МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет
імені Петра Могили
Факультет комп'ютерних наук
Кафедра інтелектуальних інформаційних систем

ДОПУЩЕНО ДО ЗАХИСТУ
Завідувач кафедри інтелектуальних
інформаційних систем, д-р техн. наук, проф.
_____Ю. П. Кондратенко
«____»_____2023 р.

БАКАЛАВРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

СИСТЕМА АНАЛІЗУ ТА ПРОГНОЗУВАННЯ ФІНАНСОВИХ ЧАСОВИХ РЯДІВ

Спеціальність 122 «Комп'ютерні науки»

122 – БКР – 402.21910222

Виконав студент 4-го курсу, групи 402
_____Р. Ю. Стрижак
«___» червня 2023 р.

Керівник: канд. пед. наук, доцент
_____Н. М. Болюбаш
«___» червня 2023 р.

Миколаїв – 2023

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет ім. Петра Могили
Факультет комп'ютерних наук
Кафедра інтелектуальних інформаційних систем

Рівень вищої освіти **бакалавр**
Спеціальність **122 «Комп'ютерні науки»**
(шифр і назва)
Галузь знань **12 «Інформаційні технології»**
(шифр і назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри інтелектуальних
інформаційних систем, д-р. техн. наук, проф.
_____ Ю. П. Кондратенко
«____» _____2022 р.

З А В Д А Н Н Я

на виконання кваліфікаційної роботи

Видано студенту групи 402 факультету комп'ютерних наук Стрижаку Роману Юрійовичу.

1.Тема кваліфікаційної роботи «Система аналізу та прогнозування фінансових часових рядів».

Керівник роботи Болюбаш Надія Миколаївна, канд. пед. наук, доцент.

Затв. наказом Ректора ЧНУ ім. Петра Могили від «___» _____ 20___ р. № _____

2. Строк представлення кваліфікаційної роботи студентом «___» _____ 20___ р.

3. Вхідні (початкові) дані до роботи: предметна сфера фінансових часових рядів курсу акцій та методів і моделей їх прогнозування, набір даних з курсом акцій компанії Microsoft.

Очікуваний результат: інформаційна система аналізу та прогнозування курсу акцій.

4. Перелік питань, що підлягають розробці (зміст пояснювальної записки):

- аналіз предметної сфери фінансових часових рядів та дослідження теоретичних засад прогнозування курсу акцій;

– обґрунтування вибору технологій і засобів розробки системи аналізу та прогнозування фінансових часових рядів;
– розробка і здійснення програмної реалізації системи аналізу та прогнозування курсу акцій, дослідження якості прогнозної моделі та точності прогнозу.

5. Перелік графічного матеріалу: презентація.

6. Завдання до спеціальної частини: «Охорона праці при роботі з екранними пристроями».

7. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис
Спеціальна частина з охорони праці	викладач А. Л. Боженко	

Керівник роботи __канд. пед. наук, доцент Болюбаш Н. М._____
(наук. ступінь, вчене звання, прізвище та ініціали)

_____
(підпис)

Завдання прийнято до виконання ___Стрижак Р. Ю._____
(прізвище та ініціали)

_____
(підпис)

Дата видачі завдання «_____» _____ 2022 р.

КАЛЕНДАРНИЙ ПЛАН
виконання бакалаврської кваліфікаційної роботи

Тема: Система аналізу та прогнозування фінансових часових рядів

№	Найменування роботи	Початок	Закінчення	Примітки
1	Визначення керівника і теми БКР. Подання заяви на затвердження теми БКР	01.09.2022	1.11.2022	Виконано
2	Отримання завдання на виконання БКР	25.11.2022	25.11.2022	Виконано
3	Складання календарного плану	26.11.2022	10.12.2022	Виконано
4	Огляд літератури за темою дослідження. Аналіз предметної сфери фінансових часових рядів, дослідження підходів до прогнозування курсу акцій, метрик оцінки прогнозних моделей	11.12.2021	31.12.2021	Виконано
5	Вибір технологій та інструментальних засобів розробки системи	1.01.2023	31.01.2023	Виконано
6	Створення дизайну, проєктування та програмна реалізація, тестування системи	1.02.2023	15.04.2023	Виконано
7	Робота над розділами фахової частини БКР	16.04.2023	31.04.2023	Виконано
8	Проходження переддипломної практики, збір та аналіз матеріалів, остаточне оформлення розділів фахової частини БКР	1.05.2023	14.05.2023	Виконано
9	Розробка спеціальної частини з охорони праці	15.05.2023	25.05.2023	Виконано
10	Обговорення отриманих результатів з керівником та попередній захист БКР	29.05.2023	1.06.2023	Виконано
11	Корегування роботи за результатами попереднього захисту	2.06.2023	6.06.2023	Виконано
12	Остаточне оформлення пояснювальної записки та слайдів доповіді до захисту	7.06.2023	11.06.2023	Виконано
13	Подання рецензенту та рецензування БКР	15.06.2023	18.06.2023	Виконано
14	Подання БКР, її електронної копії та інших документів (відгуку, рецензії) до захисту	19.06.2023	22.06.2023	Виконано
15	Захист БКР перед ЕК	28.06.2023	28.06.2023	Виконано

Розробив студент _____Стрижак Р. Ю._____ _____
(прізвище та ініціали) *(підпис)*

Керівник _____канд. пед. наук, доцент Болюбаш Н. М___ _____
(наук. ступінь, вчене звання, прізвище та ініціали) *(підпис)*

«_____» _____12_____ 2022 р.

АНОТАЦІЯ

бакалаврської кваліфікаційної роботи
студента групи 402 ЧНУ ім. Петра Могили
Стрижака Романа Юрійовича
Тема: «Система аналізу та прогнозування фінансових часових»

Бакалаврська кваліфікаційна робота присвячена розробці та здійсненню програмної реалізації системи аналізу та прогнозування фінансових часових рядів. Що є актуальним в умовах високих темпів розвитку сфери фондового ринку, оскільки прогнозування є необхідним елементом інвестиційної діяльності.

Об'єкт роботи – процес прогнозування та аналізу фінансових часових рядів.

Предмет роботи – програмні засоби та методи аналізу і прогнозування фінансових часових рядів.

Мета роботи – підвищення якості планування інвестицій на фондових ринках шляхом розробки системи прогнозування вартості комерційних компаній із вбудованими методами аналізу часових рядів курсу акцій та побудови прогнозних моделей.

Бакалаврська кваліфікаційна робота складається з фахової та спеціальної частини з охорони праці. Пояснювальна записка фахової частини складається зі вступу, трьох розділів, висновків та додатків. У першому розділі розкрито теоретичні засади аналізу та прогнозування фінансових часових рядів. У другому розділі обґрунтовано вибір технологій і засобів розробки системи. У третьому розділі описано проєктування та програмну реалізацію системи аналізу і прогнозування курсу акцій.

Бакалаврська кваліфікаційна робота містить ___ сторінку (без додатків), ___ рисунків, ___ таблиці, ___ джерел, ___ додатки.

Ключові слова: прогнозування курсу акцій, часовий ряд, тренд, автокореляційний аналіз, авторегресійна модель, адитивна модель, методи прогнозування часових рядів.

ABSTRACT

for bachelor's qualification work
of a student of 402 group at Petro Mohyla Black Sea National University
Stryzhaka Romana Yuriiovycha
Theme: «System of analysis and forecasting of financial time series»

The bachelor's thesis is devoted to the development and implementation of the software implementation of the system of analysis and forecasting of financial time series. What is relevant in the conditions of high rates of development of the stock market, since forecasting is a necessary element of investment activity.

Object of work – the process of forecasting and analysis of financial time series.

Subject of work – software tools and methods of analysis and forecasting of financial time series.

The purpose of the work is to improving the quality of investment planning on the stock markets by developing a system for forecasting the value of commercial companies with built-in methods of analyzing time series of share prices and building predictive models.

The bachelor's qualification work consists of a professional and special part on labor protection. The explanatory note of the professional part consists of an introduction, three sections, conclusions and appendices. The first chapter reveals the theoretical foundations of analysis and forecasting of financial time series. In the second section, the choice of technologies and means of system development is substantiated. The third section describes the design, software implementation and application of the system stock price analysis for forecasting.

The master's thesis contains ___ page (without appendices), ___ figures, ___ tables, ___ sources, ___ appendix.

Keywords: stock price forecasting, time series, trend, autocorrelation analysis, autoregressive model, additive model, time series forecasting methods.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ACF – Autocorrelation function

AR – Autoregressive model

CDNJS – Content Delivery Network for JavaScript

CSS – Cascading Style Sheets

ES – ECMAScript

HTML – HyperText Markup Language

JSON – JavaScript Object Notation

MACD – Moving Average Convergence Divergence

MAD – Mean Absolute Derivation

MAE – Mean Absolute Error

MAPE – Mean Absolute Percentage Error

MFE – Mean Forecast Error

MPE – Mean Percentage Error

MSE – Mean Squared Error

NPM – Node Package Manager

RSI – Relative Strength Index

SASS – Syntactically awesome style sheets

ВСТУП

Актуальність. Прогнозування фінансових часових рядів лежить в основі діяльності індустрії інвестицій. Інформатизація суспільства на сучасному етапі характеризується використання інноваційних засобів підтримки комерційної діяльності інвесторів на фондових ринках. Перспективним напрямком підвищення ефективності планування інвестицій є застосування автоматизованих систем аналізу та прогнозування курсу акцій, які дозволяють прогнозувати фондові тренди шляхом аналізу даних про динаміку зміни курсу акцій протягом певного періоду часу.

Моделювання фінансових часових рядів є складною задачею, оскільки ринкова економіка містить багато чинників, які можуть викликати ріст та падіння курсу акцій. На вибір методів аналізу та прогнозування впливає стаціонарність та не стаціонарність часового ряду. До методів аналізу часових рядів, які базуються на припущенні про стаціонарність ряду, відносять авторегресійну модель (AR-модель) та її модифікації.

Нестаціонарні часові ряди можуть мати регулярні компоненти – тренд, сезонність та/або циклічні зміни в середньому значенні рівнів ряду. Моделювання прогностичної моделі нестаціонарного часового ряду може здійснюватися як шляхом моделювання регулярних компонент у сукупності, так і шляхом розкладання часового ряду на компоненти та моделювання кожної компоненти окремо. Для підвищення ефективності планування інвестицій на фондових ринках необхідна автоматизована система, яка дозволяє здійснювати аналіз динаміки курсу акцій та серед сукупності моделей часового ряду обирати оптимальну, яка дозволяє отримувати більш точний прогноз.

Це обумовило **мету роботи**, яка полягає у підвищенні якості планування інвестицій на фондових ринках шляхом розробки системи прогнозування вартості комерційних компаній із вбудованими методами аналізу часових рядів курсу акцій та побудови прогнозних моделей.

Відповідно до поставленої мети було сформульовано **завдання**:

1) здійснити аналіз предметної сфери фінансових часових рядів та дослідити теоретичні засади прогнозування курсу акцій;

2) обґрунтувати вибір технологій та інструментальних засобів розробки системи аналізу та прогнозування фінансових часових рядів;

3) розробити, здійснити програмну реалізацію системи аналізу і прогнозування курсу акцій, дослідити якість прогнозної моделі та точність прогнозу.

Об’єкт роботи – процес прогнозування та аналізу фінансових часових рядів.

Предмет роботи – програмні засоби та методи аналізу і прогнозування фінансових часових рядів.

Методологічною основою дослідження є загальнонаукові та статистично-аналітичні методи, які дозволили комплексно вивчити предмет та об’єкт дослідження, дослідити основні підходи до побудови прогнозної моделі фінансового часового ряду.

Практичне значення отриманих результатів полягає в тому, що використання розробленої інформаційної системи дозволить підвищити ефективність планування інвестицій при купівлі часток акцій комерційних компаній на фондових ринках.

Структура бакалаврської кваліфікаційної роботи. Відповідно до мети, завдань і предмета дослідження, бакалаврська робота містить основну та спеціальну частини. Основна частина роботи складається із вступу, трьох розділів, висновку, списку використаних джерел та __ додатків. Загальний обсяг роботи – __ сторінок, із них основного тексту основної частини – __ сторінок, спеціальної – __ сторінок. Кількість використаних джерел – __.

1 ТЕОРЕТИЧНІ ЗАСАДИ АНАЛІЗУ ТА ПРОГНОЗУВАННЯ ФІНАНСОВИХ ЧАСОВИХ РЯДІВ

1.1 Предметна сфера фінансових часових рядів

Предметна сфера фінансових часових рядів включає в себе вивчення динаміки фінансових показників в часі та їх аналіз з метою прогнозування майбутніх значень. Цей напрямок досліджень має велике значення для фінансової сфери, оскільки відомості про тенденції зміни різних фінансових показників, таких як курси валют, ціни на товари та послуги, обсяги продажів та інші, є важливими для прийняття рішень про інвестування, ризик-менеджмент та планування діяльності підприємств.

Основна мета досліджень у галузі фінансових часових рядів полягає у вивченні різних параметрів цих рядів, зокрема, їхньої структури, тренду, сезонності, циклів, затримки та інших характеристик, що дозволяє зробити висновки про тенденції зміни відповідних фінансових показників.

Фінансові часові ряди – це послідовності фінансових даних, які збираються відповідно до певної часової структури, наприклад щоденно, щотижня або щомісяця [1]. Дані можуть стосуватися курсів валют, цін на акції, рівня процентних ставок та інших показників фінансової діяльності.

Фінансові часові ряди дозволяють досліджувати динаміку зміни показників фінансової діяльності в часі, аналізувати тренди та сезонність, виявляти кореляції між різними показниками та робити прогнози на майбутнє. Аналізуючи динаміку росту акцій певної компанії за допомогою часових рядів, можна визначити потенційні ризики та можливості для інвесторів.

Курс акцій компаній може змінюватися під впливом багатьох факторів, які можуть бути економічними, політичними, фінансовими, соціальними тощо [2]. Наприклад, зміни у ситуації на світових ринках, кризи, політичні кризи, катастрофи, новини про компанію та інші фактори можуть впливати на курс акцій.

Наприклад, події в світі, пов’язані з епідемією COVID-19, мали значний вплив на курс акцій компаній, особливо тих, що пов’язані з туризмом, авіацією та гостинності. За період з березня по червень 2020 року, коли велика частина світу була закрита через карантин, курс акцій багатьох компаній впав. Наприклад, курс акцій авіакомпанії American Airlines Group Inc. з 21 лютого 2020 року до 23 березня 2020 року знизився на більш ніж 62%.

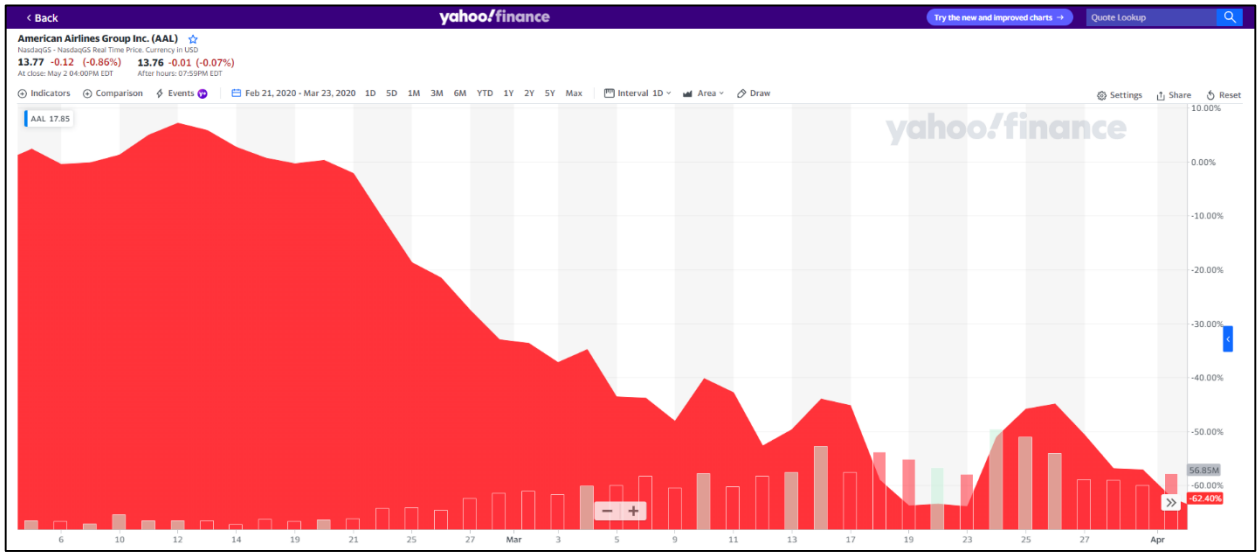

Рисунок 1.1 – Курс акцій авіакомпанії American Airlines Group Inc. з 21 лютого 2020 року до 23 березня 2020 року

Також, курс акцій може змінюватися під впливом фінансових звітів компанії, змін у керівництві компанії, змін у виробничому процесі, змін в правовому регулюванні тощо. Наприклад, коли компанія заявляє про успішність своєї діяльності та публікує позитивний фінансовий звіт, курс акцій може підвищитися, оскільки інвестори будуть вважати її більш привабливою для інвестування.

Один з реальних прикладів пов'язаний з компанією Tesla Inc. У липні 2020 року Tesla опублікувала позитивний фінансовий звіт, згідно з яким компанія заробила високі прибутки в другому кварталі, що перевершило очікування аналітиків. Як наслідок, курс акцій Tesla виросла на 10% в перший день після публікації звіту. Інвестори переконалися в успішності діяльності компанії та

побачили перспективи для її подальшого розвитку, що призвело до збільшення попиту на акції Tesla і підвищення їх курсу.

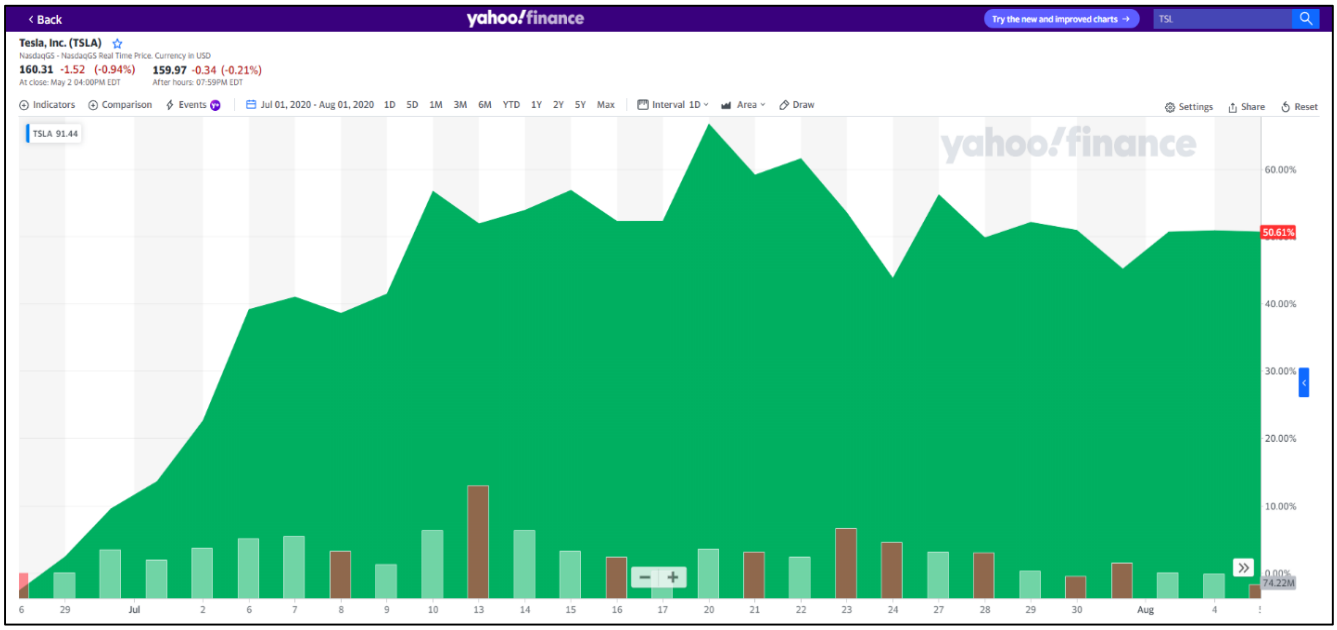

Рисунок 1.2 – Курс акцій компанії Tesla Inc. в липні 2020 року після опублікування позитивного фінансового звіту

Часові ряди є зручним інструментом для прогнозування, оскільки вони можуть допомогти виявити тренди та сезонність у даних, а також можуть допомогти у прогнозуванні майбутніх значень. Дані в часових рядах зазвичай мають відносно просту структуру, оскільки вони складаються зі значень, взятих у різні моменти часу, що дозволяє застосовувати багато різних методів аналізу [3, 4].

Курс акцій може змінюватися в залежності від таких факторів, як:

1) політичні та економічні події: курс акцій може знижуватися у зв’язку з несприятливими рішеннями уряду, такими як введення високих податків або належність до міжнародних санкцій;

2) конкуренція на ринку: курс акцій компанії може знижуватися, якщо її конкуренти представлять нові продукти або послуги, які можуть знизити прибуток компанії;

3) фінансовий здоров'я компанії: курс акцій може змінюватися в залежності від фінансового стану компанії, такого як рівень прибутку, боргові зобов'язання, або інші фінансові показники;

4) попит та пропозиція: курс акцій може змінюватися в залежності від зміни попиту та пропозиції на акції компанії. Якщо попит на акції перевищує пропозицію, то ціна зростає, а якщо пропозиція перевищує попит, то ціна падає.

Врахування цих факторів дозволяє побудувати часовий ряд курсу акцій, який містить значення курсу на різні моменти часу. Цей ряд можна проаналізувати, щоб виявити тренди та сезонність у даних, а також щоб побудувати прогноз майбутніх значень курсу акцій. Наприклад, якщо підприємство випускає новий продукт, то можна очікувати зростання попиту на акції компанії, оскільки новий продукт може привести до збільшення прибутку підприємства, що зазвичай позитивно впливає на ціну акцій.

Курси акцій можуть змінюватися через макроекономічні фактори, такі як інфляція, процентні ставки, політика центральних банків та інші подібні події. Наприклад, збільшення процентної ставки може привести до скорочення кількості грошей, доступних для інвестування, що може знизити попит на акції та зменшити їх ціну.

Оскільки курси акцій можуть змінюватися залежно від різних факторів, важливо використовувати інструменти для їх прогнозування. Програмні засоби, які дозволяють аналізувати структуру даних часового ряду курсу акцій, виявляти закономірності та здійснювати прогнозування майбутніх значень на основі цих закономірностей, є одним із таких інструментів.

Існує багато систем та програмних сервісів, що дозволяють здійснювати побудову прогнозів часових рядів [5]. Умовно їх можна поділити на кілька типів: статистичні пакети, табличні процесори, пакети візуального проєктування, ERP-системи, системи класу Business Intelligence, когнітивні та академічні системи прогнозування. Розглянемо їх більш детально [6].

Статистичні пакети. Цей клас програмного забезпечення представлено програмами, які дозволяють використовувати різні методи статистики і Data Mining та відрізняються принципами роботи користувача із пакетом [7]. STATISTICA,

IBM SPSS Statistics, Minitab – статистичні програмні комплекси, які містять багато вбудованих засобів прогнозування. Взаємодія з користувачем здійснюється за допомогою меню та панелей інструментів. Програмні комплекси MatLab, SAS, S-Plus вимагають від користувача спеціальних мов програмування. Дані статистичні пакети дозволяють здійснювати побудову моделей прогнозування різного рівня складності.

Табличні процесори є найбільш поширеними засобами економіко-математичного моделювання. Найбільш відомі програмні комплекси у цьому класі MS Excel, MS Excel 365, Coral Quattro Pro. Ці пакети дозволяють створювати та редагувати електронні таблиці, використовуючи вбудовані формули та алгоритми побудови прогнозних моделей.

Пакети візуального моделювання. Цей клас програм містить набір візуальних елементів для моделювання. Найбільш великими представниками є Microsoft Machine learning і Orange 3. Інтерфейс таких систем аналізу даних є полотном, на який користувач може наносити різні елементи, які характеризують інструменти роботи з даними. Кожен елемент налаштовується та має свої особливості. Результатом роботи з такою програмою є візуальна модель, де представлені етапи роботи з даними (потік даних).

ERP системи. Сучасні системи даного класу дозволяють проводити аналіз даних із використанням різних інструментів. Такі системи допомагають підприємствам автоматизувати основні бізнес-процеси та керувати ними для досягнення оптимальної продуктивності й можуть включати модуль методів машинного навчання з реалізованими методами прогнозування фінансових показників та курсу акцій.

Системи класу Business Intelligence. BI-системи є набір інструментів щодо аналітики, зокрема прогнозування. Прикладом таких систем є Microsoft PowerBI, IBM Cognos Analytics, QlikView, Deductor (Loginom), Contour BI. Їхня архітектура включає модуль інтелектуального аналізу даних, який дозволяє проводити побудову прогнозів досліджуваних показників.

Когнітивні системи. Найбільш яскравим представником таких систем є IBM Watson. Ця система дозволяє вирішувати складні завдання з використанням методів штучного інтелекту [8]. З її допомогою можна здійснювати прогнозування з використанням інтелектуальних моделей, а також проводити візуалізацію отриманих результатів.

