НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Інститут прикладного системного аналізу
Кафедра штучного інтелекту

До захисту допущено
В. о. завідувача кафедри
_____Ірина ДЖИГИРЕЙ
«___»_____20__ р.

Дипломна робота
на здобуття ступеня бакалавра
за освітньо-професійною програмою «Системи і методи штучного інтелекту» спеціальності 122 «Комп'ютерні науки»
на тему: «Методи інтелектуального аналізу для прогнозування фінансових даних у вигляді часового ряду »

Виконала:
студентка IV курсу, групи КІ-02
Харитонова Світлана Володимирівна _____

Керівник:
доцент кафедри ШІ, PhD
Гуськова Віра Геннадіївна _____

Консультант з економічного розділу:
професор, д.е.н., проф. кафедри ЕК ФММ,
Шевчук Олена Анатоліївна _____

Консультант з нормоконтролю:
фахівець першої категорії кафедри ШІ,
Кравець Павло Володимирович _____

Рецензент:
професор, д.т.н.,
Бідюк Петро Іванович _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студентка
_____

Київ – 2024 року

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Інститут прикладного системного аналізу

Кафедра штучного інтелекту

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 122 «Комп'ютерні науки»

Освітньо-професійна програма «Системи і методи штучного інтелекту»

ЗАТВЕРДЖУЮ

В. о. завідувачки кафедри

_____ Ірина ДЖИГИРЕЙ

«___»_____20__ р.

ЗАВДАННЯ

на дипломну роботу студентки

Харитоновій Світлані Володимирівні

1. Тема роботи «Методи інтелектуального аналізу для прогнозування фінансових даних у вигляді часового ряду», керівник роботи доцент, кандидат технічних наук Гуськова В.Г. затверджені наказом по університету від «31» травня 2024 року № 2240-с

2. Термін подання студентом роботи «10» червня 2024 року.

3. Вихідні дані до роботи: історичні дані про ціни на акції компанії Apple Inc, який є репрезентативним прикладом фінансового часового ряду.

4. Зміст роботи: аналіз предметної області, підготовка даних, їх фільтрація, вибір доцільних методів та моделей для роботи, тренування та навчання моделі обраними методами, дослідження варіантів модернізації методів, аналіз результатів.

5. Перелік ілюстративного матеріалу: використані метрики, опис набору даних, результати роботи програми

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Шевчук Олена Анатоліївна, професор, д. е. н.		

7. Дата видачі завдання:«05» лютого 2024 року.

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1.	Підготовка даних до роботи	27.03.2024	Виконано
2.	Вивчення літератури за темою роботи	29.04.2024	Виконано
3.	Підготовка першого розділу	08.05.2024	Виконано
4.	Підготовка другого розділу	18.05.2024	Виконано
5.	Розробка програмного продукту	25.05.2024	Виконано
6.	Підготовка третього розділу	17.05.2024	Виконано
7.	Підготовка четвертого розділу	24.05.2024	Виконано
8.	Підготовка економічної частини	02.06.2024	Виконано
9.	Оформлення дипломної роботи	04.06.2024	Виконано
10.	Підготовка презентації доповіді	10.06.2024	Виконано

Студентка Світлана ХАРИТОНОВА

Керівник Віра ГУСЬКОВА

РЕФЕРАТ

Дипломна робота: 92 с., 7 табл., 17 рис., 22 джерел.

МЕТОДИ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ДАНИХ ДЛЯ ПРОГНОЗУВАННЯ ФІНАНСОВИХ ДАНИХ У ВИГЛЯДІ ЧАСОВОГО РЯДУ.

Об’єкт дослідження – зміни фінансових показників акції.

Предмет дослідження – методи роботи для фільтрації даних, методи інтелектуального аналізу даних та моделі для прогнозування фінансових даних.

Мета роботи – аналіз традиційних та нових рішень для прогнозування часових ряді, розробка програмного продукту на базі цього аналізу.

Проведено ряд експериментів, в результаті отримано найбільш оптимальний метод для розв’язання задачі прогнозування.

Шляхи подальшого розвитку предмету дослідження – оптимізація продукту за рахунок тестування на більшій кількості різних даних, експериментування з розмірами досліджуваних даних, новими комбінаціями методів.

ABSTRACT

Master's thesis: 92 pages, 7 tables, 17 figures, 1 appendix, 22 sources.

METHODS OF DATA MINING FOR PREDICTING FINANCIAL DATA IN THE FORM OF A TIME SERIES.

The object of research is changes in financial indicators of stocks.

The subject of research is methods for data filtering, data mining methods, and models for predicting financial data.

The purpose of the work is to analyze traditional and new solutions for time series forecasting and to develop a software product based on this analysis.

A series of experiments were conducted, and as a result, the most optimal method for solving the prediction problem was obtained.

Ways of further development of the research subject include optimization of the product by testing on a larger amount of diverse data, experimenting with the sizes of the studied data, and new combinations of methods.

ЗМІСТ

ВСТУП

Фінансові ринки в сучасному світі характеризуються високою динамічністю, складністю та взаємопов'язаністю. В таких умовах точне прогнозування фінансових показників, таких як ціни акцій, курси валют, обсяги торгів тощо, стає критично важливим для прийняття обґрунтованих інвестиційних рішень, управління ризиками та забезпечення фінансової стабільності.

В такому контексті на перший план виходять методи інтелектуального аналізу даних, такі як машинне навчання та штучні нейронні мережі. Ці методи здатні автоматично виявляти приховані закономірності, адаптуватися до мінливих ринкових умов та забезпечувати більш точні прогнози порівняно з традиційними підходами. Особливо перспективними в цьому напрямку є моделі авторегресії з інтегрованим ковзним середнім (ARIMA) та рекурентні нейронні мережі з довгою короткочасною пам'яттю (LSTM).

Дана дипломна робота присвячена дослідженню та розробці методів прогнозування фінансових часових рядів з використанням моделей ARIMA, LSTM та їх комбінацій. В роботі проводиться комплексний аналіз предметної області, розглядаються теоретичні основи роботи з часовими рядами, та пропонується практична реалізація моделей з використанням мови програмування Python та її бібліотек.

Робота складається з трьох розділів. Перший розділ присвячений дослідженню предметної області, аналізу актуальності теми та огляду існуючих рішень. Другий розділ охоплює теоретичні основи роботи з часовими рядами, включаючи питання обробки пропущених даних, нормалізації, статистичних тестів та детальний опис моделей ARIMA та LSTM. Третій розділ містить опис програмної реалізації запропонованих методів, аналіз використаних даних та інтерпретацію отриманих результатів.

Результати даної роботи можуть бути використані в практичній діяльності фінансових аналітиків, трейдерів, інвестиційних компаній та інших учасників фінансового ринку для покращення якості прогнозування та прийняття більш обґрунтованих рішень. Крім того, робота робить внесок у розвиток методології аналізу фінансових часових рядів та демонструє потенціал застосування методів машинного навчання в цій сфері.

РОЗДІЛ 1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Актуальність

Актуальність теми обумовлена декількома ключовими факторами, що відображають зростаючу важливість та потребу у розробці ефективних інструментів для аналізу та прогнозування фінансових даних у сучасному світі.

По-перше, фінансові ринки стають все більш складними, динамічними та взаємопов'язаними. Глобалізація, технологічні інновації та швидкі зміни в економічному середовищі створюють нові виклики та можливості для інвесторів, фінансових установ та регуляторних органів. У таких умовах, точне прогнозування фінансових показників, таких як ціни акцій, курси валют, обсяги торгів, волатильність тощо, стає критично важливим для прийняття обґрунтованих інвестиційних рішень, управління ризиками та забезпечення фінансової стабільності. Традиційні методи прогнозування, засновані на статистичних моделях та експертних оцінках, часто виявляються недостатньо ефективними в умовах високої невизначеності та мінливості ринків.

По-друге, фінансові ринки генерують величезні обсяги даних у вигляді часових рядів, що відображають динаміку цін, обсягів торгів, ринкових індексів та інших показників. Ці дані мають складну структуру, часто містять шум, аномалії та нелінійні залежності. Традиційні статистичні методи, такі як регресійний аналіз та авторегресійні моделі, можуть бути обмеженими у своїй здатності виявляти приховані закономірності, адаптуватися до змін у ринкових умовах та забезпечувати точні прогнози на довгострокову перспективу. Тому виникає потреба у розробці та застосуванні більш досконалих методів аналізу даних, здатних ефективно обробляти великі обсяги інформації та виявляти складні залежності.

Методи інтелектуального аналізу даних, такі як машинне навчання та штучні нейронні мережі, демонструють значний потенціал у вирішенні завдань прогнозування фінансових даних. Ці методи здатні автоматично виявляти нелінійні залежності, адаптуватися до мінливих ринкових умов та забезпечувати більш точні прогнози порівняно з традиційними підходами. Машинне навчання дозволяє будувати моделі, які можуть навчатися на великих обсягах історичних даних, виявляти приховані закономірності та генерувати прогнози на основі нових даних. Штучні нейронні мережі, зокрема глибокі нейронні мережі та рекурентні нейронні мережі, показали вражаючі результати у задачах прогнозування часових рядів завдяки своїй здатності моделювати складні нелінійні залежності та враховувати довгострокові залежності в даних.

Застосування методів інтелектуального аналізу даних у фінансовій сфері має значну практичну цінність для різних зацікавлених сторін. Для інвесторів точні прогнози фінансових показників дозволяють приймати більш обґрунтовані рішення щодо розподілу активів, вибору інвестиційних стратегій та управління портфелем. Фінансові установи, такі як банки та інвестиційні компанії, можуть використовувати методи інтелектуального аналізу даних для оцінки ризиків, виявлення шахрайських операцій та оптимізації своїх бізнес-процесів. Регуляторні органи можуть застосовувати ці методи для моніторингу ринкових трендів, виявлення потенційних загроз та забезпечення стабільності фінансової системи. Дослідники та науковці також можуть використовувати методи інтелектуального аналізу даних для поглиблення розуміння складних фінансових процесів, розробки нових моделей та алгоритмів прогнозування.

