Санкт–Петербургский государственный университет

Ковалев Святослав Сергеевич

Выпускная квалификационная работа

Прогнозирование временных рядов методами машинного обучения

Уровень образования: бакалавриат
Направление 01.03.02 «Прикладная математика и информатика»
Основная образовательная программа СВ.5005.2015
«Прикладная математика, фундаментальная информатика и программирование»
Профиль «Математическое и программное обеспечение вычислительных машин»

Научный руководитель:
профессор кафедры математической теории игр и статистических решений, д.т.н. Буре Владимир Мансурович

Рецензент:
доцент кафедры математического моделирования энергетических систем, к.ф. - м.н. Свиркин Михаил Владимирович

Санкт-Петербург
2021 г.

Содержание

Введение

Прогнозирование временных рядов играет важную роль в задачах экономики, финансов, прогнозировании погоды, анализе электроэнергии и прочих [4].

Эта задача достаточно изучена для временных рядов, обладающих свойством стационарности - статистическим свойством, при котором основные характеристики ряда остаются неизменными со временем. Стационарные ряды успешно прогнозируются линейными моделями, такими как ARIMA, GARCH, Exponential smoothing и другими. Однако при работе с реальными данными временные ряды зачастую оказываются нестационарными. Такие ряды стараются свести к стационарным и прогнозировать их уже известными методами. В этой работе мы рассмотрим набор методов машинного обучения для прогнозирования временных рядов, не использующих свойство стационарности и применим описанные методы к данным цен биржевых торгов.

Прогнозирование цен биржевых торгов позволяет крупным компаниям принимать стратегические решения, а частным трейдерам совершать выгодные сделки. Цель этой работы - показать эффективность алгоритмов машинного обучения в вопросе прогнозирования цен биржевых торгов. Чтобы показать эффективность алгоритмов, были использованы исторические данные, предоставляемые Yahoo Finance [16], по нескольким финансовым инструментам: цены на акции компании Tesla, цена криптовалюты Bitcoin по отношению к доллару, цена доллара по отношению к евро и др.

Помимо задачи прогнозирования временных рядов, также будет рассмотрен вопрос вопроизводимости вычислений. Исследователи часто сталкиваются с двумя проблемами после завершения исследования.

Во-первых, при разработке методов машинного обучения, исследователь получает положительные результаты в специально настроенном окружении. Повторно настроить такое же окружение бывает просто невозможно и результат исследования невозможно переиспользовать для решения этой же задачи на продуктивном сервере.

Во-вторых, зачастую обнаруживается, что метод, примененный для

решения задачи, невозможно использовать для решения похожих задач. Побочная цель этого исследования - обеспечить воспроизводимость представленного решения. Для этого будут использоваться такой инструмент как Snakemake [6].

Постановка задачи

Даны временные ряды

$$X_i = \{x_{i,t}, t = \overline{1,k}\}, \quad i = \overline{1,m},$$

где m - общее количество рядов, а размер всех рядов совпадает и равен k. Цель - предсказать значение всех рядов с лагом n.

Для этого требуется построить такую модель $A(X_i)$, что

$$A(X_i) = x_{i,k+n}.$$

При этом модель должна при кросс-валидации минимизировать одну из метрик: MAE (1), MAPE (2), RMSE (3).

MAE (Mean absolute error):

$$MAE = \sum_{i=1}^{n} \frac{|y_i - x_i|}{n} \tag{1}$$

MAPE (Mean absolute percentage error):

$$MAPE = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{y_i - x_i}{y_i} \right| \tag{2}$$

RMSE (Root mean square error)

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(x_i - y_i)^2}{n}} \tag{3}$$

Обзор литературы

Научные работы о прогнозировании временных рядов появились еще в 1970 году в труде Бокса [2]. Бокс рассматривает линейные стационарные модели и некоторые системы, сводящиеся к ним. В частности, рассматривается модель ARIMA и ее производные, учитывающие сезонность и внешние факторы. Немногим раньше появляется модель экспоненциального сглаживания [15].

Модели нейронных сетей появляются в 1987 году [7] и сразу дают положительные результаты в области прогнозирования нелинейных систем. Значительный вклад в развитие прогнозирования временных рядов внесла работа о LSTM [13] в 2012 году, в которой рассматривалась модификация рекурентной нейронной сети для задач распознавания речи. Появляется множество статей, в которых модель LSTM с некоторыми модификациями превосходит в точности все ранее известные модели [5] [3] [11].

В это же время исследуется вопрос построения ансамблей моделей, когда на основе множества «простых» моделей обучается другой алгоритм регрессии, минимизирующий общую ошибку. Такой подход применяли зарубежные исследователи в своей работе о построении ансамблевых моделей на основе глубоких нейронных сетей [9]. Также бразильские исследователи сравнивали между собой ансамблевые модели, построенные на основе бэггинга, бустинга и стекинга для данных из области агрокультуры [10].

Помимо моделей прогнозирования большую роль в качество прогноза вкладывает предобработка данных и feature extraction.

Глава 1. Прогнозирование временных рядов

1.1 Методы прогнозирования

Для того, чтобы оценить качество моделей машинного обучения, необходимо построить множество моделей и сравнить их между собой для каждого конкретного ряда. Для прогнозирования использовались как классические модели, такие как ARIMA, VAR, так и регрессионные модели машинного обучения - Wavelet и Stacking. Каждая из моделей сравнивалась с «наивной» моделью, которая предсказывала последнее известное значение на весь горизонт прогнозирования.

Ниже приводится краткий обзор каждого использованного метода.

1.1.1 ARMA

Рассмотрим модель ARMA:

$$X_t = c + \varepsilon_t + \sum_{i=1}^{p} \alpha_i X_{t-i} + \sum_{i=1}^{q} \beta_i \varepsilon_{t-i}$$

где c - константа, ε_t - белый шум, а $\alpha_1, \ldots, \alpha_p$ и $\beta_1, \ldots, \beta_q$ - авторегрессионные коэффициенты и коэффициенты скользящего среднего.

Такая модель может быть использована только для прогнозирования стационарных временных рядов. Если же в задаче фигурирует нестационарный ряд, то используют модель ARIMA.

1.1.2 ARIMA

Модель ARIMA [19] является классическим методом эконометрики для прогнозирования временных рядов. Применение ARIMA для прогнозирования биржевых цен уже рассматривалось ранее [20].

Модель ARIMA является производной от модели ARMA. Для сведения нестационарного ряда к стационарному используется операция взятия разности или дифференцирование временного ряда. Количество раз, которое исходный ряд дифференцируется перед тем как попасть в модель

ARMA, является параметром, который обычно обозначают как d.