Академічні системи прогнозування. Цей клас систем розробляється спільнотою вчених і зазвичай має вузьку спеціалізацію. Серед них є системи підтримки прийняття рішення, засновану на експертному методі прогнозування економічних даних. Такі системи є застосунком для бізнес-прогнозування з використанням методів регресії та трендів у комбінації із вилученням знань. Поширені системи прогнозування на основі аналізу та прогнозування фінансових показників з використанням методів регресії, авторегресії, регресійних дерев рішень, методу опорних векторів, нейронних мереж і бустингу.

Наведені системи дозволяють здійснювати прогнозування часових рядів різної природи. Однак у них є ряд недоліків. Пакети візуального моделювання мають обмежений набір елементів, а також вимагають знання програмування для можливості гнучкого налаштування моделей. Корпоративні системи BI та ERP є дорогими та потребують трудомісткого процесу розгортання та підтримки, а також не є гнучкими. Академічні системи, як правило, вирішують вузьке коло завдань без можливості масштабування.

Когнітивні системи дозволяють провести сценарне прогнозування сукупності показників. Однак при цьому деталі обчислень приховані від аналітика і не всі виявлені моделі зв’язку є очевидними. Статистичні пакети та табличні процесори не дозволяють використовувати системний підхід, а також потребують трудомістких процесів налаштування моделей та знань мов програмування.

Таким чином, існує потреба у розробці системи, яка має зручний та зрозумілий інтерфейс, проста у налаштуванні й автоматизує усі етапи аналізу та прогнозування часових рядів курсу акцій: здійснення попередньої обробки даних,

аналіз структури, вибір оптимальної моделі, оцінку її адекватності та точності і прогнозування нових значень [9].

1.2 Основні підходи до прогнозування курсу акцій

Прогнозування фінансових часових рядів є необхідним інструментом для прийняття рішень при плануванні інвестицій. Основні методи прогнозування, які застосовують у цій сфері, є наступними [10]:

1) експертні методи – базуються на оцінці експертів;

2) методи економіко-математичного моделювання – дозволяють будувати довгострокові прогнози на основі знаходження багатофакторних закономірностей;

3) фундаментальний аналіз – побудова прогнозу здійснюється на основі аналізу фінансових показників компаній, які є власниками акцій;

4) технічний аналіз – прогноз здійснюється на основі аналізу історичних даних цін на акції – часових рядів.

У даній кваліфікаційній роботі обрано саме технічний аналіз для побудови моделі прогнозування курсу акцій.

При побудові прогностичних моделей розрізняють довгострокові, середньострокові та короткострокові прогнози [11, 12]. Середньострокові прогнози – це прогнози, які охоплюють період зазвичай не більше, ніж 3-5% обсягу спостережень, відображених у рівнях часового ряду (на 7-12 кроки уперед).

Довгострокові прогнози – це прогнози, які охоплюють період зазвичай більше, ніж 5% обсягу спостережень, відображених у рівнях часового ряду. Вони використовуються для прогнозування довгострокових тенденцій, таких як зміна демографічних тенденцій, технологічних революцій або макроекономічних циклів. Довгострокові прогнози часто базуються на аналізі трендів, циклів та сезонності, а також на прогнозуванні впливу екзогенних факторів, таких як зміна податкової політики або політики центрального банку. Довгострокові прогнози можуть бути

корисні для планування діяльності на довгостроковий період, такий як планування інвестицій або стратегічне планування підприємства.

Короткострокові прогнози – це прогнози, які охоплюють період зазвичай не більше, ніж 3% обсягу спостережень, відображених у рівнях часового ряду (на 1-2 кроки уперед). Вони використовуються для прогнозування короткострокових змін, таких як зміни в споживчих тенденціях або макроекономічному середовищі. Короткострокові прогнози можуть базуватися на аналізі технічних показників, таких як технічний аналіз цін на акції або технічний аналіз курсу валют. При аналізі фінансових часових рядів, які відображають динаміку зміни одного фактору, прогноз зазвичай можна отримати короткостроковий.

Аналіз та прогнозування фінансових часових рядів, у яких показником, що змінюється з плином часу, є курс акцій, включає сукупність методів, які реалізуються на основі базових положень про структуру часового ряду[13].

Часовий ряд представляє собою послідовність значень курсу акцій, упорядковану у хронологічному порядку: y_1, y_2, … y_n, де y_t – значення курсу акцій у певний момент часу. Основними складовими часового ряду є наступні компоненти:

1) *тренд* – основний структурний компонент, який характеризує наявність загального напряму зміни курсу акцій, що формується під впливом загальних та довгострокових тенденцій, які впливають на часовий ряд та який можна представити у вигляді функції *f(t)*, яка може бути зростаючою, спадаючою, лінійною, нелінійною тощо;

2) *сезонна компонента* – містить значення, які повторюються відносно основної тенденції протягом невеликих періодів часу: року, місяця або тижня (наприклад, споживання електроенергії може збільшуватись влітку через використання кондиціонерів або зменшуватись у вихідні дні);

3) *циклічна компонента* – містить коливання відносно основної тенденції – тренду протягом достатньо великого періоду часу, які можуть бути пов'язані з економічними чи іншими циклічними явищами, такими як бізнес-цикли;

4) *випадкова (залишкова) складова* – складова часового ряду, обумовлена впливом випадкових факторів або пов'язаних з іншими чинниками, які не враховуються в моделі.

Фінансовий часовий ряд може містити усі вище зазначені компоненти або лише деякі з них. Розрізняють також стаціонарні та нестаціонарні часові ряди [14].

До стаціонарних часових рядів відносять ті з них, значення рівнів яких коливається навколо постійного середнього значення курсу акцій. Його статистичні характеристики не змінюються з часом. Іншими словами, середнє значення, дисперсія та коваріація ряду залишаються сталі в часі. Це означає, що стаціонарні часові ряди не мають тренду, сезонної та циклічної складової. Виходячи з цього, стаціонарність та не стаціонарність – це важливі поняття в аналізі часових рядів, які впливають на вибір методів аналізу та прогнозування. Багато статистичних методів аналізу часових рядів базуються на припущенні про стаціонарність ряду. Наприклад, авторегресійна модель (AR-модель) припускає, що ряд є стаціонарним.

Існують два типи стаціонарних рядів: строго стаціонарний та узагальнений стаціонарний. Строго стаціонарний ряд має сталий розподіл ймовірності та автоковаріацію. Узагальнений стаціонарний ряд може мати змінюваний розподіл ймовірності, але автоковаріація залишається сталою.

Нестаціонарний часовий ряд – це ряд, значення рівнів якого коливаються навколо середнього значення ознаки, яке з плином часу змінюється. Такі ряди мають тренд, сезонність та/або циклічні зміни в середньому значенні. Наприклад, ряд, що відображає зростання популяції, є нестаціонарним, оскільки середнє значення зростає з часом. Для аналізу та прогнозування нестаціонарних рядів, необхідно застосовувати спеціальні методи.

Є два основних підходи до моделювання прогностичної моделі нестаціонарного фінансового часового ряду [15, 16]:

1) моделювання регулярних компонент у сукупності;

2) розкладання часового ряду на компоненти та моделювання кожної компоненти окремо.

Підхід, який базується на виокремленні складових часового ряду, включає різні способи фільтрації шуму та чітке виокремлення регулярних компонент. В залежності від зв'язку складових ряду розрізняють адитивну, мультиплікативну та змішану моделі часових рядів.

Адитивна модель є одним з найпоширеніших методів аналізу та прогнозування часових рядів. Принцип її роботи полягає у тому, що ряд розкладається на окремі компоненти, які потім можуть бути аналізовані окремо та модельовані з використанням відповідних методів та моделей. Адитивна модель виражається формулою:

$$Y_t = T_t + S_t + U_t + E_t \ , \qquad (1.1)$$

де Y_t – значення часового ряду в момент часу t;

T_t – тренд;

S_t – сезонна компонента, яка відображає періодичність у часовому ряді, що повторюється з певною періодичністю;

U_t – циклічна компонента, яка відображає зміну середнього значення в часовому ряді в залежності від довгострокових циклів;

E_t – випадкова компонента, яка відображає випадкові коливання в часовому ряді, які не піддаються систематичному опису.

Мультиплікативна модель часового ряду є ще одним підходом до розкладання часового ряду на складові компоненти та їх моделювання. На відміну від адитивної моделі, мультиплікативна модель описує часовий ряд як добуток його складових частин. Мультиплікативна модель виражається формулою:

$$Y_t = T_t \times S_t \times U_t \times E_t \ , \qquad (1.2)$$

де Y_t – значення часового ряду в момент часу *t*;

T_t – тренд;

S_t – сезонна компонента;

U_t – циклічна компонента;

E_t – випадкова компонента.

Змішана модель комбінує в собі як адитивну, так і мультиплікативну моделі, в залежності від того, який тип залежності краще підходить для кожної з компонент часового ряду. Зазвичай змішана модель містить тренд та випадкову компоненту як мультиплікативні компоненти, а сезонну та циклічну компоненту – як адитивні компоненти. Змішана модель може мати такий вигляд:

$$Y_t = T_t \times S_t + U_t + E_t\,, \tag{1.3}$$

де Y_t – значення часового ряду в момент часу *t*;

T_t – тренд;

S_t – сезонна компонента;

U_t – циклічна компонента;

E_t – випадкова компонента.

Змішана модель може дозволити враховувати взаємодію між різними компонентами часового ряду, наприклад, якщо зміна тренду впливає на зміну сезонної складової або на зміну циклічної складової. Однак змішана модель може бути складнішою у використанні та інтерпретації, оскільки містить більше параметрів, які потрібно налаштувати та оцінити. Крім того, в залежності від конкретного часового ряду та його складових, змішана модель може бути менш точною або навіть непридатною для використання.

Мультиплікативну модель можна звести до адитивної моделі шляхом логарифмування. Тому при аналізі фінансових часових рядів шляхом моделювання кожної компоненти окремо частіше усього будуть адитивну модель часового ряду.

Моделювання тренду здійснюють, відфільтровуючи випадкову компоненту та коливання. Для цього використовують наступні методи згладжування:

1) *механічне вирівнювання*, яке передбачає усереднення значень рівнів ряду на певному інтервалі й добре дозволяє виокремити сезонну та випадкову компоненти;

2) *аналітичне вирівнювання*, яке розробляється на основі регресійних методів моделювання тренду шляхом побудови аналітичної функції, яка виражає залежність рівнів ряду від часу.

Застосування регресійних методів має перевагу, яка полягає у простоті, гнучкості та узгодженості їх аналізу і проєктування.

До поширеніших методів аналізу відносять: графіки свічок, ковзні середні (MA – Moving Average), експоненційні ковзні середні (EMA – Exponential Moving Average), RSI, MACD, Bollinger Bands та інші.

Графіки свічок (англ. candlestick charts) – це графіки, які використовуються для відображення динаміки цін в певний період часу. Кожна свічка складається з тіла та вусиків. Тіло свічки відображає діапазон між відкритою та закритою ціною за певний період часу, а вусики показують максимальну та мінімальну ціну за той же період.

Ковзні середні MA – це інструмент, який використовується для визначення тренду на ринку. Ковзна середня обчислюється шляхом усереднення значень цін за певний період часу. Наприклад, 200-денна ковзна середня обчислюється шляхом усереднення значень цін за останні 200 днів.

RSI (англ. Relative Strength Index) – це індикатор, який використовується для визначення перекупленості або перепроданості активу. Індикатор RSI базується на порівнянні середнього прибутку та середньої в коливальної ціни, які обчислюються за певний період часу.

MACD (англ. Moving Average Convergence Divergence) - індикатор, який використовується для визначення зміни тенденції ціни. MACD обчислюється за допомогою різниці між двома середніми значеннями цін на різні періоди часу.

Bollinger Bands - індикатор, який відображає потенційні межі, в межах яких можуть коливатися ціни активу. Формула для обчислення Bollinger Bands включає середню ціну активу та стандартне відхилення цін за певний період часу.

Для прогнозування стаціонарних часових рядів застосовують ауторегресійні моделі (AR), які використовують попередні значення залежної змінної для прогнозування майбутніх значень. AR модель може бути визначена як лінійна комбінація попередніх значень ряду:

$$y_t = \mathrm{c} + \varphi_1 y_{t-1} + \varphi_2 y_{t-2} + \varphi_p y_{t-p} + \varepsilon_t \,, \qquad (1.4)$$

де y_t – поточне значення ряду;

c – стала складова;

φ_p – коефіцієнт регресії;

p – порядок моделі;

ε_t – випадкова компонента.

Коефіцієнти авторегресії відображають вплив попередніх значень ряду на поточне значення. Чим більший порядок моделі p, тим більшу кількість попередніх значень необхідно враховувати для прогнозування поточного значення ряду. Недоліком таких моделей є те, що для побудови точних моделей необхідно визначити порядок, що не так просто, і складно визначити границю між спрощенням моделі та точністю прогнозів.

Одним із способів покращення точності AR-моделі є поєднання її з методами механічного згладжування, такими як метод експоненційного згладжування (ES-модель) та метод низької частоти (LLF-модель). Модель ARMA поєднує автокореляційну модель AR та метод ковзного середнього MA [17]. Ще однією

модифікацією є модель ARIMA, яка застосовує методи, які приводять ряд до стаціонарного та поєднує моделі AR і ковзного середнього MA, тому її можна застосовувати для прогнозу нестаціонарних часових рядів.

Ще одним, досить поширеним у сфері прогнозування курсу акцій методом є використанні штучних нейронних мереж [18]. Після тренування нейронної мережі на історичних даних, вона може бути використана для прогнозування курсу акцій в майбутньому. Непогані результати дає використання нейронних мереж на основі глибинного навчання для аналізу великої кількості історичних даних про курс акцій, а також інших факторів, які можуть впливати на його зміну. Глибинне навчання в сфері прогнозування курсу акцій дозволяє використовувати більш складні моделі, які можуть аналізувати багато факторів, що можуть впливати на курс акцій. Це дозволяє отримати більш точні та надійні прогнози, що можуть бути корисними для інвесторів та трейдерів.

Для глибинного навчання в сфері прогнозування курсу акцій зазвичай використовуються різні типи штучних нейронних мереж: рекурентні нейронні мережі (RNN) та частіше усього згорткові нейронні мережі (CNN). RNN зазвичай використовуються для аналізу часових рядів, таких як курси валют, оскільки вони можуть зберігати попередні стани та використовувати їх для прогнозування майбутніх значень. Згорткові нейронні мережі можуть бути використані для аналізу зображень графіків цін на акції, оскільки вони можуть розпізнавати певні патерни та ознаки на зображеннях.

При глибинному навчанні в сфері прогнозування курсу акцій використовуються різні фактори, які можуть впливати на курс акцій, такі як економічні показники, новини, політичні події та інші. Всі ці фактори можуть бути оброблені та використані для покращення точності прогнозів.

Формула для глибинного навчання в сфері прогнозування курсу акцій може бути складною та залежати від типу нейронної мережі та використовуваних факторів. Однак, основна ідея полягає в тому, щоб навчити нейронну мережу визначати залежності між історичними даними про курс акцій та іншими

факторами, які можуть впливати на нього, та використовувати ці знання для прогнозування майбутнього курсу. Для тренування нейронної мережі використовуються історичні дані про курс акцій та інші фактори, які можуть впливати на нього, та відповідні значення курсу акцій в майбутньому. Під час тренування нейронна мережа встановлює зв’язки між історичними даними та майбутнім курсом акцій та вчиться прогнозувати майбутні значення курсу на основі цих зв’язків.

1.3 Етапи побудови прогнозної моделі часового ряду курсу акцій

Для аналізу та прогнозування курсу акцій було обрано підхід, який полягає у виокремленні складових часового ряду та їх моделюванні окремо. Основні етапи побудови прогнозної моделі є наступними [19, 20]:

1) проведення попереднього аналізу даних, перед початком будь-якої моделювання необхідно оцінити якість даних, виявити аномальні відхилення, пропуски значень, викиди та здійснити очищення даних;

2) виявлення структури часового ряду, виявлення наявності тренду та дослідження на стаціонарність. Цей етап включає:

– побудову графіків часових рядів та їх аналіз – дозволяє візуально визначити наявність чи відсутність тренду та інших компонент ряду;

– здійснення автокореляційного аналізу – дозволяє виявити структуру часового ряду шляхом побудови кореляційної функції та аналізу динаміки зміни значень коефіцієнта автокореляції;

– достовірне підтвердження наявності чи відсутності тренду з використанням методів критерію серій або перевірки різниць середніх рівнів;

3) здійснення згладжування, фільтрації та відокремлення компонент часового ряду – на цьому етапі робиться згладжування ряду, для чого застосовують різні методи, наприклад, ковзне середнє та експоненційне згладжування;

4) побудова трендових моделей: на цьому етапі виконують побудову моделей тренду, формуючи набір апроксимуючих функцій і чисельно оцінюючи параметри цих моделей;

5) перевірка адекватності трендових моделей, оцінка точності апроксимації та вибір кращої моделі: після побудови трендових моделей, необхідно перевірити їх адекватність та рівень апроксимації, після цього можна вибрати найкращу модель;

6) дослідження випадкової компоненти, перевірка адекватності побудованої моделі часового ряду: цей етап включає аналіз випадкової компоненти часового ряду та перевірку адекватності побудованої моделі;

7) оцінка точності прогнозу з використанням побудованої моделі часового ряду: для оцінки точності прогнозу використовують різні показники: середня похибка прогнозу, середня абсолютна похибка, середня відсоткова похибка та інші;

8) прогнозування майбутніх значень часового ряду з використанням побудованої моделі.

1.4 Попередній аналіз та виявлення структури часового ряду

Дані можуть містити проблеми, такі як аномалії, відсутні дані, викиди та інші неточності, які можуть впливати на результати аналізу та моделювання. Очищення даних є важливою складовою в обробці даних і для кожного типу проблемних даних передбачає використання різних способів їх усунення [21]. Розглянемо деякі з методів виявлення проблемних даних та їх виправлення.

Виявлення пропусків: якщо в наборі даних з рівнями часового ряду є відсутні дані, то ці дані можна видалити з аналізу або заповнити усередненими значеннями. Для фінансового часового ряду, який містить ряд значень курсу акцій через однакові часу, пропуски краще замінити значенням, яке є середнім двох сусідніх рівнів ряду. Однак, якщо відсутні дані складають значну частину даних, видалення може вплинути на точність аналізу та моделювання.

Виявлення аномальних значень – викидів: значень, які суттєво відрізняються від інших даних та можуть бути помилкою в зборі даних або представляти реальну, але рідкісну подію. Викиди можна видалити або замінити на середнє значення, медіану або інше значення, яке є більш представницьким для датасету. Для виявлення аномальних значень використовується критерій Ірвіна:

$$\lambda_t = \frac{|y_t - y_{t-1}|}{\sigma}, \tag{1.5}$$

де $\sigma = \sqrt{\sigma^2}$ – середньоквадратичне відхилення часового ряду.

Методи визначення структури часового ряду допомагають ідентифікувати та вибрати оптимальну модель для прогнозування часового ряду.

Аналіз автокореляції (ACF) є одним з ключових методів для аналізу та визначення структури часових рядів [22]. Він допомагає виявити залежність між значеннями часового ряду на певних відстанях у часі та оцінити, наскільки далеко назад в часі потрібно дивитись, щоб виявити зв'язок між значеннями ряду.

ACF вимірює кореляцію між значеннями часового ряду і його лаговими версіями. Лаг – це відстань у часі між значеннями ряду, для яких розраховується кореляція. Якщо значення ACF дорівнює 1, то це означає, що значення ряду на певній відстані у часі пов'язані повністю, тобто мають максимальну кореляцію. Якщо значення ACF дорівнює 0, то це означає, що значення ряду на певній відстані у часі незалежні одне від одного. Якщо значення ACF від'ємне, то це означає, що значення ряду на певній відстані у часі мають протилежну залежність.

Коефіцієнт автокореляції r_p з лагом p визначається за наступною формулою:

$$r_p = \frac{\sum_{t=1}^{n-p}(y_t - \overline{y}_1)(y_{t+p} - \overline{y}_2)}{\sqrt{\sum_{t=1}^{n-p}(y_t - \overline{y}_1)^2 \times \sum_{t=p+1}^{n}(y_{t+p} - \overline{y}_2)^2}}, \tag{1.6}$$

де n – довжина часового ряду;

p – лаг;

y_t – значення часового ряду в момент часу t;

$\overline{y}$ – середнє арифметичне.

ACF може бути візуалізований за допомогою графіка автокореляції. Графік автокореляції - це графік, на якому вісь X представляє лаг у часі, а вісь Y - значення ACF. На графіку автокореляції можна відслідковувати зміну залежності між значеннями ряду на різних відстанях у часі. Якщо значення ACF знижуються повільно, то це означає, що значення ряду на різних відстанях у часі мають високу кореляцію, тобто залежність між ними тривала у часі. Якщо значення ACF швидко знижуються до 0, то це означає, що значення ряду на різних відстанях у часі незалежні одне від одного. Також ACF може бути використаний для виявлення сезонності та тренду в часовому ряді.

Для перевірки наявності тренду можуть використовуватися спеціальні критерії перевірки гіпотези про наявність тренду, такі як метод перевірки різниць середніх та критерій серій.

Метод перевірки різниць середніх рівнів є одним з основних методів порівняння середніх значень двох груп спостережень і визначення, чи є різниця між ними статистично значущою. Цей метод є особливо корисним, коли ми маємо дві групи незалежних спостережень з нормальним розподілом, що мають різну середню величину.

Формально, метод перевірки різниць середніх рівнів полягає в порівнянні середніх значень двох груп спостережень і обчисленні статистики t-критерій Стьюдента. Т-критерій Стьюдента залежить від тестової статистики t, яка вимірює рівень статистичної значущості різниці між двома середніми значеннями. Якщо t-критерій перевищує критичне значення, то ми можемо стверджувати, що різниця між двома середніми значеннями є статистично значущою.

Послідовність етапів методу перевірки різниць середніх рівнів є наступною:

1) часовий ряд довжиною n розбивається на дві майже однакові частини n_1 та n_2 такі, що $n = n_1 + n_2$;

2) розраховують середні обох частин:

$$\overline{y}_1 = \frac{1}{n_1}\sum_{t=1}^{n-n_1} y_t \quad (1.7)$$

$$\overline{y}_2 = \frac{1}{n_2}\sum_{t=n_1+1}^{n_1} y_t; \quad (1.8)$$

3) розраховують дисперсії обох частин ряду:

$$\sigma_1^2 = \frac{1}{n_1-1}\sum_{t=1}^{n_1}(y_t - \overline{y}_1)^2 \quad (1.9)$$

$$\sigma_2^2 = \frac{1}{n_2-1}\sum_{t=n_1+1}^{n}(y_t - \overline{y}_2)^2; \quad (1.10)$$

4) розраховують емпіричне значення F-критерію Фішера за формулою:

$$F_{\text{емп}} = \begin{cases} \frac{\sigma_1^2}{\sigma_2^2}, \text{якщо } \sigma_1^2 > \sigma_2^2 \\ \frac{\sigma_2^2}{\sigma_1^2}, \text{якщо } \sigma_1^2 < \sigma_2^2 \end{cases}; \quad (1.11)$$

5) якщо емпіричне значення критерію Фішера $F_{\text{емп}} < F_{\text{кр}}$ менше за критичне на заданому рівні значущості α, взяте зі ступенями свободи $n_i - 1$ та $n_k - 1$ (i – індекс тієї частини ряду, яка має більшу дисперсію), то гіпотеза про однорідність дисперсій приймається і необхідно переходити до наступного пункту;

6) якщо $F_{\text{емп}} \geq F_{\text{кр}}$, то гіпотеза про однорідність дисперсій відхиляється, метод не дає відповіді на питання про наявність тренду;

7) розраховують емпіричне значення t-критерію Стьюдента за наступною формулою:

$$t_{\text{емп}} = \frac{|\overline{y}_1 - \overline{y}_2|}{\sigma\sqrt{\frac{1}{n_1}+\frac{1}{n_2}}}, \tag{1.12}$$

де σ – середньоквадратичне відхилення різниці середніх:

$$\sigma = \sqrt{\frac{(n_1-1)\sigma_1^2+(n_2-1)\sigma_2^2}{n_1+n_2-2}}; \tag{1.13}$$

8) якщо емпіричне значення критерію Стьюдента менше за табличне $t_{\text{емп}} < t_{\text{кр}}$ на заданому рівні значущості α із числом ступенів свободи $n_1 + n_2 - 2$, то гіпотеза про рівність середніх приймається – тренду немає;

9) якщо $t_{\text{емп}} \geq t_{\text{кр}}$, то гіпотеза про рівність середніх відхиляється, тренд є.

Недоліком методу перевірки різниць середніх рівнів є неможливість установити наявність тренду у разі зміни тенденції в середині ряду. Тоді для підтвердження наявності тренду застосовують інші методи.

Критерій серій є одним з методів статистичного аналізу даних, який дозволяє виявляти наявність залежності між даними у вигляді серій, тобто послідовності однакових або зростаючих/спадаючих значень. Цей метод зазвичай використовується для аналізу часових рядів та інших послідовних даних, де можуть бути присутні деякі закономірності.

Основна ідея критерію серій полягає в порівнянні кількості серій однакових або значення що збільшуються/зменшуються значень, яку можна очікувати в випадковому розподілі з даними, з кількістю серій, яку насправді спостерігається в

даних. Якщо спостережувана кількість серій перевищує кількість, яку можна очікувати в випадковому розподілі, то це свідчить про наявність певної закономірності або залежності між даними.

