Правительство Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

"Национальный исследовательский университет "Высшая школа экономики"

Факультет экономических наук

Образовательная программа «Экономика»

КУРСОВАЯ РАБОТА

«Сбор, анализ и предиктивная аналитика открытых источников данных. Предсказание цен активов Российского фондового рынка методами машинного обучения и статистического анализа"

Студент группы БЭК222
Решетнев Матвей Андреевич

Научный руководитель
Лебедев Михаил Владимирович

Москва, 2025г.

Введение

Актуальность темы

Прогнозирование цен акций остается одной из ключевых задач в финансовой аналитике и экономике. Ее важность раскрывается в следующих факторах:

Экономическая значимость фондового рынка.

- Фондовый рынок играет критическую роль в мировой экономике, обеспечивая перераспределение капитала и финансирование бизнеса. Точные прогнозы позволяют инвесторам минимизировать риски и максимизировать доходность, а компаниям — привлекать ресурсы для развития.

Практическая ценность для участников рынка.

- Для инвесторов: Прогнозы помогают в формировании портфелей и оптимальных стратегий торговли
- Для компаний: Прогнозирование волатильности и трендов способствует управлению капиталом и снижению рисков.
- Для регуляторов: Анализ рыночной динамики помогает предотвращать кризисы.

Обоснование темы проекта

Обоснование темы заключается в необходимости разработки и тестирования процесса сбора данных и построения прогнозной модели, которая сможет более точно предсказывать цену закрытия для акций фондового рынка Мосбиржи с помощью современных методов машинного обучения.

Цель и задачи исследования

Цель данного исследования заключается в разработке и обучении модели, способной прогнозировать значение цены закрытия и направление ее движения с высокой точностью, учитывая влияние традиционных экономических факторов, показателей технического анализа и неявных зависимостей во временных рядах.

Для достижения поставленной цели были сформулированы следующие задачи:

1. Провести обзор существующих подходов к прогнозированию финансовых показателей и оценить их эффективность.
2. Собрать и подготовить данные: по торгам на Мосбирже, по событиям в компаниях (сплит акций, дивиденды), показатели технического анализа.
3. Разработать модели, учитывающие временную зависимость данных и включающую ключевые финансовые и технические показатели.
4. Изучить и выбрать метрики для сравнения оценки качества предсказаний моделей.
5. Протестировать качество построенных моделей на исторических данных и выбрать наилучшую.
6. Создать удобный интерфейс для получения предсказаний модели для заинтересованных пользователей.

Методы исследования

В рамках исследования применяются методы технического анализа, эконометрические модели, современные методы классического машинного обучения и глубокого обучения.

Инструменты и технологии

Для разработки и обучения модели используются инструменты и технологии, включающие:

1. Python и библиотеки для машинного обучения и продвинутого анализа данных (Pytorch, Scikit-learn, statsforecast, mlforecast, neuralforecast и др.).
2. Платформы для сбора и обработки данных (сайты с открытыми данными и бесплатным API).
3. Средства визуализации данных (Matplotlib, Seaborn, Plotly).
4. Библиотека для создания базы данных (sqlite3)

Эти инструменты были выбраны из-за их возможностей и эффективности при работе с временными рядами и моделями машинного обучения и анализа данных.

Структура работы

Работа состоит из нескольких разделов. В первом разделе представлен обзор литературы и теоретическая база, анализируются существующие подходы к прогнозированию цен акций на фондовом рынке. Во втором разделе описана методология исследования, включающая выбор и подготовку данных, разработку архитектуры модели, методы обучения и оценку модели. Третий раздел посвящен результатам исследования, анализу данных и оценке точности модели. В заключении сформулированы основные выводы и рекомендации по использованию результатов исследования.

Раздел 1. Источники данных

Существует множество открытых источников различных данных, связанных с темой финансовых рынков. Примеры:

1. Рыночные данные и котировки

Данные о ценах акций, объемах торгов, индексах и других рыночных показателях поступают с фондовых бирж и специализированных сервисов. Они включают исторические и текущие котировки, а также информацию о сделках и заявках.
Основной источник для построения моделей временных рядов и технического анализа.
Примеры источников: Yahoo! Finance; API Мосбиржа, Т-Инвестиции и др.

2. Финансовая отчетность компаний

Финансовая отчетность является фундаментальным источником данных для анализа финансового состояния и динамики предприятий. Включает в себя бухгалтерский баланс, отчет о прибылях и убытках, отчет о движении денежных средств и прочие формы. Эти данные позволяют оценить активы, обязательства, выручку, расходы и прибыль компании за определённый период.
Примеры источников: Сайты компаний.

3. Макроэкономические и отраслевые данные

Включают статистику по экономике страны и отраслей - ВВП, инфляция, безработица, цены на сырьё и энергоносители. Эти данные предоставляются государственными статистическими службами, международными организациями и аналитическими агентствами.
Примеры источников: МВФ, Всемирный банк, ЕЦБ.

4. Новостные и аналитические источники

Финансовые новости, пресс-релизы компаний, отчеты аналитиков, публикации рейтинговых агентств и специализированных изданий (например, «Уолл-Стрит Джорнал», Bloomberg) служат источниками качественной информации о событиях, влияющих на рынок.
Примеры источников: Wall Street Journal, Bloomberg

5. Альтернативные и поведенческие данные

К ним относятся данные о поисковых запросах (Google Trends, Яндекс.Вордстат), активности в социальных сетях, мобильных приложениях, а также данные о потребительском поведении и настроениях. Используются для выявления трендов и настроений, которые могут предвосхищать изменения на финансовых рынках.

Раздел 2. Теоретический обзор

Прогнозирование финансовых показателей - это ключевая задача в экономическом и финансовом анализе, требующая использования как традиционных, так и современных методов обработки данных. В этой главе рассматриваются основные подходы, применяемые для прогнозирования финансовых показателей, включая эконометрические методы, методы технического анализа и машинное обучение, а также используемые индексы и метрики.

1. Методы эконометрики

Эконометрические модели традиционно используются для анализа и прогнозирования временных рядов. Они базируются на статистических методах и играют важную роль в экономических исследованиях.

1. **Модели ARIMA (Autoregressive Integrated Moving Average):**

- Описание: Модели ARIMA позволяют учитывать как автокорреляцию (AR), так и скользящие средние (MA) для прогнозирования временных рядов. Эти модели были подробно описаны Box и Jenkins (1970) в их книге "Time Series Analysis: Forecasting and Control", которая остается фундаментальной работой в области временных рядов.
- Применение: ARIMA хорошо справляется с временными рядами, демонстрирующими стационарное поведение, но может испытывать трудности при наличии нелинейных или структурных изменений в данных.

2. **Модели ETS (ExponenTial Smoothing):**

- Описание: Простое, двойное и тройное экспоненциальное сглаживание, такие как модель Хольта-Уинтерса, являются популярными методами прогнозирования временных рядов с трендами и сезонными колебаниями. Эти методы были впервые предложены Charles Holt (1957) и Winters (1960).
- Применение: Эти модели обладают простотой и эффективностью при наличии сезонных эффектов, что делает их особенно полезными для прогнозирования данных с выраженными сезонными паттернами.

3. **Многофакторные регрессионные модели:**

- Описание: Регрессионные модели применяются для прогнозирования зависимых переменных на основе множества независимых переменных. Например, трехфакторная модель Fama и French (1993) рассматривает рыночный риск, размер компании и её стоимость для оценки доходности активов.
- Применение: Линейные регрессии подходят для простых зависимостей, тогда как полиномиальные и логарифмические регрессии используются для моделирования более сложных взаимосвязей.