1.1.3 VAR

Модель VAR [18] позволяет находить линейные зависимости между компонентами многомерных временных рядов. VAR - обобщение модели AR на многомерный случай. Считается, что все рассматриваемые ряды зависимы, и что каждый из этих рядов нужно предсказать на n шагов вперед. Модель VAR можно записать следующим образом:

$$y_t^i = a_0^i + \sum_{j=1}^{k} a_{1j}^i y_{t-1}^j + \sum_{j=1}^{k} a_{2j}^i y_{t-2}^j + \ldots + \sum_{j=1}^{k} a_{pj}^i y_{t-p}^j + \varepsilon_t^i,$$

где $y^i, i = 1, \ldots, k - i$-й временной ряд, а a_i^j - коэффициенты авторегрессии. Более компактно можно записать это уравнение в векторной форме:

$$y_t = a_0 + \sum_{m=1}^{p} A_m y_{t-m} + \varepsilon_t,$$

где A_m - матрицы элементов a_{mj}^i

1.1.4 Wavelet

В основе модели Wavelet [14] лежит дискретное вейвлет преобразование. Исходный ряд раскладывается на k рядов, каждый из которых предсказывается по отдельности при помощи любой регрессионной модели (в нашем случае использовалась ARIMA), после чего выполняется обратное преобразование. Использовалась реализация вейвлет функций из пакета PyWavelets [8]. В качестве вейвлет-функций используются койфлеты, так как показали наилучший результат в результате полного поиска по сетке среди всех семейств вейвлетов, представленных в библиотеке. Схему разложения можно видеть на рис. 1.

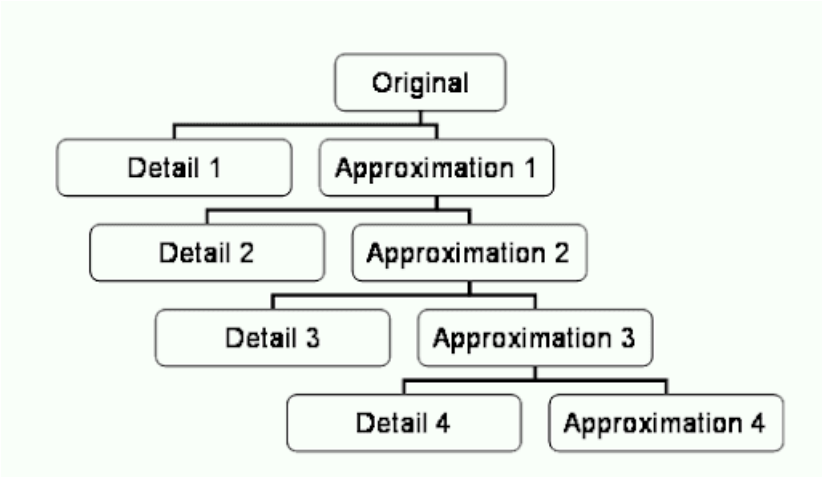

Рис. 1: Схема разложения исходного ряда. Ряды коэффициентов Approximation 4, Detail 1 - Detail 4 предсказываются регрессионной моделью.

1.1.5 Naive one step

Naive one step - «наивная» модель, которая предсказывает последнее известное значение на весь горизонт прогнозирования. Такая модель не несет в себе никакой прогностической фунции, но с помощью нее можно определить, насколько хорошо построенная модель обобщает данные. Так, если наивная модель показывает результаты какой-либо метрики лучше другой модели, это означает, что вторая модель недостаточно хорошо описывает данные в смысле выбранной метрики.

1.1.6 Naive multi step

Naive multi step - «наивная» модель, которая в качестве предсказания берет последние n значений временного ряда. Модель аналогична предыдущей модели, но лучше подходит под случай, когда в данных присутствует периодичность или сезонность.

1.1.7 Stacking

Stacking - ансамблевая модель, основанная на градиентном бустинге над решениями, описанными выше. Как было показано, использование ансамблей решений помогают построить лучшее решение.

В нашем случае в качестве модели, обобщающей предсказания, использовалась RandomForest из библиотеки scikit-learn, так как она дала лучшее качество, по сравнению с линейной регрессий (линейной комбинацией решений) и градиентным бустингом из библиотеки scikit-learn. Для каждой модели производился полный перебор гиперпараметров по сетке.

1.2 Оценка качества моделей

Оценка качества моделей прогнозирования временных рядов отличается от оценки качества моделей регрессии в случаях, когда необходимо давать предсказание больше, чем на один шаг вперед.

Пусть на вход модели поступает последние k наблюдений ряда, а на выходе она предсказывает следующие n значений. При этом, чтобы избежать переобучения и не «заглядывать в будущее», модель обучается только на данных, предшествующих последнему наблюдению, которое было передано в модель. Тогда для каждого момента времени $t, t > k$ мы можем дать предсказание и сравнить его с фактическим значением ряда в этот же момент времени, например, посчитав среднеквадратичное отклонение или любую другую метрику, которая бы описывала качество предсказания. Такой подход называется кросс-валидацией для временных рядов.

Этот подход можно обобщить на случай, когда необходимо давать прогноз на каждый из n шагов вперед.

Пусть, $X = \{x_i | i = \overline{1, m}\}$ - исходный временной ряд длины m. Обозначим срез временного ряда от i до j как $X_i^j = \{x_t | t = \overline{i, j}\}, i < j$.

Тогда формально процесс кросс-валидации можно описать так:

1. Выборка делится на тренировочную X_i^j и тестовую X_{j+1}^m.

2. Модель обучается на тренировочной выборке.

3. Тренировочная выборка прогнозируется на n точек вперед. Обозначим предсказание p_j^1, если оно строится от точки j, как в (4) на n шагов вперед.

$$p_j^1 = A(X_i^j) \tag{4}$$

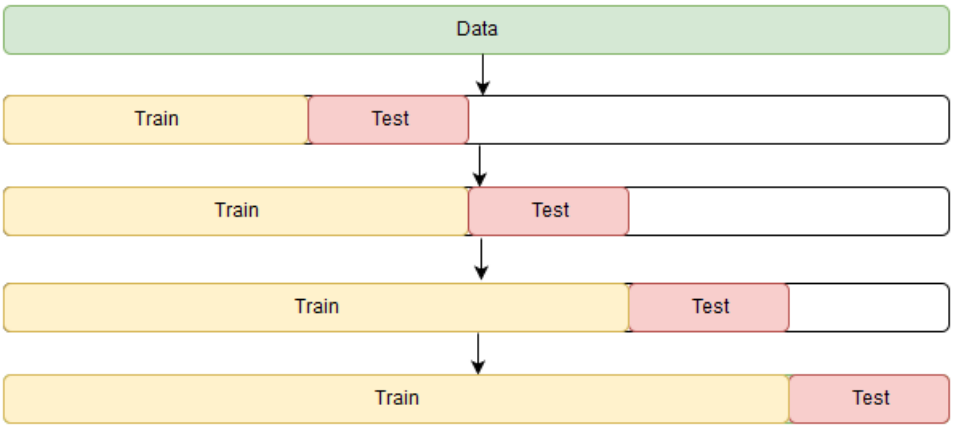

Рис. 2: Иллюстрация кросс-валидации для временных рядов.

4. Считается целевая метрика (5), сравнивая полученное предсказание с фактическими значениями в этот же момент времени.

$$M_j^1 = metric(p_j^1, X_j^{j+n}) \tag{5}$$

5. Первая точка из тестовой выборки перемещается в тренировочную:

$$i := i + 1,$$
$$j := j + 1.$$

6. Повторяем процесс до тех пор, пока $j+n < m$, то есть пока в тестовой выборке больше, чем n точек.