Слід зазначити, що застосування методів інтелектуального аналізу даних у фінансовій сфері також пов'язане з певними викликами та обмеженнями. Фінансові дані часто містять шум, викиди та аномалії, що можуть впливати на якість прогнозів. Крім того, фінансові ринки є динамічними та піддаються впливу багатьох зовнішніх факторів, таких як

політичні події, економічні новини та ринкові настрої, що ускладнює завдання прогнозування. Тому розробка надійних та стійких методів інтелектуального аналізу даних, здатних враховувати ці фактори та адаптуватися до мінливих умов, є важливим напрямком досліджень.

Методи інтелектуального аналізу даних, такі як моделі авторегресії з інтегрованим ковзним середнім (ARIMA) та рекурентні нейронні мережі (RNN), зокрема довга короткочасна пам'ять (LSTM), показали значні успіхи в задачах прогнозування часових рядів. Ці методи здатні виявляти складні нелінійні залежності, враховувати довгострокові залежності та адаптуватися до змін у динаміці часових рядів.

Крім того, комбінування різних методів інтелектуального аналізу даних, таких як поєднання LSTM з згортковими нейронними мережами (CNN), може дати додаткові переваги в прогнозуванні фінансових часових рядів. Такі комбіновані моделі здатні враховувати як довгострокові залежності, так і локальні особливості часових рядів, що може покращити точність прогнозування.

1.2 Аналіз існуючих рішень

Коли мова йде про прогнозування фінансових даних у формі часових рядів, декілька мов програмування та методів виділяються завдяки своїй ефективності та популярності в галузі. Серед них Python, безперечно, є найбільш видатним. Його широка екосистема бібліотек робить його неймовірно потужним інструментом для аналізу фінансових часових рядів. Бібліотеки, такі як `pandas` для маніпулювання даними, `numpy` для числових операцій та `matplotlib` для візуалізації, є необхідними. Крім того, бібліотеки машинного навчання Python, такі як `scikit-learn`, та фреймворки

глибокого навчання, такі як `TensorFlow` та `Keras`, спрощують реалізацію складних прогнозних моделей, що робить його фаворитом серед науковців з даних та фінансових аналітиків.

R – ще один сильний претендент, особливо популярний у статистичних та академічних колах. Його всеосяжні пакети, такі як `forecast` для аналізу часових рядів, `TTR` для технічних правил торгівлі та `quantmod` для фінансового моделювання, забезпечують надійний набір інструментів для аналізу фінансових даних. MATLAB часто використовується в академічних та дослідницьких установах завдяки своїм потужним математичним та статистичним функціям. Зі спеціалізованими наборами інструментів, такими як `Econometrics Toolbox` та `Financial Toolbox`, MATLAB пропонує надійне середовище для фінансового аналізу. Хоча SQL не використовується в першу чергу для аналізу часових рядів, він є важливим для вилучення та маніпулювання даними, особливо при роботі з даними, що зберігаються в реляційних базах даних.

Кілька методів зазвичай використовуються для прогнозування часових рядів, кожен з яких має свої сильні сторони та відповідні випадки використання. Моделі ARIMA (AutoRegressive Integrated Moving Average) є основою прогнозування часових рядів, особливо ефективні для стаціонарних даних. Вони поєднують компоненти авторегресії, диференціювання та ковзного середнього для захоплення різних закономірностей у даних. Моделі ARIMA особливо корисні для одновимірних часових рядів, забезпечуючи міцну основу для розуміння та прогнозування майбутніх значень.

Мережі довготривалої короткострокової пам'яті (LSTM), тип рекурентної нейронної мережі (RNN), розроблені для вирішення проблеми зникаючого градієнта, що робить їх ідеальними для послідовностей з довгостроковими залежностями. LSTM відмінно справляються із захопленням часових залежностей у даних фінансових часових рядів, часто забезпечуючи більш точні прогнози порівняно з традиційними методами. Експоненціальне згладжування (ETS) зосереджується на згладжуванні даних

за допомогою зважених середніх значень минулих спостережень, причому більш пізнім спостереженням надається більша вага. Цей метод простий, але ефективний, особливо для даних з чіткими трендами та сезонними закономірностями.

Facebook Prophet – відносно новий гравець у цій галузі, призначений для обробки часових рядів із сильними сезонними ефектами та відсутніми даними. Він є інтуїтивно зрозумілим, простим у використанні та добре інтегрується з Python та R, що робить його популярним вибором для багатьох аналітиків. Машини опорних векторів (SVM), які використовуються в основному для завдань класифікації, можуть бути адаптовані для регресії (SVR) для прогнозування майбутніх значень у часовому ряді. SVM добре працюють з меншими наборами даних і ефективно захоплюють складні відносини в даних.

Моделі GARCH (Generalized Autoregressive Conditional Heteroskedasticity) в основному використовуються для моделювання та прогнозування фінансової волатильності. Вони важливі для розуміння мінливості в даних часових рядів, особливо для управління ризиками та ціноутворення на похідні фінансові інструменти.

Таблиця 1.1 – Порівняння існуючих рішень

Аспект	Python	R	MATLAB
Простота використання	Висока Інтуїтивний синтаксис, обширні бібліотеки	Висока Багатий на статистичні функції	Помірна Потужний, але складний
Гнучкість	Дуже висока Широкий спектр бібліотек та інструментів	Висока Всебічні пакети для аналізу	Висока Обширні математичні функції
Підтримка спільноти	Дуже висока Велика спільнота, багато ресурсів	Висока Сильна академічна та статистична підтримка	Помірна Нішева база користувачів

Завершення табл. 1.1

Аспект	Python	R	MATLAB
Інтеграція	Дуже висока Легко інтегрується з іншими інструментами	Висока Хороша інтеграція зі статистичними інструментами	Помірна Інтегрується в своїй екосистемі
Обчислю-вальна ефектив-ність	Висока Ефективна з оптимізованими бібліотеками	Помірна Ефективна для статистичного аналізу	Висока Оптимізована для математичних обчислень

Таблиця 1.2 – Порівняння існуючих рішень

Аспект	ARIMA	LSTM	ETS	Facebook Prophet
Простота використан ня	Помірна Потребує розуміння компонентів часових рядів	Помірна Потребує знання нейронних мереж	Висока Проста реалізація	Висока Дружній до користувача, інтуїтивний дизайн
Гнучкість	Висока Добре підходить для стаціонарних даних	Дуже висока Відмінно підходить для захоплення часових залежностей	Висока Ефективна для трендів та сезонних закономірно стей	Висока Обробляє сезонність та відсутні дані
Підтримка спільноти	Висока Добре задокумен-тована, широко використо-вується	Дуже висока Сильна підтримка моделей глибокого навчання	Висока Добре задокументо вана, широко використову ється	Висока Активна розробка, зростаюча спільнота

Завершення табл. 1.2

Аспект	ARIMA	LSTM	ETS	Facebook Prophet
Інтеграція	Висока Доступна в більшості статистичних та машинного навчання бібліотек	Дуже висока Інтегрується з TensorFlow, Keras тощо	Висока Доступна в більшості статистич-них програмних пакетів	Висока Добре інтегрується з Python та R
Обчислю-вальна ефектив-ність	Помірна Обчислюва-льно трудомістка для великих наборів даних	Помірна Високі обчислюваль ні витрати для великих мереж	Висока Ефективна для великих наборів даних	Висока Оптимізована для продуктив-ності

На практиці найбільш точні прогнози часто виходять із поєднання кількох методів. Наприклад, гібридний підхід може включати використання ARIMA для захоплення лінійних закономірностей та LSTM для захоплення нелінійностей. Цей ансамблевий підхід використовує сильні сторони різних моделей, що призводить до більш надійних прогнозів.

1.3 Постановка задачі

1. Дослідження ринку на наявність подібних програмних продуктів: перший крок передбачає ретельне дослідження ринку для виявлення існуючих програмних рішень, які виконують прогнозування фінансових часових рядів. Це включає аналіз функцій, можливостей та обмежень цих продуктів, щоб зрозуміти

поточний ландшафт та визначити можливості для інновацій та вдосконалення.

2. Аналіз потенціалу розробки продукту: на основі дослідження ринку наступним кроком є оцінка потенціалу розробки нового програмного продукту. Це включає оцінку технологічних досягнень, попиту на ринку та можливості створення продукту, який може перевершити існуючі рішення з точки зору точності, зручності використання та функціональності.
3. Аналіз існуючих методів побудови інтелектуальної системи аналізу: цей крок передбачає детальний аналіз методів та алгоритмів, які можуть бути використані для побудови інтелектуальної системи фінансового прогнозування. Це включає традиційні статистичні методи, такі як ARIMA, передові методи машинного навчання, такі як LSTM, та гібридні моделі, що поєднують LSTM з CNN. Крім того, аналіз охоплюватиме основні модулі, з яких повинна складатися така система, такі як попередня обробка даних, нормалізація, фільтрація та інтеграція моделей.
4. Дослідження вимог і обмежень різних методів: кожен метод і алгоритм має свій набір вимог і обмежень. Цей крок передбачає дослідження цих аспектів, щоб зрозуміти сильні та слабкі сторони кожного підходу. Будуть розглянуті такі фактори, як обчислювальна ефективність, простота реалізації, вимоги до даних та масштабованість.
5. Вибір найбільш доцільних методів для реалізації: на основі аналізу будуть обрані найбільш підходящі методи для кожного модуля системи. Це включає вибір найкращих алгоритмів для попередньої обробки даних, моделювання часових рядів та інтеграції гібридних моделей для забезпечення оптимальної продуктивності та точності.