4. Модели VAR (Vector Autoregression):

- Описание: Модели VAR, предложенные Sims (1980), анализируют взаимосвязи между несколькими временными рядами, что позволяет моделировать систему

экономических показателей, где переменные зависят не только от своего прошлого, но и от прошлого других переменных.
- Применение: VAR используется в макроэкономике для моделирования взаимосвязанных временных рядов, таких как инфляция, ВВП и процентные ставки.

2. Методы технического анализа

Технический анализ фокусируется на использовании исторических данных о ценах и объемах торгов для прогнозирования будущих рыночных движений. В отличие от фундаментального анализа, который учитывает экономические и финансовые факторы, технический анализ основывается на графических моделях и статистических индикаторах.

1. **Индикаторы тренда:**

- Описание: Скользящие средние (SMA и EMA) используются для сглаживания временных рядов и выявления трендов. MACD, предложенный Appel (1979), сравнивает две скользящие средние и помогает определять моменты разворота тренда.
- Применение: Эти индикаторы позволяют выявлять направление тренда и предупреждать о возможных изменениях в его направлении.

2. **Индикаторы осцилляции:**

- Описание: RSI, разработанный Wilder (1978), измеряет скорость и амплитуду изменений цены, что помогает выявлять состояния перекупленности или перепроданности. Stochastic Oscillator, предложенный Lane (1950), также используется для оценки рыночных условий и выявления точек разворота.
- Применение: Эти индикаторы помогают трейдерам и аналитикам оценивать рыночные условия и прогнозировать развороты трендов.

3. **Индикаторы волатильности:**

- Описание: Bollinger Bands, созданные Bollinger (1983), измеряют волатильность актива, определяя уровни поддержки и сопротивления.
- Применение: Эти индикаторы используются для оценки рыночной волатильности и предсказания возможных экстремальных движений цен.

4. **Графические модели и паттерны:**

- Описание: Графические паттерны, такие как "Голова и плечи" или "Треугольники", и японские свечи позволяют анализировать рыночные настроения и прогнозировать дальнейшие движения цен. Исследования Edwards и Magee (1948) стали основой для применения таких паттернов в техническом анализе.
- Применение: Графические модели позволяют прогнозировать возможные развороты или продолжение трендов на рынке.

3. Методы машинного обучения

Машинное обучение предоставляет возможности для гибкого и точного прогнозирования, особенно в условиях сложных и нелинейных зависимостей, часто встречающихся в финансовых данных.

1. **Деревья решений и случайные леса:**

   - Описание: Деревья решений и случайные леса, описанные Breiman (2001), используются для построения моделей, учитывающих сложные нелинейные зависимости. Случайные леса объединяют множество деревьев решений для повышения точности и устойчивости модели.

   - Применение: Эти методы часто применяются для классификации и регрессии в задачах, связанных с финансовыми данными.

2. **Градиентный бустинг:**

- Описание: Градиентный бустинг, разработанный Friedman (2001), является мощным инструментом для построения ансамблей моделей. Современные реализации, такие как XGBoost (Chen и Guestrin, 2016), используются для прогнозирования сложных зависимостей в финансовых данных.
- Применение: Градиентный бустинг позволяет строить высокоточные модели для прогнозирования финансовых показателей.

3. **Методы опорных векторов (SVM):**

- Описание: SVM, предложенный Cortes и Vapnik (1995), используется для классификации и регрессии в задачах с нелинейными данными. SVM-регрессия часто применяется для прогнозирования временных рядов.
- Применение: Этот метод эффективен при работе с небольшими наборами данных и сложными нелинейными зависимостями.

4. **Нейронные сети:**

- Описание: Нейронные сети, такие как MLP и RNN, применяются для моделирования сложных зависимостей в данных.

- Применение: используются для прогнозирования временных рядов, где важны долгосрочные и сложные зависимости.

4. Индикаторы технического анализа

Для расчета индикаторов используются методы из библиотеки ta-lib для Python. Основные индикаторы, рассчитываемые в системе:

- **RSI (Relative Strength Index)**: определяет состояния перекупленности или перепроданности.

- **MACD (Moving Average Convergence Divergence)**: указывает на изменение силы тренда.

- **Скользящие средние (SMA и EMA)**: сглаживают временные ряды для выявления трендов.

- **Bollinger Bands**: оценивают волатильность актива.
- **Стохастический осциллятор (Stochastic Oscillator)**: анализирует соотношение цены закрытия и диапазона цен за определенный период.
- **ATR (Average True Range)**: измеряет волатильность.
- **ADX (Average Directional Index)**: оценивает силу тренда.
- **CCI (Commodity Channel Index)**: показывает отклонение текущей цены от её среднестатистического значения.
- **Williams %R**: измеряет уровни перекупленности и перепроданности.
- **Volume**: показывает объем торгов для анализа рыночной активности.

5. Метрики качества моделей временных рядов

1. Методика расчёта метрик

Для оценки качества моделей прогнозирования цен акций была использована процедура rolling validation с расширяющейся тренировочной выборкой. Валидационный период составил 14 дней. Для каждого дня валидации:

- Формировалась тренировочная выборка, включающая все данные до текущего дня валидации (инкрементальное добавление дней валидации в тренировочный набор).
- Выполнялось прогнозирование значений на текущий день валидации для всех акций.
- Для каждой акции рассчитывались метрики качества прогнозов.
- По каждой метрике вычислялись среднее (mean) и медианное (median) значения по всем акциям за данный день.
- После этого значения метрик усреднялись по всем 14 дням валидации, что позволило получить агрегированные оценки качества моделей, отражающие их стабильность и точность в динамике временного ряда.

2. Описание используемых метрик

MASE (Mean Absolute Scaled Error) — Средняя абсолютная масштабированная ошибка

MASE является масштабно-независимой метрикой, которая сравнивает среднюю абсолютную ошибку модели с ошибкой наивного сезонного прогноза. Формально:

$$\text{MASE} = \frac{\frac{1}{n}\sum_{t=1}^{n}|y_t - \hat{y}_t|}{\frac{1}{n-m}\sum_{t=m+1}^{n}|y_t - y_{t-m}|}$$

MASE позволяет объективно сравнивать модели на разных временных рядах и учитывает сезонность. Значение меньше 1 указывает на превосходство модели над наивным прогнозом.

MAPE (Mean Absolute Percentage Error) — Средняя абсолютная процентная ошибка.

MAPE измеряет среднюю абсолютную ошибку в процентах от истинных значений:

$$\text{MAPE} = \frac{100\%}{n} \sum_{t=1}^{n} \left| \frac{y_t - \hat{y}_t}{y_t} \right|$$

Метрика интуитивно понятна и широко используется в финансовом прогнозировании. Однако MAPE чувствительна к нулевым и близким к нулю значениям, что может искажать оценку.

SMAPE (Symmetric Mean Absolute Percentage Error) — Симметричная средняя абсолютная процентная ошибка

SMAPE модифицирует MAPE, учитывая сумму абсолютных значений прогноза и истинного значения в знаменателе, что снижает влияние малых значений:

$$\text{SMAPE} = \frac{100\%}{n} \sum_{t=1}^{n} \frac{|y_t - \hat{y}_t|}{(|y_t| + |\hat{y}_t|)/2}$$

SMAPE обеспечивает более сбалансированную оценку ошибок при прогнозировании финансовых временных рядов с переменной амплитудой.