Послідовність етапів критерію серій є наступною:

1) з часового ряду $y_1, y_2, \dots, y_n$, утворюють ранжований ряд та визначають його медіану m;

2) для часового ряду y_t формують послідовність „+" та „-" за правилом:

$$\delta_i = \begin{cases} +, \text{якщо } y_t > m \\ -, \text{якщо } y_t < m \end{cases}, \quad (1.14)$$

де $t = 1,2, \dots, n$, якщо значення t рівне медіані, це значення пропускають;

3) підраховують $\nu(n)$ – число серій в сукупності δ_i, де серія – це послідовність плюсів або мінусів, які ідуть підряд (один „+" та „-" також є серією);

4) підраховують $t_{max}(n)$ – довжину самої більшої серії, яка має найбільшу кількість „+" або „-";

5) розраховують критичне значення кількості серій $\nu_{\text{кр}}(n)$ для ряду довжиною n на рівні значущості 0,05:

$$\nu_{\text{кр}}(n) = \left[\frac{1}{2} \times (n + 1 - 1.96\sqrt{n-1}\right]; \quad (1.15)$$

6) розраховують критичне значення довжини найдовшої серії $t_{\text{кр}}(n)$ для ряду довжиною n на рівні значущості 0,05:

$$t_{\text{кр}}(n) = [3.3 \times (\lg n + 1)]; \quad (1.16)$$

7) перевіряють, чи виконуються нерівності:

$$\begin{cases} t_{max}(n) < t_{\text{кр}}(n) \\ \nu(n) > \nu(n) \end{cases}, \quad (1.17)$$

для підтвердження гіпотези про відсутність у структурі ряду тренду довжина самої більшої серії не повинна бути дуже великою, а кількість серій дуже малою;

8) якщо обидві нерівності виконуються – тренд відсутній, якщо хоча б одна з нерівностей не виконується – гіпотеза про відсутність тренду відхиляється: тренд є.

1.5 Побудова трендових моделей часового ряду

У разі виявлення у структурі аналізованого часового ряду тренду здійснюють його моделювання, виокремлюючи випадкову та сезонну компоненти [23, 24]. З цією метою використовують згладжування часового ряду: механічне та аналітичне вирівнювання.

Механічне вирівнювання здійснюється шляхом усереднення значень часового ряду на деякому інтервалі. З цією метою можна застосувати наступні методи: ковзне середнє та експоненційне згладжування.

Ковзне середнє MA є одним з найпростіших методів для визначення тренду в часовому ряді. Цей метод полягає у вирахуванні середнього значення (середньої лінії) попередніх значень часового ряду. Середнє значення використовується для згладжування збурень і відображення тренду. Ковзне середнє може бути виражене математично наступним чином:

$$S_t = (Y_t + Y_{t-1} + Y_{t-n+1})/n\,, \quad (1.18)$$

де S_t – середнє значення на момент часу t;

Y_t – значення часового ряду на момент часу t;

n – кількість попередніх значень, що беруться до уваги для визначення середнього значення.

Експоненційне згладжування EMA – є більш складним методом згладжування, який враховує не тільки попередні значення часового ряду, але і попередні значення згладженого ряду. Цей метод полягає у визначенні згладженого ряду як зваженої суми попереднього значення згладженого ряду і поточного значення вихідного ряду. Чим більше вага приділена поточному значенню вихідного ряду, тим більше впливу воно має на згладжений ряд. Експоненційне згладжування може бути виражене математично так:

$$S_t = \alpha Y_t + (1 - \alpha) S_{t-1}\ , \qquad (1.19)$$

де S_t – згладжене значення на момент часу t;

Y_t – спостережуване значення на момент часу t;

S_{t-1} – згладжене значення на попередньому кроці часу $t - 1$;

α – параметр згладжування, що приймає значення від 0 до 1.

Параметр згладжування α – це ваговий коефіцієнт, який визначає вплив нових значень на згладжений ряд. Чим більше значення α, тим більший вплив має поточне спостереження на згладжений ряд.

Коефіцієнти згладжування можуть бути визначені експериментально. Крім того, важливо правильно вибрати довжину сезонного періоду, щоб забезпечити ефективне згладжування сезонності.

Аналітичне вирівнювання реалізується з допомогою регресійних методів шляхом побудови аналітичної функції, яка характеризує залежність рівнів ряду від часу. При побудові системи прогнозування курсу акцій передбачено побудову наступних регресійних моделей: лінійна, логарифмічна, експоненціальна, степенева та поліноміальна.

Лінійна модель є найпростішою моделлю для аналізу часових рядів. Вона передбачає, що зміна значень часового ряду є лінійною, тобто пряма лінія може бути побудована, яка найкращим чином підходить до даних. Лінійна модель може бути виражена математично як:

$$y = m \cdot t + b\ , \tag{1.20}$$

де y – значення часового ряду;

t – період часу;

m – нахил прямої;

b – зсув.

Логарифмічна модель є корисною, коли зміна значень часового ряду не є лінійною, але може бути зведена до лінійної зміни після застосування логарифму. Логарифмічна модель може бути виражена математично як:

$$y = \ m \cdot ln(t) + b, \tag{1.21}$$

де y – значення часового ряду;

t – період часу;

m – нахил прямої;

b – зсув.

Експоненціальна модель передбачає, що зміна значень часового ряду є експоненційною. Ця модель часто використовується для прогнозування майбутніх значень часового ряду. Експоненціальна модель:

$$y = a \cdot b^t, \tag{1.22}$$

де y – значення часового ряду;

t – період часу;

a – початкове значення часового ряду;

b – коефіцієнт зміни.

Степенева модель передбачає, що зміна значень часового ряду є степеневою. Ця модель також може бути корисною для прогнозування майбутніх значень часового ряду. Степенева модель може бути виражена математично як:

$$y = a \cdot t^b, \tag{1.23}$$

де y – значення часового ряду;

t – період часу;

a і b – коефіцієнти.

Поліноміальна модель передбачає, що залежність між значеннями залежної змінної та часом є більш складною, ніж проста лінійна модель, і може бути описана поліномом більш високого порядку. Поліноміальна модель може бути виражена як поліном певного ступеня. Квадратична модель (ступінь 2) може бути виражена математично як:

$$y = a + bt + ct^2, \tag{1.24}$$

де y – значення залежної змінної;

t – період часу;

a, b та c – коефіцієнти моделі, які можуть бути знайдені методом найменших квадратів.

Для оцінки ступеня апроксимації трендовою моделлю основної тенденції зміни курсу акцій з часом розраховують *коефіцієнт детермінації*. Він вказує, яка частина загальної варіації може бути пояснена моделлю, тобто наскільки добре модель пояснює зміну рівнів курсу акцій з плином часу.

Коефіцієнт детермінації R-квадрат є квадратом коефіцієнта кореляції між спостережуваними значеннями рівнів часового ряду та прогнозованими значеннями, що отримані за допомогою побудованої моделі. Іншими словами, R-квадрат вимірює, наскільки добре модель підходить для опису спостережуваних даних. Значення R-квадрат знаходиться в діапазоні від 0 до 1, де 0 означає, що модель не пояснює жодної варіації залежної змінної, а 1 означає, що модель повністю пояснює варіацію зміни курсу акцій. Формула для обчислення R-квадрат:

$$R^2 = 1 - \frac{\frac{SSR}{n}}{\frac{SST}{n}} = 1 - \frac{SSR}{SST}, \qquad (1.25)$$

де SSR – сума квадратів відхилень прогнозованих значень рівнів ряду від її середнього значення;

SSE – сума квадратів відхилень прогнозованих значень рівнів ряду від її спостережуваних значень;

SST – загальна сума квадратів відхилень спостережуваних значень рівнів ряду від її середнього значення.

Зазвичай, прийнято вважати, що модель з коефіцієнтом детермінації більше 0,7 є досить точною для побудови прогнозу в контексті фінансових часових рядів. Незадовільною точністю для побудови прогнозу є модель з коефіцієнтом детермінації менше 0,6.

1.6 Метрики оцінки якості прогнозної моделі

Для оцінки точності прогнозу курсу акцій було використано наступну сукупність показників:

1) Mean Forecast Error (MFE) – середня похибка прогнозу є мірою середнього відхилення прогнозних значень від фактичних, показує напрям зміщення прогнозу – заниженими чи завищеними є прогнозні значення:

$$MFE = \frac{1}{n}\sum_t^n (y_t - \hat{y}_t), \quad (1.26)$$

де $\hat{y}_t$ та y_t – прогнозоване та фактичне значення рівнів ряду.

Зазвичай, вважається, що MFE повинен бути меншим за 0.5% від середнього значення фінансового часового ряду. Проте, це число може варіюватися в залежності від конкретного випадку;

2) Mean Absolute Error/Mean Absolute Derivation (MAE/MAD) – середнє абсолютне відхилення прогнозованих значень від фактичних:

$$MAE = MAD = \frac{1}{n}\sum_t^n |y_t - \hat{y}_t|, \quad (1.27)$$

де $\hat{y}_t$ та y_t – прогнозоване та фактичне значення рівнів ряду.

Вважається що MAE/MAD повинен бути меншим за 10% від середнього значення фінансового часового ряду;

3) Mean Percentage Error (MPE) – середня похибка у відсотках:

$$MPE = \frac{100}{n}\sum_t^n \left(\frac{y_t - \hat{y}_t}{y_t}\right), \quad (1.28)$$

де $\hat{y}_t$ та y_t – прогнозоване та фактичне значення рівнів ряду.

Загалом чим ближче значення MPE до 0 значення, тим менше похибок прогнозу, а отже, прогноз вважається більш точним;

4) Mean Squared Error (MSE) – середньоквадратична похибка дає загальне уявлення, чи є помилки при прогнозуванні:

$$MSE = \frac{1}{n}\sum_{t}^{n}(y_t - \hat{y}_t)^2, \quad (1.29)$$

де $\hat{y}_t$ та y_t – прогнозоване та фактичне значення рівнів ряду.

При підборі параметрів моделі та порівнянні декількох моделей обирають ті з них, для яких MSE буде мати менше значення;

5) Mean Absolute Percentage Error (MAPE) – середня абсолютна похибка у відсотках:

$$MAPE = \frac{100}{n}\sum_{t=1}^{n}\left|\frac{y_t - \hat{y}_t}{y_t}\right|, \quad (1.30)$$

де $\hat{y}_t$ та y_t – прогнозоване та фактичне значення рівнів ряду.

Точність прогнозу є: дуже високою, якщо MAPE менше 10%; високою, якщо MAPE становить 10-20%; задовільною, якщо MAPE становить 20-50%; незадовільною, якщо MAPE більше 50%.

1.7 Постановка задачі

Розробка системи аналізу та прогнозування фінансових часових рядів є актуальною задачею в умовах високих темпів розвитку сфери фондового ринку, оскільки прогнозування є необхідним елементом інвестиційної діяльності.

Об’єкт роботи – процес прогнозування та аналізу фінансових часових рядів.

Предмет роботи – програмні засоби та методи аналізу і прогнозування фінансових часових рядів.

Мета роботи – підвищення якості планування інвестицій на фондових ринках шляхом розробки системи прогнозування вартості комерційних компаній із

вбудованими методами аналізу часових рядів курсу акцій та побудови прогнозних моделей.

Для досягнення поставленої мети було поставлено такі **завдання**:

1) здійснити аналіз предметної сфери фінансових часових рядів та дослідити теоретичні засади прогнозування курсу акцій;

2) обґрунтувати вибір технологій та інструментальних засобів розробки системи аналізу та прогнозування фінансових часових рядів;

3) розробити, здійснити програмну реалізацію системи аналізу і прогнозування курсу акцій, дослідити якість прогнозної моделі та точність прогнозу.

Розроблювана системи аналізу та прогнозування фінансових часових рядів повинна мати наступні функціональні можливості:

1) здатність зчитувати дані: система повинна мати можливість імпортувати фінансові дані з файлів у форматі CSV;

2) очищення даних: система повинна здійснювати очищення даних – виявлення пропусків, аномалій дублікатів та корректно здійснювати корегування набору даних;

3) здійснення розрахунку статистичних характеристик та перевірки наявності тренду з використанням методу перевірки середніх різниць та з допомогою критерію серій;

4) проведення автокореляційного аналізу часового ряду: розрахунок коефіцієнтів автокореляції, побудова корелограми, виявляти структурних копонентфі часвого ряду, здійснення перевірки на стаціонарність;

5) для нестаціонарного часового ряду – здійснення згладжування, фільтрації, виокремлення компонент часового ряду та побудову трендових моделей, для стаціонарного часового ряду – побудову авторегресійної моделі;

6) здійснення перевірки трендових моделей на адекватність, оцінка точності апроксимації, вибір кращої моделі;

7) дослідження випадкової компоненти: система повинна проводити аналіз випадкової компоненти, перевірку на наявність шуму для перевірки адекватності побудованої моделі часового ряду;

8) розрахунок метрик оцінки якості моделі ті її точності, прогнозування нових значень.

Ці вимоги є основою для подальшого проєктування та реалізації системи.

Висновки до розділу 1

В даному розділі бакалаврської роботи розглянуто теоретичні засади аналізу та прогнозування фінансових часових рядів. Фондовий ринок є важливою складовою ринкової економіки, на якому здійснюється продаж та купівля акцій комерційних компаній. Застосування аналізу та прогнозування фінансових часових рядів дозволяє отримувати важливу інформацію про фінансові ринки та допомагає в прийнятті обґрунтованих рішень щодо інвестування коштів. Проаналізовано програмні засоби та методи, які використовуються для прогнозування курсу акцій.

Виявлено основні типи систем та програмних сервісів, які дозволяють здійснювати побудову прогнозів часових рядів: статистичні пакети (STATISTICA, IBM SPSS Statistics, Minitab, MatLab, SAS, S-Plus), табличні процесори (MS Excel, MS Excel 365), пакети візуального проєктування (MS Machine learning, Orange 3), ERP-системи, Business Intelligence системи, когнітивні та академічні системи прогнозування. Вони мають широкий функціонал для проведення аналізу часових рядів, мають вбудовані засоби для реалізації багатьох моделей та методів інтелектуального аналізу та побудови прогнозу. Однак їх використання для побудови прогнозу потребує трудомісткого процесу розгортання та підтримки, не завжди є системним та гнучким, потребує знання мов програмування та розуміння на професійному рівні усіх етапів аналізу даних та обґрунтування вибору оптимальної прогнозної моделі. Існує потреба у розробці системи, яка має зручний

та зрозумілий інтерфейс, проста у налаштуванні й автоматизує усі етапи аналізу та прогнозування часових рядів курсу акцій.

Здійснений аналіз дозволив виявити велику кількість методів та моделей, які використовують для побудови прогнозу курсу акцій. Це методи прогнозування із використанням глибинних нейронних мереж, ауторегресійні AR-моделі, MA-модель ковзних середніх, EMA-модель експоненційних ковзних середніх, регресійні методи моделювання шляхом побудови лінійної та нелінійної аналітичної функції, адитивна AM-модель та узагальнена адитивна GAM-модель, мультиплікативна модель, ARIMA-модель інтегрованої авторегресії ковзного середнього та багато інших.

Важливе значення для побудови прогнозної моделі часового ряду має виявлення основних структурних складових часового ряду: тренду, сезонної, циклічної та випадкової компонент. Для прогнозування стаціонарного часового ряду курсу акцій, які не мають тренду та коливань, а їх статистичні характеристики не змінюються часом, будують ауторегресійну AR-модель. Для нестаціонарних часових рядів є два підходи до побудови прогностичної моделі: моделювання регулярних компонент у сукупності та розкладання часового ряду на компоненти та моделювання кожної компоненти окремо.

Важливе значення має попередній аналіз даних часового ряду, який включає очищення даних, розрахунок статистичних характеристик та виявлення структури часового ряду й установлення його стаціонарності. Очищення даних включає виявлення пропусків, дублікатів та викидів – аномальних значень. Для виявлення структури часового ряду застосовують візуальний аналіз його графіку, кореляційний аналіз та критерії перевірки гіпотези про наявність тренду – метод перевірки різниць середніх та критерій серій.

Для побудови моделі нестаціонарного часового ряду при аналізі фінансових даних шляхом моделювання кожної компоненти окремо передбачено побудову адитивної моделі часового ряду (мультиплікативна модель зводиться до адитивної шляхом логарифмування). Моделювання тренду здійснюють, відфільтровуючи

випадкову компоненту та коливання із використанням сукупності методів механічного та аналітичного згладжування. Механічне згладжування передбачає побудову МА-моделі ковзних середніх та ЕМА-моделі експоненційних ковзних середніх. Аналітичне згладжування передбачає моделювання тренду шляхом побудови аналітичних регресійних лінійних та нелінійних функцій.

Розглянуті підходи до прогнозування дали змогу визначити переваги та недоліки кожного методу, а також вибрати найбільш ефективний для даної задачі. Фінансові ринки є дуже непередбачуваними, то прогнозування курсу акцій не є завжди точним та може містити певну похибку. Тому важливо використовувати не один, а декілька методів для отримання більш точного прогнозу та зменшення ризику помилкових рішень.

Для створення автоматизованої системи аналізу та прогнозування доцільно передбачити побудову сукупності моделей, визначення їх адекватності та точності і вибору оптимальної моделі для прогнозування нових значень курсу акцій. Основні етапи аналізу та прогнозування включають: як збір та попередню обробку даних,, виявлення структури часового ряду, вибір методу прогнозування, розробка моделі прогнозування та оцінка її точності.

2 ТЕХНОЛОГІЇ ТА ЗАСОБИ РОЗРОБКИ СИСТЕМИ АНАЛІЗУ ТА ПРОГНОЗУВАННЯ ФІНАНСОВИХ ЧАСОВИХ РЯДІВ

2.1 Середовище розробки Visual Studio Code

Visual Studio Code (VS Code) – середовище розробки, розроблене Microsoft для Windows, Linux та macOS [25]. Позиціонується як «легкий» редактор коду для кросплатформної розробки веб-застосунків. Він призначений для програмістів та розробників програмного забезпечення і надає широкі можливості для написання коду, редагування файлів, налагодження програм та керування проектами. Включає в себе відладчик, інструменти для роботи з Git, підсвічування синтаксису, IntelliSense та засоби для рефакторингу. Це робить його потужним інструментом для розробки програмного забезпечення на різних платформах і з різними технологіями. Має широкі можливості для кастомізації теми користувача, поєднання клавіш і файли конфігурації. Поширюється безкоштовно, розробляється як програмне забезпечення з відкритим вихідним кодом, але готові збірки поширюються під ліцензією проприетарной.

Visual Studio Code був анонсований 29 квітня 2015 року компанією Microsoft на конференції Build, і незабаром випустили бета-версію. 18 листопада 2015 Visual Studio Code був випущений під ліцензією MIT, а вихідний код був опублікований на GitHub. Анонсовано підтримку розширень. 14 квітня 2016 року Visual Studio Code вийшов із стадії бета-тестування.

VS Code відрізняється своєю легкістю, швидкодією та розширюваністю. Він підтримує багато мов програмування, включаючи JavaScript, Python, C++, Java і багато інших. Редактор має потужну систему підсвічування синтаксису, автодоповнення коду, вбудовану систему контролю версій і можливість використання розширень, які розширюють його функціональність.

Visual Studio Code є популярним вибором серед розробників завдяки своїм можливостям, легкості використання та активному співтовариству розробників, що підтримує редактор і створює розширення для його розширення.

2.2 Мова гіпертекстової розмітки HTML, каскадні таблиці стилів CSS

HyperText Markup Language (HTML) – стандартизована мова гіпертекстової розмітки документів для перегляду веб-сторінок у браузері [26]. Елементи HTML є будівельними блоками сторінок HTML. За допомогою HTML різні конструкції, зображення та інші об'єкти, такі як інтерактивна веб-форма, можуть бути вбудовані в сторінку, що відображається. HTML пропонує засоби для створення заголовків, абзаців, списків, посилань, цитат та інших елементів. Елементи HTML виділяються тегами, записаними з використанням кутових дужок. Такі теги, як <img /> і <input />, безпосередньо вводять контент на сторінку. Інші теги, такі як <p>, оточують і оформляють текст у собі і можуть включати інші теги як поделементів. Браузери не відображають HTML-теги, але використовують їх для інтерпретації вмісту сторінки.

Мова гіпертекстової розмітки HTML був розроблений британським вченим Тімом Бернерсом-Лі приблизно в 1986-1991 роках у стінах Європейської організації з ядерних досліджень в Женеві у Швейцарії. HTML створювався як мова для обміну науковою та технічною документацією, придатний для використання людьми, які не є фахівцями в галузі верстки. HTML успішно справлявся із проблемою складності SGML шляхом визначення невеликого набору структурних та семантичних елементів – дескрипторів. За допомогою HTML можна просто створити відносно простий, але красиво оформлений документ. Крім спрощення структури документа, HTML внесена підтримка гіпертексту. Мультимедійні можливості було додано пізніше.

Першим загальнодоступним описом HTML був документ "Теги HTML", вперше згаданий в Інтернеті Тімом Бернерсом-Лі наприкінці 1991. У ньому

описуються 18 елементів, що становлять початковий, відносно простий дизайн HTML. За винятком тега гіперпосилання, на них сильно вплинув SGMLguid, внутрішній формат документації, що базується на стандартній узагальненій мові розмітки SGML, в CERN. Одинадцять із цих елементів усе ще існують у HTML 4.

Спочатку мова HTML була задумана і створена як засіб структурування та форматування документів без їх прив'язки до засобів відображення. В ідеалі, текст з розміткою HTML повинен був без стилістичних та структурних спотворень відтворюватися на устаткуванні з різним технічним оснащенням. Однак сучасне застосування HTML дуже далеке від його початкового завдання. Наприклад, тег <table> призначений для створення в документах таблиць, але іноді використовується для оформлення розміщення елементів на сторінці. З часом основна ідея платформонезалежності мови HTML була принесена в жертву сучасним потребам у мультимедійному та графічному оформленні.

Текстові документи, що містять розмітку мовою, обробляються спеціальними програмами, які відображають документ у його форматованому вигляді. Такі програми, які називаються браузерами, зазвичай надають користувачеві зручний інтерфейс для запиту веб-сторінок, їх перегляду і, при необхідності, відправки введених користувачем даних на сервер. Найбільш популярними на сьогоднішній день браузерами є Google Chrome, Mozilla Firefox, Opera, Microsoft Edge та Safari.

Cascading Style Sheets (CSS) – формальна мова декорування та опису зовнішнього вигляду веб-сторінки, написаного з використанням мови розмітки HTML [27].

CSS використовується розробниками веб-сторінок для визначення кольорів, шрифтів, стилів, розташування окремих блоків та інших аспектів представлення зовнішнього вигляду цих веб-сторінок. Основною метою розробки CSS є огородження та відокремлення опису логічної структури веб-сторінки, яке робиться за допомогою HTML, від опису зовнішнього вигляду цієї веб-сторінки, яке тепер робиться за допомогою формальної мови CSS. Такий поділ збільшує

доступність веб-сторінки, надати більшу гнучкість та можливість управління його поданням, а також зменшити складність та повторюваність у структурному вмісті.

Крім того, CSS дозволяє представляти один і той же документ у різних стилях або методах виводу, таких як екранне подання, друковане подання, читання голосом або під час виведення пристроями, які використовують шрифт Брайля.

CSS – одна з широкого спектра технологій, схвалених консорціумом W3C і отримали загальну назву "стандарти Web". У 1990-х роках стала зрозумілою необхідність стандартизувати Web, створити якісь єдині правила, за якими програмісти та веб-дизайнери проектували б сайти. Так з'явилися мови HTML 4.01 і XHTML і стандарт CSS.

На початку 1990-х браузери мали свої стилі для відображення веб-сторінок. HTML розвивався дуже швидко і був здатний задовольнити всі потреби з оформлення інформації, що існували на той момент, тому CSS не отримав тоді широкого визнання.

Термін «каскадні таблиці стилів» було запропоновано Хоконом Лі у 1994 році. Разом із Бертом Босом він почав розвивати CSS.

На відміну від багатьох мов стилю, що існували на той момент, CSS використовує успадкування від батька до нащадка, тому розробник може визначити різні стилі, ґрунтуючись на вже визначених раніше стилях.

У 1990-х Консорціум Всесвітньої павутини став виявляти інтерес до CSS, і в грудні 1996 року було видано рекомендацію CSS1.

За допомогою CSS можна визначити різні аспекти зовнішнього вигляду веб-сторінки, зокрема наступні:

1) *селектори*: CSS використовує селектори для націлювання на певні елементи HTML на веб-сторінці. Селектори можуть базуватися на типах елементів, іменах класів, ідентифікаторах, атрибутах тощо.

2) *властивості*: властивості CSS визначають, як мають бути оформлені вибрані елементи. Приклади властивостей CSS включають колір тексту, розмір

тексту, колір фону, інтервал навколо елемента, рамка навколо елемента і тому подібне.

3) *значення:* властивостям CSS призначаються значення для визначення конкретного стилю. Наприклад, властивість кольору може мати значення самої назви кольору, шістнадцяткове значення кольору або значення кольору RGB.

4) *каскадування:* термін «каскадування» в CSS стосується способу об'єднання кількох таблиць стилів для оформлення веб-сторінки. CSS дотримується певного порядку пріоритету, що дозволяє застосовувати стилі з різних джерел у певній ієрархії.

5) *спадкування:* CSS підтримує концепцію успадкування, коли стилі, застосовані до батьківських елементів, можуть успадковуватися їхніми дочірніми елементами. Це спрощує процес стилізації кількох елементів зі схожими властивостями.