6. Реалізація програмного продукту: за допомогою обраних методів буде розроблено програмний продукт. Цей крок включає кодування, інтеграцію різних модулів та забезпечення функціонування системи відповідно до задуму. Реалізація здійснюватиметься з використанням Python та його потужних бібліотек, таких як Keras для нейронних мереж та ARIMA для традиційного аналізу часових рядів.
7. Тестування продукту на відповідність і продуктивність: після реалізації програмний продукт буде ретельно протестовано, щоб переконатися, що він відповідає цілям проекту. Це включає перевірку точності прогнозів, оцінку надійності системи та виявлення будь-яких потенційних проблем. Тестування допоможе зробити висновки про переваги та недоліки створеної системи.
8. Оцінка та висновки: нарешті, результати етапу тестування будуть проаналізовані, щоб зробити висновки про ефективність системи. Це включає оцінку переваг і недоліків, визначення областей для вдосконалення та розгляд перспектив подальшого розвитку та вдосконалення продукту. Дотримуючись цих кроків, проект має на меті створити найсучасніший програмний продукт, який не лише забезпечує точні фінансові прогнози, але й пропонує цінне розуміння основної динаміки фінансових ринків. Інтеграція передових методів машинного навчання з традиційними методами забезпечує комплексне та надійне рішення, здатне задовольнити вимоги сучасного фінансового аналізу.

Висновки до розділу 1

У цьому розділі був проведений огляд та аналіз існуючих підходів до прогнозування фінансових часових рядів. Ми розглянули актуальність теми та її важливість для сучасних фінансових ринків, які стають все більш складними та динамічними.

Було виявлено, що традиційні методи прогнозування, засновані на статистичних моделях та експертних оцінках, часто виявляються недостатньо ефективними в умовах високої невизначеності та мінливості ринків. Тому виникає потреба у використанні сучасних методів інтелектуального аналізу даних, таких як машинне навчання та штучні нейронні мережі, які здатні автоматично виявляти складні нелінійні залежності та адаптуватися до змін ринкових умов.

Також було проаналізовано різні мови програмування та інструменти, які використовуються для аналізу фінансових часових рядів, такі як Python, R та MATLAB, зроблено порівняння цих інструментів з точки зору їх ефективності, гнучкості та підтримки спільноти.

Ми розглянули основні методи прогнозування часових рядів, включаючи моделі ARIMA, мережі LSTM, експоненціальне згладжування (ETS), та сучасні інструменти, такі як Facebook Prophet. Кожен з цих методів має свої переваги та недоліки, що визначає їх застосовність у різних сценаріях.

На основі проведеного аналізу можна зробити висновок, що найбільш точні прогнози часто виходять із поєднання кількох методів. Гібридні підходи, що комбінують традиційні статистичні методи з сучасними нейронними мережами, показують високу ефективність у прогнозуванні фінансових часових рядів.

Подальші дослідження мають бути спрямовані на вдосконалення цих методів та адаптацію їх до специфічних викликів фінансових ринків.

РОЗДІЛ 2 ТЕОРЕТИЧНІ ОСНОВИ АНАЛІЗУ ТА ПРОГНОЗУВАННЯ ЧАСОВИХ РЯДІВ

2.1 Основні положення

2.1.1 Часовий ряд та його види

Часовий ряд – це послідовність числових значень, що спостерігаються протягом певних періодів часу. Кожне значення має свій індекс, що відповідає періоду спостереження.

Спостережуваний часовий ряд можна розкласти на три компоненти: тренд (довгостроковий напрям), сезонний (систематичні, пов'язані з календарем рухи) і нерегулярний (безсистемні, короткострокові коливання).

2.1.2 Адитивна та мультиплікативна модель

Будь-який часовий ряд можна представити як суму детермінованого та випадкового компонентів:

$$y_t = d_t + r_t \ .$$

Детермінований компонент складається з трьох частин: трендового, сезонного та циклічного компонентів:

$$d_t = tr_t + s_t + c_t.$$

Таким чином, будь-який часовий ряд можна розглядати як суму:

$$y_t = tr_t + s_t + c_t + r_t.$$

Мультиплікативна модель базується на використанні логарифмів компонентів:

$$\ln y_t = \ln t\, r_t + \ln s_t + \ln c_t + \ln r_t \text{ або } y_t = tr_t \cdot s_t \cdot c_t \cdot r_t.$$

2.1.3 Компоненти часових рядів

Детермінований компонент змінюється за певними правилами, які можна визначити за допомогою досліджень та аналізу часових рядів. Зазвичай одним з основних параметрів, від яких залежить детермінований компонент, є час.

Аналіз часових рядів починається з виділення трендового компонента (рис. 1.1). Його присутність легко виявити, аналізуючи графік часових даних. Дослідники можуть описувати такі зміни за допомогою кривих, які мають аналітичне представлення.

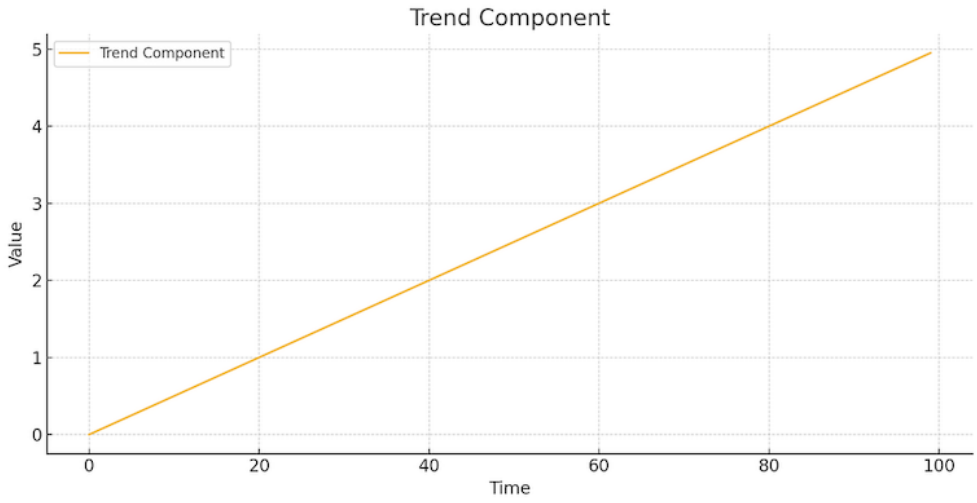


Рисунок 1.1 – Компонента тренду

Сезонний компонент (рис. 1.2) показує коливання навколо трендового компонента. Для економічних часових рядів сезонність пояснюється

регулярними змінами у виробництві та споживанні. Основна ідея виділення сезонних коливань полягає у порівнянні даних за відповідні періоди, а не за минулі проміжки часу.

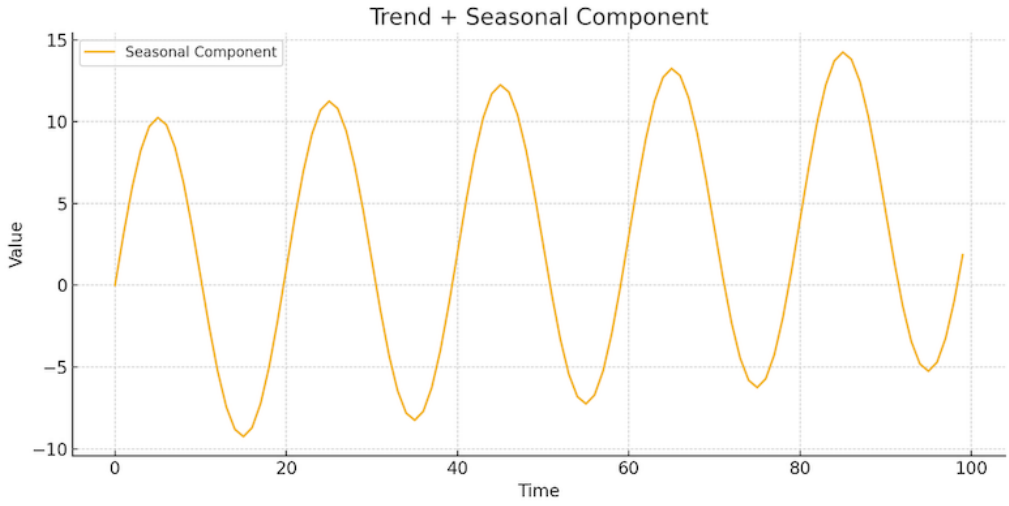


Рисунок 1.2 – Компонента тренду і сезонна компонента

Циклічний компонент (рис. 1.3) займає проміжне місце між трендом і сезонним компонентом. Тренд – це плавна зміна, яка проявляється на великому проміжку часу. Сезонний компонент – це періодична функція з коротким періодом повторюваності. Циклічний компонент є плавною зміною, залежною від часу, але він не включається ні до тренду, ні до сезонного компонента.

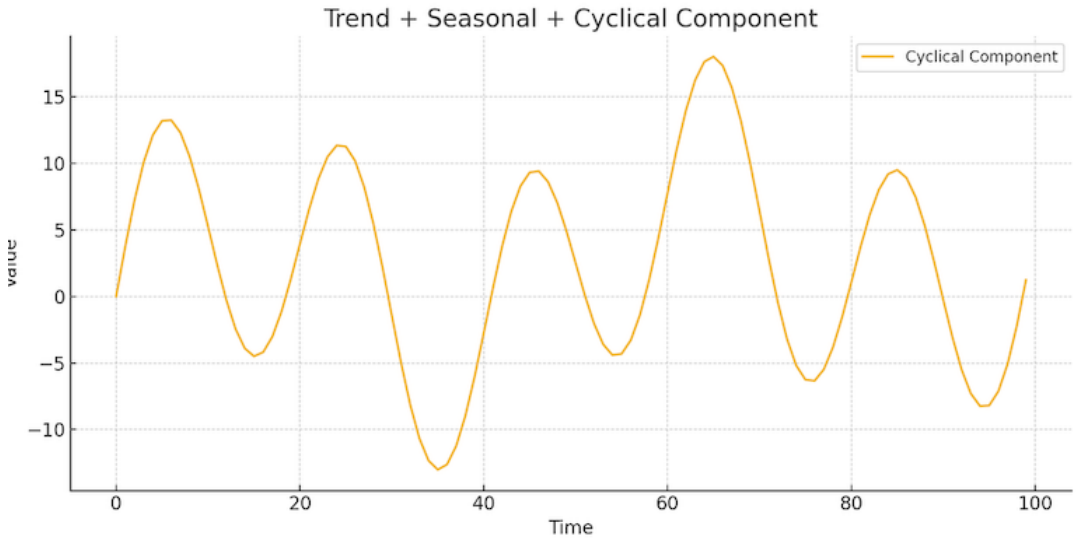


Рисунок 1.3 – Компонента тренду, сезонна і циклічна

Випадковий компонент (рис. 1.4) представляє те, що залишилося від часових даних після виключення тренду, циклічного та сезонного компонентів. Частина таких ефектів може бути пов'язана з непередбачуваними природними катаклізмами (землетруси, пожежі тощо), інша частина — з випадковими діями людей. Наявність випадкового компонента робить неможливим прогнозування значень часових рядів без помилки. Але будь-який реальний процес включає випадковий компонент.