7. Усредним значение полученного массива метрик.

В предложенном алгоритме особое внимание стоит уделить процессу подсчета метрики. Можно считать близость двух векторов (вектор предсказаний и вектор фактических цен), либо можно считать близость предсказания в последней точке прогноза.

В первом случае мы оценим модель на то, насколько точно она прогнозирует общую картину изменения величины (скачки цен внутри периода, если говорить о биржевых цена).

При этом, если задача состоит в предсказании n значения, стоит считать метрику именно по последней точке предсказания. В таком случае мы оценим насколько точно модель прогнозирует цену на заданный горизонт, игнорируя общую картину.

В этой работе будем использовать второй подход, так как его проще визуализировать и интерпретировать. То есть, оцениваться будет только прогноз модели в последний день прогноза.

В качестве метрики будем использовать:

- Mean Absolute Error (MAE) (1);
- Mean Absolute Percentage Error (MAPE) (2);
- Root Mean Squared Error (RMSE) (3).

1.3 Воспроизводимость вычислений

1.3.1 Snakemake

Для того, чтобы полученные результаты были достоверными и воспроизводимыми, а также чтобы полученные модели можно было использовать как фреймворк для произвольных данных в будущем, был построен конвейер (workflow) для обработки данных. Для реализации конвейера использовался инструмент Snakemake [6]. Помимо Snakemake существует множество других инструментов, например Luigi [12], Airflow [1] и многие другие. Выбор пал на Snakemake, так как он очень прост в изучении, расширяет язык Python, но не связан с ним напрямую, и позволяет строить гибкие конвейеры, используя высокий уровень абстракции.

Snakemake по своей сути очень похож на CMake, но более прост в использовании и адаптирован под создание конвейеров для анализа данных. Большое внимание уделено вопросам масштабирования и параллельного исполнения, так как фрейворк часто используется для исследования задач биоинформатики, в которых зачастую фигурируют большие данные. Как известно, в Python нет возможности эффективно производить параллельные вычисления средствами самого языка из-за GIL (Global Interpreter

Lock). Обычно эта проблема решается использованием отдельных процессов (workers), каждый из которых решает свою задачу, и которые способны общаться между собой через асинхронное хранилище (например, Redis). Snakemake решает проблему масштабирования на своей стороне и предоставляет простой интерфейс для реализации параллельных вычислений как в рамках одной машины, так и для вычислительных кластеров.

Snakemake позволяет определять «правила» (rules) для обработки данных. Каждое правило может иметь входные и выходные файлы. Правило должно генерировать из входного файла выходной. Если snakemake не может найти входной файл в файловой системе, то ищется другое правило, которое генерирует этот файл. Таким образом, получается ациклический граф работ (DAG), при помощи которого snakemake планирует и исполняет все необходимые работы для того, чтобы получить результат.

Для конвейера можно определить параметры в специальном конфигурационном файле. Например, это может быть URL-адрес с которого будут загружаться данные, параметры моделей, перечисление названий моделей, метрики для оценки качества моделей и прочее. Параметризация конвейера позволяет быстро и качественно проводить исследование, не нарушая логику других экспериментов.

Каждое правило в конвейере - python-скрипт, которому через аргументы командной строки передают пути к файлам и прочие параметры.

Для решения задачи использовался граф работ, изображенный на рис. 3.

Весь исходный код можно использовать как внешнюю python-библиотеку, поверх которой возможно создать API и использовать как микросервис для предсказаний.

1.3.2 Описание конвейера

Можно выделить следующие шаги, исполняемые конвейером:

1. Загружаются данные через API Yahoo finance по интересующим тикерам (котировкам) за указанный временной интервал. Для работы с API Yahoo finance используется python-библиотека yfinance.

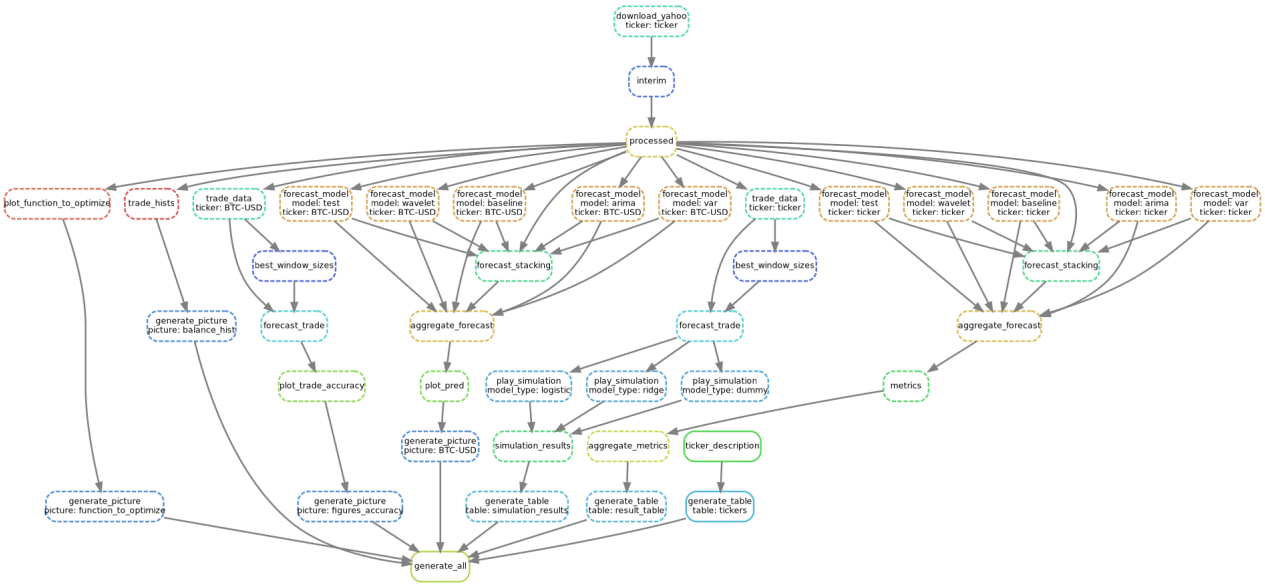

Рис. 3: Ациклический граф работ.

2. Преобразование исходных данных в удобный формат. Обработку исходных данных можно поделить на следующие шаги:

   (a) raw - «сырые» данные, только что загруженные из источника.

   (b) interim - данные, которые прошли первичную обработку. В данных отсутствуют лишние колонки и строки. Набор исходных файлов агрегируется в один, или наоборот, исходный файл разбивается на несколько логических файлов.

   (c) processed - готовый датасет, который готов к загрузке в модель. Такие данные получаются из interim данных, в которых заполняются пропуски и генерируются дополнительные признаки.

   Использование таких шагов для обработки позволяет реже перезапускать весь конвейер и упрощает обработку данных.

