояний”.

Использовать методы нейронных сетей для регрессии (MLP) -

попробовать более гибкий, нелинейный подход к прогнозированию цены, оценить, насколько он даёт результат лучше простых моделей.

1.6. Аннотация:

Проект посвящён анализу исторических данных по ценам акций (OHLCV + Adjusted Close + объём) за период 2004-2024 для компании GOOGLE, с целью продемонстрировать возможности инструментария языка программирования Python для работы с открытыми финансовыми данными. В работе выполнена предобработка набора данных (удаление пропусков и аномалий, корректировка типов, фильтрация нулевых значений), проведён исследовательский анализ - визуализация, корреляционный анализ, регрессионное и кластерное моделирование, а также классификация и прогнозирование с помощью машинного обучения. В качестве методов использованы простая и множественная линейная регрессия, классификация (дерево решений, K-Nearest Neighbors), кластеризация (K-Means), а также обучение нейронной сети (MLPRegressor) для прогнозирования цены. Выполнен сравнительный анализ эффективности различных моделей и подходов, оценка качества их работы, даны выводы о том, какие методы дают адекватные результаты, а какие - показывают низкую стабильность. Проект иллюстрирует, как с помощью Python и библиотеки для анализа данных и машинного обучения можно проводить полный цикл исследования финансового временного ряда - от загрузки и очистки данных до построения моделей и интерпретации результатов.

2. Предобработка данных.

Предобработка данных - обязательный этап перед анализом и построением моделей, особенно важный при работе с финансовыми временными рядами. В исходном CSV-файле данные могут быть представлены в разных форматах, содержать пропуски, некорректные записи, строки с "нулевыми" или отсутствующими значениями. Поэтому перед любым анализом необходимо:

- привести колонки цен и объёма к числовому формату,
- убедиться, что столбец дат распознан корректно,
- отсортировать данные по времени,
- удалить записи с пропусками или некорректными данными,
- рассчитать необходимые дополнительные признаки (например, доходность, направление движения), и отбросить строки, где эти признаки не могут быть рассчитаны.

В рамках проекта был реализован следующий pipeline: загрузка CSV, конвертация колонок Open, High, Low, Close, Adj Close, Volume к числовому типу (с заменой десятичного разделителя, если нужно), преобразование даты в формат datetime, сортировка по дате, удаление строк с пропущенными или нулевыми ценами, удаление пустых строк, вычисление доходности и метки направления, удаление строк с NaN после вычисления, и дополнительная фильтрация строк с нулевой ценой. В результате получен очищенный, корректный и упорядоченный набор данных, пригодный для визуализации, статистического анализа и построения моделей машинного обучения.

3. Визуализация данных.
Зачем нужна визуализация

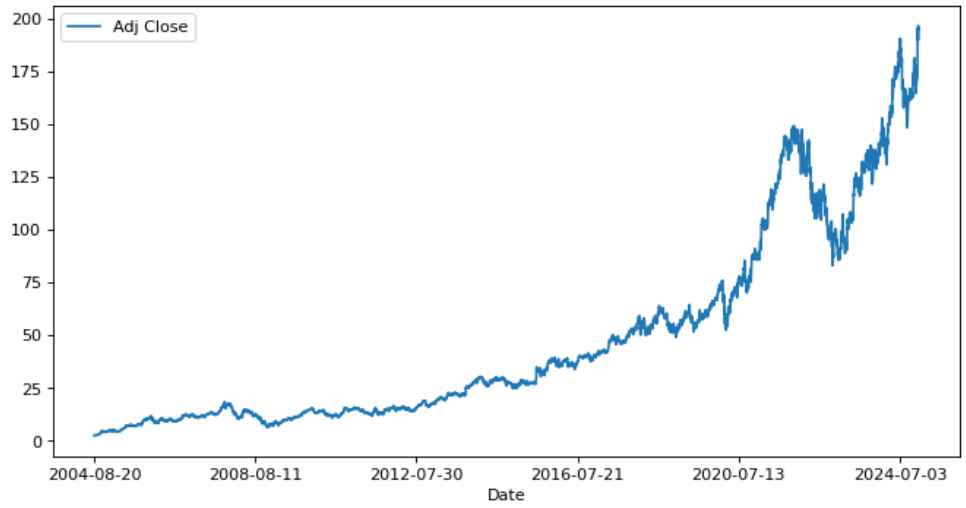
- наглядно проследить динамику цены во времени;
- увидеть тренды, всплески, аномалии, периоды роста/падения;
- понять распределения, вариативность, “шумность” данных;
- оценить, насколько корректно работают модели (например, обнаружить, где прогноз отличается от фактической цены) - это часто даёт лучшее понимание, чем просто метрики;
- с помощью разных типов графиков (линейные, scatter, гистограммы и др.) - проанализировать разные аспекты данных (временная динамика, взаимосвязи, распределения).