6) *медіа-запити:* CSS дозволяє створювати медіа-запити для застосування різних стилів на основі характеристик пристрою чи екрана, на якому відображається веб-сторінка. Медіа-запити забезпечують адаптивний дизайн, дозволяючи веб-сторінкам адаптуватися та забезпечувати оптимальну взаємодію з користувачами на різних пристроях і розмірах екрана.

CSS широко підтримується сучасними веб-браузерами та відіграє вирішальну роль у веб-розробці. Він забезпечує гнучкість, можливість багаторазового використання та підтримку стилів, полегшуючи створення візуально привабливих і узгоджених веб-сторінок.

2.3 Препроцесор SASS

Syntactically awesome style sheets (SASS) – це мова сценаріїв препроцесора, яка інтерпретується або компілюється в каскадні таблиці стилів CSS [28]. Він представляє додаткові функції та вдосконалення, щоб зробити написання та керування таблицями стилів CSS більш ефективним і організованим.

Sass складається з двох синтаксисів. Оригінальний синтаксис, який називається «синтаксис із відступами». Він використовує відступи для розділення блоків коду та символи нового рядка для розділення правил. Новий синтаксис SCSS використовує блочне форматування, подібне до CSS. Він використовує дужки для позначення блоків коду та крапки з комою для розділення правил у блоці.

SCSS є надмножиною CSS, тобто будь-який дійсний код CSS також є дійсним кодом SCSS. Він додає нові функції, такі як змінні, вкладення, міксини та успадкування, які недоступні у звичайному CSS. Ці функції підвищують читабельність, можливість повторного використання та обслуговування коду CSS. Ключові особливості SCSS є такими:

1) дозволяє використовувати змінні для зберігання значень, які повторно використовуються в таблиці стилів. Це забезпечує узгодженість і спрощує технічне обслуговування, дозволяючи легко змінювати значення в одному місці;

2) підтримує вкладеність селекторів, що забезпечує більш інтуїтивне та структуроване представлення стилів. Це зменшує кількість повторень і полегшує читання та розуміння коду;

3) дозволяє використовувати міксини. Міксини – це багаторазові блоки стилів, які можна включати в кілька селекторів. Вони дозволяють розробникам легко визначати та застосовувати набори стилів, зменшуючи дублювання коду та покращуючи зручність обслуговування;

4) дозволяє розбивати таблиці стилів на декілька файлів, які називаються частками. Частки потім імпортуються в головний файл SCSS, полегшуючи модульну та організовану структуру коду;

5) надає можливість створювати успадкування стилів за допомогою директиви @extend. Це дозволяє застосовувати стилі з одного селектора до іншого, сприяючи повторному використанню коду та зменшуючи дублювання;

6) представляє математичні оператори, такі як +, -, * та /, які можна використовувати для виконання обчислень числових значень у таблицях стилів;

7) підтримує керуючі директиви, такі як @if, @for та @each, які дозволяють розробникам писати умовні оператори та цикли у своїх таблицях стилів. Це забезпечує більшу гнучкість і динамічність у створенні стилів.

Щоб використовувати SCSS, його потрібно скомпілювати у звичайний CSS, перш ніж його можна буде використовувати у веб-проекті. Існують різні інструменти та системи збірки, які можуть компілювати файли SCSS у CSS, наприклад Sass, node-sass або libSass.

Загалом SCSS розширює можливості CSS, роблячи його потужнішим і зручнішим у роботі, особливо для великих і складніших проектів. Це покращує продуктивність, організацію коду та повторне використання коду, що зрештою призводить до більш зручних і ефективних таблиць стилів.

2.4 Мова програмування JavaScript. Транскомпілятор Babel

JavaScript (JS) – мультипарадигмова мова програмування [29]. Підтримує об'єктно-орієнтований, імперативний та функціональний стилі.

Є реалізацією специфікації ECMAScript та в основному використовується для додавання інтерактивності та динамічної поведінки веб-сторінок. JavaScript підтримується всіма сучасними веб-браузерами та широко використовується у веб-розробці. Також використовується як вбудована мова для програмного доступу до об'єктів застосунків.

На JavaScript вплинули багато мов, при розробці була мета зробити мову схожою на Java. Мовою JavaScript не володіє будь-яка компанія або організація, що відрізняє його від низки мов програмування, що використовуються у веб-розробці. Найперша реалізація JavaScript була створена Бренданом Ейхом у компанії Netscape, і з того часу оновлюється, щоб відповідати ECMA-262 Edition 5 і пізнішим версіям. Ключові функції та аспекти JavaScript є наступними:

1) *сценарії на стороні клієнта*: JavaScript в основному використовується як мова сценаріїв на стороні клієнта, тобто він працює у веб-браузері користувача, а

не на веб-сервері. Це дозволяє розробникам створювати інтерактивні веб-елементи, перевіряти дані форми, маніпулювати об’єктною моделлю документа, обробляти події та виконувати різні дії безпосередньо на пристрої користувача;.

2) *мова сценаріїв*: JavaScript часто називають мовою сценаріїв, оскільки вона не потребує явної компіляції. Замість цього він виконується рядок за рядком під час виконання. Це дозволяє швидко розвиватися та експериментувати;.

3) *об’єктно-орієнтований:* JavaScript підтримує принципи об’єктно-орієнтованого програмування, дозволяючи розробникам створювати та керувати об’єктами, визначати класи та використовувати успадкування та поліморфізм. Він надає такі функції, як інкапсуляція, абстракція та модульність;

4) *програмування, кероване подіями:* JavaScript використовує парадигму програмування, кероване подіями, де код виконується у відповідь на певні події або дії користувача. Розробники можуть визначити обробники подій або прослуховувачі для відповіді на такі події, як натискання кнопок, надсилання форм, рухи миші та взаємодії з клавіатурою;

5) *універсальність*: JavaScript – це універсальна мова, яку можна використовувати не лише для веб-розробки, але й для програмування на стороні сервера з Node.js, розробки мобільних додатків з використанням фреймворків, таких як React Native та Ionic і навіть розробки додатків для настільних ПК з такими фреймворками як Electron;

6) *бібліотеки та фреймворки сторонніх розробників*: JavaScript має величезну екосистему бібліотек і фреймворків, які розширюють його функціональні можливості та спрощують завдання розробки. Деякі популярні фреймворки включають React.js, Angular.js і Vue.js, які допомагають створювати масштабовані та складні веб-додатки;

7) *інтеграція з HTML і CSS*: JavaScript легко інтегрується з HTML і CSS, дозволяючи динамічно маніпулювати елементами та стилями веб-сторінки. Його можна вставити безпосередньо в HTML-код за допомогою тегів сценарію або додати із зовнішніх файлів JavaScript.

JavaScript став важливою мовою для веб-розробки, забезпечуючи покращену взаємодію з користувачем, інтерактивність і швидкість реагування на веб-сайтах. Він продовжує розвиватися з новими мовними функціями та вдосконаленнями, що робить його потужним інструментом для розробки сучасних веб-додатків.

Популярний ***транскомпілятор JavaScript Babel*** з відкритим кодом дозволяє розробникам писати сучасний код JavaScript, використовуючи найновіші мовні функції та синтаксис, а потім перетворює цей код у зворотно сумісні версії, які можна запускати в старіших браузерах або середовищах, які не підтримують останні функції JavaScript [30]. Розглянемо ключові особливості Babel.

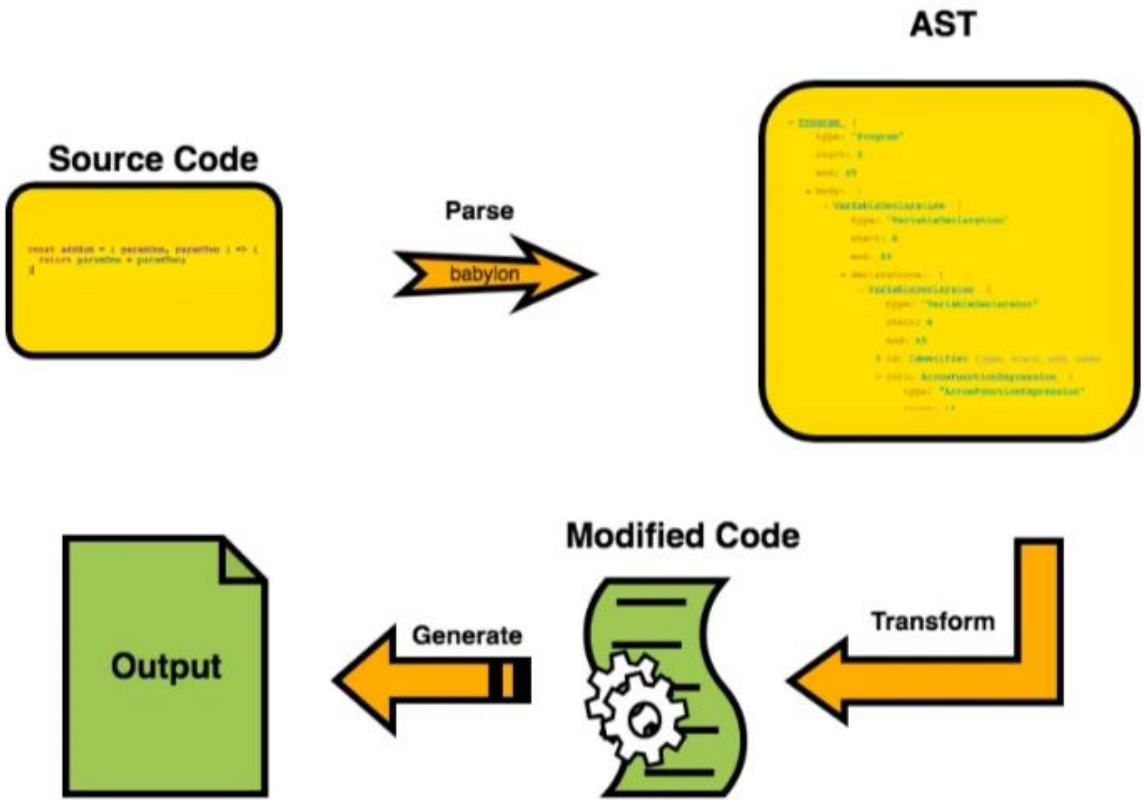

Рисунок 2.2 – Транскомпілятор JavaScript Babel

Транскомпіляція JavaScript. Babel використовує сучасний код JavaScript, зазвичай написаний з використанням найновіших стандартів ECMAScript (ES), таких як ES6 або ESNext, і транспілює його в еквівалентний код, який можна виконувати в старіших середовищах JavaScript. Це гарантує, що додатки, створені

з використанням найновіших мовних функцій, можуть працювати в браузерах і платформах з обмеженою підтримкою нових версій JavaScript.

Мовні функції та синтаксис. Babel підтримує широкий спектр мовних можливостей і синтаксису JavaScript, включаючи функції зі стрілками, класи, шаблонні літерали, призначення деструктуризації, async/await тощо. Це дозволяє розробникам писати код за допомогою цих сучасних мовних конструкцій, не турбуючись про проблеми сумісності.

Плагіни та пресети. Babel можна налаштовувати та розширювати. Він пропонує архітектуру на основі плагінів, яка дозволяє розробникам увімкнути або вимкнути певні перетворення або мовні функції відповідно до вимог проекту. Пресети Babel — це попередньо налаштовані набори плагінів, які забезпечують зручний спосіб увімкнути певний набір трансформацій для певного випадку використання, наприклад React або TypeScript.

Інтеграція інструментів збірки. Babel добре інтегрується з популярними інструментами збирання та програмами запуску завдань, такими як webpack, Rollup і Gulp. Його можна плавно включити в процес збірки, дозволяючи автоматичну транспіляцію коду JavaScript під час етапу збирання або об’єднання.

Полізаповнення та сумісність із веб-переглядачами. Окрім транспіляції сучасних мовних функцій, Babel також може включати полізаповнення або бібліотеки часу виконання, які забезпечують функції, яких бракує в старіших браузерах. Ці полізаповнення допомагають забезпечити узгоджену поведінку коду в різних середовищах і емулювати функції, які не підтримуються браузером.

Екосистема та спільнота. Babel має процвітаючу екосистему з широким набором плагінів і попередніх налаштувань, розроблених спільнотою. Це дозволяє розробникам використовувати існуючі рішення для конкретних випадків використання або налаштовувати Babel відповідно до потреб свого проекту.

Babel відіграв важливу роль у розвитку JavaScript, дозволивши прийняти сучасні функції мови та забезпечивши зворотну сумісність. Він став важливим

інструментом в екосистемі JavaScript, що дає змогу розробникам писати чистіший і зручніший код, підтримуючи при цьому широкий діапазон середовищ.

2.6 Збірник модулів Webpack, формат обміну даними JSON

Webpack – це потужний і широко використовуваний збирач модулів для програм JavaScript [31]. Він бере кілька модулів із залежностями та об’єднує їх в один оптимізований файл або набір файлів, які можна завантажити веб-браузером.

Webpack спрощує керування модулями JavaScript, створюючи графік залежностей, який відображає всі залежності між різними модулями в програмі. Потім він використовує завантажувачі та плагіни для обробки та трансформації цих модулів, уможливлюючи використання різноманітних ресурсів і мов у екосистемі JavaScript, таких як CSS, зображення, TypeScript тощо.

Ключові функції Webpack включають:

1) об’єднання модулів – Webpack може обробляти не лише файли JavaScript, але й інші ресурси, такі як CSS, зображення та шрифти, що дозволяє імпортувати ці ресурси безпосередньо у ваші модулі JavaScript і об’єднувати їх разом для ефективної доставки;

2) керування залежностями – Webpack аналізує залежності між модулями та автоматично вирішує їх, гарантуючи, що кожен модуль включено в пакет у правильному порядку;

3) розбиття коду – Webpack дає змогу розділити ваш код на кілька фрагментів, забезпечуючи кращу продуктивність і відкладене завантаження модулів, що особливо корисно для великих програм, де попереднє завантаження всього коду може призвести до повільного початкового завантаження;

4) сервер розробки та гаряча заміна модулів – Webpack надає сервер розробки, який обслуговує вашу програму і автоматично оновлює браузер, коли вносяться зміни та підтримує гарячу заміну модулів, що дозволяє оновлювати модулі без повного оновлення сторінки, зберігаючи стан програми;

5) розширюваність – Webpack дуже розширюється за допомогою завантажувачів і плагінів: завантажувачі перетворюють певні типи файлів, як-от транспіляція TypeScript у JavaScript або застосування препроцесорів CSS, а плагіни пропонують додаткові функції, такі як оптимізація коду, керування активами та налаштування середовища.

Загалом, Webpack допомагає оптимізувати процес розробки, надаючи комплексне рішення для керування та об’єднання модулів JavaScript та їхніх залежностей, оптимізуючи продуктивність і ефективність веб-додатків. Webpack – це потужний і широко використовуваний модульний об’єднувач для додатків JavaScript. Він бере кілька модулів із залежностями та об’єднує їх в один оптимізований файл або набір файлів, які можна завантажити веб-браузером (рис. 2.2).

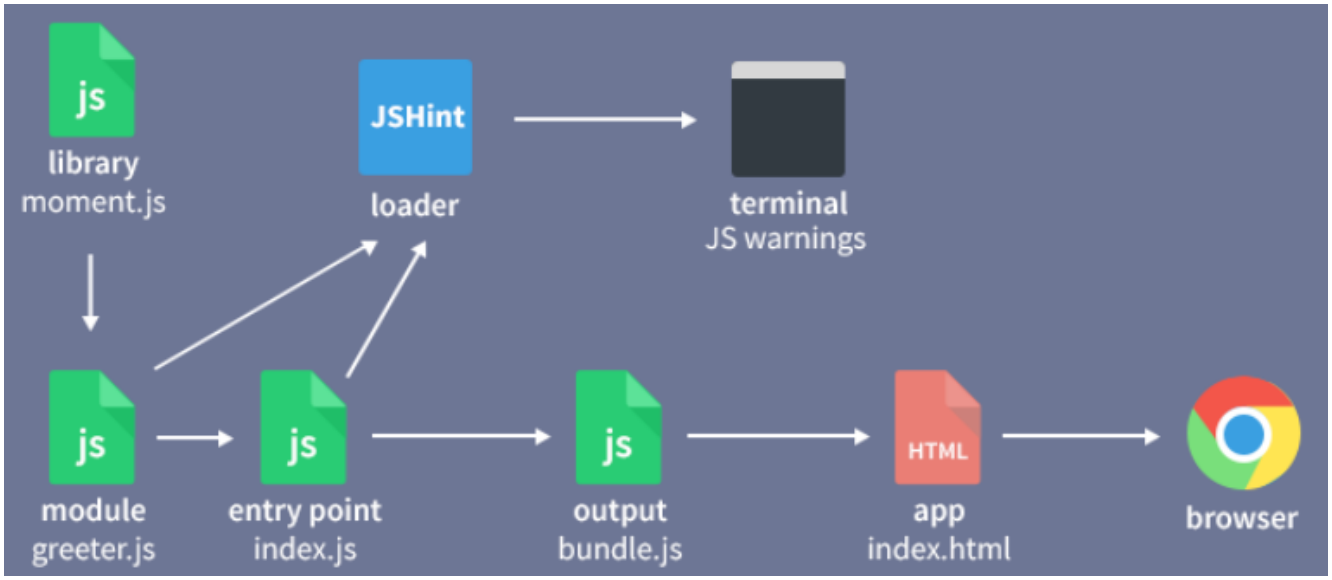

Рисунок 2.2 – Схема збірки модулів Webpack

JavaScript Object Notation (JSON) є легким форматом обміну даними, який використовується для передачі структурованих інформаційних об'єктів між клієнтськими додатками та серверами [32]. Він базується на синтаксисі JavaScript, але може бути використаний в багатьох інших мовах програмування.

JSON виник із потреби в протоколі сеансу зв’язку між сервером і браузером у реальному часі без використання плагінів браузера, таких як Flash або Java-аплети, домінуючих методів, які використовувалися на початку 2000-х.

Крокфорд першим визначив і популяризував формат JSON. Абревіатура виникла в компанії State Software, співзасновником якої були Крокфорд та інші в березні 2001 року. Співзасновники погодилися побудувати систему, яка б використовувала стандартні можливості браузера та забезпечувала рівень абстракції для веб-розробників для створення веб-додатків із збереженням стану, які мали постійне дуплексне з’єднання з веб-сервером, утримуючи два з’єднання протоколу передачі гіпертексту (HTTP) відкритими та повторюючи їх до стандартного тайм-ауту браузера, якщо подальший обмін даними не відбувався.

Співзасновники провели круглий стіл і проголосували, чи називати формат даних JavaScript Markup Language (JSML) чи JavaScript Object Notation (JSON), а також під яким типом ліцензії зробити його доступним. Веб-сайт JSON.org був запущений у 2001 році. У грудні 2005 року Yahoo! почав пропонувати деякі свої веб-сервіси в JSON. У жовтні 2013 року Ecma International опублікувала перше видання свого стандарту JSON ECMA-404.

JSON представляє дані у вигляді текстових рядків із строго визначеною синтаксичною структурою. Він використовує прості типи даних, такі як рядки, числа, логічні значення, масиви та об'єкти, для опису інформації. Дані у JSON виглядають подібно до асоціативних масивів у багатьох мовах програмування.

Основні переваги JSON:

1) простота читання та запису – JSON використовує зрозумілу людям текстову форму для представлення даних, що робить його легким у використанні та розумінні;

2) незалежність від мови – JSON може бути використаний у багатьох мовах програмування, що робить його універсальним форматом обміну даними;

3) підтримка різних типів даних – JSON підтримує рядкові дані, числа, логічні значення, масиви та об'єкти, що дозволяє представляти різнорідні дані у структурованому форматі;

4) легкість інтеграції з веб-додатками – багато сучасних веб-служб та API використовують JSON для передачі даних між клієнтом та сервером, що робить його популярним вибором для роботи з даними веб-додатків.

JSON забезпечує простий і стандартизований спосіб обміну даними між різними системами, оскільки він підтримується широким спектром мов програмування та платформ. Він зазвичай використовується в API для надсилання та отримання даних у структурованому форматі, що забезпечує взаємодію між різними системами. Дані JSON можна легко серіалізувати (перетворити на рядок) і десеріалізувати (розібрати з рядка) різними мовами програмування, що робить їх популярним вибором для передачі та зберігання даних.

2.7 Бібліотеки для роботи з даними та побудови і аналізу трендових моделей

При розробці системи аналізу та прогнозування фінансових часових рядів було використано бібліотеки SheetJS, Regression-js, Chart.js.

Chart.js - це потужна бібліотека JavaScript для візуалізації даних у веб-додатках. Вона надає простий і зрозумілий спосіб створення різноманітних графіків, діаграм і діагностичних засобів, що допомагають представити дані у зрозумілій і привабливій формі. Основні особливості Chart.js є наступними:

1) легкість використання: Chart.js має простий та зрозумілий API, що дозволяє швидко створювати графіки та діаграми без глибоких знань у графічному дизайні або веб-розробці;

2) гнучкість: бібліотека надає широкий набір налаштувань та параметрів для керування виглядом графіків. Надає можливість змінювати кольори, типи ліній, шрифти, масштаби та багато іншого, щоб налаштувати графіки під свої потреби;

3) підтримка різних типів графіків: Chart.js підтримує широкий спектр типів графіків, включаючи лінійні, стовпчасті, кругові, ареольні, розсіяні та багато інших. Це дає можливість вибрати найбільш підходящий тип графіка для представлення ваших даних;

4) анімація: візуальні ефекти та анімація можуть покращити враження користувача. Chart.js надає плавні анімаційні переходи між станами графіка, що допомагають зрозуміти зміни в даних;

5) відповідний дизайн: графіки, створені за допомогою Chart.js, виглядають сучасно та стильно. Бібліотека має вбудовані теми оформлення, які допомагають створювати привабливі графіки з мінімальними зусиллями;

6) мобільна підтримка: Chart.js добре працює на мобільних пристроях і адаптується до різних розмірів екранів. Надає можливість створювати адаптивні графіки, які зручно переглядати на будь-якому пристрої;

7) розширюваність: Chart.js можна розширювати за допомогою додаткових модулів, які додають нові функціональні можливості. Надає більшу гнучкість та можливості для створення специфічних типів графіків або діаграм.

Загалом, Chart.js є потужним інструментом для візуалізації даних. Вона дозволяє швидко створювати привабливі та інтерактивні графіки, що допомагають зрозуміти та аналізувати дані.

Regression-js - це бібліотека JavaScript, призначена для побудови і аналізу регресійних моделей, яка застосовувалася для моделювання трендових моделей часового ряду курсу акцій. Regression-js надає простий та потужний інтерфейс для виконання різних видів регресійного аналізу, включаючи просту лінійну регресію, множинну лінійну регресію, поліноміальну регресію та інші.

Основні особливості regression-js є такими:

1) простота використання. Бібліотека надає зрозумілий та легкий у використанні API для побудови регресійних моделей. Надає можливість легко визначити незалежні та залежні змінні, виконати побудову моделі та отримати результати аналізу;

2) різні типи регресій. Regression-js підтримує різні типи регресійних моделей. Надає можливість використовувати лінійну регресію для прогнозування залежності між двома змінними, або використовувати поліноміальну регресію для апроксимації складніших залежностей. Також є підтримка множинної лінійної регресії для аналізу багатовимірних даних;

3) підтримка розріджених даних. Regression-js може працювати з розрідженими даними, де деякі значення можуть бути відсутніми. Вона автоматично обробляє відсутні значення і використовує наявні дані для побудови моделі;

4) статистичні характеристики. Бібліотека надає різні статистичні характеристики для аналізу регресійних моделей. Надає можливість отримати значення коефіцієнтів регресії, стандартні помилки, коефіцієнти детермінації та інші показники, які допоможуть вам оцінити якість моделі;

5) візуалізація результатів. Regression-js надає можливість візуалізувати результати аналізу регресійних моделей за допомогою графіків. Надає можливість відобразити реальні та прогнозовані значення, а також графік залишків для оцінки точності моделі;

6) масштабованість. Бібліотека добре масштабується і може використовуватися для обробки великих обсягів даних. Вона ефективно оптимізована для швидкої обробки регресійних аналізів навіть з великою кількістю точок даних.

Загалом, regression-js є потужним інструментом для аналізу регресійних моделей в JavaScript. Вона дозволяє легко побудувати регресійні моделі, аналізувати їх та отримувати результати, які допоможуть зрозуміти залежності між змінними та прогнозувати значення на основі цих залежностей.

Бібілотеку SheetJS – потужну бібліотеку JavaScript, було вирішено застосовувати для роботи з вхідним набором даних, що містить часовий ряд, представлений у різних форматах. SheetJS дозволяє зчитувати, записувати і

маніпулювати даними в електронних таблицях у форматі Excel з використанням файлів XLSX, XLS, CSV та інших форматів.