3. Использование моделей для построения прогнозов по заданным датам и генерации тестовой выборки. Модели дают предсказание от стартовой даты до конечной даты, которые указываются в конфигурационном файле. Также для моделей указывается горизонт прогнозирования. Итоговое предсказание модели представляет из себя

таблицу, у которой в строках указаны даты, от которых ведется предсказание, а порядковый номер столбца означает горизонт прогнозирования. Например, если предсказание необходимо дать с 1 января 2019 года по 30 января 2020 года с горизонтом в 3 дня, то в таблице будет 30 строк и 3 столбца. Тестовые значения генерируются аналогичным образом, только вместо прогноза используются настоящие значения временных рядов.

4. Полученные данные используются для генерации графиков и подсчета метрик. На этом этапе читаются все предсказания, полученные на прошлом этапе и идет оценка результатов прогнозирования. По результатам проверки сохраняется таблица со всеми полученными метриками. Также отрисовываются графики по каждому прогнозу 4.

5. Генерируется код LaTeX для вставки графиков и таблиц с результатами. На этом шаге используются графики и таблицы с прошлого шага. Таблицы из формата csv транслируются в LaTeX. Пути до изображений также вставляются в LaTeX. Параметры таблиц и изображений (например, ширина столбцов, расположение на странице, label) лежат в отдельном конфигурационном файле.

6. LaTeX шаблон компилируется и генерирует pdf-файл с оформленными графиками и таблицами. В случае проведения нового эксперимента, добавления новой модели или использования новых данных, запуском одной команды pdf-файл собирается заново, из-за чего ошибки в цифрах в итоговом документе исключены.

Также используется примитивное версионирование: результаты каждого запуска сохраняются в отдельную папку и архивируются, сохраняя временную метку и автора запуска конвейера.

В случае работы нескольких человек над проектом возможна организация версионирования через облачное хранилище. После каждого запуска идет загрузка полученного архива в облачное хранилище, а на локальную машину загружаются все отсутствующие исследования, которые были про-

ведены до этого момента. Такой подход решает проблему совместной работы над проектом и позволяет не потерять полученные результаты.

Глава 2. Эксперимент

2.1 Описание исходных данных

Были взяты данные о ценах и объемах биржевых торгов с Yahoo Finance [16]. Yahoo Finance предоставляет данные о «тикерах» (ticker) - торговых парах. Каждый тикер имеет свой код. Например, акции компании Tesla, торгующиеся в долларах, имеют код TSLA. Тикеры, которые были взяты для анализа, представлены в таблице 1.

Таблица 1: Биржевые показатели, взятые для анализа

Код инструмента	**Показатель**
CL=F	Цена за баррель сырой нефти в долларах
EURUSD=X	Цена евро к доллару
BTC-USD	Цена биткоина в долларах
GC=F	Цена золота в долларах
TSLA	Цена акции компании Tesla в долларах

API Yahoo Finance предоставляет по умолчанию данные за последние три года с частотой в один день. За каждый день предоставляются данные о минимальной и максимальной цене сделки, об открывающей и закрывающей цене торгов, а также об объеме торгов. Будем использовать эти данные для обучения.

Для предсказания был выбран промежуток с 1 февраля 2021 года по 30 марта 2021 года. Прогноз строился на 10 дней вперед.

2.2 Подготовка данных

2.2.1 Этап предобработки raw

Этот этап необходим для того, чтобы сохранить загруженные данные. В случае, если дальше в конвейере произойдет ошибка, можно будет использовать уже загруженные данные.

2.2.2 Этап предобработки interim

На этом этапе выбирается нужный временной промежуток в данных. Например, если были загружены данные за последние 10 лет, а для прогнозирования необходимо использовать данные за последний год, то это отсечение ненужных данных произойдет на этом этапе.

Если в данных есть пропуски, то они заполняются последним известным значением. Если пропущены первые значения в ряде, они заполняются первым известным значением.

2.2.3 Этап предобработки processed

На этом этапе из данных выделяются необходимые столбцы, а именно:

1. Open - цена первой сделки за выбранный промежуток торгов.
2. Close - цены последней сделки за выбранный промежуток торгов.
3. High - наивысшая цена за выбранный промежуток торгов.
4. Low - наименьшая цена за выбранный промежуток торгов.

Это необходимо сделать, так как в исходных данных есть лишние столбцы, в которых нет нужды.

2.3 Результаты

Для каждого временного ряда применялась каждая из моделей. Модель VAR применялась сразу для всех временных рядов. Сравнение результатов происходило по метрикам MAE (1), MAPE (2) и RMSE (3).

Результаты можно наблюдать в таблице 2. Также результат прогноза для тикера BTC-USD можно видеть на рис. 4.

Как видно из таблицы и графика, наивная модель дает результат, который не всегда получается превзойти. Это объясняется тем, что на ряд влияют внешние неизвестные факторы, которыми нельзя пренебречь.

По тикеру CL=F лучшим предсказанием оказалась модель baseline. Это объясняется тем, что по этому инструменту цена изменяется медленно, и больше зависит от внешних факторов, чем от самого себя.

При анализе результатов по тикеру EURUSD=X следует использовать метрику MAPE, так как она использует относительные значения. Значения метрик MAE и RMSE близки к нулю, так как цена тикера близка к нулю, и их сравнивать проблематично.

Таблица 2: Результаты

ticker	metric	model	value
CL=F	MAE	wavelet	2.7
CL=F	MAPE	wavelet	4.3
CL=F	RMSE	wavelet	3.2
CL=F	MAE	baseline	2.4
CL=F	MAPE	baseline	3.9
CL=F	RMSE	baseline	3.0
CL=F	MAE	arima	4.0
CL=F	MAPE	arima	6.5
CL=F	RMSE	arima	4.5
CL=F	MAE	var	2.5
CL=F	MAPE	var	4.0
CL=F	RMSE	var	3.1
CL=F	MAE	stacking	2.6
CL=F	MAPE	stacking	4.2
CL=F	RMSE	stacking	3.2
EURUSD=X	MAE	wavelet	0.0
EURUSD=X	MAPE	wavelet	0.7
EURUSD=X	RMSE	wavelet	0.0
EURUSD=X	MAE	baseline	0.0
EURUSD=X	MAPE	baseline	0.8
EURUSD=X	RMSE	baseline	0.0
EURUSD=X	MAE	arima	0.0
EURUSD=X	MAPE	arima	1.2
EURUSD=X	RMSE	arima	0.0
EURUSD=X	MAE	var	0.0
EURUSD=X	MAPE	var	0.6
EURUSD=X	RMSE	var	0.0
EURUSD=X	MAE	stacking	0.0
EURUSD=X	MAPE	stacking	0.8