Directional Accuracy (DA) — Точность направления изменения
DA измеряет долю случаев, когда модель правильно предсказала направление изменения временного ряда (рост или падение) относительно предыдущего значения. Формально:

$$\text{DA} = \frac{1}{n-1} \sum_{t=2}^{n} \mathbf{1}\left(\text{sign}(y_t - y_{t-1}) = \text{sign}(\hat{y}_t - y_{t-1})\right)$$

где $\mathbf{1}$ — индикатор совпадения знаков изменений.

DA важна для финансовых приложений, где правильное предсказание направления движения цены зачастую важнее точности числового значения.

Таким образом, методика оценки качества прогнозов включала:

- Инкрементальное расширение тренировочной выборки на каждый день валидации.
- Расчёт метрик MASE, MAPE, SMAPE и Directional Accuracy для каждого дня и каждой акции.
- Агрегацию метрик по акциям (среднее и медиана) для каждого дня.
- Усреднение значений метрик по всем 14 дням валидации для получения итоговых характеристик моделей.
- Такой подход обеспечивает всестороннюю и надёжную оценку качества прогнозов, учитывая как абсолютную точность, так и правильность направления изменения цен.

Раздел 3. Сбор, обработка и хранение данных

Сбор данных

В силу внешних обстоятельств акции Российских компаний больше недоступны иностранным инвесторам, поэтому данные по их котировкам за даты позже 2022 года недоступны на **yfinance.** Поэтому было принято решение собирать данные о торгах на Мосбирже через бесплатное API Мосбиржи.

Были собраны данные 44 российских компаний за даты с 2022-04-01 по данный момент (2025-05-27) .

Пример кода для сбора данных по торгам на Мосбирже:

```
# ПАРСИНГ ЦЕН С MOEX
def fetch_prices_data(ticker, start_date, end_date, engine='stock', market='shares', board='TQBR'):
    url = f"https://iss.moex.com/iss/history/engines/{engine}/markets/{market}/boards/{board}/securities/{ticker}.json"
    all_data = []
    start = 0
    expected_columns = ['SECID', 'TRADEDATE', 'WAPRICE', 'OPEN', 'CLOSE', 'LOW', 'HIGH', 'VOLUME']

    while True:
        params = {'from': start_date, 'till': end_date, 'start': start}
        response = requests.get(url, params=params)
        if response.status_code != 200:
            print(f"Ошибка для {ticker}: {response.status_code}")
            break

        json_data = response.json()
        history = json_data.get('history', {})
        data = history.get('data', [])

        if not data:
            break

        df = pd.DataFrame(data, columns=history.get('columns', []))
        df = df.reindex(columns=expected_columns).dropna(how='all')

        if not df.empty:
            all_data.append(df)
            start += len(data)
        else:
            break

    if all_data:
        dtypes = {
            'SECID': 'str',
            'TRADEDATE': 'str',
```

```
            'WAPRICE': 'float64',
            'OPEN': 'float64',
            'CLOSE': 'float64',
            'LOW': 'float64',
            'HIGH': 'float64',
            'VOLUME': 'int64'
        }
        return pd.concat(all_data, ignore_index=True).astype(dtypes)
    return pd.DataFrame()
```

Также API Мосбиржи предоставляет доступ к данным по планируемым и выплаченным дивидендам российских компаний, они также были собраны:

```
# ПАРСИНГ ДИВИДЕНДОВ С MOEX
def fetch_dividends_data(ticker):
    url = f"https://iss.moex.com/iss/securities/{ticker}/dividends.json"

    params = {
        'iss.json': 'extended',
        'iss.meta': 'off'
    }

    response = requests.get(url, params=params)
    if response.status_code != 200:
        print(f"Ошибка для {ticker}: {response.status_code}")
        return pd.DataFrame()

    data = response.json()[1].get('dividends', [])
    if not data:
        return pd.DataFrame()

    expected_columns = ['secid', 'isin', 'registryclosedate', 'value', 'currencyid']
    df = pd.DataFrame(data).reindex(columns=expected_columns)

    dtypes = {
        'secid': 'str',
        'isin': 'str',
        'registryclosedate': 'str',
        'value': 'float64',
        'currencyid': 'str'
    }
    return df.astype(dtypes)
```

Также для правильной обработки и учета рыночных изменений были собраны данные по сплитам акций российских компаний:

```
# ПАРСИНГ СПЛИТОВ С MOEX
def fetch_splits_data(engine='stock'):
    url = f"https://iss.moex.com/iss/statistics/engines/{engine}/splits.json"

    params = {
        'iss.json': 'extended',
        'iss.meta': 'off'
    }

    response = requests.get(url, params=params)
```

```
    if response.status_code != 200:
        print(f"Ошибка при получении сплитов: {response.status_code}")
        return pd.DataFrame()

    data = response.json()[1].get('splits', [])
    if not data:
        return pd.DataFrame()

    expected_columns = ['tradedate', 'secid', 'before', 'after']
    df = pd.DataFrame(data).reindex(columns=expected_columns)

    dtypes = {
        'tradedate': 'str',
        'secid': 'str',
        'before': 'int32',
        'after': 'int32'
    }
    return df.astype(dtypes)
```

Далее данные были подвергнуты предобработке.

Обработка данных

Сплиты акций

Часто в данных некоторых компаний наблюдались пропуски – дни, когда их акции не торговались на бирже. В большинстве случаев это было связано с изменением структуры акций компаний – проходили сплиты акций, после которых актив был недоступен некоторое время. Как известно, при сплите акций цена акции меняется вместе с объемом торгов. Для корректной работы моделей такие случаи было решено обрабатывать следующим образом: приводить цены акций до сплита к уровню цен после сплита. Например, таким образом цены акций VTBR до сплита были умножены на 5000, для поддержания единого уровня.

Пропуски

Для поддержания единой структуры датасета и выровненности временных рядов было принято решение интерполировать пропущенные фрагменты данных через квадратичный полином:

```
for ticker in list_of_tickers_with_gaps[list_of_tickers_with_gaps > 0].index:
    df_ticker = df[df['SECID'] == ticker]
    df_ticker = df_ticker.interpolate(method='polynomial', order=2)

    df[df['SECID'] == ticker] = df_ticker
```

База данных

Для реализации эффективного хранения, обработки и анализа данных, поступающих из различных источников, в рамках данного исследования была разработана архитектура

базы данных. Она является ключевым компонентом в автоматизации процессов обработки данных, включая сбор и хранение исторических данных по акциям российских компаний а также индикаторов технического анализа, дивидендов и сплитов. В результате база данных выступает центральным хранилищем для дальнейшей работы с данными, их анализа, визуализации и подготовки для использования в моделях машинного обучения.

1. Выбор базы данных

В рамках данной работы была выбрана **SQLite** как платформа для реализации базы данных. Выбор обоснован следующими факторами:

- **Легкость внедрения**: SQLite — это встроенная база данных, которая не требует установки и настройки серверов.
- **Производительность**: SQLite отлично подходит для небольших и средних проектов, где не требуется многопользовательский доступ или сложная конфигурация.
- **Поддержка SQL**: Поддержка стандартного языка SQL позволяет гибко работать с данными, осуществлять сложные запросы и агрегировать информацию.