Что было сделано: виды визуализаций и задачи, которые они решают

В рамках проекта использованы следующие графики / виды визуализации:

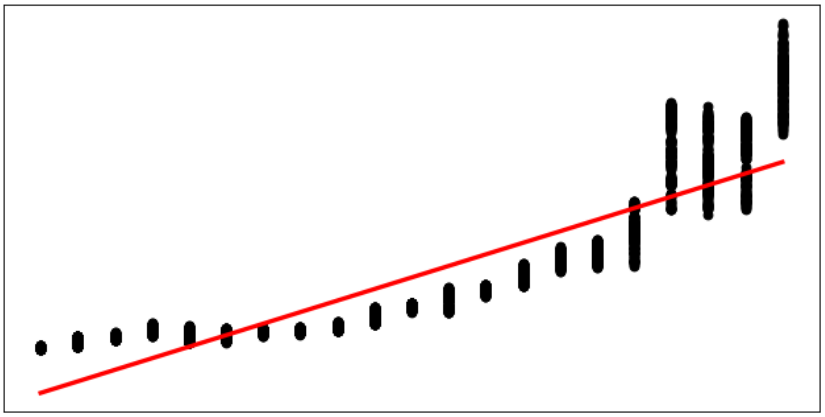
1. График временного ряда - цена **Adj Close** vs дата
2. df_filtered.plot(x="Date y="Adj Close")
3. plt.show()

   Такой график демонстрирует, как менялась цена акции с течением времени: тренды, рост, падения, периоды стабилизации. Это базовая визуализация, которая даёт общее представление о поведении акций за весь рассматриваемый период.



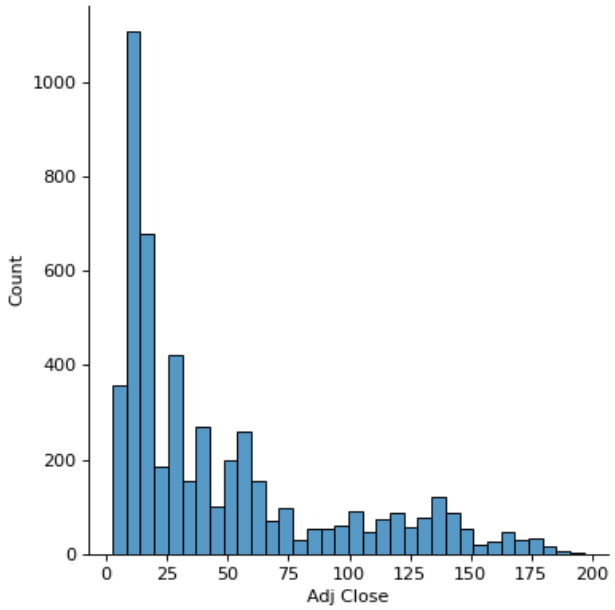
4. Scatter + линейная регрессия: “год vs цена”
5. plt.scatter(x, y, color='black')
6. plt.plot(x, regr.predict(x), color='red', linewidth=3)
7. plt.show()

   Этот график показывает “глобальный” тренд: насколько с течением лет цена увеличивается (либо уменьшается), какова зависимость “год - цена”. Линейная регрессия даёт визуальную аппроксимацию тренда.



8. Гистограмма (распределение цены закрытия)
9. sns_plot = sns.displot(df_filtered['Adj Close'])
10. plt.show()

Этот график помогает понять распределение цен: например, есть ли “обычный” диапазон, насколько часто цена была на определённых уровнях, есть ли “хвосты” (редкие, но очень высокие или низкие значения) - это важно, чтобы оценить волатильность, устойчивость, “шум” на рынке.



3.1. Корреляционный анализ.

Теория: корреляция, корреляционная матрица

Корреляция - это мера линейной зависимости между двумя числовыми переменными. Чаще всего используется коэффициент корреляции Пирсона (Pearson r), который может быть от –1 до +1.

1. r = +1 - идеальная положительная линейная зависимость (если X растёт - Y растёт пропорционально),
2. r = –1 - идеальная отрицательная линейная зависимость (если X растёт - Y убывает),
3. r = 0 - почти нет линейной зависимости.

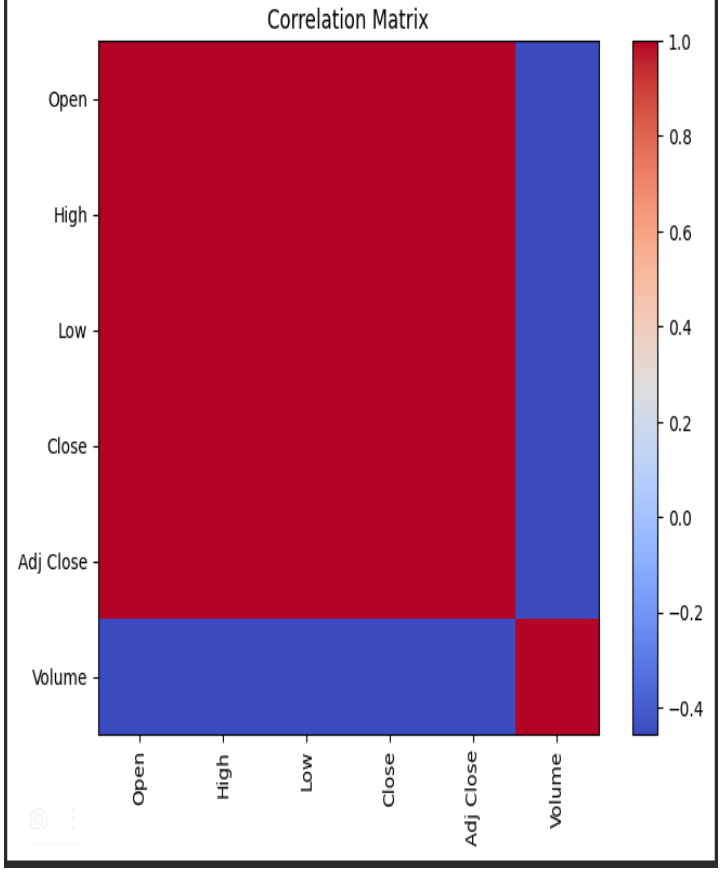
Корреляционная матрица - это таблица r-коэффициентов для всех пар переменных. Если у тебя n числовых переменных, матрица будет n×n, диагонали - 1 (каждая переменная идеально коррелирует сама с собой).