EURUSD=X	RMSE	stacking	0.0
BTC-USD	MAE	wavelet	4747.1
BTC-USD	MAPE	wavelet	9.5
BTC-USD	RMSE	wavelet	5909.0
BTC-USD	MAE	baseline	4730.2
BTC-USD	MAPE	baseline	9.2
BTC-USD	RMSE	baseline	5702.4
BTC-USD	MAE	arima	5273.8
BTC-USD	MAPE	arima	12.2
BTC-USD	RMSE	arima	7535.5
BTC-USD	MAE	var	5588.7
BTC-USD	MAPE	var	9.6
BTC-USD	RMSE	var	7019.4
BTC-USD	MAE	stacking	4523.9
BTC-USD	MAPE	stacking	8.9
BTC-USD	RMSE	stacking	5451.9
GC=F	MAE	wavelet	28.7
GC=F	MAPE	wavelet	1.6
GC=F	RMSE	wavelet	37.7
GC=F	MAE	baseline	29.3
GC=F	MAPE	baseline	1.7
GC=F	RMSE	baseline	39.2
GC=F	MAE	arima	48.1
GC=F	MAPE	arima	2.7
GC=F	RMSE	arima	61.1
GC=F	MAE	var	29.8
GC=F	MAPE	var	1.7
GC=F	RMSE	var	39.8
GC=F	MAE	stacking	30.3
GC=F	MAPE	stacking	1.7
GC=F	RMSE	stacking	41.2

TSLA	MAE	wavelet	58.2
TSLA	MAPE	wavelet	8.5
TSLA	RMSE	wavelet	68.8
TSLA	MAE	baseline	62.2
TSLA	MAPE	baseline	9.0
TSLA	RMSE	baseline	72.7
TSLA	MAE	arima	82.6
TSLA	MAPE	arima	10.6
TSLA	RMSE	arima	105.2
TSLA	MAE	var	73.3
TSLA	MAPE	var	10.3
TSLA	RMSE	var	85.0
TSLA	MAE	stacking	62.5
TSLA	MAPE	stacking	8.8
TSLA	RMSE	stacking	74.4

2.4 Моделирование торговли на бирже

2.4.1 Постановка задачи

Помимо классической задачи прогонозирования временных рядов может быть полезно использовать альтернативный подход. Будем моделировать торговлю на бирже. Пусть в нулевой момент имеется C долларов, на которые можно купить актив по цене $price_i$ в момент времени i. Торговля продолжается на протяжении определенного периода (неделя, месяц, год), в конце которого все активы продаются по текущей цене. Задача - максимизировать итоговое количество долларов. Также для удобства будем пользоваться формулой (6)

$$profit = \frac{revenue - cost}{cost} * 100 \tag{6}$$

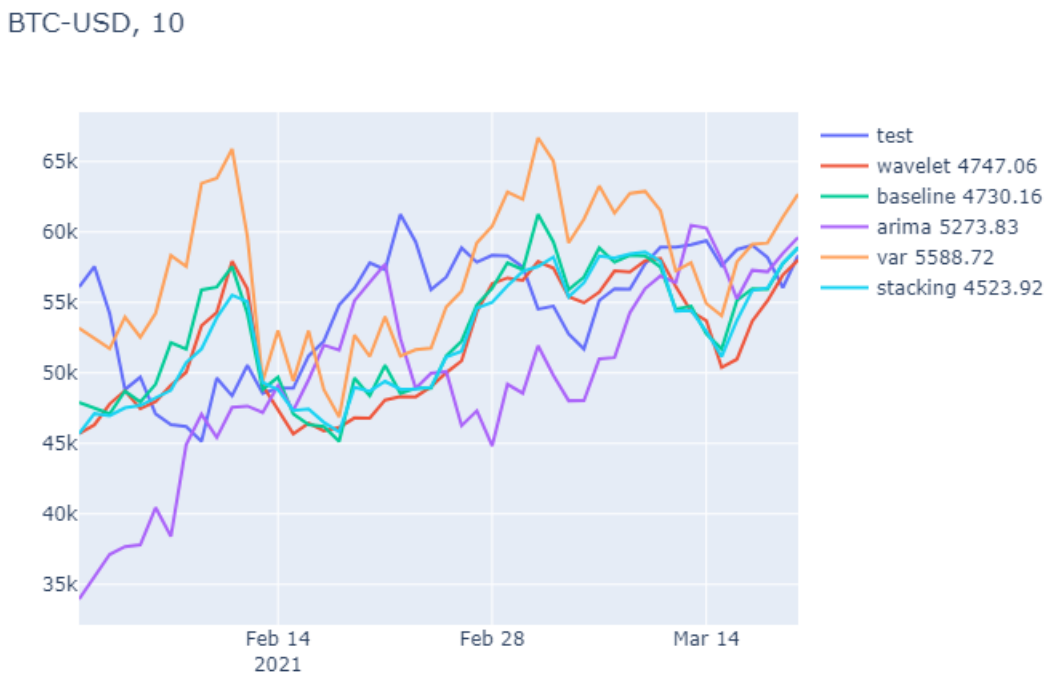

Рис. 4: Цена биткоина в долларах. На легенде числа обозначают точность модели (MAE).

Сведем задачу к задаче классификации. Каждую точку исходного временного ряд разметим на три класса: «sell», «buy», «hold»:

- sell - означает, что скоро цена значительно снизится, а следовательно нужно продавать активы;
- buy - означает, что скоро цена значительно повысится, а следовательно нужно покупать активы;
- hold - означает, что цена в ближайшее время значительно не изменится, а значит ничего делать не нужно.

Выберем горизонт прогнозирования n и сформируем обучающую выборку. Для каждой точки посчитаем изменение цены в процентах в момент i (7)

$$\Delta p_i = \frac{price_{i+n} - price_i}{price_i} * 100 \tag{7}$$

Выберем порог t, начиная с которого будем считать, что изменение цены значительное и выберем класс для каждой точки по правилу (8)

$$class_i = \begin{cases} sell & \text{if } \Delta p_i < -t \\ buy & \text{if } \Delta p_i > t \\ hold & \text{if } |\Delta p_i| < t \end{cases} \tag{8}$$

Порог t можно выбирать вручную в зависимости от целей трейдера, но на случай, когда необходимо проанализировать большое количество временных рядов, хотелось бы иметь алгоритм автоматического подбора порога t.

Для того, чтобы в дальнейшем было проще оценивать результаты задачи классификации, будем исходить из предпосылки, что данные разбиты на равные классы.

Будем решать задачу одномерной оптимизации, в которой t выступает в качестве независимой переменной.

Введем понятие сбалансированности. Пусть есть выборка из N элементов, каждый из который принадлежит одному из $k \in K$ классов. Для каждого класса можно посчитать его долю от общего числа элементов:

$$\{\frac{n_1}{N}, \frac{n_2}{N}, \dots, \frac{n_K}{N}\}$$

Будем называть классы сбалансированными, когда

$$\frac{n_1}{N} \approx \frac{n_2}{N} \approx \dots \approx \frac{n_K}{N}$$

В нашем случае имеем 3 класса, соответственно стремимся к тому, чтобы доля каждого класса составляла $\frac{1}{3}$ от общего числа примеров.

Введем функционал качества:

$$J = \sum (\frac{n_i}{N} - \frac{1}{K})^2$$

где n_i - количество элементов, принадлежащих классу $i, i < K$. Будем считать, что разбиение на классы зависит от порога t. Тогда можем автоматически выбирать t как решение задачи оптимизации:

$$t_{opt} = argmin_t J$$

Полученная оптимизационная задача решается методом нулевого порядка (например, методом золотого сечения).