Также преимуществом является возможность легко перенести архитектуру на другие реляционные базы данных (например, PostgreSQL, MySQL) при необходимости масштабирования системы.

2. Проектирование базы данных

Архитектура базы данных была спроектирована с учётом специфики финансовых данных и технического анализа. В базе данных было создано несколько таблиц, каждая из которых отвечает за хранение определённого типа данных.

Основные таблицы:

prices — таблица для хранения исторических данных по торгам на Мосбирже

```
# ТАБЛИЦА ЦЕН
cursor.execute('''
CREATE TABLE IF NOT EXISTS prices (
    SECID TEXT,
    TRADEDATE TEXT,
    WAPRICE REAL,
    OPEN REAL,
    CLOSE REAL,
    LOW REAL,
    HIGH REAL,
    VOLUME INTEGER,
    PRIMARY KEY (SECID, TRADEDATE)
)
''')
```

- SECID (TEXT) – тикер компании на бирже
- TRADEDATE (TEXT / DATE) – дата торгового дня
- WAPRICE (REAL) – средневзвешенная цена за торговый день

- OPEN (REAL) – цена открытия
- CLOSE (REAL) – цена закрытия
- LOW (REAL) – минимальная цена за торговый день
- HIGH (REAL) - максимальная цена за торговый день
- VOLUME (REAL) – объем торгов

dividends — таблица для хранения данных о дивидендах компаний

```
    # ТАБЛИЦА ДИВИДЕНДОВ
    cursor.execute('''
    CREATE TABLE IF NOT EXISTS dividends (
        secid TEXT,
        isin TEXT,
        registryclosedate TEXT,
        value REAL,
        currencyid TEXT,
        PRIMARY KEY (secid, registryclosedate)
    )
    ''')
```

- SECID (TEXT) – тикер компании на бирже
- ISIN (TEXT) – идентификатор акции
- REGISTRYCLOSEDATE (TEXT) – последняя дата для возможности получения дивидендов
- VALUE (REAL) – размер дивидендов на акцию
- CURRENCYID (TEXT) – валюта

splits — таблица для хранения данных о сплитах акций

```
    # ТАБЛИЦА СПЛИТОВ
    cursor.execute('''
    CREATE TABLE IF NOT EXISTS splits (
        tradedate TEXT,
        secid TEXT,
        before INTEGER,
        after INTEGER,
        PRIMARY KEY (secid, tradedate)
    )
    ''')
```

- TRADEDATE (TEXT / DATE) – дата торгового дня
- SECID (TEXT) – тикер компании на бирже
- BEFORE (INTEGER) – вес акции до сплита
- AFTER (INTERGER) – вес акции после сплита

clean_prices – таблица обработанных данных, пригодных для обучения модели

```
        # Создаем таблицу
        cursor = conn.cursor()
        cursor.execute('''
        CREATE TABLE IF NOT EXISTS clean_prices (
```

```
            SECID TEXT,
            TRADEDATE TEXT,
            WAPRICE REAL,
            OPEN REAL,
            CLOSE REAL,
            LOW REAL,
            HIGH REAL,
            VOLUME REAL,
            PRIMARY KEY (SECID, TRADEDATE)
        )
        ''')
```

- SECID (TEXT) – тикер компании на бирже
- TRADEDATE (TEXT / DATE) – дата торгового дня
- WAPRICE (REAL) – средневзвешенная цена за торговый день
- OPEN (REAL) – цена открытия
- CLOSE (REAL) – цена закрытия
- LOW (REAL) – минимальная цена за торговый день
- HIGH (REAL) - максимальная цена за торговый день
- VOLUME (REAL) – объем торгов

predictions – обновляемая таблица с предсказаниями моделей

```
    # Подключаемся к базе данных
    with sqlite3.connect(db_path) as conn:
        conn.execute('''
        CREATE TABLE IF NOT EXISTS predictions (
            unique_id TEXT,
            ds TEXT,
            prediction REAL,
            model_name TEXT,
            prediction_dttm TEXT,
            last_y REAL,
            delta REAL,
            delta_perc REAL,
            direction TEXT,
            PRIMARY KEY (unique_id, ds, model_name, prediction_dttm)
        )
        ''')
```

- unique_id (TEXT) – тикер
- ds (TEXT) – дата торгового дня,
- prediction (REAL) – предсказанная цена закрытия,
- model_name (TEXT) – название предсказывающей модели,
- prediction_dttm (TEXT) – дата-время создания предсказания,
- last_y (REAL) – прошлое значение таргета,
- delta (REAL) – разница prediction и last_y,
- delta_perc (REAL) - разница prediction и last_y, в процентах,
- direction (TEXT) – направление изменения цены,

Код сбора и обработки данных: github

Организация структуры данных

База данных была организована таким образом, чтобы поддерживать несколько типов финансовой информации в разных таблицах. Это позволяет эффективно осуществлять различные запросы, комбинировать данные из разных таблиц и производить сложные вычисления на уровне базы данных.

Например, данные о дивидендах, сплитах и исторических котировках акций можно легко комбинировать для анализа с помощью SQL-запросов. Также данная структура упрощает расширение функционала базы данных и её масштабирование.

Преимущества выбранной архитектуры

1. **Универсальность**: Архитектура базы данных поддерживает широкий спектр финансовых данных, от исторических котировок акций до отчетов о доходах и движении денежных средств. Это позволяет использовать базу данных для различных видов анализа.

2. **Расширяемость**: В случае необходимости структура базы данных может быть легко дополнена новыми таблицами или колонками без нарушения текущей логики работы системы.

3. **Простота интеграции**: Благодаря использованию SQL и стандартных библиотек Python, таких как **Pandas**, база данных легко интегрируется с другими компонентами системы, такими как алгоритмы машинного обучения или инструменты визуализации.

Раздел 4. Разработка моделей

Код исследования моделей: github

Рассмотренные модели

1. Бейзлайн-модели

Небольшие и наивные модели в задачах прогнозирования временных рядов часто могут оказаться намного эффективнее и качественнее сложных и требовательных ко времени и памяти моделей машинного обучения.

В качестве бейзлайн-вариантов были рассмотрены следующие алгоритмы из библиотеки statsforecast:

```
# Инициализируем модели для каждого прогноза
models = [
    stfm.Naive(),
    stfm.AutoARIMA(),
    stfm.AutoCES(),
    stfm.AutoETS(),
    stfm.AutoRegressive(lags=list(range(1, 8))),
    stfm.RandomWalkWithDrift(),
    stfm.WindowAverage(window_size=2, alias='WindowAverage(2)'),
    stfm.WindowAverage(window_size=3, alias='WindowAverage(3)'),
    stfm.WindowAverage(window_size=4, alias='WindowAverage(4)'),
    stfm.WindowAverage(window_size=5, alias='WindowAverage(5)'),
    stfm.WindowAverage(window_size=6, alias='WindowAverage(6)'),
    stfm.WindowAverage(window_size=7, alias='WindowAverage(7)'),
    stfm.WindowAverage(window_size=14, alias='WindowAverage(14)'),
]
```

1. Naive Model (Наивная модель)
Наивная модель предполагает, что прогноз на следующий период равен последнему наблюдаемому значению ряда. Несмотря на свою простоту, она служит базовым ориентиром для оценки качества более сложных моделей. В задачах с высокой волатильностью и шумом, характерных для фондового рынка, наивная модель часто показывает достаточно сильный базовый уровень, особенно для краткосрочных прогнозов. Использование наивной модели позволяет оценить, насколько сложные модели превосходят простейший подход.