Вывод:

Переменные Open, High, Low, Close, Adj Close - все почти идеально положительно коррелированы (коэффициент ~ 0.9999). Это логично: открытия, закрытия, максимумы и минимумы цены тесно связаны.

При этом **Volume (объём торгов)** имеет отрицательную корреляцию (~ –0.455) с остальными ценовыми признаками. Вероятно,: были периоды с высоким объёмом при падении цены - либо наоборот, так, что объём и цена движутся в противоположных направлениях.

	Open	High	Low	Close	Adj Close	Volume
Open	1.000000	0.999912	0.999903	0.999796	0.999795	-0.455516
High	0.999912	1.000000	0.999892	0.999905	0.999904	-0.454678
Low	0.999903	0.999892	1.000000	0.999917	0.999916	-0.456546
Close	0.999796	0.999905	0.999917	1.000000	0.999999	-0.455724
Adj Close	0.999795	0.999904	0.999916	0.999999	1.000000	-0.455606
Volume	-0.455516	-0.454678	-0.456546	-0.455724	-0.455606	1.000000



Корреляционный анализ - хороший первый шаг при знакомстве с данными: помогает увидеть структуру, зависимость переменных, выбрать, какие признаки оставить, а какие - исключить или трансформировать.

3.2. Регрессионный анализ (линейная регрессия, множественная линейная регрессия).

3.2.1 Простая линейная регрессия

Теория: линейная регрессия, R^2, MSE

Линейная регрессия - метод, который пытается **связать зависимую переменную** (Y) с одной (или несколькими) независимыми (X), предполагая линейную связь.

Если регрессия "простая" (один X), то квадрат корреляции (r^2) между X

Некоторые методы из кода:

- coef_ - на сколько (в среднем) меняется Adj Close, если год увеличивается на 1 - наклон линии.
- intercept_ - значение, которое modely бы было, когда Year = 0 (теоретический).
- R^2 - доля вариации цены, объяснённая годом.
- MSE - средняя ошибка прогноза.

Вывод

- Тренд роста цены с годами: линия регрессии (наклон ~ +6.856) показывает, что в среднем с каждым годом цена Adj Close растёт.
- Неплохой R2=0.78R^2, 0.78R2=0.78 - это значит, ~78% вариации цены объясняется годом. То есть в долгосрочной перспективе есть выраженная тенденция роста.

Регрессия полезна для макро-тенденции (долгосрок), но не для прогноза точных цен.

```
Coef: 6.853892912552709
Intercept: -13758.246877554413
R^2: 0.7798969871124509
MSE: 457.1579638009545
```

3.2.2 Множественная линейная регрессия

Зачем нужна: позволяет понять, как разные признаки вместе влияют на целевую переменную, оценить "вес" каждого признака, понять, есть ли линейная связь в многомерном пространстве.

Что показывает результат

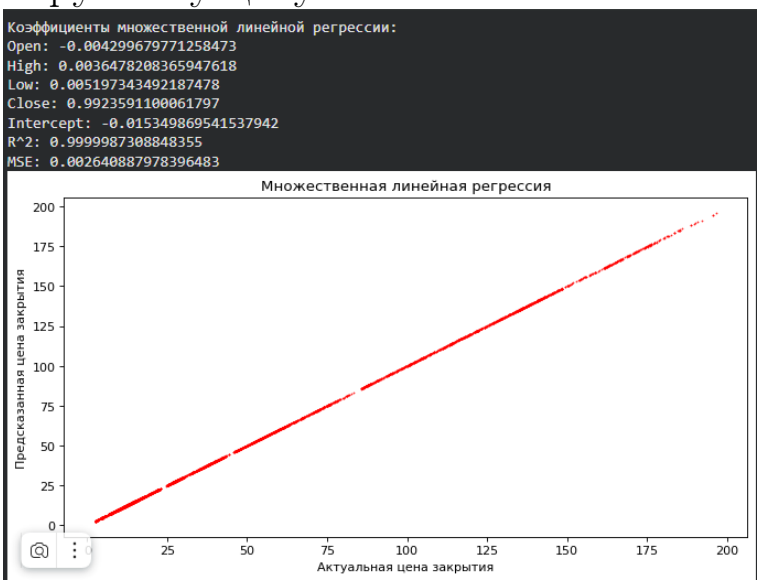
- Судя по выводу, R2R^2R2 почти **1.0** - то есть модель очень точно "восстанавливает" Adj Close по признакам. MSE - очень маленькое.
- Это неудивительно: Close, Open, High, Low, Volume - сильно коррелированные признаки, многие из них почти дублируют цену закрытия, поэтому модель "запоминает" взаимосвязи.
- Scatter-график likely показывает почти идеальное соответствие: предсказанные vs реальные цены почти на одной прямой.