Основні особливості SheetJS є такими:

1) зчитування та запис даних. Бібліотека дозволяє легко зчитувати дані з електронних таблиць у форматі Excel і зберігати їх у JavaScript-об'єктах. Надає можливість отримати доступ до окремих комірок, рядків або стовпців, а також здійснювати різні операції з даними, такі як фільтрація, сортування, обчислення та інші. Після внесення змін може зберігати дані у файлі Excel або іншому підтримуваному форматі;

2) підтримка різних форматів. SheetJS підтримує різні формати файлів, включаючи XLSX (Excel 2007+), XLS (Excel 97-2003), CSV (розділений комами) та інші. Ви можете зчитувати дані з файлів у цих форматах і зберігати дані у вибраному форматі. Бібліотека також підтримує роботу зі складними функціями, формулами, стилізацією та іншими атрибутами електронних таблиць;

3) розширені можливості. SheetJS надає розширені можливості для роботи з електронними таблицями. Надає можливість створювати нові робочі книги, додавати нові аркуші, об'єднувати комірки, вставляти зображення, застосовувати стилі до даних і багато іншого. Бібліотека також підтримує маніпулювання даними з використанням формату JSON, що дає можливість працювати з даними в зручному для JavaScript форматі;

4) крос-платформеність. SheetJS працює як на клієнтській стороні (в браузері), так і на серверній стороні (з використанням Node.js). Це дозволяє використовувати бібліотеку на різних платформах і в різних середовищах розробки;

5) легкість використання. SheetJS має простий та зрозумілий API, який дозволяє легко інтегрувати її в ваші проекти. Вона надає широкі можливості для маніпулювання даними в електронних таблицях, не вимагаючи складних налаштувань або глибоких знань роботи з Excel.

Загалом, бібліотека SheetJS є потужним інструментом для роботи з електронними таблицями в форматі Excel у JavaScript. Вона надає багато функцій для зчитування, запису та маніпулювання даними, що дозволяє легко працювати з електронними таблицями у вашому проекті. Бібліотека є корисною для розробників, які потребують обробки та аналізу даних з Excel-файлів, а також для створення зручного інтерфейсу для користувачів для взаємодії з електронними таблицями.

2.8 Мережа доставки контенту CDNJS

Content Delivery Network for JavaScript (CDNJS) — це безкоштовна мережа доставки вмісту з відкритим кодом, спеціально розроблена для розміщення та доставки популярних бібліотек і ресурсів JavaScript [33]. Він надає величезну колекцію бібліотек JavaScript, фреймворків CSS, шрифтів та інших веб-ресурсів, які розробники можуть легко включити у свої проекти.

Станом на травень 2021 року він обслуговує 4013 бібліотек JavaScript і CSS, які публічно зберігаються на GitHub. Його включено до мільйонів веб-сайтів, або 12,4% веб-сайтів в Інтернеті, що робить його першим за популярністю CDN для JavaScript.

У січні 2011 року Райан Кіркман і Томас Девіс створили сервіс, запустивши його на GitHub 25 лютого 2011 року. Спочатку він обслуговував контент через Amazon CloudFront. 15 червня 2011 року cdnjs співпрацює з Cloudflare, який надав CDN і субдомен cdnjs.cloudflare.com для проекту. 1 листопада 2019 року засновники передали контроль над cdnjs Cloudflare.

CDNJS пропонує зручний спосіб доступу та використання бібліотек, які часто використовуються, без необхідності їх розміщення на власних серверах. Посилаючись на URL-адреси CDNJS у своїй веб-програмі, можна використовувати переваги глобально розподіленої мережі серверів, щоб ефективніше надавати ці ресурси своїм користувачам.

Розглянемо основні переваги використання CDNJS:

1) покращена продуктивність. CDNJS використовує мережу серверів, розташованих у різних регіонах світу. Коли користувачі заходять на ваш веб-сайт, ресурси CDNJS обслуговуються з найближчого до їхнього розташування сервера, що зменшує затримку та покращує загальний час завантаження сторінки;

2) спрощене налаштування. Замість того, щоб завантажувати та розміщувати файли бібліотеки на власному сервері, ви можете просто посилатися на URL-адреси CDNJS у своєму коді HTML або JavaScript. Це усуває необхідність ручного завантаження, оновлення та підтримки цих бібліотек;

3) контроль версій та оновлення. CDNJS надає різні версії популярних бібліотек JavaScript, що дозволяє легко вказати потрібну версію у вашій програмі. Це допомагає забезпечити сумісність і дозволяє скористатися перевагами нових функцій або виправлення помилок без необхідності вручну оновлювати файли на вашому сервері;

4) керований спільнотою. CDNJS – це проект, керований спільнотою, який спирається на внески розробників і авторів бібліотек. Він має на меті надати вичерпну та оновлену колекцію популярних бібліотек JavaScript, що робить його надійним і надійним ресурсом для розробників.

Таким чином, CDNJS – це широко використовувана мережа доставки вмісту, яка пропонує велику колекцію бібліотек і ресурсів JavaScript. Це спрощує процес включення цих ресурсів у веб-додатки, покращує продуктивність завдяки глобальному розповсюдженню та забезпечує контроль версій і оновлення для розміщених бібліотек.

2.9 Менеджер пакетів NPM

Node Package Manager (NPM) є пакетним менеджером для платформи Node.js [34]. Він дозволяє розробникам легко встановлювати, оновлювати та керувати залежностями (бібліотеками та іншими пакетами) у своїх проектах.

Основні функції та можливості NPM включають:

1) управління пакетами. NPM дозволяє встановлювати пакети з великого репозиторію публічних пакетів npm. Розробники можуть шукати пакети за іменем, переглядати їхні версії, встановлювати певні версії пакетів та автоматично встановлювати їх залежності;

2) локальні та глобальні пакети. NPM дозволяє встановлювати пакети локально (для конкретного проекту) або глобально (доступні для всіх проектів на комп'ютері). Це дозволяє розробникам керувати залежностями на рівні проекту і використовувати спільні пакети між різними проектами;

3) команди для розробки. NPM надає ряд команд, які спрощують процес розробки Node.js-проектів. Це включає команди для створення нового проекту, запуску скриптів, тестування, збирання та публікації пакетів;

4) управління залежностями. Кожен проект має файл package.json, в якому зберігаються метадані проекту, включаючи залежності, скрипти, авторів та іншу інформацію. npm дозволяє керувати цим файлом, додавати, видаляти та оновлювати залежності автоматично;

5) інструменти спільної роботи. NPM надає можливості спільної роботи та публікації пакетів. Розробники можуть публікувати свої пакети в репозиторій NPM, ділитися ними з іншими розробниками, установлювати пакети з приватних репозиторіїв, керувати доступом до пакетів та організовувати спільну роботу над проектами.

Крім того, NPM має широку спільноту розробників та підтримується активною командою, що дозволяє швидко знаходити рішення проблем та отримувати підтримку у разі потреби.

Загалом, NPM є потужним інструментом для керування пакетами та залежностями у проектах, що допомагає розробникам ефективно працювати з екосистемою та використовувати сторонні пакети для розширення функціональності своїх проектів.

Висновки до розділу 2

У цьому розділі було розглянуто та обґрунтовано вибір технологій та інструментальних засобів розробки системи аналізу та прогнозування фінансових часових рядів, застосування яких дозволяє створити потужну та ефективну платформу для розробки, реалізувати потужний набір функціональності, поліпшує продуктивність та допомагає забезпечити ефективну і гнучку розробку.

Для розробки системи аналізу та прогнозування фінансових часових рядів обґрунтовано використання середовища розробки Visual Studio Code, яке є потужним інструментом для розробки вебзастосунків, підтримує обрані для розробки мови програмування, надає зручний інтерфейс, має розширену функціональність, редактор коду та інші корисні інструменти для розробників.

Основною мовою для створення веб-сторінок застосунку системи та структурування контенту було обрано мову гіпертекстової розмітки HTML. Каскадні таблиці стилів CSS використовуються для оформлення та стилізації візуального вигляду веб-сторінок. Для полегшення процесу розробки та підтримки стилей CSS та розширення їх можливостей використано препроцесор SASS, який дозволяє використовувати змінні, вкладені стилі, міксіни та інші функції.

Для реалізації динамічної функціональності на веб-сторінках, створення інтерактивних елементів, взаємодії з користувачем та виконання обробки даних використано мову програмування JavaScript. Застосування транскомпілятора Babel дозволяє використовувати сучасний синтаксис JavaScript та нові функції, які можуть не бути підтримувані в старіших версіях браузерів. Він перетворює сучасний код JavaScript в сумісний з різними браузерами.

Візуалізація даних у вебзастосунку системи аналізу та прогнозування часових рядів реалізована із використанням бібліотеки JavaScript Chart.js.

Бібліотека SheetJS застосовувалася для роботи з вхідним набором даних, що містить часовий ряд: зчитування, запису і маніпулювати даними в електронних таблицях у форматі Excel з використанням файлів XLSX, XLS, CSV та інших

форматів. Для моделювання трендових моделей часового ряду курсу акцій було вирішено використовувати бібліотеку Regression-js, призначену для побудови і аналізу регресійних моделей.

Для збирання та оптимізації коду вебзастосунку використано збірник модулів Webpack – потужний інструмент організації модульної структури коду, що автоматично визначає залежності та оптимізує завантаження ресурсів. Мережа доставки контенту CDNJS дозволяє ефективно доставляти статичні ресурси, такі як бібліотеки JavaScript, до користувачів. Це сприяє прискоренню завантаження сторінок та поліпшує продуктивність.

Передачу даних між клієнтом та сервером доцільно реалізувати з використанням простого у використанні, легко зрозумілого формат обміну даними JSON. Центральним репозиторієм пакетів для проектів JavaScript є менеджер пакетів NPM. Він дозволяє швидко встановлювати, оновлювати та керувати залежностями проєкту.

3 ПРОЄКТУВАННЯ І ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ ДЛЯ АНАЛІЗУ ТА ПРОГНОЗУВАННЯ КУРСУ АКЦІЙ

3.1 Інформаційно-логічна схема побудови прогнозної моделі

На етапі проєктування розроблено інформаційно-логічну схему побудови прогнозних моделей (рис. 3.1). Основна частина функціональності системи полягає в аналізі часових рядів та побудові за його результатами прогнозної моделі.

На початку роботи із системою завантажують набір даних зі значеннями часового ряду курсу акцій. На етапі попередньої обробки даних здійснюється перевірка наявності проблемних даних: пропусків, дублікатів та викидів – аномальних значень. У разу необхідності робиться очищення даних.

Далі відбувається перевірка часового ряду на стаціонарність шляхом проведення автокореляційного аналізу та підтвердження наявності чи відсутності тренду з використанням критерію серій та методу перевірки різниць середніх.

Для стаціонарного часового ряду у системі передбачено побудову AR-моделей (1-го та 2-го порядків). Для нестаціонарного часового ряду виявляють наявність у його структурі інших складових – сезонної та випадкової компонент. Після чого здійснюють виокремлення регулярних компонент та фільтрацію шуму (випадкової компоненти). Здійснивши декомпозицію, передбачено побудову сукупності трендових моделей, дослідження рівня апроксимації та вибір трендової моделі, яка найкраще апроксимує тренд часового ряду.

Для перевірки адекватності прогнозної моделі часового ряду здійснюється дослідження випадкової компоненти та розрахунок метрик оцінки якості та точності побудованої моделі.

Після чого побудована прогнозна модель готова до її застосування з метою здійснення прогнозу курсу акцій, який можна враховувати при плануванні інвестицій на фондовому ринку.

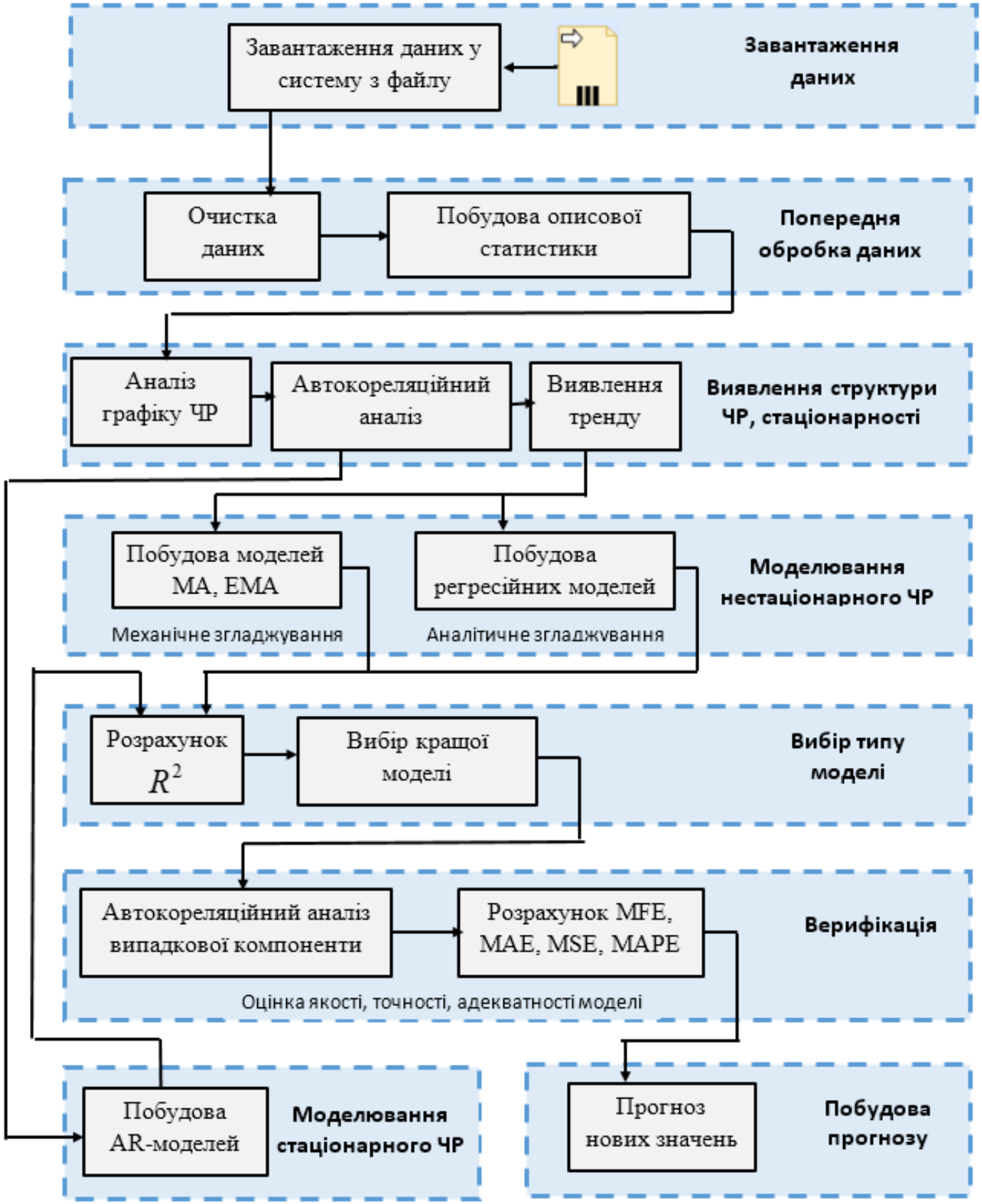

Рисунок 3.1 – Інформаційно-логічна схема побудови прогнозних моделей

3.2 Програмна розробка системи

Програмна реалізація здійснювалася у середовищі розробки Visual Studio Code з використанням мови програмування JavaScript. Основним класом системи є клас timeSeries, який надає широкий функціонал для аналізу та прогнозування фінансових часових рядів. Загальна структура класу зображена на рис. 3.2.

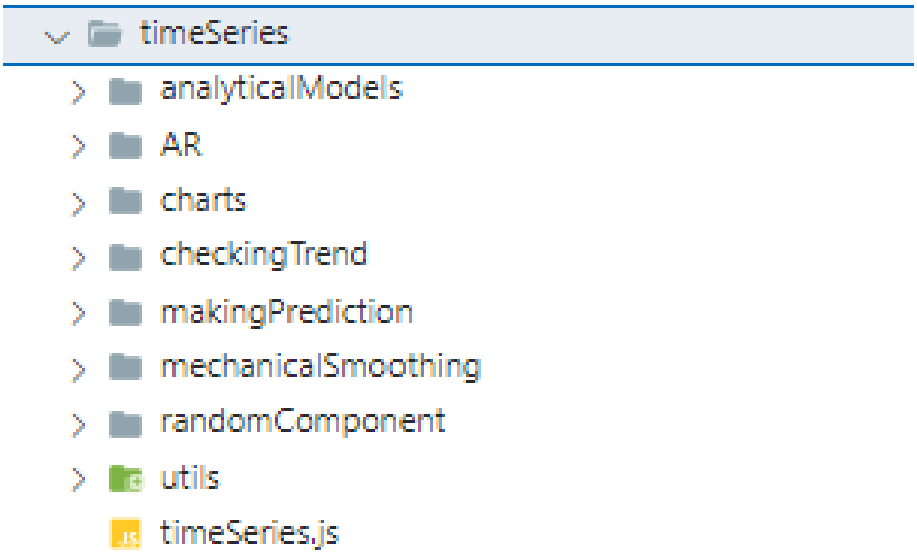

Рисунок 3.2 – Загальна структура класу timeSeries

Клас timeSeries здійснює аналіз та прогнозування цін акцій на фондових ринках та дозволяє проводити згладжування, виокремлення сезонної компоненти, моделювання тренду, прогнозування і візуалізацію даних.

Весь функціонал реалізовано у різних файлах, тому всі функції імпортуються у цей клас, як зображено на рис. 3.3.

Основні функції та їх призначення описано у таблиці 3.1. Розглянемо їх більш детально.

timeSeries.js

projectV2 > timeSeries > timeSeries.js > timeSeries > constructor

```
"use strict";

import calculateUnbiasedEstimate from "./utils/unbiasedEstimate.js";
import calculateBiasedEstimate from "./utils/biasedEstimate.js";
import calculateMathematicalExpectation from "./utils/mathematicalExpectation.js";
import calculateOrderAutocovariance from "./utils/orderAutocovariance.js";
import calculateAutocorrelationCoefficient from "./utils/autocorrelationCoefficient.js";
import detectAbnormalValues from "./utils/abnormalValues.js";
import checkTrend from "./checkingTrend/trend.js";
import analyzeCorelogram from "./utils/analyzeCorelogram.js";
import { initializeProperties, fillProperties } from "./utils/initProperies.js";
import calculateLinearModel from "./analyticalModels/linearModel.js";
import calculateExponentialModel from "./analyticalModels/exponentialModel.js";
import calculateLogarithmicModel from "./analyticalModels/logarithmicModel.js";
import calculatePowerModel from "./analyticalModels/powerModel.js";
import calculatePolynomialModel from "./analyticalModels/polynomialModel.js";
import {
  createDataChart,
  createDataCorrelogram,
  createAnalyticalModelsCharts,
  createMechanicalModelsCharts,
  createTrendsComparasionChart,
  createRandomComponentChart,
  createRandomComponentCorrelogram,
  createPredictionChart,
} from "./charts/charts.js";
import { calculateAR } from "./AR/ar.js";
import mechanicalSmoothing from "./mechanicalSmoothing/mechanicalSmoothing.js";
import compareMechanicalSmoothings from "./mechanicalSmoothing/comparasionSmoothings.js";
import compareAnalyticalModels from "./analyticalModels/comparasionAnalyticalModels.js";
import calculateRandomComponent from "./randomComponent/randomComponent.js";
import predict from "./makingPrediction/prediction.js";
```

Рисунок 3.3 – Імпорт файлів із основним функціоналом системи у клас timeSeries

Таблиця 3.1 – Назва функцій та їх призначення

Назва функції	**Призначення**
calculateUnbiasedEstimate	Розрахунок зміщеної оцінки дисперсії
calculateBiasedEstimate	Розрахунок незміщеної оцінки дисперсії
calculateMathematicalExpectation	Розрахунок середнього арифметичного
calculateOrderAutocovariance	Розрахунок значень порядку автоковаріації

Закінчення таблиці 3.1

calculateAutocorrelationCoefficient	Розрахунок значень коефіцієнтів автокореляції
detectAbnormalValues	Пошук аномальних значень
checkTrend	Перевірка наявності тренду
analyzeCorelogram	Аналіз корелограми
initalizeProperties	Ініціалізація початкових значень
fillProperties	Заповнення старими значеннями
calculateLinearModel	Розрахунок лінійної моделі тренду
createDataChart	Створення графіку часового ряду
createDataCorrelogram	Створення корелограми часового ряду
createAnalyticalModelsCharts	Створення графіків аналітичних моделей тренду
createTrendsComparasionChart	Створення графіку порівняння трендових моделей
createRandomComponentCorrelogram	Створення корелограми випадкової компоненти
createPredictionChart	Створення графіку прогнозу
calculateAR	Розрахунок коефіцієнтів AR моделей
mechanicalSmoothing	Механічне згладжування часового ряду
compareAnalyticalModels	Порівняння аналітичних моделей тренду
calculateRandomComponent	Розрахунок значень випадкової компоненти
predict	Прогнозування

Функція handleFileUpload здійснює завантаження набору даних та його обробку (рис. 3.4). Спочатку здійснюється визначення, який файл було обрано, а потім зчитування його з формату excel файлів в формат JSON, використовуючи бібліотеку SheetJS. SheetJS XLSX дозволяє читати дані з Excel файлів, записувати дані в Excel файли, а також виконувати різноманітні маніпуляції з електронними таблицями, такі як додавання/видалення стовпців та рядків, зміна значень комірок, об'єднання комірок, форматування таблиць тощо. Після зчитування створюється об’єкт класу timeSeries та передаються дані про часовий ряд.

```
function handleFileUpload(event) {
  const file = event.target.files[0];
  const reader = new FileReader();

  reader.onload = (e) => {
    const data = e.target.result;
    const workbook = XLSX.read(data, {
      type: "binary",
      cellDates: true,
      dateNF: "yyyy-mm-dd",
    });

    const sheetName = workbook.SheetNames[0];
    const worksheet = workbook.Sheets[sheetName];

    const stockPrices = XLSX.utils.sheet_to_json(worksheet, {
      raw: false,
    });

    msft = new timeSeries(stockPrices);
    createTable(msft, table);

    event.target.value = "";
  };

  reader.readAsBinaryString(file);
}
```

Рисунок 3.4 – Функція handleFileUpload

Для очищення даних створено дві функції clearEmptyDates та clearEmptyPrices для виявлення та редагування пропусків дат та цін (рис. 3.5). Пропуски заміняються значеннями, які розраховують як середнє значення рівнів ряду (дат), суміжних з ними.

```
function clearData() {
  clearEmptyDates();
  clearEmptyPrices();
}

function clearEmptyDates() {
  let dates = msft.dates;

  for (let i = 0; i < dates.length - 1; i++) {
    let currentDate = new Date(dates[i]);
    let nextDate = new Date(dates[i + 1]);
    if (nextDate.getMonth() - currentDate.getMonth() == -1) {
      areEmptyValuesPresent = true;
      nextDate = new Date(currentDate);
      nextDate.setMonth(currentDate.getMonth() + 1);
      dates[i + 1] = nextDate.toISOString().slice(0, 10);
    }
  }

  msft.dates = dates;
}

function clearEmptyPrices() {
  let prices = msft.stockPrices;

  for (let i = 0; i < prices.length; i++) {
    if (Object.is(prices[i], null) || isNaN(prices[i])) {
      areEmptyValuesPresent = true;
      prices[i] = (prices[i - 1] + prices[i + 1]) / 2;
    }
  }

  msft.stockPrices = prices;
}
```

Рисунок 3.5 – Функції clearEmptyDates та clearEmptyPrices

Метод класу preliminaryAnalysis здійснює: розрахунок статистичних характеристик часового ряду, коефіцієнтів автоковаріації та автокореляції; виявлення аномальних значень; перевірку наявності тренду; побудову та аналіз корелограми (рис. 3.6).