2. AutoARIMA (Автоматическая ARIMA)
ARIMA (AutoRegressive Integrated Moving Average) — классическая модель временных рядов, учитывающая авторегрессию, интегрирование (устранение нестационарности) и скользящее среднее. AutoARIMA автоматически подбирает оптимальные параметры (p, d, q) и сезонные компоненты, что особенно важно при работе с большим числом временных рядов и отсутствием экспертных знаний. ARIMA хорошо моделирует линейные зависимости и сезонность, что часто встречается в финансовых данных.

3. AutoCES (Автоматическая модель CES)
CES (Complex Exponential Smoothing) — метод экспоненциального сглаживания, учитывающий тренд и сезонность с возможностью адаптации к сложным паттернам. AutoCES автоматически подбирает параметры модели, что позволяет эффективно работать с разнородными временными рядами. Данный метод хорошо справляется с нелинейными и изменяющимися во времени трендами, что актуально для динамичных финансовых рынков.

4. AutoETS (Автоматическая ETS)
ETS (Error-Trend-Seasonality) — семейство моделей экспоненциального сглаживания, учитывающих ошибку, тренд и сезонность. AutoETS автоматически выбирает структуру модели (аддитивная или мультипликативная компонента) и параметры сглаживания. Это обеспечивает гибкость и устойчивость к различным типам временных рядов. ETS-модели широко применяются для прогнозирования с сезонностью и трендами, что характерно для многих финансовых инструментов.

5. AutoRegressive (Автоматическая авторегрессия с лагами 1–7)
Авторегрессионные модели используют предыдущие значения ряда для прогнозирования текущего. Включение лагов с 1 по 7 позволяет учесть недельные циклы и краткосрочные зависимости, что важно для ежедневных данных фондового рынка. Автоматический подбор параметров позволяет адаптироваться к особенностям каждого ряда, обеспечивая баланс между сложностью и обобщающей способностью модели.

6. RandomWalkWithDrift (Случайное блуждание с дрейфом)
Модель случайного блуждания с дрейфом предполагает, что изменения ряда являются случайными, но со смещением (дрейфом), отражающим общий тренд. Эта модель часто используется в финансовой эконометрике как базовая для оценки эффективности более сложных моделей. Она хорошо отражает свойства финансовых временных рядов, где цены часто следуют случайному процессу с трендом.

7. WindowAverage (Скользящее среднее с окнами от 2 до 14 дней)
Скользящее среднее — простой, но эффективный метод сглаживания временного ряда, позволяющий устранить краткосрочные колебания и выявить тренды. Использование различных размеров окна (от 2 до 14 дней) позволяет исследовать влияние краткосрочной и среднесрочной динамики на прогноз. Такие модели полезны для улавливания локальных закономерностей и снижения шума в данных.

Обоснование

Выбор перечисленных моделей обусловлен их разнообразием по сложности, способности моделировать различные компоненты временных рядов (тренды, сезонность, шум), а также исторической и практической эффективностью в задачах финансового прогнозирования. Использование ансамбля моделей с разной природа позволяет повысить устойчивость и точность прогнозов цен акций на фондовом рынке.

Результат сравнения моделей

	MASE_mean	MAPE_mean	SMAPE_mean	Directional_Accuracy_mean	MASE_median	MAPE_median	SMAPE_median	Directional_Accuracy_median
y	0.000000	0.000000	0.000000	1.000000	0.000000	0.000000	0.000000	1.000000
AutoARIMA	0.887251	0.014194	0.007080	0.443182	0.707527	0.012067	0.006022	0.321429
RWD	0.882467	0.014074	0.007019	0.509740	0.707623	0.011901	0.005945	0.535714
Naive	0.883129	0.014119	0.007039	0.426948	0.712339	0.011933	0.005957	0.392857
AutoETS	0.885894	0.014173	0.007068	0.405844	0.714716	0.011920	0.005953	0.285714
CES	0.908864	0.014511	0.007233	0.485390	0.758950	0.012322	0.006152	0.464286
AutoRegressive	0.917551	0.014782	0.007369	0.423701	0.763405	0.012482	0.006233	0.357143
WindowAverage(2)	1.049940	0.016748	0.008339	0.428571	0.865799	0.014057	0.007021	0.392857
WindowAverage(3)	1.183505	0.019035	0.009462	0.451299	0.997822	0.016812	0.008389	0.357143
WindowAverage(4)	1.284377	0.020791	0.010321	0.459416	1.090919	0.018399	0.009180	0.428571
WindowAverage(5)	1.364199	0.022275	0.011035	0.501623	1.152022	0.019888	0.009906	0.500000
WindowAverage(6)	1.424808	0.023368	0.011545	0.524351	1.190863	0.021308	0.010587	0.500000
WindowAverage(7)	1.468713	0.024129	0.011889	0.548701	1.235269	0.021859	0.010831	0.500000
WindowAverage(14)	1.713490	0.028568	0.013994	0.538961	1.454360	0.026854	0.013244	0.535714

Лучшими по MASE стали алгоритмы AutoARIMA и RWD, но RWD лучше работала по метрике Directional Accuracy, поэтому было принятно решение считать RWD – бейзлайном.

2. ML-модели

CatBoostRegressor

В работе для повышения качества прогнозирования цен акций использовалась модель CatBoostRegressor с оптимизацией гиперпараметров с помощью библиотеки Optuna.

Обоснование выбора CatBoostRegressor

CatBoost — это современный алгоритм градиентного бустинга на решающих деревьях, созданный в компании Яндекс. Он обладет высокой точностью и устойчивостью к переобучению, особенно при работе с категориальными и временными признаками. Он эффективно обрабатывает разнородные данные и демонстрирует превосходство над классическими моделями в задачах регрессии и прогнозирования временных рядов.

2. Роль гиперпараметров и необходимость их оптимизации

Гиперпараметры модели (например, количество итераций, глубина деревьев, скорость обучения, коэффициент регуляризации) существенно влияют на качество и стабильность прогнозов. Неправильный выбор гиперпараметров может привести к переобучению или недообучению модели.

Оптимизация гиперпараметров — процесс поиска их наилучших значений, минимизирующих ошибку модели на валидационных данных. В работе применялся автоматизированный байесовский подход с помощью библиотеки Optuna, который позволяет эффективно исследовать пространство параметров и быстро находить оптимальные настройки.

3. Методика оптимизации с Optuna

Целевая функция (objective):
Определяется функция, которая принимает набор гиперпараметров, обучает модель CatBoostRegressor на тренировочных данных и возвращает метрику ошибки (например, RMSE) на валидационном наборе. Задача оптимизации — минимизировать эту метрику.

Поиск гиперпараметров:

Optuna использует методы байесовской оптимизации и алгоритмы поиска (например, Tree-structured Parzen Estimator) для выбора значений гиперпараметров из заданных диапазонов. В частности, оптимизировались параметры:

iterations — количество деревьев (100–2000),

learning_rate — скорость обучения (логарифмический масштаб от 0.001 до 0.3),

depth — максимальная глубина деревьев (4–10),

l2_leaf_reg — коэффициент L2-регуляризации (0.1–10),

border_count — количество границ для разбиения признаков (32–255),

bagging_temperature и random_strength — параметры стохастичности.