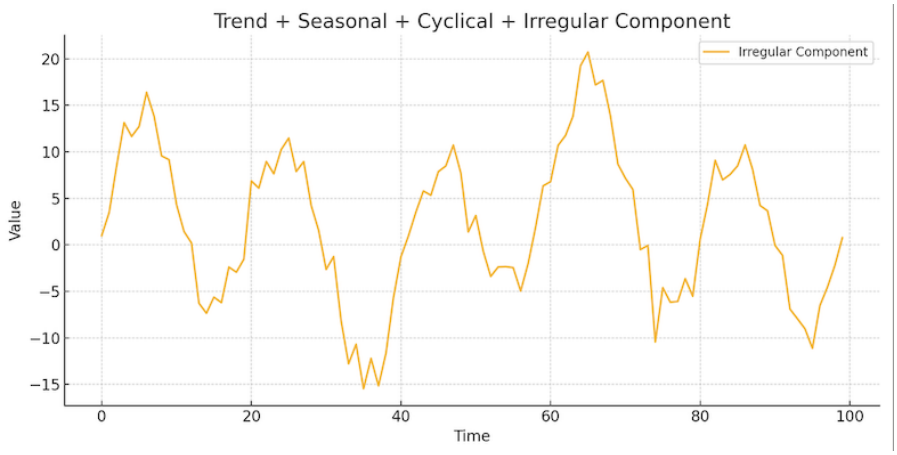


Рисунок 1.4 – Випадкова компонента

Значення кожного компонента впливає на значення часових даних у кожний період спостережень. Проте не завжди можна окремо характеризувати кожний компонент. Іноді краще робити прогноз відносно всієї моделі, ніж намагатися виділити кожний компонент окремо.

2.1.4 Міри точності прогнозів

Розробка прогнозу потребує оцінки його точності та надійності. Точність прогнозу визначається розміром помилки, яка є різницею між прогнозованим і фактичним значенням. Існує кілька методів для оцінки точності прогнозів:

– MSE (Mean Squared Error): середнє квадратів похибок прогнозу:

$$MSE = \frac{1}{n}\sum_t (y_t - \hat{y}_t)^2.$$

– RMSE (Root Mean Squared Error): середньоквадратична похибка прогнозу:

$$\mathrm{RMSE} = \sqrt{\mathrm{MSE}}.$$

– MAD (Mean Absolute Deviation): середня абсолютна похибка:

$$\mathrm{MAD} = \frac{1}{n}\sum_t |y_t - \hat{y}_t|.$$

– RMSPE (Root Mean Squared Percentage Error): середньоквадратичне значення відносних похибок.

$$RMSPE = 100\sqrt{\frac{1}{n}\sum_t\left(\frac{y_t-\widehat{y_t}}{y_t}\right)^2}.$$

– MAPE (Mean Absolute Percentage Error): середнє значення абсолютних величин відносних похибок:

$$MAPE = \frac{100}{n}\sum_t\left|\frac{y_t-\widehat{y_t}}{y_t}\right|.$$

Ці показники дозволяють оцінити відповідність моделі реальним даним та порівняти різні моделі між собою. Прогнози можуть оцінюватись за точністю: високоточні (MAPE < 10%), добрі (10% < MAPE < 20%), задовільні (20% < MAPE < 40%), погані (40% < MAPE < 50%) або незадовільні (MAPE > 50%).

2.2 Робота з пропущеними даними

Найпростішим рішенням обробки даних, що мають прогалини, є виключення некомплектних спостережень, що містять пропуски, і подальша робота з такими «повними» даними. За такого підходу можемо спостерігати сильну відмінність між результатами, що були отримані з початкових даних і вже оброблених. В такому разі маємо звернутися до інших методів – методів заповнення пропусків перед аналізом масиву даних.

Для того, щоб належно елімінувати пропуски, необхідно зрозуміти механізми їх формування. Відповідно до загальноприйнятих визначень, описують три види формування пропусків.

1. MCAR (Missing Completely At Random) – вид наявності пропусків, в якому кожне поле має однакову ймовірність

пропуску. За такого механізму, ігнорування чи вилучення записів, зазвичай, не веде до сильного погіршення результатів, а часто і зовсім не впливає.

2. MAR (Missing At Random) – вид наявності пропусків, в якому ймовірність пропуску може бути визначена на основі іншої наявної інформації без пропусків. Таким чином це не випадково пропущені характеристики і причиною їм є певні закономірності, тому як і в MCAR, це не веде до суттєвого спотворення результатів.
3. MNAR (Missing Not At Random) – вид наявності пропусків, коли невідомі чинники визначають існування чи не існування прогалин в наборі даних. В наслідок, на основі даних неможливо якось оцінити ймовірність прогалин.

Існують різні методи заповнення пропущених значень.

1. Метод Hot Deck – цей метод використовує заміну пропущеного значення найближчим доступним інформаційним об'єктом. Пропущені дані можна відновлювати з усієї групи повних спостережень або з певної підгрупи, такої як кластер, до якого належить цільовий об'єкт. Для заповнення пропуску використовується значення цієї характеристики з найближчого об'єкта до цільового. Вибір типу функції відстані для визначення найближчого спостереження залежить від типу досліджуваних даних та характеру зв'язку між змінними, а також від постановки конкретного дослідження.
2. Метод Барлета – він включає два етапи: спочатку на першому етапі використовується заміщення пропущених значень початковими згенерованими значеннями; на другому етапі проводиться коваріаційний аналіз цільової змінної та побудова дихотомічного індикатора повноти спостережень за цією змінною. Індикатор повноти спостережень завжди дорівнює 0,

крім одного випадку: якщо і-те значення є пропущеним для цільової змінної, то індикатор набуває значення 1.

3. Алгоритм ZET – цей алгоритм використовує частину сукупності спостережень, відому як компонентна матриця, для заповнення пропущених значень. Компонентна матриця складається з компонентних рядків і стовпців, і вона обмежена лише на деяку частину даних, а не на всю сукупність спостережень. Кожен компонентний рядок має величину, яка залежить від декартової відстані до цільового рядка, який має пропущене значення. За допомогою компонентної матриці будується функціональна залежність між прогнозним значенням і відповідним значенням в компонентній матриці, що дозволяє прогнозувати пропущені значення.

2.3 Нормалізація та масштабування

Необхідність нормалізації та масштабування випливає з природи реальних даних, які часто мають різні діапазони та розподіли для різних ознак. Коли ознаки мають різні одиниці вимірювання або масштаби, вони можуть непропорційно впливати на результат алгоритмів машинного навчання. Це особливо проблематично для алгоритмів, які спираються на метрики відстані, таких як k-найближчих сусідів (KNN) та опорні вектори (SVM).

Розглянемо набір даних, що містить вік (від 0 до 100) та дохід (від 0 до 100000). Без нормалізації ознака доходу домінуватиме в аналізі, спотворюючи результати. Нормалізуючи дані, ми забезпечуємо рівний внесок кожної ознаки, підвищуючи ефективність та точність моделі.

Для нормалізації та масштабування даних можна використовувати кілька методів, кожен з яких має свої переваги та застосування. Найбільш поширеними методами є Min-Max масштабування, Z-Score нормалізація та Robust масштабування.

Min-Max масштабування, також відоме як масштабування ознак, перетворює дані таким чином, щоб вони відповідали певному діапазону, зазвичай між 0 та 1. Цей метод корисний, коли мінімальні та максимальні значення ознаки відомі та постійні. Формула для Min-Max масштабування:

$$X' = \frac{X - X_{\min}}{X_{\max} - X_{\min}},$$

де X – представляє оригінальні дані;
$X_{\min}$ та $X_{\max}$ – мінімальне та максимальне значення ознаки відповідно.

Ця методика забезпечує рівний внесок усіх ознак в аналіз.

Z-Score нормалізація (стандартизація). Z-Score нормалізація, або стандартизація, масштабує дані на основі середнього значення та стандартного відхилення ознаки. Цей метод особливо корисний, коли дані мають гауссівський розподіл. Формула для Z-Score нормалізації:

$$X' = \frac{X - \mu}{\sigma},$$

де μ – середнє значення ознаки;
σ – стандартне відхилення.

Ця методика перетворює дані таким чином, щоб вони мали середнє значення 0 та стандартне відхилення 1, що робить її придатною для алгоритмів, які припускають нормально розподілені дані.

Robust масштабування призначене для ефективного опрацювання викидів, використовуючи медіану та міжквартильний розмах (IQR) замість середнього значення та стандартного відхилення. Формула для Robust масштабування:

$$X' = \frac{X - \text{median}}{\text{IQR}}.$$

Цей метод особливо корисний, коли дані містять значні викиди, оскільки він зменшує їх вплив на процес масштабування.

2.4 Статистичні тести

В аналізі часових рядів розуміння властивостей даних є вирішальним для ефективного моделювання та прогнозування. Для аналізу характеристик даних часових рядів, таких як стаціонарність, автокореляція та часткова автокореляція, використовується кілька статистичних тестів і інструментів. Розглянемо чотири основні тести та інструменти: розширений тест Дікі-Фуллера (ADF), тест Квятковського-Філліпса-Шмідта-Шина (KPSS), автокореляційна функція (ACF) і часткова автокореляційна функція (PACF).