Функція calculateMathematicalExpectation здійснює розрахунок середнього арифметичного (рис. 3.7). Ця функція перевантажена використовуючи логічні конструкції if else та розраховує середнє арифметичне для різних випадків, що зустрічаються в програмі.

```
preliminaryAnalysis() {
  this.mathematicalExpectation = calculateMathematicalExpectation(
    this,
    this.stockPrices
  );
  calculateBiasedEstimate(this);
  calculateUnbiasedEstimate(
    this,
    this.stockPrices,
    this.mathematicalExpectation
  );
  calculateOrderAutocovariance(this);
  calculateAutocorrelationCoefficient(this);

  detectAbnormalValues(this);
  checkTrend(this);
  analyzeCorelogram(this, this.autocorrelationCoefficient);
}
```

Рисунок 3.6 – Функція preliminaryAnalysis яка робить попередній аналіз даних

```
export default function calculateMathematicalExpectation(
  timeSeries,
  arr,
  isDoubleArr = false,
  timeLag = 0
) {
  let sum = 0;
  let mathExpect = null;
  if (timeLag === 0 && isDoubleArr === false) {
    sum = 0;
    mathExpect = 0;

    arr.forEach((price) => (sum += price));
    mathExpect = (1 / timeSeries.numberOfTimeSeries) * sum;

    return mathExpect;
```

Рисунок 3.7 – Функція calculateMathematicalExpectation

Функція calculateBiasedEstimate здійснює розрахунок зміщеної оцінки дисперсії (рис. 3.8). Для знаходження незміщеної оцінки використовується схожа функція.

```
export default function calculateBiasedEstimate(timeSeries) {
  let sum = 0;

  timeSeries.stockPrices.forEach((timeRow) => {
    sum += Math.pow(timeRow - timeSeries.mathematicalExpectation, 2);
  });

  timeSeries.dispersion.biasedEstimate =
    (1 / timeSeries.numberOfTimeSeries) * sum;
}
```

Рисунок 3.8 – Функція calculateBiasedEstimate

Функція calculateOrderAutocovariance знаходить значення порядку автоковаріації (рис. 3.9). Ця функція перевантажена і в залежності від розміру часового ряду, обирає як розраховувати порядок автоковаріації.

```
export default function calculateOrderAutocovariance(timeSeries) {
  if (timeSeries.numberOfTimeSeries > 99) {…
  } else if (timeSeries.numberOfTimeSeries < 100) {
    timeSeries.orderAutocovariance.push(timeSeries.dispersion.biasedEstimate);
    let sum = 0;
    let timeLag = 1;

    while (timeSeries.numberOfTimeSeries - timeLag != 1) {
      calculateMathematicalExpectation(
        timeSeries,
        timeSeries.stockPrices,
        true,
        timeLag
      );

      for (let i = 0; i < timeSeries.numberOfTimeSeries - timeLag; i++) {
        sum +=
          (timeSeries.stockPrices[i] - timeSeries.mathematicalExpectationY1) *
          (timeSeries.stockPrices[i + timeLag] -
            timeSeries.mathematicalExpectationY2);
      }
      timeSeries.orderAutocovariance.push(
        (1 / (timeSeries.numberOfTimeSeries - timeLag)) * sum
      );

      sum = 0;
      timeLag++;
    }
  }
}
```

Рисунок 3.9 – Функція calculateOrderAutocovariance

Функція calculateAutocorrelationCoefficient здійснює розрахунок коефіцієнтів автокореляції (рис. 3.10). Вона також перевантажена і розраховує значення для різних випадків, як для самого часового ряд так і для випадкової компоненти.

```
export default function calculateAutocorrelationCoefficient(
  timeSeries,
  values = [],
  mathExpect = 0
) {
  if (
    timeSeries.numberOfTimeSeries > 99 &&
    values.length == 0 &&
    mathExpect == 0
  ) {…
  } else if (timeSeries.numberOfTimeSeries > 99 && values.length != 0) {…
  } else if (timeSeries.numberOfTimeSeries < 100 && values.length == 0) {
    let numerator = 0;
    let denominator = 0;
    let timeLag = 1;
    while (timeSeries.numberOfTimeSeries - timeLag >= 1) {
      console.log(
        timeSeries.numberOfTimeSeries - timeSeries.numberOfTimeSeries + 1
      );
      for (let i = 0; i < timeSeries.numberOfTimeSeries - timeLag; i++) {
        numerator +=
          (timeSeries.stockPrices[i] - timeSeries.mathematicalExpectation) *
          (timeSeries.stockPrices[i + timeLag] -
            timeSeries.mathematicalExpectation);
      }
      for (let i = 0; i < timeSeries.numberOfTimeSeries; i++) {
        denominator += Math.pow(
          timeSeries.stockPrices[i] - timeSeries.mathematicalExpectation,
          2
        );
      }

      timeSeries.autocorrelationCoefficient.push(numerator / denominator);
      timeLag++;
    }
  } else if (timeSeries.numberOfTimeSeries < 100 && values.length != 0) {…
  }
}
```

Рисунок 3.10 – Функція calculateAutocorrelationCoefficient для знаходження коефіцієнтів автокореляції

Функція detectAbnormalValues виявляє аномальні значення використовуючи метод Ірвіна за формулою 1.5(рис. 3.11).

```
export default function detectAbnormalValues(timeSeries) {
  let res = [];
  let hasAbnormalValues = false;
  let counter = 0;

  for (let i = 0; i < timeSeries.numberOfTimeSeries - 1; i++) {
    res.push(
      Math.abs(
        (timeSeries.stockPrices[i + 1] - timeSeries.stockPrices[i]) /
          Math.sqrt(timeSeries.dispersion.unbiasedEstimate)
      )
    );
  }

  for (let i = 0; i < res.length; i++) {
    if (res[i] > 1.2) {
      timeSeries.hasAbnormalValues = true;
      counter++;
      if (counter == 2) {
        timeSeries.stockPrices[i] =
          (timeSeries.stockPrices[i - 1] + timeSeries.stockPrices[i + 1]) / 2;
        counter = 0;
      }
      console.log(`Abnormal value i=${i} : ${res[i]}`);
      hasAbnormalValues = true;
    } else if (i === res.length - 1 && hasAbnormalValues === false) {
      console.log(`There are no abnormal values!`);
    }
  }
}
```

Рисунок 3.11 – Функція для виявлення аномальних значень

Функція analyzeCorelogram здійснює аналіз корелограми часового ряду: перевіряє наявність тренду, стаціонарності та наявність сезонної компоненти використовуючи коефіцієнти автокореляції, вміст функції наведено на рисунку 3.12.

Для достовірного підтвердження наявності чи відсутності тренду використовуються функції seriesCriterion, яка реалізує критерій серій, та checkingAverageLevelDifferences, яка реалізує метод перевірки різниць середніх рівнів (рис. 3.13, рис. 3.14).

```
export default function analyzeCorelogram(timeSeries, autoCorelCoefValues) {
  let highestValueIndex = 0;
  let isStationary = true;
  for (let i = 0; i < autoCorelCoefValues.length / 3; i++) {
    if (
      Math.abs(autoCorelCoefValues[highestValueIndex]) <
      Math.abs(autoCorelCoefValues[i])
    ) {
      highestValueIndex = i;
    }
  }

  for (let i = 0; i < autoCorelCoefValues.length / 3; i++) {
    if (
      autoCorelCoefValues[i] < timeSeries.confidenceInterval[0] ||
      autoCorelCoefValues[i] > timeSeries.confidenceInterval[1]
    ) {
      isStationary = false;
    }
  }

  if (highestValueIndex == 0) {
    timeSeries.hasSeasonalComponent = false;
  } else if (isStationary) {
    timeSeries.isStationary = true;
  } else if (highestValueIndex != 0) {
    timeSeries.hasSeasonalComponent = true;
  }
}
```

Рисунок 3.12 – Функція кореляційного аналізу часового ряду

```
export default function seriesCriterion(timeSeries) {
  let median = calculateMedian(timeSeries.stockPrices);
  console.log(`median = ${median}`);

  let sequence = formSequence(timeSeries.stockPrices, median);
  console.log(`sequence: ${sequence}`);

  let series = [];
  let numberSeries = 0;
  let largestSeries = 0;
  [series, numberSeries, largestSeries] = calculateNumberSeries(sequence);

  let criticalValueNumberSeries = calculateCriticalValueOfNumberSeries(
    sequence.length
  );

  let criticalValueLargestSeries =
    calculateCriticalValueOfLargestSeries(largestSeries);

  if (
    checkingInequalities(
      timeSeries,
      largestSeries,
      criticalValueLargestSeries,
      numberSeries,
      criticalValueNumberSeries
    )
  ) {
    return true;
  } else {
    return false;
  }
}
```

Рисунок 3.13 – Функція виявлення тренду за критерієм серій

```
  [mathExpectY1, mathExpectY2] = calculateMathematicalExpectation(
    timeSeries,
    twoArrays,
    true
  );

  [unbiasedEstimateN1, unbiasedEstimateN2] = calculateUnbiasedEstimate(
    timeSeries,
    twoArrays,
    [mathExpectY1, mathExpectY2],
    true
  );

  empiricalValueF = calculateEmpiricalFisherValue(
    unbiasedEstimateN1,
    unbiasedEstimateN2
  );

  criticalValueF = findFisherCriticalValue(firstArraySize, secondArraySize);

  canFindTrend = compareEmpiricalAndCriticalFisherValue(
    empiricalValueF,
    criticalValueF
  );

  if (canFindTrend === true) {
    empiricalValueS = calculateEmpiricalStudentValue(
      [firstArraySize, secondArraySize],
      [unbiasedEstimateN1, unbiasedEstimateN2],
      [mathExpectY1, mathExpectY2]
    );

    criticalValueS = findStudentCriticalValue(firstArraySize + secondArraySize);

    timeSeries.hasTrend = compareEmpiricalAndCriticalStudentValue(
      empiricalValueS,
      criticalValueS
    );
    if (timeSeries.hasTrend) {
      return true;
    }
  }

  timeSeries.hasTrend = false;
  return false;
}
```

Рисунок 3.14 – Функція виявлення тренду за методом перевірки різниць середніх рівнів

Функція mechanicalSmoothing здійснює механічне згладжування: згладжування простого ковзного середнього та експоненційного згладжування (рис. 3.15).

```
export default function mechanicalSmoothing(
  timeSeries,
  interval,
  data = timeSeries.stockPrices
) {
  let sma = timeSeries.mechanicalSmoothing.simpleMovingAverage;
  let ema = timeSeries.mechanicalSmoothing.exponentialMovingAverage;

  sma.values = simpleMovingAverage(data, interval);
  sma.predict = (timeSeries, period) =>
    predictSimpleMovingAverageValues(timeSeries, period);

  ema.values = exponentialSmoothing(data, interval);
  ema.predict = (timeSeries, period) =>
    predictExponentialValues(timeSeries, period);

  removeEmptyValues(timeSeries, sma.values, interval);
  removeEmptyValues(timeSeries, ema.values, interval);

  sma.determinationCoefficient = round(
    calculateDeterminationCoefficient(data, sma.values),
    3
  );
  ema.determinationCoefficient = round(
    calculateDeterminationCoefficient(data, ema.values),
    3
  );
}
```

Рисунок 3.15 – Функція mechanicalSmoothing

Функція compareMechanicalSmoothings здійснює аналіз рівня апроксимації моделей механічного згладжування, розраховуючи коефіцієнт детермінації за формулою 1.25 (рис. 3.16).

```
export default function compareMechanicalSmoothings(
  timeSeries,
  isFindingTrend = false
) {
  const [sma, ema] = [
    timeSeries.mechanicalSmoothing.simpleMovingAverage.determinationCoefficient,
    timeSeries.mechanicalSmoothing.exponentialMovingAverage
      .determinationCoefficient,
  ];

  const result =
    sma >= ema
      ? {
          model: "Просте ковзнє середнє",
          parameters: timeSeries.mechanicalSmoothing.simpleMovingAverage,
        }
      : {
          model: "Експоненційне згладжування",
          parameters: timeSeries.mechanicalSmoothing.exponentialMovingAverage,
        };

  if (isFindingTrend) {
    timeSeries.trend.mechanical = result;
  } else {
    timeSeries.smoothedInputData.mechanical = result;
  }
}
```

Рисунок 3.16 – Функція для порівняння моделей механічного згладжування

Функція calculateSeasonalComponent реалізує відокремлення сезонної компоненти часового ряду (рис. 3.17).

```
calculateSeasonalComponent() {
  let seasonalComponent = [];
  for (let i = 0; i < this.numberOfTimeSeries; i++) {
    if (this.hasSeasonalComponent) {
      seasonalComponent.push(
        this.stockPrices[i] - this.trend.mechanical.parameters.values[i]
      );
    }
  }
  this.seasonalComponent.values = seasonalComponent;
}
```

Рисунок 3.17 – Розрахунок значень сезонної компоненти

Для розрахунку коефіцієнтів аналітичних трендових моделей використовується бібліотека regression-js, яка підключається на сторінку за допомогою CDNJS (рис. 3.18).

```
<script
  src="https://cdnjs.cloudflare.com/ajax/libs/underscore.js/1.13.6/underscore-min.js"
  integrity="sha512-2V49R8ndaagCOnwmj8QnbT1Gz/rie17UouD9Re5WxbzRVUGoftCu5IuqqtAM9+UC3fwfHCSJR1hkzNQh/2wdtg=="
  crossorigin="anonymous"
  referrerpolicy="no-referrer"
></script>
```

Рисунок 3.18 – Підключення бібліотеки regression-js через CDNJS

Загалом передбачено побудову наступних моделей: лінійної, експоненційної, логарифмічної, степеневої та поліноміальної. Для виявлення найкращої аналітичної трендової моделі здійснюється розрахунок коефіцієнта детермінації за формулою 1.25.

Для побудови графіків у системі передбачено використання бібліотеки Chart.js. Вона також підключається через CDNJS. Для спрощення побудови графіків було створено спеціальні функції: createDataset, createChart та createCorrelogram. Вони оптимізовані під будь-який набір даних та налаштування.

Функція createDataset створює набір даних, який може прийняти графік, зазвичай це масив даних зі значеннями по віссю x та y. При створені набору даних

для графіку, також вказуються опції, які використовуються для візуального налаштування (рис. 3.19).

```
function createDataset(label, data, options) {
  let dataset = {
    label: label,
    data: data,
    options: options,
  };
  return dataset;
}
```

Рисунок 3.19 – Функція створення набору даних для побудови графіку

Функції createChart та createCorrelogram здійснюють побудову графіків та корелограми часового ряду, тренду та інших компонент (рис. 3.20, рис. 3.21). В програмі побудова кожного графіку реалізується через виклик відповідної функції, у якій використовуються раніше створені функції createDataset, createChart.

```
function createChart(chartId, title, labels, datasets) {
  Chart.defaults.color = "#0a0a0a";
  Chart.defaults.font.color = "black";
  Chart.defaults.font.family = "Roboto";
  let chart = document.getElementById(chartId);
  new Chart(chart, {
    type: "line",
    data: {
      labels: labels,
      datasets: datasets.map((dataset) => {
        return {
          label: dataset.label,
          data: dataset.data,
          ...dataset.options,
        };
      }),
    },
    options: {…
    },
  });
}
```

Рисунок 3.20 – Функція createChart

```
function createCorrelogram(chartId, title, labels, datasets) {
  Chart.defaults.color = "#0a0a0a";
  Chart.defaults.font.color = "black";
  Chart.defaults.font.family = "Roboto";
  let chart = document.getElementById(chartId);
  new Chart(chart, {
    type: "bar",
    data: {
      labels: labels,
      datasets: datasets.map((dataset) => {
        return {
          label: dataset.label,
          data: dataset.data,
          ...dataset.options,
        };
      }),
    },
    options: {…
    },
  });
}
```

Рисунок 3.21 – Функція createCorrelogram

Функція calculateRandomComponent здійснює розрахунок значень випадкової компоненти (див. рис. 3.22).

```
export default function calculateRandomComponent(
  timeSeries,
  smoothedValues = null
) {
  let randomComponent = [];
  let hasAbnormalValues = false;

  for (let i = 0; i < timeSeries.numberOfTimeSeries; i++) {
    let stockPrice = timeSeries.stockPrices[i];
    let smoothValue = smoothedValues ? smoothedValues[i] : null;
    let seasonalComp = timeSeries.seasonalComponent.values[i] || 0;
    let trendComp =
      timeSeries.trend.analytical.parameters.determinationCoefficient <
      timeSeries.trend.mechanical.parameters.determinationCoefficient
        ? timeSeries.trend.mechanical.parameters.values[i]
        : timeSeries.trend.analytical.parameters.values[i];

    if (!timeSeries.hasSeasonalComponent && smoothedValues == null) {
      randomComponent.push(stockPrice - trendComp);
    } else if (!timeSeries.hasSeasonalComponent && smoothValue !== null) {
      randomComponent.push(smoothValue - trendComp);
    } else if (timeSeries.hasSeasonalComponent && smoothedValues == null) {
      randomComponent.push(stockPrice - seasonalComp - trendComp);
    } else if (timeSeries.hasSeasonalComponent && smoothValue !== null) {
      randomComponent.push(smoothValue - seasonalComp - trendComp);
    }
  }

  timeSeries.randomComponent.values = randomComponent;
```

Рисунок 3.22 – Функція calculateRandomComponent

Функція predict здійснює прогнозування майбутніх значень курсу акцій за побудованою прогнозною моделлю часового ряду (рис. 3.23).

```
export default function predict(timeSeries, number) {
  if (
    timeSeries.trend.analytical.parameters.determinationCoefficient <
    timeSeries.trend.mechanical.parameters.determinationCoefficient
  ) {
    timeSeries.trend.mechanical.parameters.predict(timeSeries, number);
  } else {
    for (let i = 1; i < number + 1; i++) {
      timeSeries.predictedFuturePrices.analytical.push(
        timeSeries.trend.analytical.parameters.predict(
          timeSeries.numberOfTimeSeries + i
        )[1]
      );
    }
  }

  let predictedFuturePrices =
    timeSeries.predictedFuturePrices.analytical.length == 0
      ? timeSeries.predictedFuturePrices.mechanical
      : timeSeries.predictedFuturePrices.analytical;

  calculateForecastAccuracy(timeSeries, number, predictedFuturePrices);
}
```

Рисунок 3.23 – Функція для здійснення прогнозу

Для оцінки якості прогнозу за побудованою прогнозною моделлю часового ряду використовується ряд метрик (формули 1.26 – 1.30).

3.3 Збірка проєкту

Вебзастосунок системи для аналізу та прогнозування фінансового часового ряду використовує п’ять сторінок та п’ять js-файлів для кожної сторінки (рис. 3.24). Використовуючи SASS всі прописані стилі зберігаються в scss-файлі, цей файл потім компілюється в звичайний css-файл, який використовують усі сторінки.

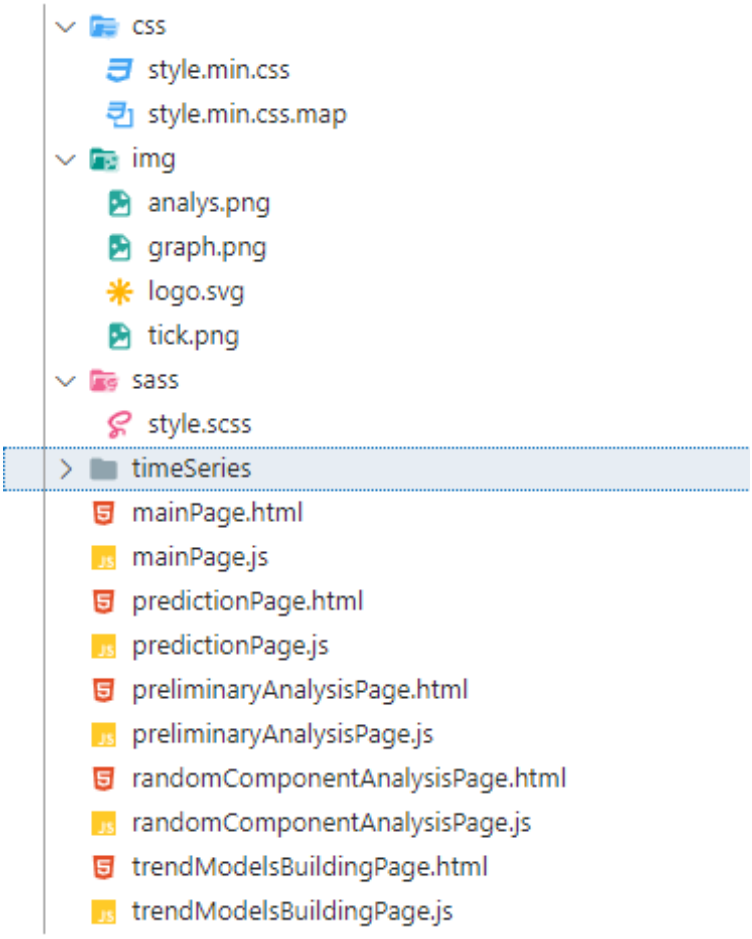

Рисунок 3.24 структура вебзастосунку

Всі бібліотеки неможливо завантажити через CDNJS, деякі з них знаходяться в NPM, тому вони були завантажені за допомогою NPM. Для того щоб NPM працював, його було налаштовано та завантажено (рис. 3.25).

Деякі бібліотеки мають або старий синтаксис або синтаксис, який не підтримується браузерами. Для вирішення цієї проблеми використовується збирач модулів webpack та babel транскомпілятор (рис. 3.25).

Babel переводить старий синтаксис у новий, а webpack підключає всі модулі таким чином, щоб модулі старого і нового синтаксису не конфліктували між собою і працювали в браузері.

```json
{
  "name": "diplom",
  "version": "3.0.0",
  "description": "Diploma",
  "main": "scipt.js",
  ▷ Отладка
  "scripts": {
    "build": "./node_modules/.bin/webpack",
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "Strizhak Roman",
  "license": "ISC",
  "dependencies": {
    "chart.js": "^4.1.1",
    "csv-parser": "^3.0.0",
    "regression": "^2.0.1",
    "timeseries-analysis": "^1.0.12",
    "xlsx": "^0.18.5"
  },
  "devDependencies": {
    "@babel/cli": "^7.21.5",
    "@babel/core": "^7.21.8",
    "@babel/preset-env": "^7.21.5",
    "babel-loader": "^9.1.2",
    "babel-preset-env": "^1.7.0",
    "babelify": "^10.0.0",
    "webpack": "^5.82.1",
    "webpack-cli": "^5.1.1"
  }
}
```

Рисунок 3.25 – Налаштування NPM

```js
const path = require("path");

module.exports = {
  mode: "development",
  // entry: "./projectV2/dataUploadPage.js",
  entry: {
    mainPage: "./projectV2/mainPage.js",
    predictionPage: "./projectV2/predictionPage.js",
    preliminaryAnalysisPage: "./projectV2/preliminaryAnalysisPage.js",
    randomComponentAnalysisPage: "./projectV2/randomComponentAnalysisPage.js",
    trendModelsBuildingPage: "./projectV2/trendModelsBuildingPage.js",
    timeSeries: "./projectV2/timeSeries/timeSeries.js",
  },
  output: {
    filename: "[name].bundle.js",
    path: path.resolve(__dirname, "dist"),
  },
  module: {
    rules: [
      {
        test: /\.js$/,
        exclude: /node_modules/,
        use: {
          loader: "babel-loader",
          options: {
            presets: ["@babel/preset-env"],
          },
        },
      },
    ],
  },
  resolve: {
    extensions: [".js"],
    alias: {
      "@": path.resolve(__dirname, "dist"),
    },
  },
};
```

Рисунок 3.26 – Налаштування webpack та babel

Файли які використовуються при збірці, збираються в нові оптимізовані js-файли в окремій папці (рис. 3.27). Саме ці файли підключаються до сторінок, щоб система працювала корректно в браузері.

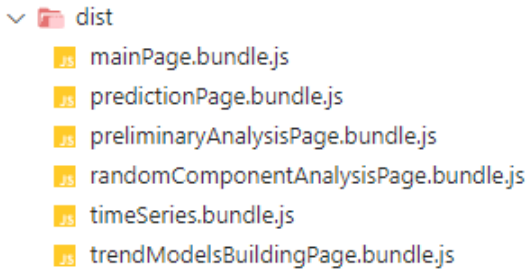

Рисунок 3.27 – Зібрані модулі, які підключаються до сторінок

3.4 Інтерфейс системи. Аналіз та прогнозування курсу акцій

Початкова сторінка розробленої системи аналізу та прогнозування зображена на рисунку 3.28.

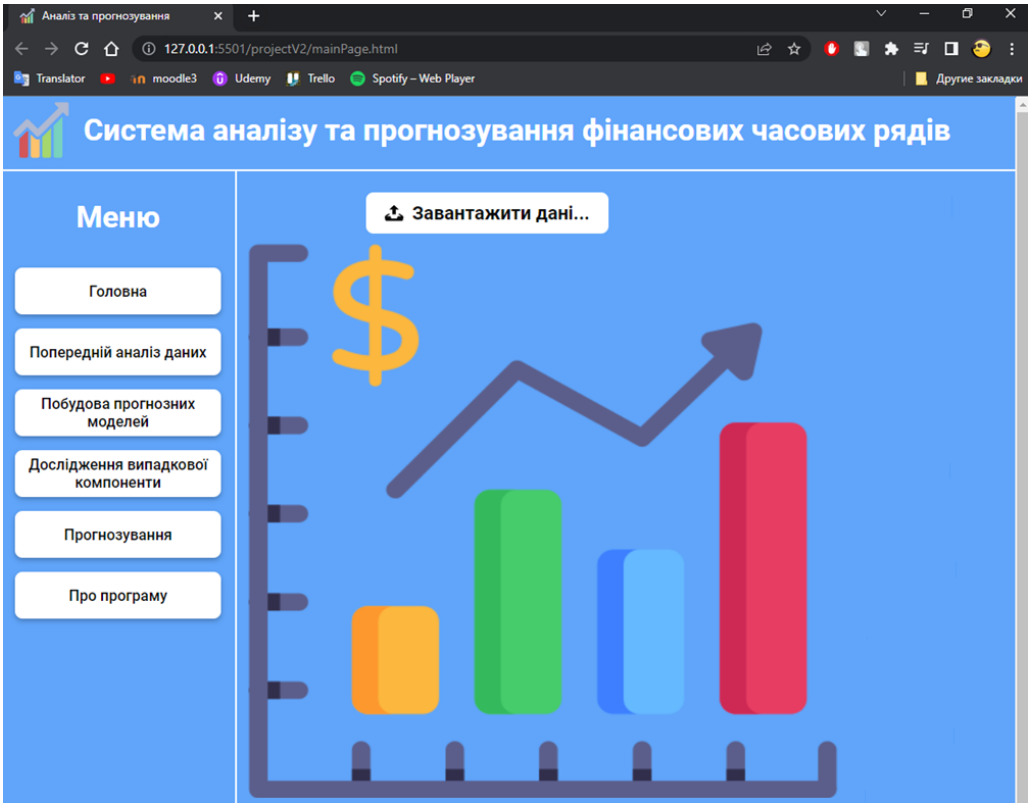

Рисунок 3.28 – Початкова сторінка системи

Система автоматизовано здійснює побудову прогнозної моделі часового ряду курсу акцій. Зліва розміщена панель із опціями для перегляду інформації кожного етапу аналізу часового ряду та побудови прогнозної моделі.

Спочатку потрібно завантажити дані, натиснувши на кнопку Завантажити дані у центральній частині вікна та обрати для цього необхідний файл формату Microsoft Excel. Доступні формати можна побачити у області, яка з’явиться у центральній частині вікна системи (рис. 3.29).