Однако для целевой функции не была доказана выпуклость, поэтому могут возникнуть случаи, когда в результате оптимизации будет достигнут локальный минимум. Пример целевой функции приведен на рис. 5

Рис. 5: Функция для оптимизации.

Обычно локальный минимум достигается, когда вся обучающая выборка состоит из одного и того же класса, а параметр t либо отрицателен, либо очень большой. Чтобы избежать таких случаев рекомендуется ставить ограничения на оптимизационную задачу. Из постановки задачи следует, что параметр t лежит в промежутке $(0, 100)$. Однако чаще всего разумные значения параметра лежат в промежутке $(0, 50)$. Наложим эти ограничения на оптимизационную задачу, а также будем оценивать, какое количество классов выделил алгоритм. Если не был выделен какой-то класс, то будем

сужать промежуток вдвое (уменьшая верхнюю границу).

Полученный метод был применем ко всем выбранным временным рядам. Пример приведен на рис. 6

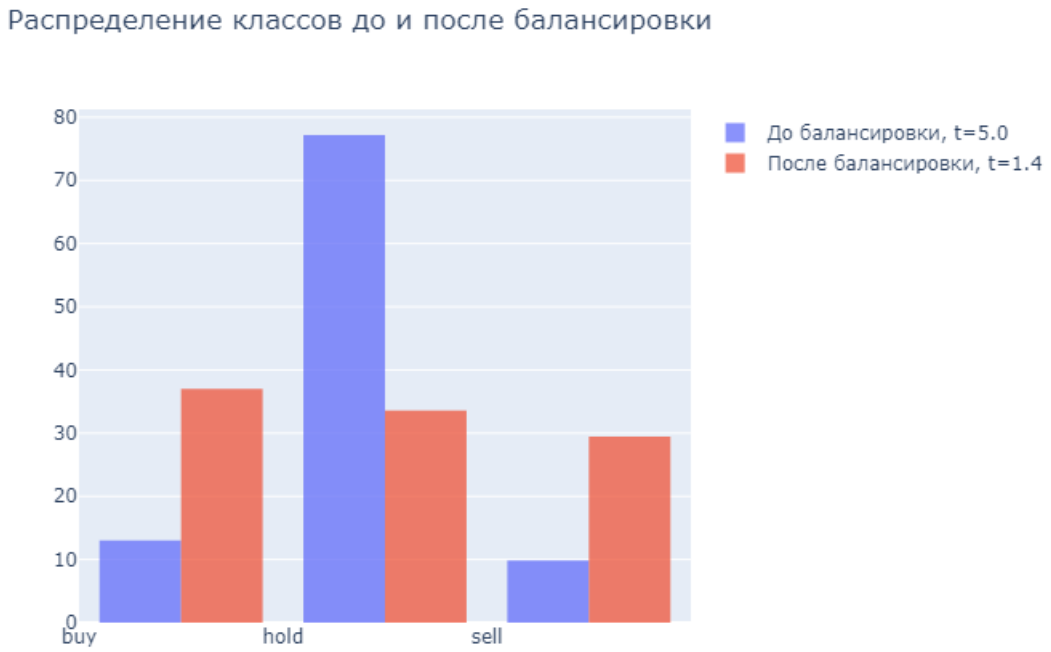

Рис. 6: Диаграмма сбалансированности классов.

Таким образом, мы получили размеченную выборку и теперь можем решать задачу классификации.

2.4.2 Построение моделей классификации

Поставим задачу обучения с учителем для полученного датасета. Необходимо построить функцию $f(x) = d, d \in \{sell, buy, hold\}$, которая на вход получает информацию о временом ряде, а на выходе принимает решение о продаже, покупке или удержании активов.

Задачу классификации можно решать множеством методов. При проведении эксперимента использовались модели линейной регрессии, случайного леса и логистической регрессии.

Остается открытым вопрос о том, какую информацию об исходном

ряде передавать в модель. Обычно берут окно фиксированной длины w и в качестве входных данных передают последние w значений временного ряда. Длина окна - гиперпараметр модели, который необходимо подобрать.

При подборе гиперпараметра будем исходить из горизонта прогнозирования n. Будем оценивать качество модели для каждого значения w (9)

$$w \in \left[n - \frac{n}{2}; n + \frac{n}{2}\right]. \tag{9}$$

Оценивать качество моделей будем при помощи метрики accuracy, используя кросс-валидацию для временных рядов, как это было описано ранее.

Сравнивать качество метрики можно со случайным угадыванием, то есть считаем, что какая-то модель в каждой точке кросс-валидации дала точность $\frac{1}{3}$.

При прогнозировании с 1 мая 2020 года по 30 марта 2021 года моделями Logistic Regression и Random Forest на 3 дня вперед, были получены результаты, отображенные на рис. 7.

Как видно из рисунка, модель Random Forest стабильно правильно предсказывает 2 из 3 направлений тренда. В то же время логистическая регрессия правильно предсказывает направление цены примерно в половине случаев.

2.4.3 Симуляция биржевых торгов

Будем считать, что у трейдера есть возможность купить или продать активы в конце торгового дня (ориентируемся на показатель Close).

Модель предсказывает одно из трех значений: sell, buy, hold. Трейдер действует точно по инструкции:

- sell - продать все имеющиеся активы, если они есть;
- buy - купить активы на все деньги;
- hold - ничего не делать.

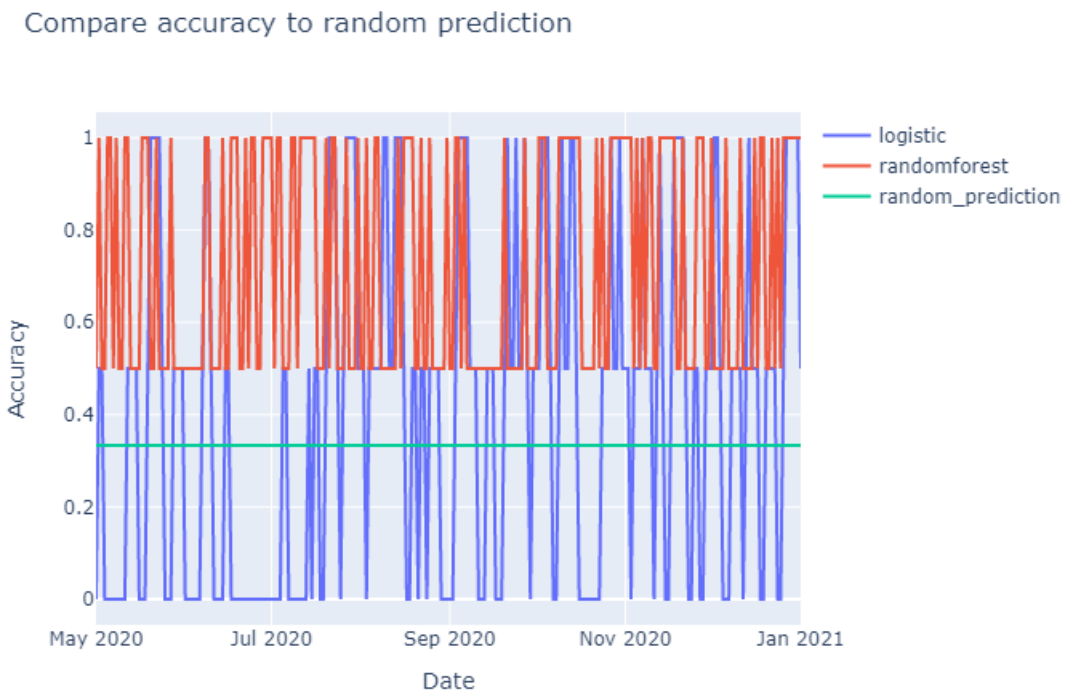

Рис. 7: Точность модели определения направления цены.