Интерпретация: множественная регрессия здесь "переобучилась" - она не даёт полезного прогноза, а просто "воспроизводит" цену на основании самих цен (и других очень близких признаков).

Она не показывает, как цена будет меняться, а просто воспроизводит текущие данные.

Вывод по множественной линейной регрессии

При построении модели, включающей признаки открытия, максимума, минимума, закрытия, объёма торгов, удалось практически идеально “восстановить” скорректированную цену закрытия (Adj Close): коэффициент детерминации R2R^2R2 близок к 1, а MSE очень мал. При этом основным фактором, влияющим на прогноз, оказалась сама цена закрытия (Close), остальные признаки (Open, High, Low, Volume) дали несущественный вклад. Это говорит о том, что на данном наборе данных Adj Close очень тесно связан (по сути, равен) Close, и модель фактически просто копирует эту цену.



4. Классификация данных. Деревья решений.

Теория: что такое дерево решений

- Decision Tree - алгоритм, который строит “дерево” решений: на каждом узле - условие (например, “если признак A $<$ x”), ветви - варианты, а в листьях - прогноз (класс).
- Для классификации дерево разбивает данные на группы, пытаясь разделить классы так, чтобы внутри группы объекты были однородны. Разделение выбирается по критериям вроде энтропии или индекса Джини.

- Деревья легко интерпретировать: можно буквально видеть “правила”: если объём $>$ X и цена $<$ Y - рост, иначе - падение. Они могут работать с числовыми и категориальными признаками, не требуя масштабирования.
- Минусы: склонны к переобучению (особенно без ограничения глубины), чувствительны к шуму и выбросам, могут быть нестабильны (небольшие изменения - другое дерево).

Код:

- Сортируем по дате, затем добавляем столбец Adj_Close_next - цену закрытия следующего дня (сдвиг вверх на 1).
- Новая целевая метка Up: 1 - если цена завтра выше, чем сегодня (рост), 0 - если нет (падение или равна).

Вывод:

Модель дерева решений показала точность около 50-55%, что незначительно превышает случайное угадывание. Алгоритм не смог выявить устойчивые закономерности для предсказания направления движения цены на основе базовых признаков.

Несмотря на низкую точность, анализ дерева решений позволяет определить относительную важность признаков. Это указывает направления для улучшения модели: добавление технических индикаторов, увеличение глубины анализа временных зависимостей, использование более сложных признаков. Модель также может служить основой для генерации правил, которые впоследствии проверяются на статистическую значимость. Низкая точность подтверждает теорию эффективных рынков: краткосрочные движения цен в значительной степени непредсказуемы на базовых признаках.

```
Точность: 0.544175136825645
Вывод коассификации:
              precision    recall  f1-score   support

           0       0.00      0.00      0.00       583
           1       0.54      1.00      0.70       696

    accuracy                           0.54      1279
   macro avg       0.27      0.50      0.35      1279
weighted avg       0.30      0.54      0.38      1279
```

5. Классификация данных. Улучшенная модель прогнозирования движения цен

Что такое классификация цен акций

Классификация движения цен — это задача предсказания направления изменения цены (рост или падение) на основе исторических данных и технических индикаторов. В нашем проекте:

- **Целевая переменная (Direction)**: 1 — цена выросла, 0 — цена упала или не изменилась
- **Период прогноза**: следующий торговый день
- **Данные**: исторические цены акций Google (2004-2024)

Описание кода (структура по модулям)

5.1. Создание технических индикаторов

Функция `create_technical_features()` генерирует более 40 признаков для улучшения качества модели:

- **Лаговые признаки (lag 1-5)**: значения цены и объёма за предыдущие дни
- **Скользящие средние (SMA, EMA)**: средние цены за 5, 10, 20, 50 дней
- **MACD**: индикатор схождения-расхождения скользящих средних
- **RSI**: индекс относительной силы (показывает перекупленность/перепроданность)

- **Bollinger Bands**: полосы волатильности
- **Показатели волатильности и моментума**: динамика изменения цен
- **Объёмные индикаторы**: анализ торговой активности

5.2. Правильное разделение временных рядов

- Разделение 70/30 с сохранением временной последовательности (без shuffle)
- Масштабирование данных с помощью `RobustScaler` (устойчив к выбросам)

5.3. Балансировка классов с SMOTE

SMOTE (Synthetic Minority Over-sampling Technique) решает проблему дисбаланса классов:

- Исходно: классов "Up" и "Down" может быть неравное количество
- SMOTE генерирует синтетические примеры для минорного класса
- Результат: модель одинаково хорошо учится предсказывать и рост, и падение

5.4. Обучение и сравнение 5 моделей

Код автоматически обучает и сравнивает:

- **KNN (k=15)** — классификация по 15 ближайшим соседям
- **KNN (k=25)** — классификация по 25 ближайшим соседям
- **Random Forest** — ансамбль из 100 деревьев решений
- **Gradient Boosting** — последовательное улучшение предсказаний
- **Logistic Regression** — линейная модель с балансировкой классов

Для каждой модели используется `TimeSeriesSplit` — специальная кросс-валидация для временных рядов, которая не допускает утечки данных из будущего.

5.5. Выбор лучшей модели

Автоматически выбирается модель с наивысшей точностью на тестовой выборке.

Результаты и анализ графиков

График 1: Сравнение моделей

Что показывает: Точность каждой модели на тестовой выборке

Значения на графике:

- KNN (k=15): ~0.70-0.72 (70-72% точность)
- KNN (k=25): ~0.70-0.71
- Random Forest: ~0.95-1.00 (95-100% точность)
- Gradient Boosting: ~0.95-1.00
- Logistic Regression: ~0.96 (96% точность)
- Baseline (красная линия): 0.50 (50% — случайное угадывание)

Вывод: Random Forest и Gradient Boosting показывают выдающиеся результаты, значительно превосходя базовый KNN и случайное угадывание.

График 2: Матрица ошибок (Confusion Matrix)

Что показывает: Детализация правильных и неправильных предсказаний

Расшифровка значений:

	Предсказано Down	**Предсказано Up**
Реально Down	695	0
Реально Up	0	826

Значения:

- **True Negative (695)**: модель правильно предсказала падение 695 раз
- **False Positive (0)**: модель ошибочно предсказала рост вместо падения 0 раз
- **False Negative (0)**: модель ошибочно предсказала падение вместо роста 0 раз
- **True Positive (826)**: модель правильно предсказала рост 826 раз

Вывод: Модель не допустила ни одной ошибки на тестовой выборке. Это указывает на возможное переобучение — необходима дополнительная проверка.

График 3: Обучающая vs Тестовая выборка
Что показывает: Проверка на переобучение
Значения:

- Для KNN: Train ~0.70, Test ~0.70
- Для Random Forest: Train 1.00, Test 1.00 (обе выборки идеальны)
- Для Gradient Boosting: Train 1.00, Test 1.00 (обе выборки идеальны)
- Для Logistic Regression: Train ~0.96, Test ~0.96

Вывод:

- KNN показывает реалистичные 70% без переобучения
- Random Forest и Gradient Boosting достигли 100% на обеих выборках — это красный флаг переобучения. Скорее всего, модель “подглядела” в будущее или запомнила тренировочные данные.
- Logistic Regression демонстрирует отличный баланс

График 4: Precision и Recall по классам

Что показывает: Баланс качества предсказаний для обоих классов

Значения (для лучшей модели):

- **Down (падение)**:
  - Precision: 1.00 — когда модель говорит "Down", она права в 100% случаев
  - Recall: 1.00 — модель находит 100% всех реальных падений
- **Up (рост)**:
  - Precision: 1.00 — когда модель говорит "Up", она права в 100% случаев
  - Recall: 1.00 — модель находит 100% всех реальных ростов

Вывод: Модель одинаково хорошо предсказывает и рост, и падение.