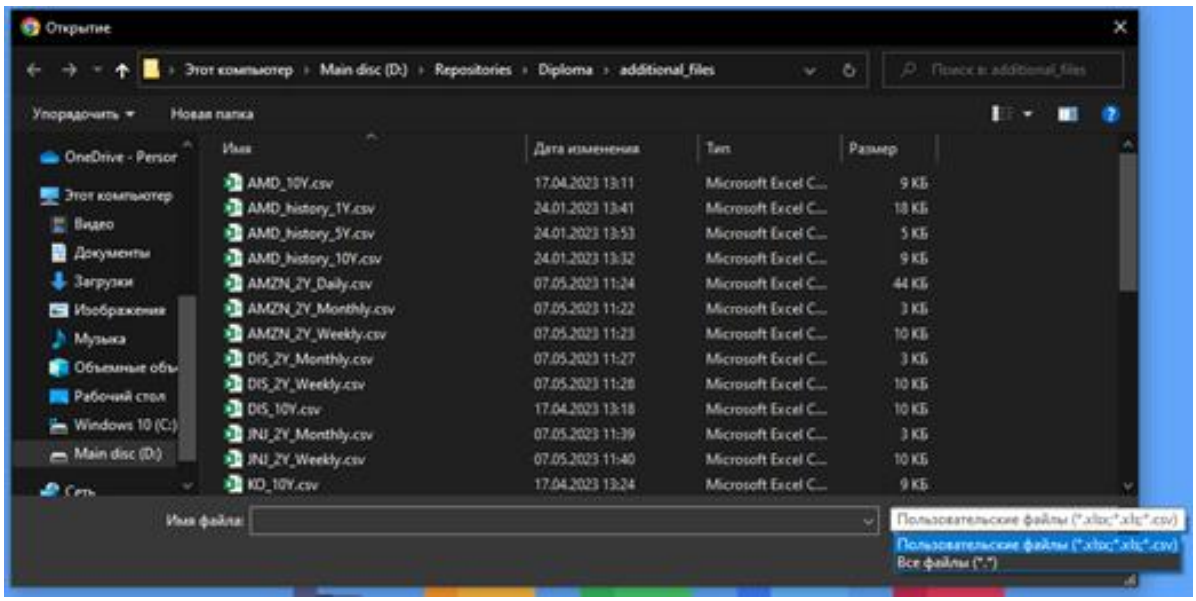

Рисунок 3.29 – Доступні формати файлів для завантаження даних

Після завантаження даних у вікні застосунку з’являється таблиця з даними часового ряду, який необхідно аналізувати (рис. 3.30).

Для перегляду результатів попереднього аналізу даних у лівій панелі вікна необхідно обрати опцію Попередній аналіз даних. У центральній частині вікна з’явиться область із результатами проведеного аналізу: очищення даних, розрахованими характеристиками описової статистики (рис. 3.31), графіком часового ряду, проведеним кореляційним аналізом та розрахованими критеріями підтвердження гіпотези наявності тренду за методом середніх різниць та критерієм серій.

Рисунок 3.30 – Завантажені дані

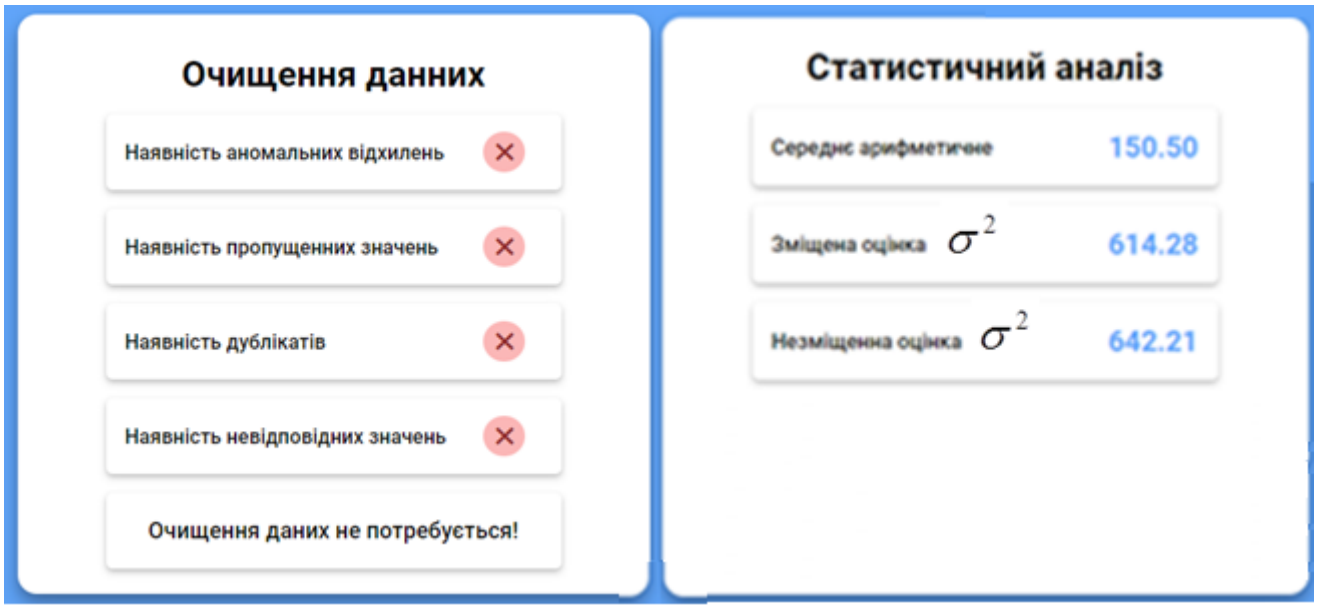

Рисунок 3.31 – Результати перевірки наявності проблемних значень та розрахунку статистичних характеристик

Далі розглянемо аналіз та побудову прогнозної моделі на прикладі даних із цінами акцій компанії Microsoft помісячно з 1.01.2020 року по 1.10.2021 року. DataSet було взято з сайту https://finance.yahoo.com/quote/MSFT/history?p=MSFT. На етапі попереднього аналізу було виявлено проблемні дані, здійснено їх очищення та розраховано статистичні характеристики (рис. 3.32).

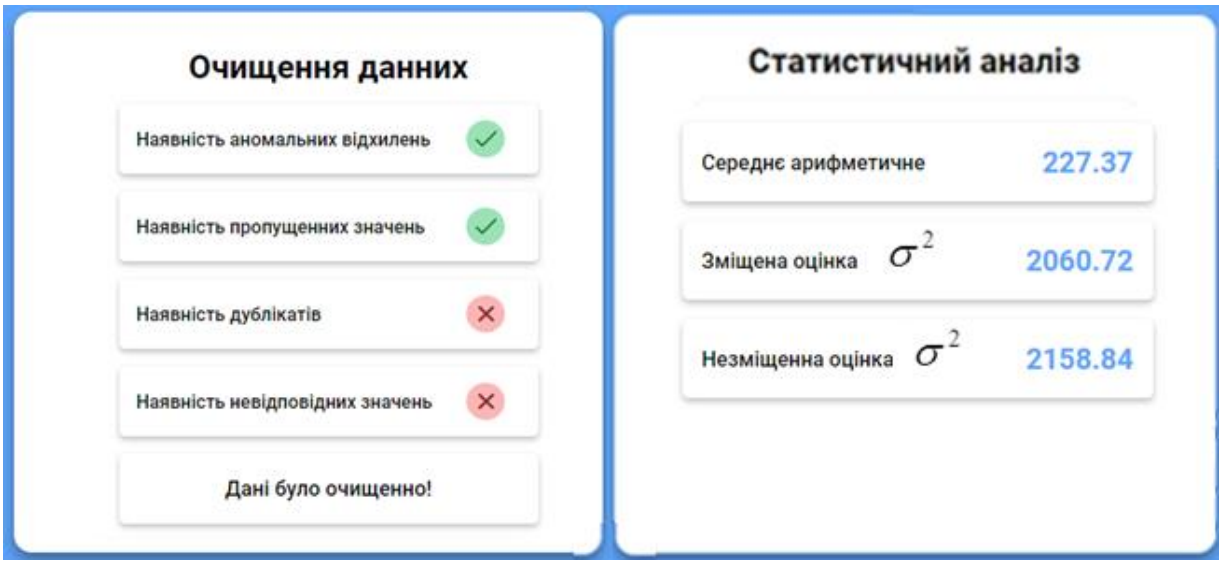

Рисунок 3.32 – Результат очищення даних

На рисунку 3.33 зображено побудований графік часового ряду. Візуальний аналіз графіку на цьому рисунку дозволяє стверджувати, що ряд не має сезонних коливать і має тренд.

Рисунок 3.33 – Графік часового ряду

Побудована корелограма часового ряду зображена на риснуку 3.34. Її аналіз дає підстави стверджувати, що даний ряд не є стаціонарним. Для стаціонарного часового ряду характерним є чергування затухаючих додатних та від'ємних

статистично незначущих значень коефіцієнтів автокореляції. На рисунку бачимо, що частина коефіцієнтів автокореляції виходить за межі довірчого інтервалу, тому аналіз корелограми дає підстави стверджувати, що даний ряд не є стаціонарним.

Коефіцієнт автокореляції r_p із ймовірністю 95% є значимим, якщо виходить за межі інтервалу (відміченого на рисунку червоними пунктирними лініями):

$$-1{,}96 \cdot \frac{1}{\sqrt{n}} \le r_p \le 1.96 \cdot \frac{1}{\sqrt{n}}, \qquad (3.1)$$

де n – довжина часового ряду;

p – порядок автокореляції.

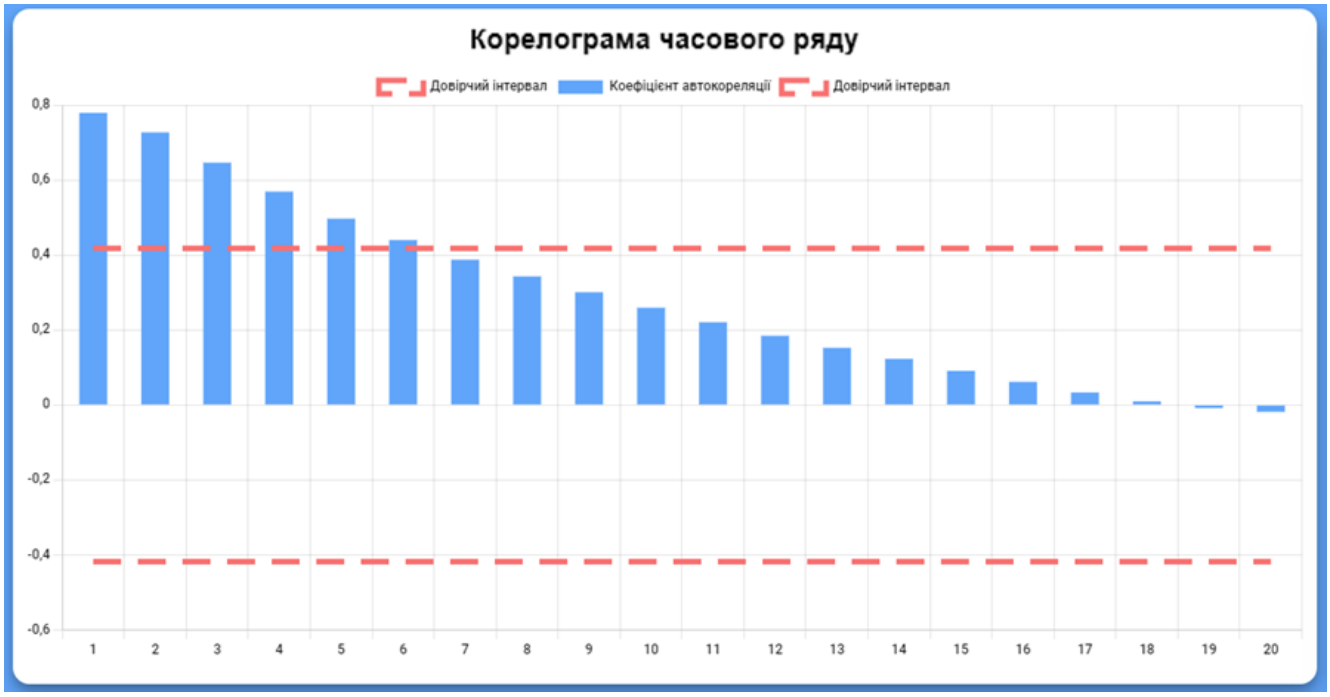

Рисунок 3.34 – Графік корелограми часового ряду

Таким чином, часовий ряд є нестаціонарним і має тренд, оскільки найбільшим по модулю виявилося значуще значення коефіцієнта автокореляції 1-го порядку. У разі наявності сезонних коливань корелограма містить багато максимальних і мінімальних значень коефіцієнтів автокореляцій, а даний графік

містить плавно спадаючі значення коефіцієнтів автокореляції. Тому зроблено висновок, що сезонної компоненти у структурі часового ряду немає (рис. 3.35).

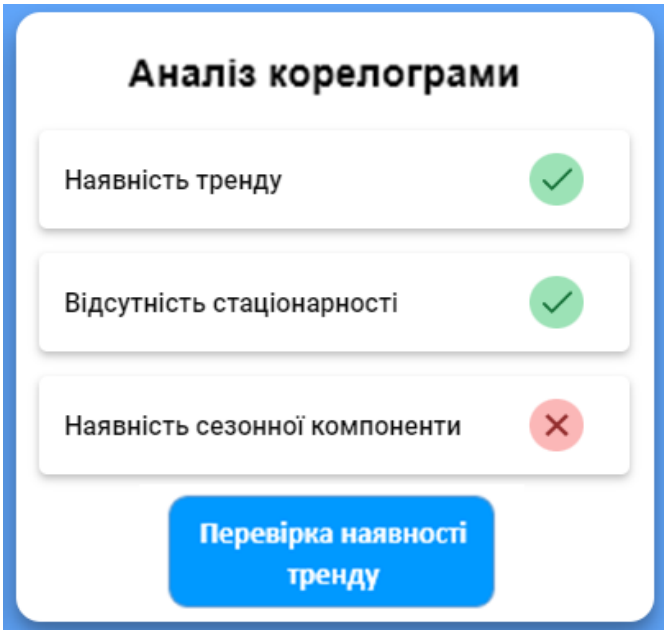

Рисунок 3.35 – Результати кореляційного аналізу

Для відображення результатів виявлення наявності тренду за методом середніх різниць та критерєм серій необхідно натиснути кнопку Перевірка наявності тренду. Нижче з’явиться результат перевірки за методом середніх різниць та критерієм серій (рис. 3.36).

За результатами здійсненого аналізу та виявленої структури часового ряду система у автоматичному режимі здійснює побудову сукупності прогнозних моделей часового ряду, обирає найбільш оптимальну модель та оцінює її якість, точність і адекватність. Користувач має можливість здійснити налаштування, вказавши моделі, серед яких буде здійснено вибір кращої при побудові прогнозної моделі та вказати показники, які буде розраховано для оцінки якості моделі (рис. 3.37). Для цього необхідно на лівій панелі обрати опцію Побудова прогнозних моделей. По замовчуванню обрано усі параметри, однак користувач може здійснити вибір однієї або декількох моделей та параметрів оцінки якості.

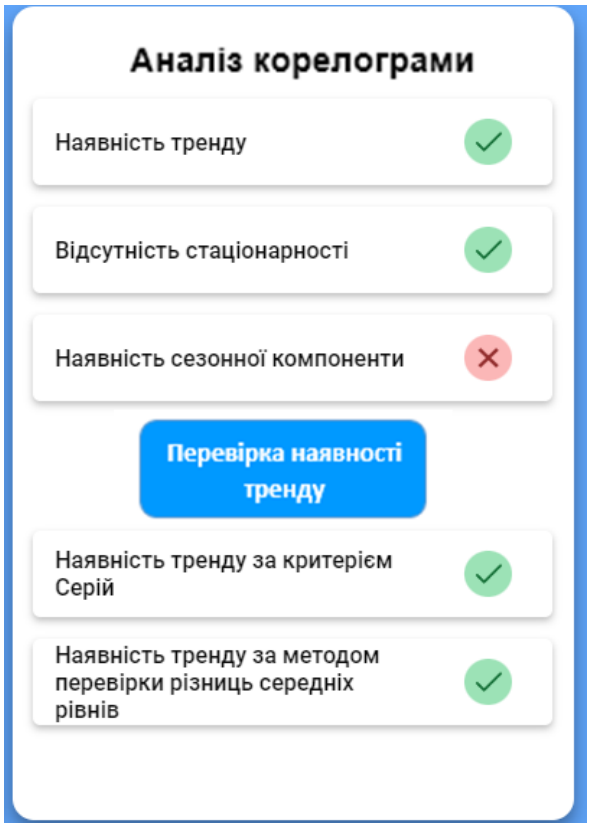

Рисунок 3.36 – Результат перевірки наявності тренду за критерієм серій та методом середніх різниць

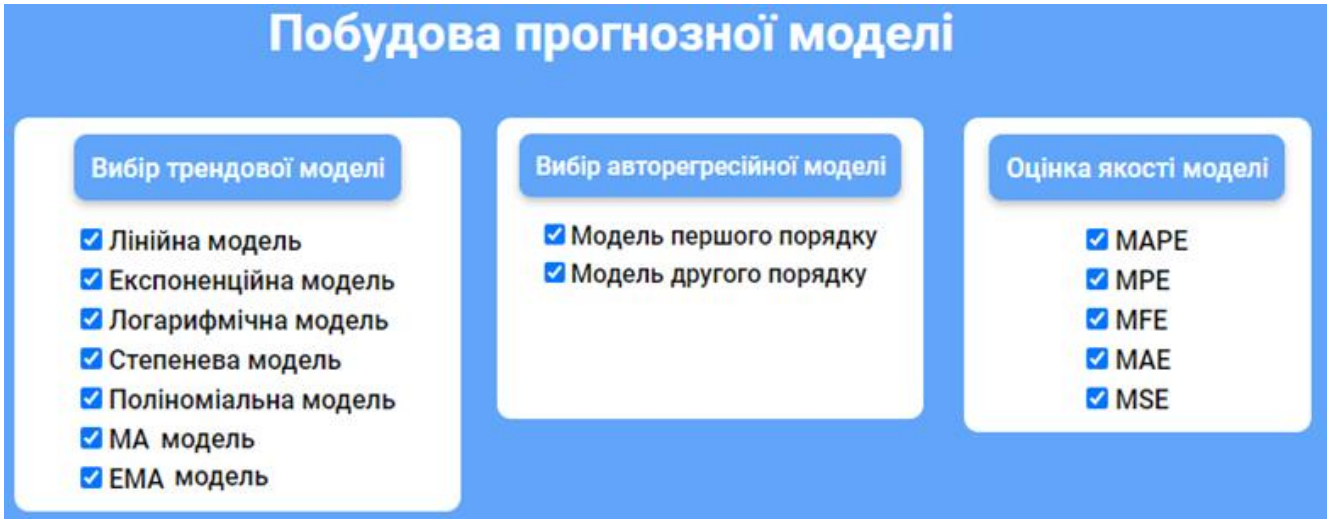

Рисунок 3.37 – Налаштування параметрів побудови прогнозної моделі

При побудові MA та EMA моделей кількість спостережень, що входять до інтервалу згладжування дорівнювала трьом: $n = 3$.

Згладжування значень часового ряду в МА-моделі здійснювалося за формулою:

$$S_t = m_{t-2} + \frac{1}{n}(Y_{t-1} - Y_{t-2}), \quad (3.2)$$

де m – ковзне середнє за 2 періоди до прогнозного: $m_t = (Y_{t-1} + Y_t + Y_{t+1})/3$.

Згладжування значень часового ряду в ЕМА-моделі здійснювалося за формулою:

$$S_t = \alpha Y_t + (1+\alpha) \cdot S_{t-1}, \quad (3.3)$$

де $\alpha = \frac{2}{n+1}$.

У вікні також виводяться графіки побудованих моделей, розраховуються їх коефіцієнти детермінації та метрики оцінки їх якості і точності із вказівкою кращої моделі. Часовий ряд курсу акцій, який аналізується, є нестаціонарним і має тренд. Тому було побудовано сукупність регресійних трендових моделей та моделі МА і ЕМА. Серед них було обрано поліноміальну модель, оскільки вона мала найвищий коефіцієнт детермінації, рівний 0,936 (рис. 3.38). А значить вона має найвищий ступінь апроксимації емпіричних даних часового ряду курсу акцій.

На рисунку 3.39 зображено розраховані метрики оцінки якості і точності побудованої моделі. Отримані значення свідчать про високу якість побудованої моделі. Середня похибка прогнозу MFE рівна -0,01: прогнозовані значення в середньому будуть відрізнятися від фактичних у більшу або меншу сторону, що є добрим показником. Середнє абсолютне відхилення прогнозованих значень від фактичних MAE рівне 8,98, становить 5,9% від середнього значення часового ряду і є меншим за 10%, що також є добрим показником. Середньоквадратична похибка MSE становить 131,21. Середня відсоткова похибка MPE рівна -0,24 та має від'ємний знак, що означає що прогнозні значення будуть трохи заниженими. Середня абсолютна похибка у відсотках MAPE становить 3,87%, що свідчить про високу точність прогнозу.

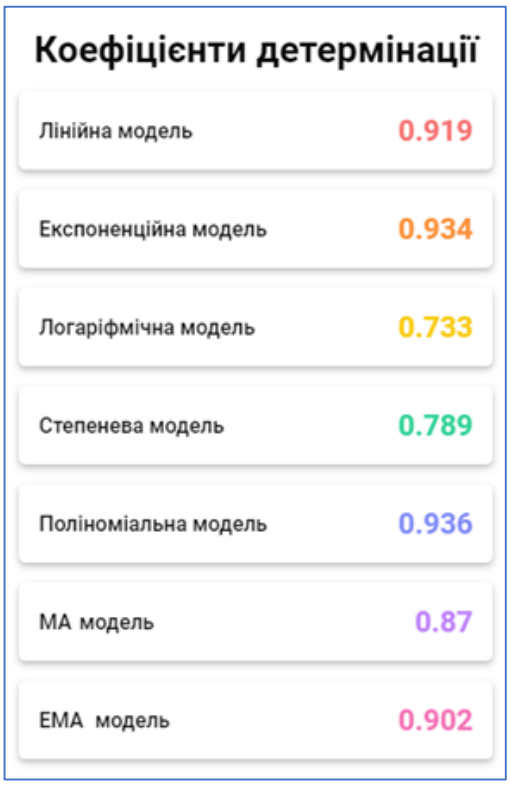

Рисунок 3.38 – Розраховані коефіцієнти детермінації

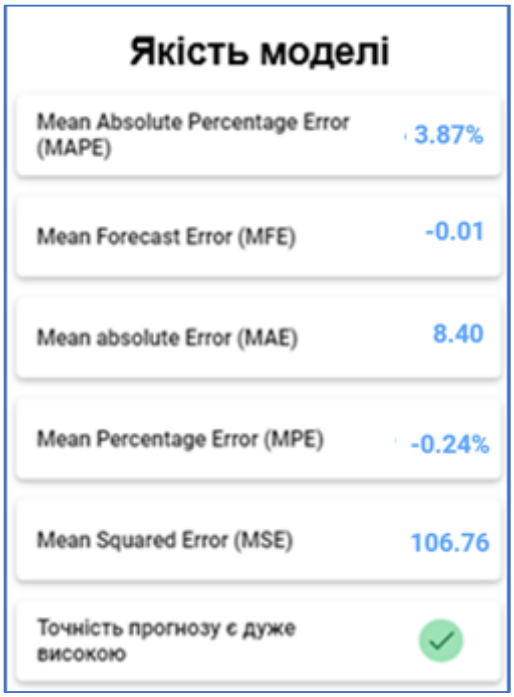

Рисунок 3.39 – Оцінка якості та точності моделі

На рисунку 3.40 зображено графік побудованої прогнозної моделі.

Рисунок 3.40 – Побудована прогнозна модель часового ряду

Для перевірки побудованої моделі на адекватність здійснюється автокореляційний аналіз випадкової компоненти. Для цього необхідно на лівій панелі обрати Дослідження випадкової компоненти. У вікні буде виведена корелограма випадкової компоненти та результат її аналізу (рис. 3.41, рис. 3.42).

Рисунок 3.41 – Графік корелограми випадкової компоненти прогнозної моделі

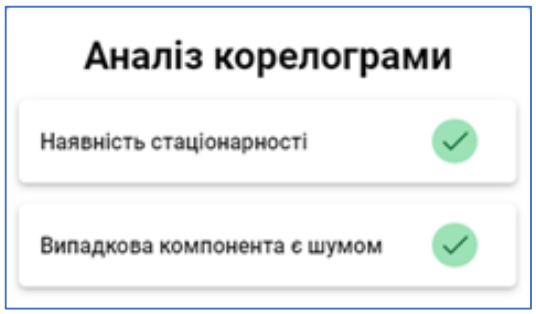

Рисунок 3.42 – Аналіз корелограми випадкової компоненти

Здійснений аналіз графіку автокореляційної функції випадкової компоненти показує, що корелограма містить чергування затухаючих додатних та від'ємних значень коефіцієнтів автокореляції. Жоден із коефіцієнтів автокореляції не є значимим, тобто не виходить за межі довірчого інтервалу, зображеного на графіку двома червоними пунктирними лініями. Отже, випадкова компонента є стаціонарним рядом – шумом, а побудована модель часового ряду є адекватною.