Розширений тест Дікі-Фуллера (ADF). Тест ADF є широко використовуваним статистичним тестом, який перевіряє наявність одиничного кореня в часовому ряді, тим самим тестуючи стаціонарність ряду. Стаціонарність є критичним припущенням в аналізі часових рядів, оскільки багато моделей, таких як ARIMA, вимагають, щоб дані були стаціонарними.

Стаціонарний часовий ряд має постійне середнє значення, дисперсію та автокореляцію з плином часу. Ця властивість спрощує процес моделювання та

покращує точність прогнозів. Нестаціонарні ряди можуть призвести до помилкових регресій і ненадійних результатів.

Нульова гіпотеза (H0): часовий ряд має одиничний корінь (тобто він нестаціонарний). Альтернативна гіпотеза (H1): часовий ряд не має одиничного кореня (тобто він стаціонарний). Якщо p-значення тесту нижче певного порогу (зазвичай 0,05), ми відкидаємо нульову гіпотезу і робимо висновок, що часовий ряд є стаціонарним.

Тест ADF може бути застосований у трьох різних формах залежно від характеру даних.

1. Тест без константи та тренду: цей варіант підходить для рядів, які, як вважається, не мають тренду та мають нульове середнє значення.
2. Тест з константою: цей варіант включає константу (зсув) у тестове рівняння і підходить для рядів із ненульовим середнім значенням.
3. Тест з константою та трендом: цей варіант включає як константу, так і лінійний часовий тренд у тестове рівняння. Він підходить для рядів, які виявляють тенденцію з плином часу.

Вибір правильної форми тесту ADF залежить від візуального аналізу даних і теоретичного розуміння процесу, що генерує дані.

Тест Квятковського-Філліпса-Шмідта-Шина (KPSS). Тест KPSS доповнює тест ADF, надаючи інший підхід до перевірки стаціонарності. У той час як тест ADF перевіряє наявність одиничного кореня, тест KPSS перевіряє відсутність одиничного кореня.

Нульова гіпотеза (H0): часовий ряд є стаціонарним. Альтернативна гіпотеза (H1): часовий ряд не є стаціонарним (тобто він має одиничний корінь).

У тесті KPSS, якщо p-значення нижче порогу, ми відкидаємо нульову гіпотезу і робимо висновок, що ряд не є стаціонарним. Цей тест особливо корисний у поєднанні з тестом ADF для підтвердження статусу стаціонарності часового ряду.

Подібно до тесту ADF, тест KPSS також має різні форми залежно від припущень про дані.

1. Тест рівня: цей варіант перевіряє стаціонарність навколо постійного рівня (середнього значення). Він підходить для рядів без явної тенденції.
2. Тест тренду: цей варіант перевіряє стаціонарність навколо детермінованого лінійного тренду. Він підходить для рядів, які демонструють тенденцію з плином часу.

Вибір між тестом рівня та тестом тренду KPSS залежить від візуальних характеристик даних і очікуваної поведінки процесу, що генерує дані.

Автокореляційна функція (ACF). ACF вимірює кореляцію між часовим рядом і його лаговими значеннями. Вона допомагає визначити ступінь, в якому поточні значення ряду пов'язані з його минулими значеннями.

Інтерпретація графіка ACF.

1. Значущі лаги: піки, які перевищують поріг значущості, вказують на значну автокореляцію на цих лагах.
2. Розпізнавання патернів: повільний спад вказує на нестаціонарність, тоді як різке обрізання вказує на відповідного кандидата для моделі MA.

Застосування ACF у моделюванні часових рядів включає кілька ключових етапів. По-перше, ідентифікація моделі: графік ACF допомагає визначити порядок компоненти MA в моделях ARIMA, де значні піки на певних лагах вказують на необхідність включення цих лагів у модель. По-друге, діагностична перевірка: після підгонки моделі ACF залишків використовується для перевірки наявності залишкової автокореляції; значні піки вказують на те, що модель не повністю охоплює структуру автокореляції в даних. Нарешті, прогнозування: ACF допомагає оцінювати майбутні значення часового ряду на основі історичних даних, що особливо корисно для короткострокових прогнозів у моделях MA.

Часткова автокореляційна функція (PACF). PACF вимірює кореляцію між часовим рядом і його лаговими значеннями після видалення впливу проміжних лагів. Вона ізолює прямий зв'язок між спостереженнями на різних лагах. PACF використовується для визначення порядку авторегресійної (AR) частини моделі ARIMA. Зосереджуючись на прямих зв'язках, вона допомагає визначити лаги, які мають найбільш значний вплив на поточні значення ряду.

Інтерпретація графіка PACF.

1. Значущі лаги: піки на графіку PACF, які перевищують поріг значущості, вказують на значну часткову автокореляцію на цих лагах.
2. Розпізнавання патернів: різке обрізання вказує на відповідного кандидата для моделі AR, тоді як повільний спад може вказувати на необхідність диференціювання або більш складної моделі.

Застосування PACF у моделюванні часових рядів є важливим, особливо в контексті моделей ARIMA. По-перше, графік PACF допомагає визначити порядок компоненти AR у моделях ARIMA, де значні піки на певних лагах вказують на необхідність включення цих лагів у модель AR. По-друге, PACF допомагає визначити необхідність диференціювання часового ряду; повільний спад PACF може свідчити про необхідність диференціювання для досягнення стаціонарності. Нарешті, PACF залишків використовується для оцінки адекватності підігнаної моделі; значні піки в PACF залишків вказують на необхідність додаткового налаштування моделі.

2.5 ARIMA

Модель авторегресійного інтегрованого ковзного середнього (ARIMA) є наріжним каменем прогнозування часових рядів. Вона поєднує три компоненти – авторегресію (AR), інтеграцію (I) та ковзне середнє (MA) – для аналізу та прогнозування майбутніх точок часового ряду.

Моделі ARIMA моделюють дані часових рядів за допомогою трьох параметрів: *p, d,* і *q* (рис. 2.1), які відповідають порядку авторегресії, ступеню диференціювання та порядку ковзного середнього відповідно. Ось як кожен компонент робить свій внесок у модель.

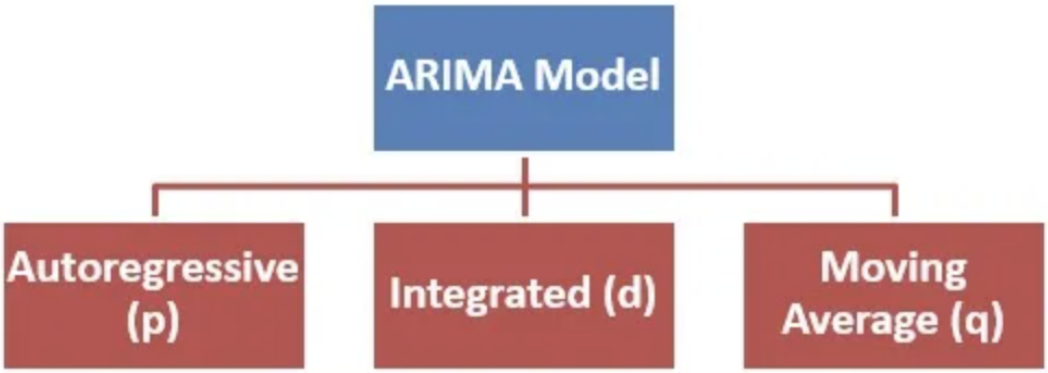


Рисунок 2.1 – Компоненти моделі ARIMA

Авторегресійний компонент включає регресію часового ряду на його власні лагові значення. Параметр p вказує кількість лагових спостережень, включених у модель. Частина AR відображає зв'язок між спостереженням і певною кількістю лагових спостережень.

Наприклад, модель AR(1) (де $p = 1$) виглядає так:

$$Y_t = \phi_1 Y_{t-1} + \epsilon_t,$$

де Y_t – поточне значення;

Y_{t-1} – лагове значення;

ϕ_1 – коефіцієнт;

а ϵ_t – член похибки.

Компонент інтеграції відноситься до диференціювання часового ряду для досягнення стаціонарності. Стаціонарність означає, що статистичні властивості ряду, такі як середнє значення та дисперсія, є постійними в часі. Параметр d представляє кількість разів, коли ряд потрібно диференціювати, щоб стати стаціонарним.

Наприклад, якщо $d = 1$, операція диференціювання буде виглядати так:

$$Y't = Y_t - Y_{t-1}.$$

Диференціювання допомагає видалити тренди та сезонність, стабілізуючи середнє значення часового ряду.

Ковзне середнє (MA). Компонент ковзного середнього моделює похибку часового ряду як лінійну комбінацію минулих членів похибки. Параметр q вказує кількість лагових похибок прогнозу в рівнянні прогнозування.

Наприклад, модель MA(1) (де $q = 1$) виглядає так:

$$Y_t = \epsilon_t + \theta_1 \epsilon_{t-1},$$

де ϵ_t – поточний член похибки;

θ_1 – коефіцієнт для лагового члена похибки ϵ_{t-1}.

Поєднання AR, I та MA. Модель ARIMA поєднує ці три компоненти в одне рівняння:

$$Y_t = \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \cdots + \phi_p Y_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \cdots + \theta_q \epsilon_{t-q},$$

де Y_t – диференційований ряд, якщо $d > 0$.

Це поєднання дозволяє ARIMA охоплювати широкий спектр поведінки часових рядів. ARIMA особливо добре підходить для прогнозування часових рядів з кількох ключових причин.