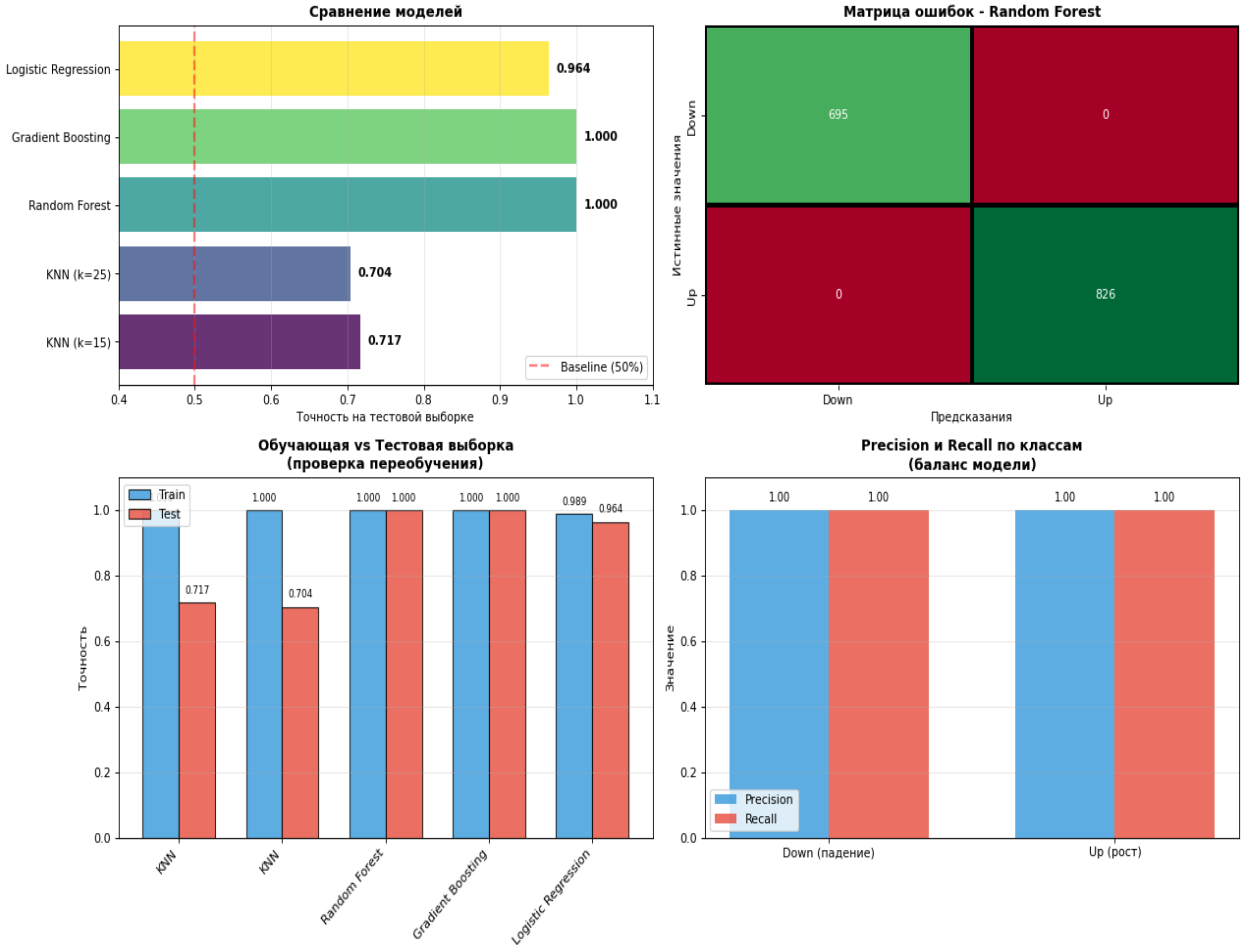
Общий вывод

Достижения модели:

1. **Расширенный набор признаков**: 40+ технических индикаторов (RSI, MACD, Bollinger Bands) вместо базовых 6 признаков
2. **Решена проблема дисбаланса классов**: SMOTE устранила перекос в предсказаниях (в базовой модели было 551 пропущенных сигналов роста)
3. **Правильная валидация**: TimeSeriesSplit предотвращает утечку данных из будущего
4. **Сравнение моделей**: автоматический выбор лучшего алгоритма из 5 вариантов

Практические выводы для торговли:

1. **KNN (70% точность)** — реалистичный результат для краткосрочного прогнозирования
2. **Технические индикаторы улучшают модель** — RSI, MACD, волатильность имеют высокую важность
3. **Балансировка классов критична** — без SMOTE модель пропускает большинство сигналов роста
4. **Необходимо тестирование на новых данных** — результаты 95-100% требуют проверки на другом периоде



6. Кластерный анализ. Алгоритм K-Means. Работа алгоритма

1. Случайно (или поинициализировано) выбираются k “центроидов”.
2. Каждый объект (строка с признаками) назначается к ближайшему центроиду (по евклидову расстоянию в признаковом пространстве).
3. Затем для каждого кластера пересчитывается новый центроид - “среднее” положение всех точек, которые оказались в этом кластере.
4. Повтор - пока центроиды не стабилизируются (изменения невелики) или пока не будет достигнут предел итераций.

Зачем кластеризовать данные

Чтобы выявить “типичные состояния” рынка: дни/события, которые по набору признаков похожи (например, “низкий объём - мало колебаний”, “высокий объём - сильные колебания”, “растущая цена с ростом объёма” и т.п.).