Обучение и оценка:
Для каждого набора параметров модель обучается на тренировочном подмножестве, затем прогнозируется на валидационном, и вычисляется ошибка. Optuna последовательно улучшает параметры, опираясь на результаты предыдущих испытаний.

4. Результаты

- Качество прогнозирования модели было недостаточным – дисперсия прогнозов модели была высокой и численная точность прогнозов была низкой.

- Оптимизация гиперпараметров позволила существенно снизить ошибку прогноза по сравнению с базовыми настройками.

- Использование Optuna обеспечило автоматизацию и ускорение процесса подбора параметров, что особенно важно при большом числе временных рядов и ограниченных ресурсах.

3. Deep Learning модели

Chronos

Chronos — это фреймворк от Amazon, адаптирующий архитектуры языковых моделей для задач прогнозирования временных рядов. Основная идея - токенизация непрерывных значений временного ряда в дискретные категории, что позволяет использовать мощные языковые модели без существенных изменений архитектуры. Chronos демонстрирует высокую эффективность благодаря предобучению на больших наборах данных и способности моделировать сложные последовательности с минимальными модификациями. Такой подход особенно актуален для финансовых временных рядов с их сложной динамикой и шумом.

Chronos была предобучена на больших и разнообразных наборах временных рядов из различных областей, что обеспечило её способность к обобщению и zero-shot прогнозированию — точному предсказанию на новых данных без дополнительного обучения. Предобучение включало обучение на миллионах временных рядов с различной структурой, что позволило модели эффективно выявлять сложные паттерны и зависимости.

В исследование было использовано два варианта Chronos:

- zero-shot pretrained-модель, которая по передаваемому контексту делает прогноз без обучения.
- Был проведен тюнинг small-версии модели на имеющихся исторических биржевых данных.

Результат:
Недостатком модели стало то, что она принимала похожие решения для каждой акции в датасете, запомнив общую конъюнктуру рынка по историческим данным, но не адаптировав информацию об отдельных тикерах.

RNN (Рекуррентная нейронная сеть)

RNN — класс нейросетевых моделей, специально разработанных для обработки последовательных данных. Благодаря внутренней памяти, RNN способны учитывать временную зависимость и контекст предыдущих состояний, что важно для моделирования автокорреляций и трендов в финансовых временных рядах. В работе RNN применялись для захвата нелинейных и долгосрочных зависимостей, характерных для цен акций, что улучшает качество краткосрочных прогнозов

Обоснование выбора гиперпараметров для RNN-модели

Конфигурация RNN-модели в рамках библиотеки NeuralForecast была оптимизирована с учётом специфики финансовых временных рядов.

- input_size=24, inference_input_size=24

  Размер входного окна (24 шага) позволяет модели анализировать данные за ~24 дня (при дневной частоте), что соответствует месячному торговому циклу. Это обеспечивает баланс между:
  - Улавливанием среднесрочных паттернов (например, реакция на квартальные отчёты)
  - Избеганием избыточной сложности, которая может привести к переобучению на шуме.

- loss=MASE(seasonality=1), valid_loss=MASE(seasonality=1)

  MASE (Mean Absolute Scaled Error) выбрана как масштабно-независимая метрика, сравнивающая ошибку модели с наивным прогнозом. Параметр seasonality=1 исключает сезонность, так как дневные данные цен акций редко демонстрируют явные сезонные паттерны из-за влияния внешних факторов (новости, макроэкономика). Это позволяет объективно оценить качество модели в условиях нестационарности.

- scaler_type='standard'

  Нормировка данных критична для финансовых временных рядов, где разные акции имеют различный ценовой диапазон. Стандартизация была выбрана

как наиболее распространенный и интерпретируемый способ нормализации данных.

- encoder_n_layers=2, decoder_layers=2

  Двухслойная архитектура кодировщика и декодировщика обеспечивает:

  - Достаточную глубину для моделирования нелинейных зависимостей (тренды, автокорреляция).

  - Умеренную сложность, предотвращающую переобучение на ограниченных финансовых данных.

- encoder_hidden_size=128, decoder_hidden_size=128

  Размер скрытого состояния (128 нейронов) выбран как компромисс между:
  - Емкостью модели — возможность улавливать сложные паттерны (например, корреляции между активами).
  - Вычислительной эффективностью — обучение на множестве временных рядов (акций) требует оптимизации ресурсов.

- max_steps=200

  Ограничение числа шагов обучения предотвращает переобучение, что особенно важно для финансовых данных с их низким соотношением сигнал/шум. Эмпирические тесты показали, что 200 шагов достаточно для сходимости при использовании оптимизатора Adam.

- hist_exog_list содержал календарные признаки и показатели технического анализа
  Включение экзогенных переменных (например, объёмы торгов, скользящие средние, RSI) позволяет модели учитывать дополнительные факторы, влияющие на цены акций. Это соответствует практике фундаментального и технического анализа.

Результат:
По результатам исследования RNN модель, получающая в контекст признаки на основе технического анализа и календарных особенностей, показала себя одной из лучших по метрикам Directional Accuracy и MASE, а также по визуальному анализу.

SOFTS (Singular Spectrum Analysis with Forecasting and Trend Separation)

SOFTS — метод, основанный на сингулярном спектральном анализе (SSA), который разлагает временной ряд на сумму компонент с разными частотами и трендами. Этот подход позволяет выделять скрытые паттерны и тренды, что особенно полезно для нестационарных и шумных финансовых данных. SOFTS сочетает в себе преимущества SSA и методов прогнозирования, обеспечивая более точное моделирование сложной структуры временных рядов.

Обоснование выбора гиперпараметров для SOFTS **-модели**

- hidden_size=256, d_core=256 (размер скрытого состояния модели и размерность ядра, используемого в сингулярном спектральном анализе)

  Значение 256 обеспечивает достаточную ёмкость для моделирования сложных нелинейных зависимостей, характерных для финансовых данных, без чрезмерного увеличения вычислительной сложности.

- e_layers=2 (Количество слоёв кодировщика)

  Двухслойная архитектура кодировщика позволяет:

  - Улавливать иерархические зависимости в данных (например, локальные тренды и глобальные рыночные сдвиги).

  - Избегать переобучения, которое возможно при большем числе слоёв на ограниченных финансовых данных.

- d_ff=64 (Размер внутреннего слоя)

  В трансформерных блоках SOFTS этот параметр определяет размерность полносвязного слоя внутри механизма внимания. Значение 64 выбрано как компромисс между способностью моделировать нелинейности и вычислительной эффективностью.

- dropout=0.1
  Дропаут применяется для регуляризации и предотвращения переобучения. Уровень 0.1 был подобран эмпирически.

Результат:

Оказалась худшей из рассматриваемых моделей. Значительно недооценивала все прогнозы, занижая их для каждой акции.

TimesNet

TimesNet — современная архитектура, ориентированная на обработку временных рядов с помощью специализированных нейросетевых блоков, которые эффективно улавливают локальные и глобальные зависимости. Благодаря модульной структуре и способности к масштабированию, TimesNet хорошо подходит для анализа больших объемов финансовых данных с высокой частотой обновления, обеспечивая баланс между точностью и вычислительной эффективностью.

Обоснование выбора гиперпараметров для TimesNet-модели

- hidden_size=32 и conv_hidden_size=32 (размер скрытого состояния модели и размер скрытого слоя в сверточных блоках модели)

  Значения 32 выбраны как компромисс между способностью модели улавливать сложные зависимости и вычислительной эффективностью, что особенно важно при работе с большим числом временных рядов.