Будем считать, что на бирже всегда есть необходимый объем активов и что цена не меняется в зависимости от приобретенных активов.

Так продолжается на протяжении какого-то периода (неделя, месяц или год), а в конце продаются все активы, вне зависимости от цены, и подсчитывается итоговая стоимость портфеля. Чем дольше выбирается период, тем более точно можно оценить качество алгоритма.

Для построения прогноза использовались данные по тем же тикерам, что и ранее. Был выбран промежуток с 1 мая 2020 года по 30 марта 2021 года. Прогноз давался на 3 дня вперед. При построении обучающей выборки использовался автоматический способ подбора порога, исходя из сбалансированности классов.

Результаты представлены в таблице 3.

Из таблицы видно, что все алгоритмы на всех рядах данных дали положительную доходность.

Логистическая регрессия дала доходность значительно меньше, чем случайный лес. Особое внимание стоит обратить на тикер TSLA. Если про-

анализировать логи торгов логистической регрессии, то будет видно, что алгоритм совершил покупку в самом начале, и продал все активы ближе к концу. Таким образом, логистическая регрессия совершила всего 2 сделки за этот промежуток. Если же проанализировать действия случайного леса, то он активно торговал внутри выделенного временного промежутка и доходность получилась в разы больше.

В целом, результаты можно оценить как положительные. В среднем, торговые роботы, использующие консервативные стратегии торговли, достигают доходности от 5% до 15% в месяц. Результаты, полученные моделью RandomForest, сравнимы с этими данными.

Так как сейчас алгоритм ограничен одной сделкой за день, то внутри фиксированного промежутка доходность наибольшим образом определяется общим движением рынка, а не эффективностью алгоритма. Полученную доходность можно увеличить, если проанализировать данные внутри каждого торгового дня (интрадей данные), однако это выходит за рамки текущего исследования.

Таблица 3: Результаты симуляции биржевых торгов

ticker	**models**	**start budget**	**budget**	**profit**
CL=F	Logistic	15000	27178.6	81.2
CL=F	Random Forest	15000	89352.2	495.7
EURUSD=X	Logistic	15000	15000.0	0.0
EURUSD=X	Random Forest	15000	19491.6	29.9
BTC-USD	Logistic	15000	33405.1	122.7
BTC-USD	Random Forest	15000	72609.6	384.1
GC=F	Logistic	15000	17039.3	13.6
GC=F	Random Forest	15000	25257.0	68.4

TSLA	Logistic	15000	74933.0	399.6
TSLA	Random Forest	15000	335608.8	2137.4

Глава 3. Прогнозирование цен биржевых торгов на бирже СПбМТСБ

Санкт-Петербургская Международная Товарно-Сырьевая Биржа (СПбМТСБ) [17] является основной площадкой для торгов нефтепродуктами в России, на которой происходит подавляющее число операций по их покупке и продаже. Так, по итогам 2018 года, каждая четвертая тонна бензина, реализованная в РФ, торговалась на СПбМТСБ. Помимо бензина разных марок, здесь также продаются дизельное топливо, керосин, мазут, нефть, газ и некоторое другое сырье. СПбМТСБ функционирует пять дней в неделю с понедельника по пятницу, с выходными в субботу и воскресенье, а также в государственные праздники.

Биржевая торговля осуществляется в режиме двойного встречного анонимного аукциона, при котором продавец и покупатель не видят друг друга и не могут сговориться о цене. Сделка заключается автоматически при пересечении условий во встречных анонимных заявках. Благодаря особой организации биржевых торгов, они позволяют определять справедливую рыночную стоимость товаров.

Компания ПАО «Газпром нефть» была заинтересована в том, чтобы прогнозировать средневзвешенные цены реализации ряда нефтепродуктов на 14 дней вперед.

Каждый из нефтепродуктов представлен на нескольких базисах поставки. По сути, базис поставки - это завод, который реализует нефтепродукт (или любую другую продукцию) посредством биржи. Пару из нефтепродукта и базиса поставки в рамках этой задачи можно называть торговым инструментом. Заказчик был заинтересован прогнозировать предложенные нефтепродукты по трем базисам поставки. Все нефтепродукты представлены на всех базисах поставки, кроме авиакеросина ТС-1, который не представлен на базисе поставки №2.

В качестве исходных данных были предоставлены данные о средневзвешенных ценах торгов по необходимым инструментам, а также некоторые внешние данные, в которых содержится информация об экономических процессах, влияющих на ценообразование.

Для решения задачи были использованы описанные выше алгоритмы. Для оценки результатов использовалась метрика MAPE (2).

Результаты представлены в таблице 4.

Здесь в качестве базового решения можно взять модель ARIMA, как наиболее простую модель. Отчетливо наблюдается, что модель VAR предсказывает значительно хуже остальных моделей. Модели Wavelet и ARIMA предсказывают примерно с одинаковой точностью. Лучшую точность продемонстрировала модель Stacking. Для большинства рядов удалось предсказать цену с отклонением менее чем в 2%, такая точность устраивала заказчика.

Отдельно стоит обратить внимание на предсказания цены мазута. Отклонение прогноза от фактических значений по этому нефтепродукту значительно выше, чем по остальным. Это объясняется тем, что в абсолютных значениях мазут стоит значительно дешевле других нефтепродуктов. Для сравнения, один биржевой лот мазута стоит от 15 до 25 тысяч рублей, а один биржевой лот бензина стоит от 40 до 55 тысяч рублей, в зависимости от марки и базиса поставки. Также стоит обратить внимание на то, что по дизельному топливу прогноз немного точнее, чем по остальным нефтепродуктам. Это объясняется тем, что рынок дизельного топлива наименее волатилен, в сравнении с остальными нефтепродуктами.