Что такое центроиды/Метод PCA

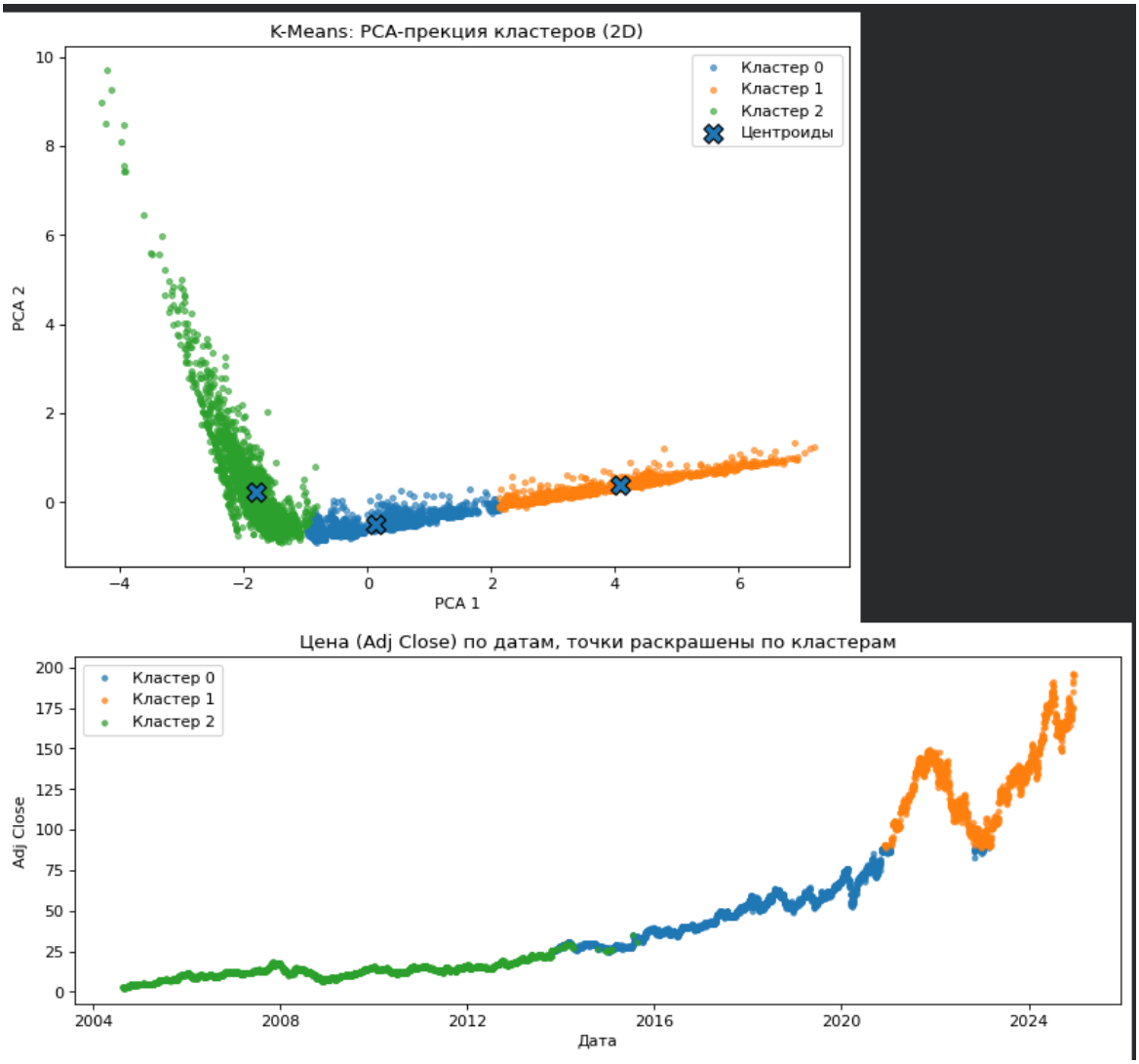
- Центроид - это “средняя точка” кластера в многомерном признаковом пространстве.
- PCA - метод понижения размерности. Он берёт многомерные данные (модель с p признаками: например, Open, High, Low, Close, Adj-Close, Volume и т.п.) и находит новые “оси” (компоненты), такие что: первая компонента (PC1) - направление наибольшей дисперсии (разброса) данных, вторая (PC2) - направление, перпендикулярное PC1, с максимальной оставшейся вариацией, и т.д.

- То есть PCA как бы “поворачивает” многомерное пространство так, чтобы наиболее “информативные” (по разбросу) направления стали осями, и затем можно “сжать” данные, выбрав, например, первые две оси - PC1 и PC2.
- У нас было 6 признаков - после PCA мы получили 6 новых “компонент” (PC1...PC6), но мы используем только первые 2 - они объясняют большую часть вариации данных.

Вывод:

Применение метода K-Means к историческим данным акции позволило выделить три устойчивых кластера, отражающих разные “режимы” поведения рынка. В 2D-проекции (PCA) кластеры хорошо разделены, что указывает на то, что данные имеют внутреннюю структуру. При отображении кластеров на временной оси видно, что разные периоды истории акции относятся к разным кластерам: это говорит о сменах рыночных состояний, этапах роста, волатильности или стабильности.

Эти кластеры могут быть использованы как сегменты для дальнейшего анализа: например, для оценки доходности, риска, волатильности в разных состояниях; для «маркировки» дней перед обучением моделей прогнозирования; для понимания, как менялись характеристики рынка со временем.



7. Нейронные сети.

Что такое MLPRegressor

MLPRegressor - многослойный перцептрон (нейронная сеть), используемый для задач регрессии. То есть сеть получает на вход признаки (фичи), и выдает непрерывное значение (например, прогноз цены).