1. Гнучкість і адаптивність: ARIMA може моделювати різні типи даних часових рядів, включаючи ті, що мають тренди, сезонність (з сезонними розширеннями, такими як SARIMA) та нестаціонарні характеристики. Ця гнучкість робить її застосовною до широкого спектру наборів даних.
2. Відображення залежностей: компонент AR відображає залежності між спостереженням та його минулими значеннями, тоді як компонент MA відображає залежності між спостереженням та минулими похибками. Це подвійне відображення залежностей дозволяє ARIMA ефективно моделювати складні структури часових рядів.
3. Диференціювання для стаціонарності: завдяки включенню диференціювання, ARIMA може обробляти нестаціонарні дані. Ця здатність перетворювати нестаціонарні дані на стаціонарні є вирішальною, оскільки багато статистичних методів, включаючи саму ARIMA, краще працюють зі стаціонарними даними.
4. Оцінка та вибір моделі: моделі ARIMA можна оцінювати та порівнювати за допомогою критеріїв, таких як інформаційний критерій Акаіке (AIC) та байєсівський інформаційний критерій (BIC). Ці критерії допомагають вибрати модель, яка найкраще підходить, забезпечуючи точні та надійні прогнози.
5. Аналіз залишків: після налаштування моделі ARIMA можна провести аналіз залишків, щоб перевірити, чи залишки

поводяться як білий шум. Цей діагностичний крок гарантує, що модель зафіксувала основні закономірності в даних, залишивши лише випадковий шум.

6. Універсальність у застосуванні: ARIMA можна застосовувати в різних областях, таких як фінанси (для цін на акції та економічних показників), роздрібна торгівля (для прогнозування попиту), охорона здоров'я (для аналізу даних пацієнтів) та метеорологія (для прогнозування погоди). Її універсальність робить її методом вибору для аналізу та прогнозування часових рядів.

2.6 LSTM

Довгострокова короткострокова пам'ять (Long Short-Term Memory, LSTM) – це тип архітектури штучних рекурентних нейронних мереж (RNN), що використовується у сфері глибинного навчання. На відміну від стандартних прямопередатних нейронних мереж, LSTM мають зворотні зв'язки, що дозволяє їм використовувати тимчасові залежності в послідовностях даних. LSTM розроблені для вирішення проблеми зникаючих або вибухових градієнтів, які можуть виникати під час навчання традиційних RNN на послідовностях даних. Це робить їх особливо підходящими для задач, що включають послідовні дані, такі як обробка природної мови (NLP), розпізнавання мови та прогнозування часових рядів.

LSTM включають елементи пам'яті, які можуть зберігати інформацію протягом довгих послідовностей. Кожен елемент пам'яті має три основні компоненти: вхідні ворота, ворота забуття і вихідні ворота. Ці ворота допомагають регулювати потік інформації в і з елемента пам'яті.

Input Gate. Вхідні ворота визначають, яка частина нової інформації має бути збережена в елементі пам'яті. Формула для вхідних воріт виглядає так:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i),$$

де σ – сигмоїдна функція активації;

W_i – ваги;

h_{t-1} – прихований стан з попереднього часу;

x_t – вхідний вектор у поточний час;

b_i – зміщення.

Forget Gate.Ворота забуття вирішують, яку інформацію видалити з елемента пам'яті. Формула для воріт забуття виглядає так:

$$f_t = \sigma\big(W_f \cdot [h_{t-1}, x_t] + b_f\big).$$

Output Gate. Вихідні ворота контролюють, скільки вмісту елемента пам'яті повинно бути використано для обчислення прихованого стану. Формула для вихідних воріт виглядає так:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o).$$

Оновлений стан комірки обчислюється за допомогою вхідних і забуттєвих воріт:

$$\tilde{C}t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C).$$

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \widetilde{C_t}.$$

Прихований стан обчислюється з використанням вихідних воріт і нового стану комірки (рис. 2.2):

$$h_t = o_t \cdot \tanh(C_t).$$

—

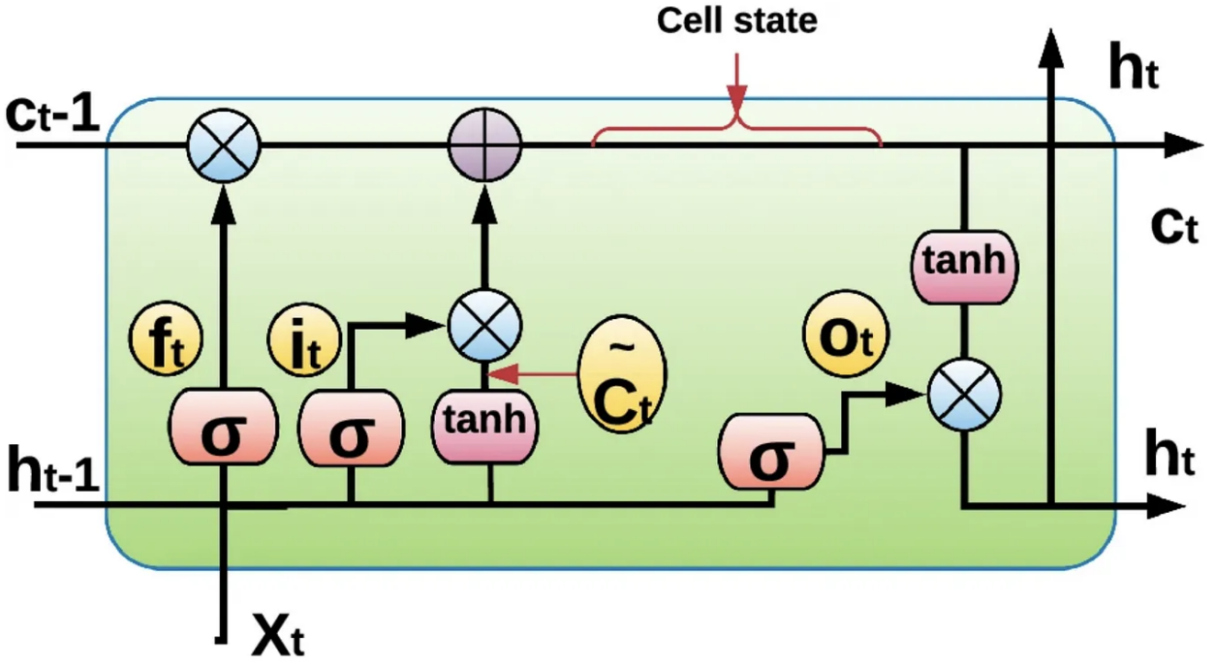


Рисунок 2.2 – Комірка LSTM

Підготовка вхідних даних для LSTM включає організацію даних у формат, який модель LSTM може ефективно обробляти. Основні кроки включають.

1. Формування послідовностей: організуйте вхідні дані у послідовності фіксованої довжини. Наприклад, у контексті часових рядів, якщо у вас є щоденні дані, створіть послідовності з декількох днів даних як одну вхідну послідовність.
2. Переформатування даних у тривимірний формат: (зразки, часові кроки, ознаки).

Time Steps. Часові кроки відносяться до окремих точок у послідовності. У часових рядах або послідовних даних кожен елемент або спостереження в конкретний момент розглядається як часовий крок. Наприклад, якщо ви працюєте з часовим рядом цін на акції і передбачаєте ціну на наступний день

на основі попередніх п'яти днів, кожен день у цих п'яти днях представляє собою часовий крок.

Sequence Data. Послідовні дані, в контексті LSTM, – це набір даних, що структурований у послідовному порядку. Це можуть бути будь-які дані, що впорядковані і залежать від своїх минулих значень для прогнозування майбутніх значень. Це можуть бути часові ряди, текст природної мови, послідовності ДНК або будь-який впорядкований набір даних, де порядок має значення.

Висновки до розділу 2

У розділі 2 було детально розглянуто аналіз часових рядів, їх моделювання та обробку даних. Основні положення включають поняття часового ряду та його видів, адитивні та мультиплікативні моделі, а також компоненти часових рядів. Ми визначили, що часові ряди можна розкласти на трендовий, сезонний та циклічний компоненти, що дозволяє краще розуміти і прогнозувати їх поведінку.

Також було розглянуто міри точності прогнозів, такі як MSE, RMSE, MAD, RMSPE та MAPE, які дозволяють оцінювати точність моделей та порівнювати їх між собою. Було зазначено важливість роботи з пропущеними даними та методи їх заповнення, що є критичним для точності аналізу.

Нормалізація та масштабування даних виявились необхідними для підвищення ефективності алгоритмів машинного навчання, зокрема для тих, що спираються на метрики відстані. Основні методи, такі як Min-Max масштабування, Z-Score нормалізація та Robust масштабування, допомагають зменшити вплив різних масштабів даних на результат моделювання.

Статистичні тести, включаючи тест Дікі-Фуллера (ADF) та тест Квятковського-Філліпса-Шмідта-Шина (KPSS), є ключовими для перевірки

стаціонарності часових рядів. Автокореляційна функція (ACF) та часткова автокореляційна функція (PACF) використовуються для ідентифікації моделей ARIMA, які є основними для прогнозування часових рядів.

Нарешті, було розглянуто сучасні методи прогнозування, такі як моделі ARIMA та LSTM. Модель ARIMA поєднує компоненти авторегресії, інтеграції та ковзного середнього для аналізу та прогнозування часових рядів. LSTM, тип рекурентних нейронних мереж, забезпечує роботу з довгими послідовностями даних та вирішує проблеми зникаючих градієнтів.

У цілому, розділ 2 надав комплексне розуміння підходів до аналізу часових рядів, що є фундаментальним для точного моделювання та прогнозування у різних галузях.

РОЗДІЛ 3 ОГЛЯД ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТА АНАЛІЗ ОТРИМАНИХ РЕЗУЛЬТАТІВ

3.1 Обґрунтування вибору програмного забезпечення

Python є мовою вибору для науки про дані завдяки своїй простоті, читабельності та обширній екосистемі бібліотек. Його універсальність прискорює розробку та спрощує складні завдання маніпулювання даними та машинного навчання. Синтаксис Python сприяє написанню та розумінню складних моделей, що робить його галузевим стандартом для застосунків штучного інтелекту та машинного навчання.

Для підготовки даних бібліотеки Python надають потужні інструменти для очищення та маніпулювання даними:

1. pandas: надає потужні структури даних, такі як DataFrame, які спрощують очищення та маніпулювання даними. Він необхідний для роботи з табличними даними та виконання таких операцій, як об'єднання, фільтрація та агрегація даних.
2. numpy: підтримує великі багатовимірні масиви та матриці, а також пропонує математичні функції для роботи з цими масивами. Він є важливим для числових обчислень і формує основу для багатьох інших бібліотек.
3. scikit-learn: пропонує ефективні інструменти для інтелектуального аналізу даних та аналізу даних, включаючи утиліти попередньої обробки, які є важливими для підготовки даних до моделей машинного навчання. Він включає функції для масштабування, кодування та розділення даних, які є фундаментальними кроками в процесі підготовки даних.