- learning_rate=0.001

  Значение скорости обучения 0.001 является стандартным для нейросетевых моделей, обеспечивая баланс между скоростью обучения и стабильностью процесса оптимизации.

StemGNN (Spatio-Temporal Graph Neural Network)

StemGNN — графовая нейросеть, учитывающая пространственно-временную структуру данных. В контексте финансовых рынков она позволяет моделировать взаимосвязи между различными активами (например, акциями), учитывая как временные зависимости, так и корреляции между инструментами. Это особенно важно для комплексного анализа портфелей и выявления системных рисков. StemGNN демонстрирует высокую способность к прогнозированию сложных многомерных временных рядов.

Обоснование выбора гиперпараметров для StemGNN**-модели**

- input_size=24

  Окно входных данных в 24 дня позволяют модели улавливать краткосрочные динамики и тренды, типичные для фондового рынка.

- Остальные гиперпараметры стандартные, как и для других вышеперечисленных моделей

Результаты сравнения моделей

На rolling-валидационном периоде в 14 дней, были замерены показатели MASE, MAPE, SMAPE и Directional_Accuracy.

Для выбора между моделями был сделан приоритет для MASE и Directional Accuracy. Была выведена новая метрика **Harmonic Metric** равная среднему гармоническому MASE (mean) и (1 - Directional Accuracy (mean)).

Таблица с результатами моделей отсортирована по возрастанию **Harmonic Metric.** Чем ниже модель, тем она оказалась хуже на валидации.

	MASE_median	MASE_mean	MAPE_median	MAPE_mean	SMAPE_median	SMAPE_mean	Directional_Accuracy_median	Directional_Accuracy_mean	Type	harmonic_metric
CatBoostRegressor	1.021628	17.127108	0.015493	0.353800	0.007735	0.066256	1.000000	0.741883	ML/DL	0.254285
Chronos_zero_shot	0.869229	1.022899	0.013832	0.016740	0.006924	0.008344	0.642857	0.548701	ML/DL	0.313142
RWD	0.707623	0.882467	0.011901	0.014074	0.005945	0.007019	0.535714	0.509740	Baseline	0.315167
RNN_many_exog	0.790539	0.940294	0.012786	0.014699	0.006400	0.007354	0.571429	0.524351	ML/DL	0.315867
CES	0.758950	0.908864	0.012322	0.014511	0.006152	0.007233	0.464286	0.485390	Baseline	0.328570
RNN_no_exog	0.776577	0.936866	0.013084	0.014841	0.006536	0.007404	0.357143	0.491883	ML/DL	0.329442
AutoARIMA	0.707527	0.887251	0.012067	0.014194	0.006022	0.007080	0.321429	0.443182	Baseline	0.342115
WindowAverage(7)	1.235269	1.468713	0.021859	0.024129	0.010831	0.011889	0.500000	0.548701	Baseline	0.345221
Naive	0.712339	0.883129	0.011933	0.014119	0.005957	0.007039	0.392857	0.426948	Baseline	0.347538
AutoRegressive	0.763405	0.917551	0.012482	0.014782	0.006233	0.007369	0.357143	0.423701	Baseline	0.353974
StemGNN	1.092148	1.476071	0.018691	0.022710	0.009290	0.011201	0.642857	0.534091	ML/DL	0.354131
AutoETS	0.714716	0.885894	0.011920	0.014173	0.005953	0.007068	0.285714	0.405844	Baseline	0.355636
WindowAverage(6)	1.190863	1.424808	0.021308	0.023368	0.010587	0.011545	0.500000	0.524351	Baseline	0.356603
WindowAverage(14)	1.454360	1.713490	0.026854	0.028568	0.013244	0.013994	0.535714	0.538961	Baseline	0.363290
Chronos_finetuned	0.971523	1.152041	0.015734	0.018233	0.007899	0.009155	0.428571	0.469156	ML/DL	0.363396
WindowAverage(5)	1.152022	1.364199	0.019888	0.022275	0.009906	0.011035	0.500000	0.501623	Baseline	0.365024
WindowAverage(2)	0.865799	1.049940	0.014057	0.016748	0.007021	0.008339	0.392857	0.428571	Baseline	0.370037
RNN_few_exog	0.789637	0.970301	0.013206	0.015296	0.006589	0.007624	0.285714	0.389610	ML/DL	0.374685
WindowAverage(3)	0.997822	1.183505	0.016812	0.019035	0.008389	0.009462	0.357143	0.451299	Baseline	0.374892
WindowAverage(4)	1.090919	1.284377	0.018399	0.020791	0.009180	0.010321	0.428571	0.459416	Baseline	0.380454
SOFTS	11.647752	13.301091	0.200546	0.200199	0.112395	0.112183	0.607143	0.573052	ML/DL	0.413670
TimesNet	1.714129	1.978888	0.031019	0.033339	0.015213	0.016268	0.321429	0.441558	ML/DL	0.435534

CatBoostRegressor

При визуальном анализе прогнозов модели CatBoostRegressor была замечена высокая дисперсия выдаваемых ответов, большие ошибки на конкретных тикерах. Это видно по очень высокому значению MASE_mean (значение, соответствующее наивному алгоритму – 1).

Эта модель предсказывает нереалистичные значения цен и не угадывает их значения, хотя верно предсказывает направление движения цен. Было решено отбросить эту модель в пользу других.

Chronos_zero_shot

Полностью предобученная модель Chronos оказалась на втором месте по гармонической метрике. Тем не менее, визуальный анализ показал, что модель не адаптируется к конкретным акциям, а запоминает лишь всю конъюнктуру рынка, а также имеет большие нереалистичные промахи с предсказаниями для акций с низкой ценой (например, акции FEES в районе 1 рубля за акцию). В среднем модель показывает неплохие результаты, но из-за случаев очевидно неправильных ответов было решено отбросить ее.

RWD

Бейзлайн-алгоритм. Визуальный анализ предсказаний показал, что ответы этой модели стабильны, адаптивны для каждой акции. Модель также выделяется хорошим сочетанием угадывания направления движения цены и ее конкретного значения.

RNN

Рекуррентная нейронная сеть, получающая на вход исторические данные торгов, признаки на основе технического анализа и на основе календарных дат. Так же показала себя хорошо и стабильно при визуальном анализе.

В процессе сравнения моделей было принято решение ансамблировать модели для получения более совершенной.

Ансамблированная модель

Ансамблированная модель является сочетанием моделей RWD и RNN. Ее предсказание – это среднее значение предсказаний двух представленных моделей.