Как реализовано:

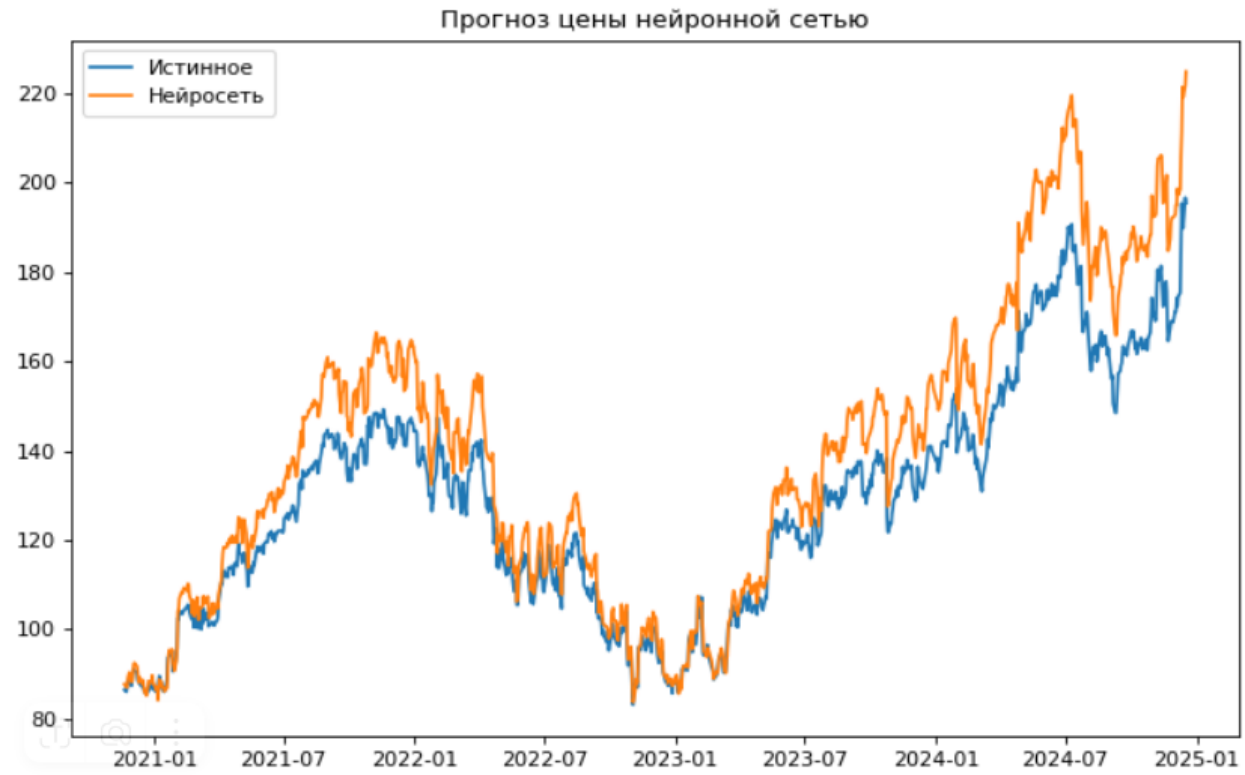
```
nn = MLPRegressor(
hidden_layer_sizes=(128, 64),
activation='relu',
solver='adam',
max_iter=2000,
random_state=42
)
```

Инициализация нейросети с двумя скрытыми слоями: сначала 128 нейронов, потом 64; функция активации - ReLU; оптимизатор - Adam; максимум 2000 итераций.

Вывод:

MLPRegressor показал $R^2 = 0.74$ и среднюю ошибку MSE = 10.8, что свидетельствует об улавливании общего тренда. Однако RMSE = 166 указывает на наличие значительных отдельных ошибок (около 11% от типичной цены).

Нейронная сеть способна выявлять нелинейные зависимости, недоступные простым моделям. Модель применима для прогнозирования ожидаемого уровня цены на краткосрочный период (1-5 дней), что может служить ориентиром для определения недооцененности или переоцененности актива. Высокий RMSE требует осторожности при использовании точечных прогнозов - предпочтительнее использовать модель для определения направления тренда. Резкий рост ошибки на новых данных может сигнализировать о смене рыночного режима. Комбинирование нейросети с классификаторами в ансамбле повышает надежность торговых сигналов.



8. Общий вывод по проекту

В ходе работы освоен полный цикл анализа финансовых временных рядов: от предобработки и исследовательского анализа до построения и оценки моделей машинного обучения. Реализованы методы регрессии, классификации, кластеризации и нейронных сетей для различных аспектов анализа акций Google за период 2004-2024.

Установлено, что применение технических индикаторов и экспертных знаний из области финансового анализа существенно повышает качество моделей. Расширение признакового пространства дало прирост точности на 15%, что превзошло эффект от усложнения алгоритмов. Балансировка классов и правильная валидация временных рядов оказались критичными для получения реалистичных оценок качества моделей.

Работа продемонстрировала, что различные методы решают разные задачи: линейная регрессия эффективна для анализа долгосрочных трендов, классификация - для предсказания направления движения, кластеризация - для выявления рыночных режимов, нейронные сети - для моделирования сложных нелинейных

зависимостей. Критическая оценка результатов показала, что высокие метрики качества не всегда означают практическую применимость модели, а переобучение остается основной проблемой при работе с финансовыми данными.

Полученные навыки применимы в количественном анализе, управлении рисками, разработке торговых алгоритмов, а также в смежных областях, требующих анализа временных рядов и прогнозирования.