Для побудови та навчання моделей машинного навчання обширна підтримка бібліотек Python є незамінною:

1. TensorFlow: відкритий фреймворк глибокого навчання, який надає всеосяжну, гнучку екосистему інструментів і бібліотек для машинного навчання. Він дозволяє будувати та навчати широкий спектр моделей з високою ефективністю та масштабованістю.
2. Keras: високорівневий API нейронних мереж, що працює поверх TensorFlow, розроблений для зручності користування та модульності. Він спрощує процес створення та навчання складних моделей нейронних мереж, надаючи чистий і простий у використанні інтерфейс.
3. statsmodels: надає класи та функції для оцінки багатьох різних статистичних моделей, а також для проведення статистичних тестів і статистичного дослідження даних. Він особливо корисний для аналізу та прогнозування часових рядів.

3.2 Опис використаних даних та алгоритму роботи програми

3.2.1 Опис даних

Дані, що використовуються в, завантажуються з файлу CSV з назвою AAPL.csv. Набір даних містить історичні дані про ціни акцій Apple Inc. (AAPL).

Типова структура даних.

1. Date: дата запису даних.
2. Open: ціна відкриття акції на цю дату.
3. High: найвища ціна акції на цю дату.
4. Low: найнижча ціна акції на цю дату.
5. Close: ціна закриття акції на цю дату.
6. Adj Close: скоригована ціна закриття акції на цю дату, з урахуванням дивідендів і дроблення акцій.

7. Volume: кількість акцій, проданих на цю дату.

	Date	Open	High	Low	Close	Volume	Adj Close
0	2008-10-14	116.26	116.40	103.14	104.08	70749800	104.08
1	2008-10-13	104.55	110.53	101.02	110.26	54967000	110.26
2	2008-10-10	85.70	100.00	85.00	96.80	79260700	96.80
3	2008-10-09	93.35	95.80	86.60	88.74	57763700	88.74
4	2008-10-08	85.91	96.33	85.68	89.79	78847900	89.79
...	...	...	...	...	...	...	...
6076	1984-09-13	27.50	27.62	27.50	27.50	7429600	3.14
6077	1984-09-12	26.87	27.00	26.12	26.12	4773600	2.98
6078	1984-09-11	26.62	27.37	26.62	26.87	5444000	3.07
6079	1984-09-10	26.50	26.62	25.87	26.37	2346400	3.01
6080	1984-09-07	26.50	26.87	26.25	26.50	2981600	3.02

Рисунок 3.1 – Приклад датасету

3.2.2 Фільтрація даних

З'ясуємо, чи є пропуски в даних та отримати загальну інформацію про типи даних і кількість записів.

1. Використовуємо метод info(), щоб отримати загальну інформацію про DataFrame, включаючи кількість ненульових значень у кожному стовпчику та типи даних.
2. Підраховуємо кількість пропущених значень у кожному стовпчику за допомогою методу isnull().sum().

```
RangeIndex: 6081 entries, 0 to 6080
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Date       6081 non-null   object
 1   Open       6081 non-null   float64
 2   High       6081 non-null   float64
 3   Low        6081 non-null   float64
 4   Close      6081 non-null   float64
 5   Volume     6081 non-null   int64
 6   Adj Close  6081 non-null   float64
dtypes: float64(5), int64(1), object(1)
memory usage: 332.7+ KB
None
Missing Values:
Date         0
Open         0
High         0
Low          0
Close        0
Volume       0
Adj Close    0
dtype: int64
```

Рисунок 3.2 – Результат перевірки даних на пропуски

Опис результату: Інформація показує, що дані містять 6081 записів, кожен з яких має 7 стовпчиків: 'Date', 'Open', 'High', 'Low', 'Close', 'Volume', 'Adj Close'. Всі стовпчики не мають пропущених значень. Типи даних: 'Date' - object, інші стовпчики – числові (float64, int64). Отже дані повні та готові до аналізу без необхідності обробки пропущених значень, типи даних відповідають очікуваним для фінансових даних.

Підготуємо дані до візуалізації, оцінимо поведінку даних з часом та визначити можливі аномалії чи тренди. Алгоритм дії.

1. Перетворюємо стовпчик 'Date' у формат datetime для зручності роботи з часовими рядами.
2. Візуалізуємо оригінальні ціни, щоб побачити тренди та сезонність.

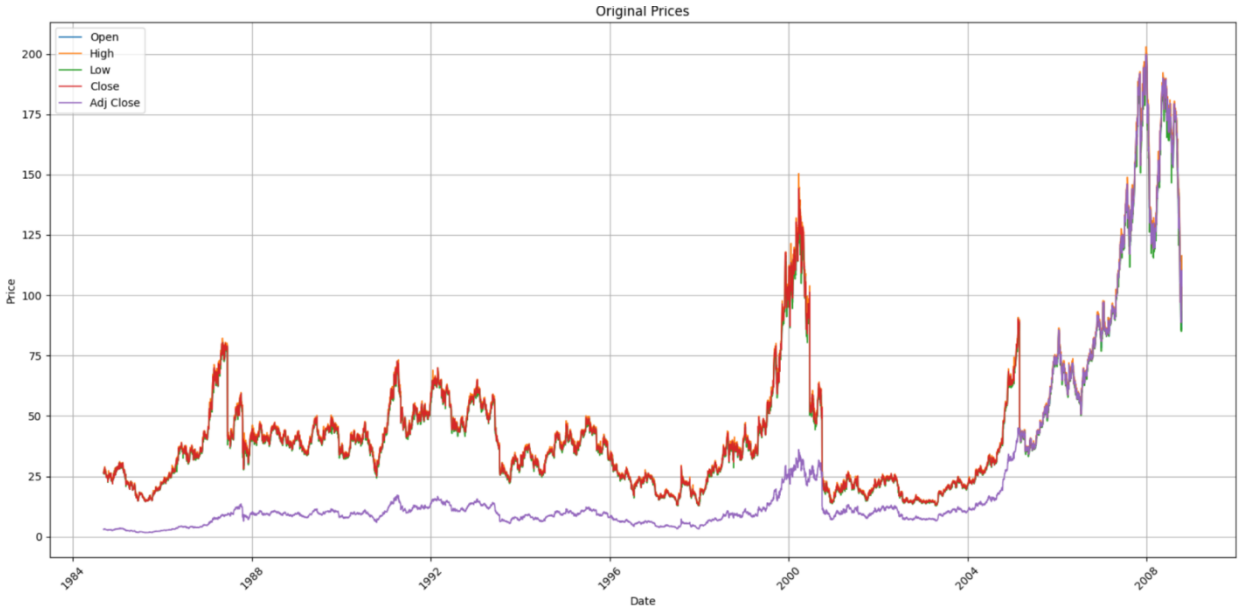


Рисунок 3.3 – Графік історичних цін дата сету

Опис результату: лінійний графік показує історичні дані цін акцій Apple. Різні кольори представляють різні ціни: 'Open', 'High', 'Low', 'Close', 'Adj Close'. Графік показує довгостроковий тренд зростання цін акцій Apple, де видно значні піки і падіння, які можуть бути пов'язані з економічними подіями або корпоративними новинами.

Переходимо до нормалізації даних, нам потрібно звести всі значення до одного масштабу для забезпечення рівних умов при подальшій обробці та аналізі даних.

1. Ініціалізуємо MinMaxScaler.
2. Нормалізуємо вибрані стовпчики до діапазону [0, 1] за допомогою методу fit_transform().

І перевіряємо, як змінився розподіл даних після нормалізації, для цього створюємо гістограми для нормалізованих стовпчиків, щоб оцінити розподіл даних.

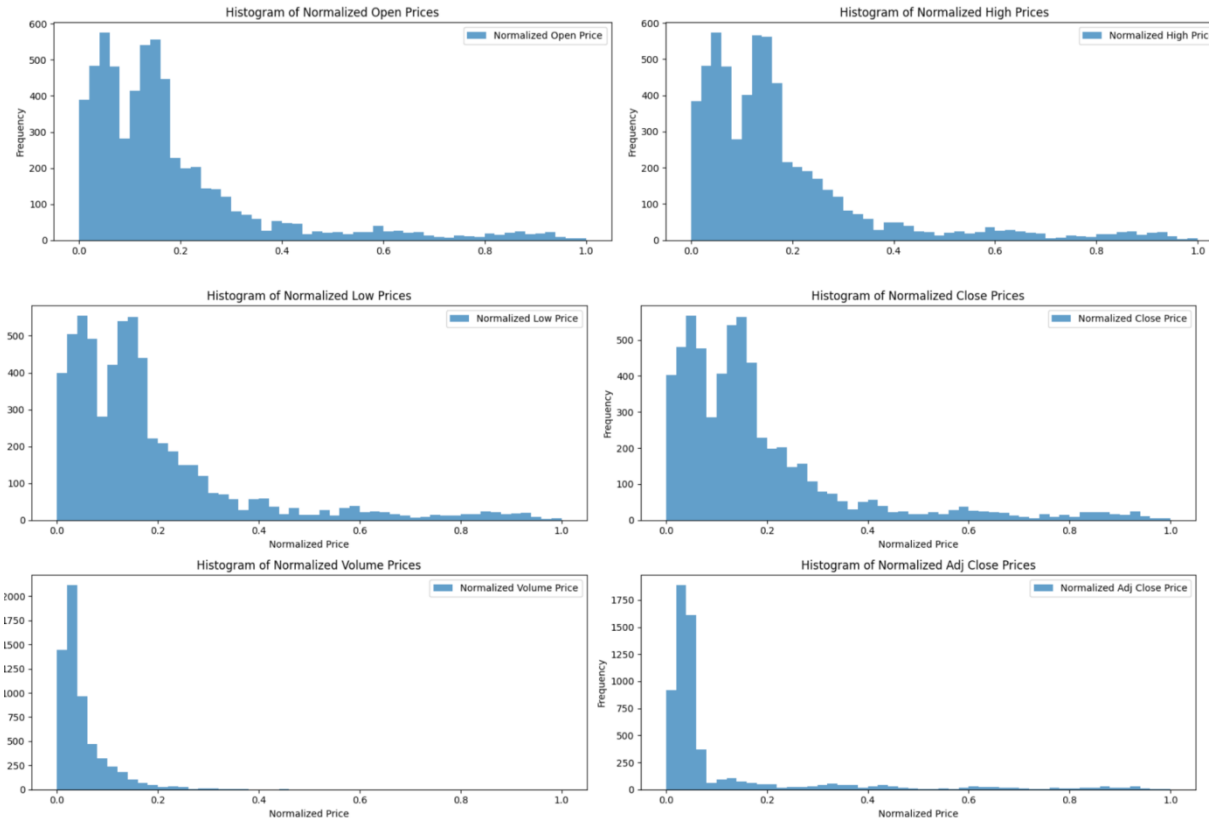


Рисунок 3.4 – Гістограми розподілу нормалізованих цін 'Open', 'High', 'Low', 'Close', 'Volume', 'Adj Close'