Побудована модель була застосована для побудови короткострокового прогнозу значень часового ряду (рис. 3.43). Прогнозні значення порівнювалися з реальними цінами акцій на два періоди уперед. Для відображення прогнозу необхідно на лівій панелі обрати Прогнозування.

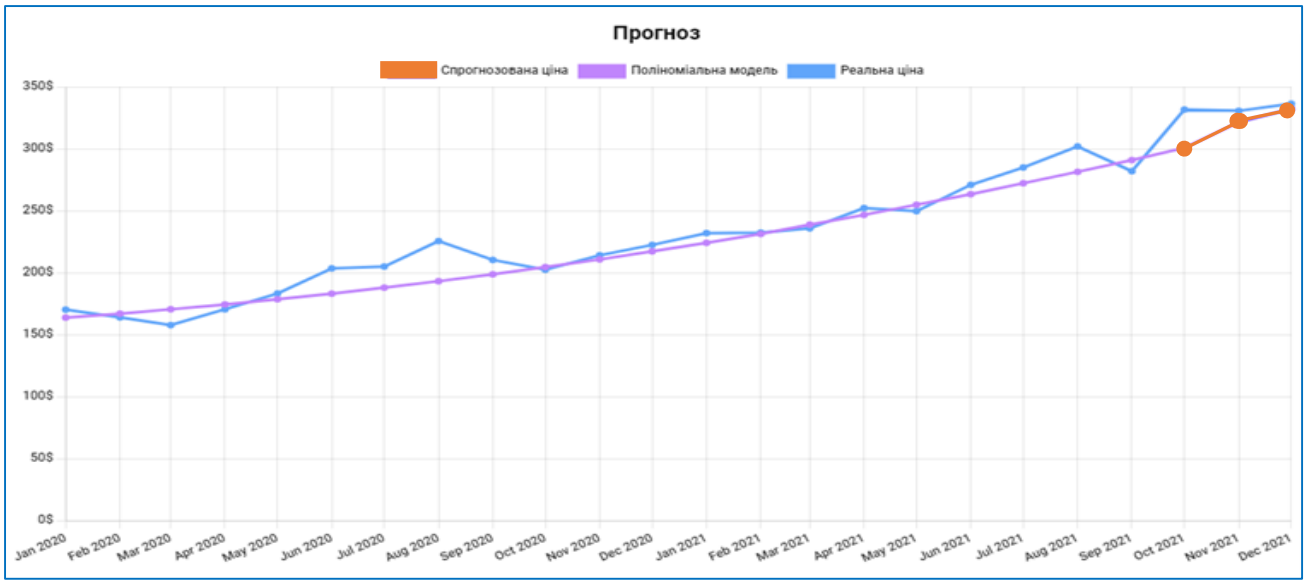

Рисунок 3.43 – Прогнозування курсу акцій

У таблиці 3.2 показано порівняння метрик оцінки якості моделей, серед яких здійснювався вибір кращої. Як бачимо, поліноміальна модель має найкращі показники: самий високий рівень апроксимації та точності. Трохи нижчий, але також високий рівень апроксимації мають лінійна, експоненціальна, ЕМ-модель та ЕМА-модель. І кореляційний аналіз їх випадкової компоненти підтверджує адекватність цих моделей.

Нижчим, але також прийнятним є рівень апроксимації та точність експоненційної та степеневої моделей. Однак корелограма випадкової компоненти у випадку вибору цих моделей не є стаціонарним рядом. Тому ці моделі не є адекватними.

Таблиця 3.2 – Порівняння моделей, серед яких здійснювався вибір кращої

Модель	R^2	MAPE	MFE	MAE	MPE	MSE
Лінійна $y = 6.399t + 152.019$	0,919	4,04	-0,00	8,86	-0,21	115,22
Експоненційна $y = 159.612e^{0.029t}$	0,934	3,93	-0,58	8,48	-0,0049	106,29
Логарифмічна $y = 1287.032 + 44.136 \ln t$	0,733	7,33	-0,00	15,85	-0,61	376,57
Степенева $y = 140.15t^{0.206}$	0,789	6,25	0,88	13,78	-0,29	294,20
Поліноміальна $y = 0.089t^2 + 4.442t + 159.52$	0,936	3,87	-0,01	8,40	-0,24	106,76
МА-модель	0,87	4,25	8,85	10,01	3,72	171,03
ЕМА-модель	0,902	3,99	7,92	9,16	3,26	124,05

У цілому реалізований у системі аналізу та прогнозування підхід забезпечує високу якість побудованої моделі та високу точність короткострокового прогнозу і може бути застосована для прогнозування курсу акцій та інших фінансових показників.

Висновки до розділу 3

У даному розділі було описано розробку, програмну реалізацію системи аналізу і прогнозування курсу акцій. Розроблена система дозволяє здійснювати очистку та попередню обробку набору даних зі значеннями часового ряду курсу акцій, виявляти структурні компоненти, наявність тренду, стаціонарність. На основі здійсненого аналізу є можливість побудови авторегресійної моделі для стаціонарного часового ряду. Для нестаціонарного часового ряду у системі передбачено здійснення декомпозиції, фільтрації компонент та моделювання тренду з розрахунком показників, які визначають якість прогнозної моделі та точність прогнозу. На основі здійсненого аналізу є можливість обрати більш якісну та точну модель, яку можна використовувати для прогнозування курсу акцій при плануванні інвестицій на фондовому ринку.

ВИСНОВКИ

Проведене дослідження дозволяє зробити наступні висновки. Фондовий ринок є важливою складовою ринкової економіки, на якому здійснюється продаж та купівля акцій комерційних компаній. Застосування аналізу та прогнозування фінансових часових рядів дозволяє отримувати важливу інформацію про фінансові ринки та допомагає в прийнятті обґрунтованих рішень щодо інвестування коштів.

Виявлено основні типи систем та програмних сервісів, які дозволяють здійснювати побудову прогнозів часових рядів: статистичні пакети (STATISTICA, IBM SPSS Statistics, Minitab, MatLab, SAS, S-Plus), табличні процесори (MS Excel, MS Excel 365), пакети візуального проєктування (MS Machine learning, Orange 3), ERP-системи, Business Intelligence системи, когнітивні та академічні системи прогнозування. Вони мають широкий функціонал для проведення аналізу часових рядів, мають вбудовані засоби для реалізації багатьох моделей та методів інтелектуального аналізу та побудови прогнозу. Однак їх використання для побудови прогнозу потребує трудомісткого процесу розгортання та підтримки, не завжди є системним та гнучким, потребує знання мов програмування та розуміння на професійному рівні усіх етапів аналізу даних та обґрунтування вибору оптимальної прогнозної моделі. Тому існує потреба у розробці системи, яка має зручний та зрозумілий інтерфейс, проста у налаштуванні й автоматизує усі етапи аналізу і прогнозування часових рядів курсу акцій.

Здійснений аналіз дозволив виявити велику кількість методів та моделей, які використовують для побудови прогнозу курсу акцій. Це методи прогнозування із використанням глибинних нейронних мереж, ауторегресійні AR-моделі, MA-модель ковзних середніх, EMA-модель експоненційних ковзних середніх, регресійні методи моделювання шляхом побудови лінійної та нелінійної аналітичної функції, адитивна AM-модель та узагальнена адитивна GAM-модель, мультиплікативна модель, ARIMA-модель інтегрованої авторегресії ковзного середнього та багато інших.

Важливе значення для побудови прогнозної моделі часового ряду має виявлення основних структурних складових часового ряду: тренду, сезонної, циклічної та випадкової компонент. Для прогнозування стаціонарного часового ряду будують ауторегресійну AR-модель. Для нестаціонарних часових рядів є два підходи до побудови прогностичної моделі: моделювання регулярних компонент у сукупності та розкладання часового ряду на компоненти та моделювання кожної компоненти окремо.

Основні етапи аналізу та прогнозування часового ряду включають: збір та попередню обробку даних, виявлення структури часового ряду, вибір методу прогнозування, розробка моделі прогнозування та оцінка її точності. Фінансові ринки є дуже непередбачуваними, тому важливо використовувати не один, а декілька методів для отримання більш точного прогнозу та зменшення ризику помилкових рішень. Для створення системи аналізу та прогнозування доцільно передбачити побудову сукупності моделей, визначення їх адекватності та точності і вибору оптимальної моделі для прогнозування нових значень курсу акцій

Попередній аналіз даних часового ряду включає очищення даних, розрахунок статистичних характеристик та виявлення структури часового ряду й установлення його стаціонарності. Для виявлення структури часового ряду застосовують візуальний аналіз його графіку, автокореляційний аналіз та критерії перевірки гіпотези про наявність тренду – метод перевірки різниць середніх та критерій серій.

Для побудови моделі нестаціонарного часового ряду передбачено побудову адитивної моделі часового ряду. Моделювання тренду здійснюють, відфільтровуючи випадкову компоненту та коливання із використанням сукупності методів механічного й аналітичного згладжування. Механічне згладжування передбачає побудову MA-моделі ковзних середніх та EMA-моделі експоненційних ковзних середніх. Аналітичне згладжування передбачає моделювання тренду шляхом побудови аналітичних регресійних лінійних та нелінійних функцій.

Для розробки системи аналізу та прогнозування фінансових часових рядів обґрунтовано використання середовища розробки Visual Studio Code, яке є

потужним інструментом для розробки вебзастосунків, підтримує обрані для розробки мови програмування. Основною мовою для створення веб-сторінок та структурування контенту обрано мову HTML. CSS використовуються для оформлення та стилізації візуального вигляду веб-сторінок. Для полегшення процесу розробки та підтримки стилей CSS та розширення їх можливостей використано препроцесор SASS.

Для реалізації динамічної функціональності на веб-сторінках використано мову програмування JavaScript. Застосування транскомпілятора Babel дозволяє перетворює сучасний код JavaScript в сумісний з різними браузерами. Візуалізація даних реалізована із використанням бібліотеки JavaScript Chart.js. Бібліотека SheetJS застосовувалася для роботи з вхідним набором даних із використанням форматів XLSX, XLS, CSV та інших. Для моделювання трендових моделей часового ряду курсу акцій використано бібліотеку Regression-js.

Для збирання та оптимізації коду вебзастосунку використано збірник модулів Webpack. Мережа доставки контенту CDNJS застосовувалася для доставки до користувачів статичних ресурсів – бібліотек JavaScript, що сприяло прискоренню завантаження сторінок та поліпшило продуктивність. Передачу даних між клієнтом та сервером було реалізовано з використанням формату обміну даними JSON. Центральним репозиторієм пакетів для проектів JavaScript є менеджер пакетів NPM. Він дозволяє швидко встановлювати, оновлювати та керувати залежностями проєкту.

Здійснено розробку та програмну реалізацію системи аналізу і прогнозування курсу акцій. Розроблена система дозволяє здійснювати очистку та попередню обробку набору даних зі значеннями часового ряду курсу акцій, виявляти структурні компоненти, наявність тренду, стаціонарність. На основі здійсненого аналізу є можливість побудови авторегресійної моделі для стаціонарного часового ряду. Для нестаціонарного часового ряду у системі передбачено здійснення декомпозиції, фільтрації компонент та моделювання тренду з розрахунком показників, які визначають якість прогнозної моделі та точність прогнозу. На

основі здійсненого аналізу є можливість обрати більш якісну та точну модель, яку можна використовувати для прогнозування курсу акцій при плануванні інвестицій на фондовому ринку.

Поставлені завдання виконано, однак у подальшому функціонал системи може бути розширено за рахунок розширення моделей, серед яких робиться вибір при побудові прогнозної моделі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Часові ряди : підручник. Прикарпаття, 2018. 82 с. URL: https://kstat.pnu.edu.ua/wp-content/uploads/sites/63/2018/04/Часові-ряди.pdf (дата звернення: 02.05.2023).

2. Мармоза А.Т. Теорія статистики : підручник. К.: Центр учбової літератури, 2013. 592 с.

3. Мороз О., Міловська К. Ветвлейний аналіз та прогнозування фінансових часових рядів. *Вісник Національного технічного університету «Харківський політехнічний інститут»*. 2020. Т. 1 №1-2. URL: http://pim.khpi.edu.ua/article/view/270110 (дата звернення: 03.05.2023).

4. Hamilton J.D. Time Series Analysis. New Jersey : Princeton University Press, 1994. 820 p.

5. Берзлев О.Ю. Сучасний стан інформаційних систем прогнозування часових рядів. *Управління розвитком складних систем*. 2013. Вип. 13. С. 78-82.

6. Андрусенко Ю.О. Аналіз основних моделей прогнозування часових рядів. *Збірник наукових праць Харківського національного університету Повітряних сил*. 2020. № 3(65). С. 91-96.

7. Ляшенко О.І., Кравець Т.В., Хрущ Л.З. Застосування пакетів прикладних програм в економетричному моделюванні фінансових часових рядів. *Економіко-математичне моделювання соціально-економічних систем*. 2017. Вип. 22. С. 33-59. URL: http://dspace.nbuv.gov.ua/bitstream/handle/123456789/132550/02-Liashenko.pdf?sequence=1 (дата звернення: 04.05.2023).

8. Огляд методів машинного навчання в задачі прогнозування фінансових часових рядів / Гурєєва К.М., Кудін О.В. Лісняк Ф.О. *Вісник Запорізького національного університету. Фізико-метематичні науки*. 2018. №2. С. 18-28. URL: http://nbuv.gov.ua/UJRN/Vznu_mat_2018_2_5 (дата звернення: 05.05.2023).

9. Котова О., Савинова В. Структура и информационно-логическая схема работы системы прогнозирования экономических временных рядов «СГМ Горизонт». *Modern Economy Success*. 2020. №5. С. 190-197.

10. Долгіх А.О., Байбуз О.Г. Аналіз методів, моделей та програмних засобів прогнозування часових рядів. *Відкриті інформаційні та комп'ютерні інтегровані технології*. 2018. №79. С. 74-85.

11. Прогнозування та аналіз часових рядів : методичні вказівки / укл. Юрченко М. Є. Чернігів : ЧНТУ, 2018. 88 с.

12. Фінансове прогнозування : електронний посібник. Луцьк, 2018. URL: https://elib.lntu.edu.ua/sites/default/files/elib_upload/Електронний%20посібник%20ФІНАНСОВЕ%20ПРОГНОЗУВАННЯ/index.html (дата звернення: 06.05.2023).

13. Моделювання і прогнозування по часових рядах : вебсайт. URL: https://stud.com.ua/75017/statistika/modelyuvannya_prognozuvannya_chasovih_ryadah (дата звернення: 07.05.2023).

14. Стаціонарні та нестаціонарні часові ряди. Основні характеристики часових рядів : вебсайт. URL: https://studfile.net/preview/2398204/page:22/ (дата звернення: 08.05.2023).

15. Фінансове моделювання : електронний посібник. Луцьк, 2018. URL: https://elib.lntu.edu.ua/sites/default/files/elib_upload/Електронний%20посібник%20ФМ/index.html (дата звернення: 09.05.2023).

16. Harvey A. C. Time Series Models. 2nd Edition. Cambridge : The MIT Press, 1993. 308 p.

17. Different types of Time-series Forecasting Models. Data Analytics : website. URL: https://vitalflux.com/different-types-of-time-series-forecasting-models/ (дата звернення: 10.05.2023).

18. Nielsen A. Practical Time Series Analysis: Prediction with Statistics and Machine Learning. California : O'Reilly Media, 2019. 497 p.

19. The Complete Guide to Time Series Models. Built-in. Data Science : website. URL: https://builtin.com/data-science/time-series-model (дата звернення 11.05.2023).

20. Introduction to the Fundamentals of Time Series Data and Analysis. APTECH : website. URL: https://www.aptech.com/blog/introduction-to-the-fundamentals-of-time-series-data-and-analysis/ (дата звернення: 12.05.2023).

21. Woodward W. A., Sadler B. P., Robertson S. Time Series for Data Science: Analysis and Forecasting. Boca Raton : Chapman and Hall/CRC, 2022. 506 p.

22. Shmueli G. Practical Time Series Forecasting: A Hands-On Guide. 3nd Edition. Florida : Axelrod Schnall Publishers, 2016. 208 p.

23. Montgomery D. C., Jennings C. L., Kulahci M. Introduction to Time Series Analysis and Forecasting. 2nd Edition. New Jersey : Wiley-Interscience, 2015. 515 p.

24. Time Series Analysis: Forecasting and Control / Box G. E., Jenkins G., Reinsel G., Ljung G. New Jersey : Wiley, 2015. 681 p.

25. Visual Studio Code - Code Editing. Visual Studio Code : вебсайт. URL: https://code.visualstudio.com/ (дата звернення: 13.05.2023).

26. HTML5 specification. W3C Working Draft : вебсайт. URL: https://www.w3.org/TR/2011/WD-html5-20110405/ (дата звернення: 13.05.2023).

27. CSS3 specification. W3C Group Draft Note : вебсайт. URL: https://www.w3.org/TR/css3-roadmap/#intro (дата звернення: 14.05.2023).

28. Syntactically Awesome Style Sheets. SASS : вебсайт. URL: https://sass-lang.com/ (дата звернення: 14.05.2023).

29. ECMAScript® 2024 Language Specification. TC39 : вебсайт. URL: https://tc39.es/ecma262/ (дата звернення: 15.05.2023).

30. What is Babel? Babel : вебсайт. URL: https://babeljs.io/docs (дата звернення: 15.05.2023).

31. Webpack Concepts. Webpack : вебсайт. URL: https://webpack.js.org/ (дата звернення: 16.05.2023).

32. ECMA-404 The JSON Data Interchange Standard. Json.org : вебсайт. URL: https://www.json.org/json-en.html (дата звернення: 16.05.2023).

33. CDNJS API Documentation. Cndjs : вебсайт. URL: https://cdnjs.com/api (дата звернення: 17.05.2023).

34. NPM Documentation. Npm Docs : вебсайт. URL: https://docs.npmjs.com/about-npm (дата звернення: 18.05.2023).

ДОДАТОК А

Функції побудови трендових моделей

```
//Побудова лінійної моделі тренду
linearModel.js:
function calculateLinearModel(timeSeries, inputData) {
  let data = transformDataForRegression(timeSeries, inputData);
  calculateLinearModelCoefficients(timeSeries, data);
  calculateLinearModelValues(
    timeSeries,
    timeSeries.analyticalSmoothing.linearModel.m,
    timeSeries.analyticalSmoothing.linearModel.c
  );
}
function calculateLinearModelCoefficients(timeSeries, data) {
  let result = regression.linear(data, { precision: 3 });
  timeSeries.analyticalSmoothing.linearModel.m = result.equation[0];
  timeSeries.analyticalSmoothing.linearModel.c = result.equation[1];
  timeSeries.analyticalSmoothing.linearModel.determinationCoefficient =
    result.r2;
  timeSeries.analyticalSmoothing.linearModel.predict = result.predict;
}
function calculateLinearModelValues(timeSeries, m, c) {
  timeSeries.analyticalSmoothing.linearModel.values = Array.from(
    { length: timeSeries.numberOfTimeSeries },
    (_, index) => {
      return m * index + c;
    }
  );
}
//Побудова експоненційної моделі тренду
exponentialModel.js:
function calculateExponentialModel(timeSeries, inputData) {
  let data = transformDataForRegression(timeSeries, inputData);
  calculateExponentialModelCoefficients(timeSeries, data);
  calculateExponentialModelValues(
    timeSeries,
    timeSeries.analyticalSmoothing.exponentialModel.a,
    timeSeries.analyticalSmoothing.exponentialModel.b
  );
}
function calculateExponentialModelCoefficients(timeSeries, data) {
  let result = regression.exponential(data, { precision: 3 });
  timeSeries.analyticalSmoothing.exponentialModel.a = result.equation[0];
  timeSeries.analyticalSmoothing.exponentialModel.b = result.equation[1];
  timeSeries.analyticalSmoothing.exponentialModel.determinationCoefficient =
    result.r2;
  timeSeries.analyticalSmoothing.exponentialModel.predict = result.predict;
```

```
}
function calculateExponentialModelValues(timeSeries, a, b) {
  timeSeries.analyticalSmoothing.exponentialModel.values = Array.from(
    { length: timeSeries.numberOfTimeSeries },
    (_, index) => {
      return a * Math.exp(b * index);
    }
  );
}
```

//Побудова логарифмічної моделі тренду

logarithmicModel.js:

```
function calculateLogarithmicModel(timeSeries, inputData) {
  let data = transformDataForRegression(timeSeries, inputData);
  calculateLogarithmicModelCoefficients(timeSeries, data);
  calculateLogarithmicModelValues(
    timeSeries,
    timeSeries.analyticalSmoothing.logarithmicModel.a,
    timeSeries.analyticalSmoothing.logarithmicModel.b
  );
}
function calculateLogarithmicModelCoefficients(timeSeries, data) {
  let result = regression.logarithmic(data, { precision: 3 });
  timeSeries.analyticalSmoothing.logarithmicModel.a = result.equation[0];
  timeSeries.analyticalSmoothing.logarithmicModel.b = result.equation[1];
  timeSeries.analyticalSmoothing.logarithmicModel.determinationCoefficient =
    result.r2;
  timeSeries.analyticalSmoothing.logarithmicModel.predict = result.predict;
}
function calculateLogarithmicModelValues(timeSeries, a, b) {
  timeSeries.analyticalSmoothing.logarithmicModel.values = Array.from(
    { length: timeSeries.numberOfTimeSeries },
    (_, index) => {
      return a + b * Math.log(index);
    }
  );
}
```

//Побудова степеневої моделі тренду

powerModel.js:

```
function calculatePowerModel(timeSeries, inputData) {
  let data = transformDataForRegression(timeSeries, inputData);
  calculatePowerModelCoefficients(timeSeries, data);
  calculatePowerModelValues(
    timeSeries,
    timeSeries.analyticalSmoothing.powerModel.a,
    timeSeries.analyticalSmoothing.powerModel.b
  );
}
function calculatePowerModelCoefficients(timeSeries, data) {
  let result = regression.power(data, { precision: 3 });
  timeSeries.analyticalSmoothing.powerModel.a = result.equation[0];
```

```
  timeSeries.analyticalSmoothing.powerModel.b = result.equation[1];
  timeSeries.analyticalSmoothing.powerModel.determinationCoefficient =
    result.r2;
  timeSeries.analyticalSmoothing.powerModel.predict = result.predict;
}
function calculatePowerModelValues(timeSeries, a, b) {
  timeSeries.analyticalSmoothing.powerModel.values = Array.from(
    { length: timeSeries.numberOfTimeSeries },
    (_, index) => {
      return a * Math.pow(index, b);
    }
  );
}
//Побудова степеневої моделі тренду
polynomialModel.js:
function calculatePolynomialModel(timeSeries, inputData) {
  let data = transformDataForRegression(timeSeries, inputData);
  calculatePolynomialModelCoefficients(timeSeries, data);
  calculatePolynomialModelValues(
    timeSeries,
    timeSeries.analyticalSmoothing.polynomialModel.a
  );
}

function calculatePolynomialModelCoefficients(timeSeries, data) {
  let result = regression.polynomial(data, { precision: 3 });
  result.equation.forEach((res) => {
    timeSeries.analyticalSmoothing.polynomialModel.a.push(res);
  });
  timeSeries.analyticalSmoothing.polynomialModel.determinationCoefficient =
    result.r2;
  timeSeries.analyticalSmoothing.polynomialModel.predict = result.predict;
}

function calculatePolynomialModelValues(timeSeries, a) {
  timeSeries.analyticalSmoothing.polynomialModel.values = Array.from(
    { length: timeSeries.numberOfTimeSeries },
    (_, index) => {
      return a[0] * Math.pow(index, 2) + a[1] * index + a[2];
    }
  );
}
```

ДОДАТОК Б

Функції розрахунку метрик оцінки якості прогнозної моделі

```
//MFE:
function meanForecastError(timeSeries, predictLength, predictedFuturePrices) {
  let mfe = 0;
  for (let i = 0; i < predictLength; i++) {
    mfe += timeSeries.actualFuturePrices[i] - predictedFuturePrices[i];
  }
  return (mfe /= predictLength);
}
//MAE:
function meanAbsoluteError(timeSeries, predictLength, predictedFuturePrices) {
  let mae = 0;
  for (let i = 0; i < predictLength; i++) {
    mae += Math.abs(
      timeSeries.actualFuturePrices[i] - predictedFuturePrices[i]
    );
  }
  return (mae /= predictLength);
}
//MAPE:
function meanAbsolutePercentageError(
  timeSeries,
  predictLength,
  predictedFuturePrices
) {
  let mape = 0;
  for (let i = 0; i < predictLength; i++) {
    mape += Math.abs(
      (timeSeries.actualFuturePrices[i] - predictedFuturePrices[i]) /
        timeSeries.actualFuturePrices[i]
    );
  }
  return (mape *= 100 / predictLength);
}
//MPE:
function meanPercentageError(timeSeries, predictLength, predictedFuturePrices) {
  let mpe = 0;
  for (let i = 0; i < predictLength; i++) {
    mpe +=
      (timeSeries.actualFuturePrices[i] - predictedFuturePrices[i]) /
      timeSeries.actualFuturePrices[i];
  }
  return (mpe *= 100 / predictLength);
}
//MSE:
function meanSquaredError(timeSeries, predictLength, predictedFuturePrices) {
  let mse = 0;
```

```
  for (let i = 0; i < predictLength; i++) {
    mse += Math.pow(
      timeSeries.actualFuturePrices[i] - predictedFuturePrices[i],
      2
    );
  }
  return (mse *= 1 / predictLength);
}
```