Результат на валидационной выборке:

	MASE_median	MASE_mean	MAPE_median	MAPE_mean	SMAPE_median	SMAPE_mean	Directional_Accuracy_median	Directional_Accuracy_mean	Type	harmonic_metric
CatBoostRegressor	1.021628	17.127108	0.015493	0.353800	0.007735	0.066256	1.000000	0.741883	ML/DL	0.254285
ensemble_mean	0.765434	0.906407	0.012192	0.014203	0.006098	0.007094	0.607143	0.524351	ML/DL	0.311950
Chronos_zero_shot	0.869229	1.022899	0.013832	0.016740	0.006924	0.008344	0.642857	0.548701	ML/DL	0.313142
RWD	0.707623	0.882467	0.011901	0.014074	0.005945	0.007019	0.535714	0.509740	Baseline	0.315167
RNN_many_exog	0.790539	0.940294	0.012786	0.014699	0.006400	0.007354	0.571429	0.524351	ML/DL	0.315867
CES	0.758950	0.908864	0.012322	0.014511	0.006152	0.007233	0.464286	0.485390	Baseline	0.328570
RNN_no_exog	0.776577	0.936866	0.013084	0.014841	0.006536	0.007404	0.357143	0.491883	ML/DL	0.329442
AutoARIMA	0.707527	0.887251	0.012067	0.014194	0.006022	0.007080	0.321429	0.443182	Baseline	0.342115
WindowAverage(7)	1.235269	1.468713	0.021859	0.024129	0.010831	0.011889	0.500000	0.548701	Baseline	0.345221
Naive	0.712339	0.883129	0.011933	0.014119	0.005957	0.007039	0.392857	0.426948	Baseline	0.347538
AutoRegressive	0.763405	0.917551	0.012482	0.014782	0.006233	0.007369	0.357143	0.423701	Baseline	0.353974
StemGNN	1.092148	1.476071	0.018691	0.022710	0.009290	0.011201	0.642857	0.534091	ML/DL	0.354131
AutoETS	0.714716	0.885894	0.011920	0.014173	0.005953	0.007068	0.285714	0.405844	Baseline	0.355636
WindowAverage(6)	1.190863	1.424808	0.021308	0.023368	0.010587	0.011545	0.500000	0.524351	Baseline	0.356603
WindowAverage(14)	1.454360	1.713490	0.026854	0.028568	0.013244	0.013994	0.535714	0.538961	Baseline	0.363290
Chronos_finetuned	0.971523	1.152041	0.015734	0.018233	0.007899	0.009155	0.428571	0.469156	ML/DL	0.363396
WindowAverage(5)	1.152022	1.364199	0.019888	0.022275	0.009906	0.011035	0.500000	0.501623	Baseline	0.365024
WindowAverage(2)	0.865799	1.049940	0.014057	0.016748	0.007021	0.008339	0.392857	0.428571	Baseline	0.370037
RNN_few_exog	0.789637	0.970301	0.013206	0.015296	0.006589	0.007624	0.285714	0.389610	ML/DL	0.374685
WindowAverage(3)	0.997822	1.183505	0.016812	0.019035	0.008389	0.009462	0.357143	0.451299	Baseline	0.374892
WindowAverage(4)	1.090919	1.284377	0.018399	0.020791	0.009180	0.010321	0.428571	0.459416	Baseline	0.380454
SOFTS	11.647752	13.301091	0.200546	0.200199	0.112395	0.112183	0.607143	0.573052	ML/DL	0.413670
TimesNet	1.714129	1.978888	0.031019	0.033339	0.015213	0.016268	0.321429	0.441558	ML/DL	0.435534

Ансамблированная модель имеет второй лучший результат по гармонической метрике. При этом визуальный анализ подтверждает реалистичность и низкий разброс предсказаний модели.

Результаты ансамблированной модели на тестовых данных:

	MAPE	Directional_Accuracy
ensemble_mean	1.80%	38.64%

Код с инференсом модели: github

Раздел 5. Разработка телеграм-бота

Ссылка на доступ к боту: Stock Price Predictor bot

Для сервиса по прогнозированию цен акций разработка телеграм-бота была обусловлена необходимостью предоставить пользователям удобный и интуитивно понятный интерфейс для быстрого доступа к актуальным прогнозам.

Для реализации телеграм-бота использовалась популярная и удобная библиотека python-telegram-bot. Эта библиотека обеспечивает простой и интуитивно понятный интерфейс для взаимодействия с Telegram Bot API, поддерживает обработку команд, сообщений, создание кнопок и клавиатур.

Основные смысловые части кода бота:

Обработка команд и сообщений
С помощью обработчиков (CommandHandler, MessageHandler) бот реагирует на команды пользователя, такие как /start, и на текстовые сообщения. Это позволяет реализовать диалоговый интерфейс, где пользователь выбирает интересующие акции и получает прогнозы.

Доступ к данным прогнозов из базы данных
Для быстрого и надёжного получения прогнозов бот подключается к локальной базе данных SQLite. При запросе пользователя бот выполняет SQL-запрос, извлекает актуальные прогнозы и исторические данные по выбранному тикеру, что обеспечивает минимальные задержки и автономность работы.

Генерация и отправка графиков
Для визуализации прогнозов бот строит графики с помощью библиотек визуализации (например, matplotlib), где отображаются исторические цены и прогнозируемое значение с цветовой индикацией направления движения. График конвертируется в изображение и отправляется пользователю.

Интерактивный интерфейс с кнопками
Используются встроенные клавиатуры Telegram, позволяющие пользователю легко выбирать акции, переключаться между ними и получать дополнительную информацию без необходимости вводить текст вручную.

Логирование и обработка ошибок
В коде предусмотрено логирование действий и ошибок, что помогает отслеживать работу бота и быстро устранять возможные сбои.

Асинхронность
Поддержка async/await позволяет обрабатывать множество запросов одновременно, что важно для масштабируемости.

Заключение

По результатам проекта были выполнены все поставленные задачи:

- Построены различные эконометрические и модели машинного обучения
- Выбрана, обучена и готова к предсказаниям лучшая модель
- Разработан телеграм-бот для обеспечения ежедневного удобного доступа к результатам модели

Итоговая модель является комбинацией статистической модели RandomWalkWithDrift и Deep Learning модели RNN. Модель использует исторические показатели технического анализа, календарные признаки и данные по биржевым торгам.

Список литературы

Теоретические статьи:

1. Ansari A. F., Stella L., Turkmen C., Zhang X., Mercado P., Shen H., Shchur O. et al. Chronos: Learning the language of time series // arXiv preprint arXiv:2403.07815. 2024. Режим доступа: https://www.amazon.science/code-and-datasets/chronos-learning-the-language-of-time-series (дата обращения: 29.05.2025).

2. Han L., Chen X.-Y., Ye H.-J., Zhan D.-C. SOFTS: Efficient multivariate time series forecasting with series-core fusion // arXiv preprint arXiv:2404.14197. 2024. Режим доступа: https://arxiv.org/pdf/2404.14197 (дата обращения: 29.05.2025).

3. Wu H., Hu T., Liu Y., Zhou H., Wang J., Long M. TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis // arXiv preprint arXiv:2210.02186. 2022. Режим доступа: https://arxiv.org/abs/2210.02186 (дата обращения: 29.05.2025).

4. Cao D., Wang Y., Duan J., Zhang C., Zhu X., Huang C., Tong Y. et al. Spectral temporal graph neural network for multivariate time-series forecasting // Advances in Neural Information Processing Systems. 2020. Vol. 33. P. 17766–17778. Режим доступа: https://proceedings.neurips.cc/paper/2020/file/cdf6581cb7aca4b7e19ef136c6e601a5-Paper.pdf (дата обращения: 29.05.2025).

Ресурсы по библиотекам и инструментам:

1. Официальная документация pandas: https://pandas.pydata.org/
2. Официальная документация PyTorch: https://pytorch.org/
3. Официальная документация API MOEX: https://pypi.org/project/apimoex/
4. Официальная документация StatsForecast, MLForecast, NeuralForecast: https://nixtlaverse.nixtla.io/

Приложение

Код на Python, использованный для проведения всех этапов исследования, доступен для просмотра и скачивания по ссылке на репозиторий GitHub:

Ссылка на репозиторий GitHub