Таблица 4: Результаты прогнозирования цен на бирже СПбМТСБ

Инструмент	**ARIMA**	**Wavelet**	**VAR**	**Stacking**
АИ-92 Б.П. №1	2.95	2.54	3.33	1.83
АИ-92 Б.П. №2	3.61	3.11	2.93	1.85
АИ-92 Б.П. №3	4.46	2.92	3.25	1.98
АИ-95 Б.П. №1	3.63	3.39	3.81	2.05
АИ-95 Б.П. №2	4.91	3.66	3.54	2.11
АИ-95 Б.П. №3	4.9	3.65	3.94	1.91
ДТ Б.П. №1	1.88	1.65	2.14	1.08
ДТ Б.П. №2	5.71	1.74	2.07	1.13
ДТ Б.П. №3	2.38	1.86	2.29	1.18

Мазут Б.П. №1	16.75	14.5	15.94	6.39
Мазут Б.П. №2	13.3	12.58	12.97	5.58
Мазут Б.П. №3	11.86	11.79	11.32	5.33
ТС-1 Б.П. №1	3.46	3.0	3.01	1.96
ТС-1 Б.П. 3	2.29	2.16	2.88	1.14

Выводы

Таким образом, использованные методы прогнозирования в некоторых случаях дают прогноз лучше, чем наивное или случайное предсказание. Учитывая высокую волатильность рынков, этот результат является значимым. Использованные подходы прогнозирования универсальны и их можно применять для предсказания любых других временных рядов, например, в задачах прогнозирования погоды, городских пробок и различных экономических показателей. Полученные результаты можно улучшить, если учитывать внешние факторы, объясняющие исходный ряд.

Подход, связанный с симуляцией биржевых торгов, также дает положительный результат для большинства рядов.

В дальнейшем исследование может получить несколько векторов развития.

Во-первых, могут быть рассмотрены другие прогнозные модели. Например, можно использовать нейросетевые методы, или другие техники построения ансамблей.

Во-вторых, можно глубже исследовать вопрос предобработки исходных данных. Данные можно сглаживать при помощи скользящей средней, медианного фильтра или вейвлет-фильтра. Также можно рассмотреть различные декомпозиции временных рядов, например CEEMDAN [3].

В-третьих, результаты можно значительно улучшить, если обогатить модели внешними данными, которые бы объясняли исходный ряд. Например, акции Tesla связаны с высказываниями основателя компании в Twitter. Можно провести сентимент-анализ сообщений в Twitter, и использовать полученные результаты как экзогенные признаки для модели Stacking.

Что касается моделей классификации, их результаты могут быть улучшены всеми пунктами, перечисленными выше, а также если строить модели, которые совершали бы сделки внутри торговой сессии.

Весь исходный код выложен на GitHub: `https://github.com/svkov/time-series-forecasting`.

Список литературы

[1] *Apache Airflow*. URL: https://airflow.apache.org/ (дата обр. 04.05.2021).

[2] G.E.P. Box, G.M. Jenkins и WISCONSIN UNIV MADISON Dept. of STATISTICS. *Time Series Analysis: Forecasting and Control*. Holden-Day series in time series analysis and digital processing. Holden-Day, 1970. ISBN: 9780816210947. URL: https://books.google.ru/books?id=5BVfnXaq03oC.

[3] Jian Cao, Zhi Li и Jian Li. "Financial time series forecasting model based on CEEMDAN and LSTM". В: *Physica A: Statistical Mechanics and its Applications* 519 (2019), с. 127—139. ISSN: 0378-4371. DOI: https://doi.org/10.1016/j.physa.2018.11.061. URL: https://www.sciencedirect.com/science/article/pii/S0378437118314985.

[4] Jan G. De Gooijer и Rob J. Hyndman. "25 years of time series forecasting". В: *International Journal of Forecasting* 22.3 (2006). Twenty five years of forecasting, с. 443—473. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2006.01.001. URL: https://www.sciencedirect.com/science/article/pii/S0169207006000021.

[5] Felix Gers, Douglas Eck и Jürgen Schmidhuber. "Applying LSTM to Time Series Predictable through Time-Window Approaches". В: авг. 2001, с. 669—676. ISBN: 978-3-540-42486-4. DOI: 10.1007/3-540-44668-0_93.

[6] Johannes Köster и Sven Rahmann. "Snakemake—a scalable bioinformatics workflow engine". В: *Bioinformatics* 28.19 (авг. 2012), с. 2520—2522. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/bts480. eprint: https://academic.oup.com/bioinformatics/article-pdf/28/19/2520/819790/bts480.pdf. URL: https://doi.org/10.1093/bioinformatics/bts480.

[7] A Lapedes и R Farber. "Nonlinear signal processing using neural networks: Prediction and system modelling". В: (июнь 1987). URL: https://www.osti.gov/biblio/5470451.

[8] Gregory R. Lee и др. “PyWavelets: A Python package for wavelet analysis”. В: *Journal of Open Source Software* 4.36 (2019), с. 1237. DOI: 10.21105/joss.01237. URL: https://doi.org/10.21105/joss.01237.

[9] Xueheng Qiu и др. “Ensemble deep learning for regression and time series forecasting”. В: *2014 IEEE Symposium on Computational Intelligence in Ensemble Learning (CIEL)*. 2014, с. 1—6. DOI: 10.1109/CIEL.2014.7015739.

[10] Matheus Henrique Dal Molin Ribeiro и Leandro dos Santos Coelho. “Ensemble approach based on bagging, boosting and stacking for short-term prediction in agribusiness time series”. В: *Applied Soft Computing* 86 (2020), с. 105837. ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2019.105837. URL: https://www.sciencedirect.com/science/article/pii/S1568494619306180.

[11] Sima Siami-Namini, Neda Tavakoli и Akbar Siami Namin. “A Comparison of ARIMA and LSTM in Forecasting Time Series”. В: *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*. 2018, с. 1394—1401. DOI: 10.1109/ICMLA.2018.00227.

[12] *spotify/luigi: Luigi is a Python module that helps you build complex pipelines of batch jobs. It handles dependency resolution, workflow management, visualization etc. It also comes with Hadoop support built in.* URL: https://github.com/spotify/luigi (дата обр. 04.05.2021).

[13] Martin Sundermeyer, Ralf Schlüter и Hermann Ney. “LSTM Neural Networks for Language Modeling”. В: сент. 2012.

[14] Fei Wang и др. “Wavelet Decomposition and Convolutional LSTM Networks Based Improved Deep Learning Model for Solar Irradiance Forecasting”. В: *Applied Sciences* 8.8 (2018). ISSN: 2076-3417. DOI: 10.3390/app8081286. URL: https://www.mdpi.com/2076-3417/8/8/1286.

[15] Peter R. Winters. “Forecasting Sales by Exponentially Weighted Moving Averages”. В: *Management Science* 6.3 (1960), с. 324—342. ISSN: 00251909, 15265501. URL: http://www.jstor.org/stable/2627346.

[16] *Yahoo Finance [Электронный ресурс].* URL: `https://finance.yahoo.com/` (дата обр. 04.05.2021).

[17] *АО СПбМТСБ, Санкт-Петербургская Международная Товарно-сырьевая Биржа.* URL: `https://spimex.com/` (дата обр. 04.05.2021).

[18] Носко В.П. *Эконометрика. Введение в регрессионный анализ временных рядов.* Т. 312 страниц. 2004. ISBN: 5-94010-322-7.

[19] Магнус Я. Р., Катышев П. К. и Пересецкий А. А. *Эконометрика. Начальный курс.* 2004.

[20] Ковалев С.С. "Прогнозирование средневзвешенной цены торгов нефтепродуктами на бирже классическими методами анализа временных рядов." В: *Процессы управления и устойчивость. №51.* (2020).