Опис результату: Всі значення були нормалізовані до діапазону [0, 1]. Ці гістограми дозволяють оцінити розподіл нормалізованих даних. Нормалізовані ціни мають різний розподіл, що може вказувати на різні характерні риси цих даних:

- Normalized Open Prices: більшість значень зосереджена ближче до нуля, що вказує на те, що більшість відкритих цін були нижчими.
- Normalized High Prices: схоже розподіл, що вказує на максимальні ціни, які не виходили за певні межі.
- Normalized Low Prices: низькі ціни мають подібний розподіл, зосереджуючись ближче до нижньої межі.
- Normalized Close Prices: розподіл закритих цін схожий на розподіл відкритих цін.

– Normalized Volume: обсяги мають значний пік біля нуля, що вказує на високий обсяг торгів у певні дні.
– Normalized Adj Close Prices: розподіл подібний до закритих цін, що логічно.

Фільтруємо дані і візуалізуємо результати для розуміння успішності виконаного завдання.

1. Обчислюємо перший (Q1) та третій (Q3) квартили для кожного стовпчика.
2. Обчислюємо інтерквартильний розмах (IQR).
3. Фільтруємо дані, видаляючи значення, що знаходяться за межами діапазону [Q1 - 1.5*IQR, Q3 + 1.5*IQR].
4. Створюємо діаграми розмаху для кожного стовпчика, щоб візуально оцінити видалення викидів.

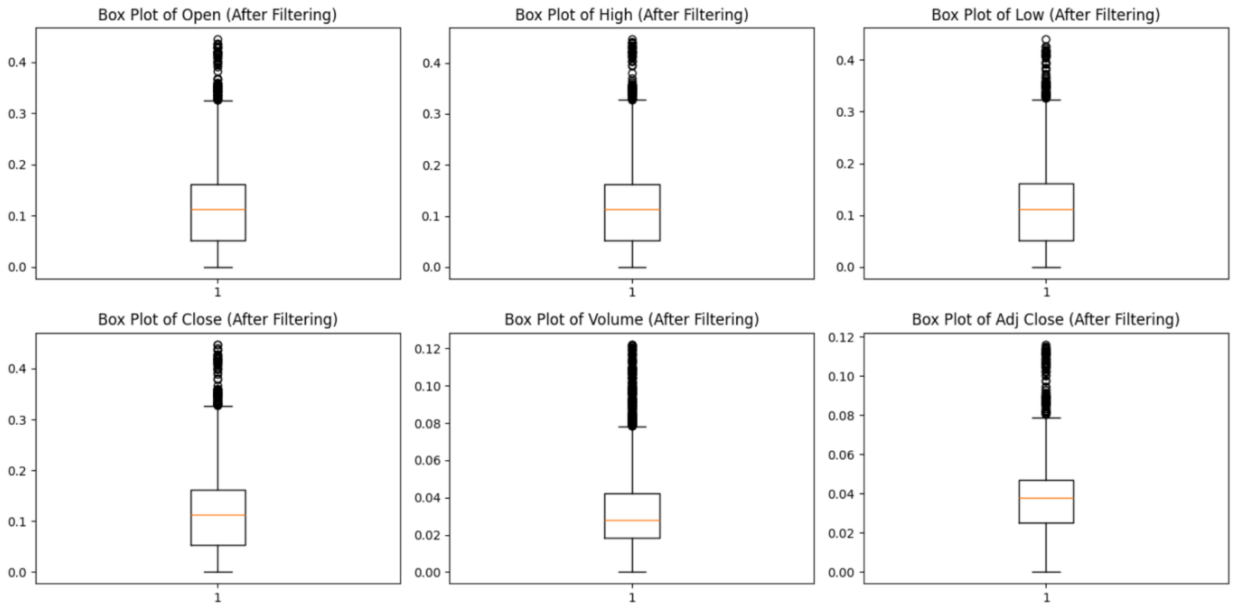


Рисунок 3.5 – Розподіл даних і викиди після фільтрації

Опис результату: боксплоти для кожного показника ('Open', 'High', 'Low', 'Close', 'Volume', 'Adj Close') після фільтрації показують розподіл даних і викиди. На кожному графіку відображені середні значення, межі квартилів (Q1, Q3), міжквартильний розмах (IQR), а також викиди. Після фільтрації, більшість даних зосереджені всередині IQR, викиди поза межами верхніх і нижніх вусів все ще присутні, але їх кількість зменшена, фільтрація допомогла зменшити кількість аномальних значень.

Наступний крок – перевірка на стаціонарність. Для цього проводимо відповідні тести: Діккі-Фуллера (ADF), KPSS, будуємо графіки автокореляційної функції (ACF) та часткової автокореляційної функції (PACF) для кожного стовпчика.

ADF визначатиме, чи є часова серія стаціонарною (не має одиничного кореня). Нульова гіпотеза (H0) – часова серія має одиничний корінь (нестабільна, нестаціонарна).

KPSS також визначить, чи є часова серія стаціонарною. Нульова гіпотеза (H0) – часова серія стаціонарна.

Алгоритм дії.

1. Виконуємо тест Аугментованої Діккі-Фуллера (ADF) для перевірки стаціонарності ряду.
2. Виводимо статистику тесту та p-значення.
3. Виконуємо тести ADF та KPSS для кожного стовпчика, щоб мати більш повну картину стаціонарності.
4. Будуємо графіки автокореляційної функції (ACF) та часткової автокореляційної функції (PACF) для кожного стовпчика.

ADF Statistic: -3.913009040229648

p-value: 0.001941166553474393

Open - ADF Test

ADF Statistic: -3.8819839461622387

p-value: 0.0021715095125103847

Critical Values: {'1%': -3.431724255247682, '5%': -2.862147265506281, '10%': -2.5670932493178116}

Open - KPSS Test

KPSS Statistic: 2.1192895796664724

p-value: 0.01

Critical Values: {'10%': 0.347, '5%': 0.463, '2.5%': 0.574, '1%': 0.739}

High - ADF Test

ADF Statistic: -4.030957555725225

p-value: 0.0012568074560584746

Critical Values: {'1%': -3.431723966504064, '5%': -2.8621471379437025, '10%': -2.5670931814107942}

High - KPSS Test

KPSS Statistic: 2.0861627277260872

p-value: 0.01

Critical Values: {'10%': 0.347, '5%': 0.463, '2.5%': 0.574, '1%': 0.739}

Low - ADF Test

ADF Statistic: -3.9313910533253824

p-value: 0.0018155953961649215

Critical Values: {'1%': -3.4317245441126865, '5%': -2.862147393122474, '10%': -2.567093317253372}

Low - KPSS Test

KPSS Statistic: 2.150282099399603

p-value: 0.01

Critical Values: {'10%': 0.347, '5%': 0.463, '2.5%': 0.574, '1%': 0.739}

Close - ADF Test

ADF Statistic: -3.913009040229648

p-value: 0.001941166553474393

Critical Values: {'1%': -3.431720223848477, '5%': -2.8621454844937517, '10%': -2.5670923012088482}

Close - KPSS Test

KPSS Statistic: 2.108277360059734

p-value: 0.01

Critical Values: {'10%': 0.347, '5%': 0.463, '2.5%': 0.574, '1%': 0.739}

Volume - ADF Test

ADF Statistic: -5.94249849623888

p-value: 2.243717954953032e-07

Critical Values: {'1%': -3.431723966504064, '5%': -2.8621471379437025, '10%': -2.5670931814107942}

Volume - KPSS Test

KPSS Statistic: 5.172922876417975

p-value: 0.01

Critical Values: {'10%': 0.347, '5%': 0.463, '2.5%': 0.574, '1%': 0.739}

Adj Close - ADF Test

ADF Statistic: -3.478907187899742

p-value: 0.008545617183961114

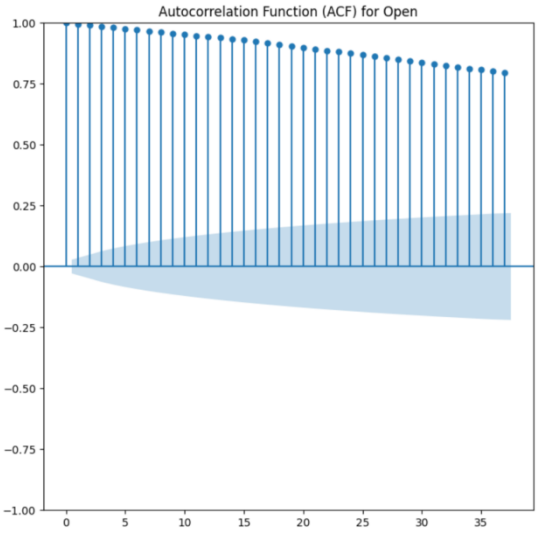
Critical Values: {'1%': -3.431722812741928, '5%': -2.862146628228856, '10%': -2.567092910067799}

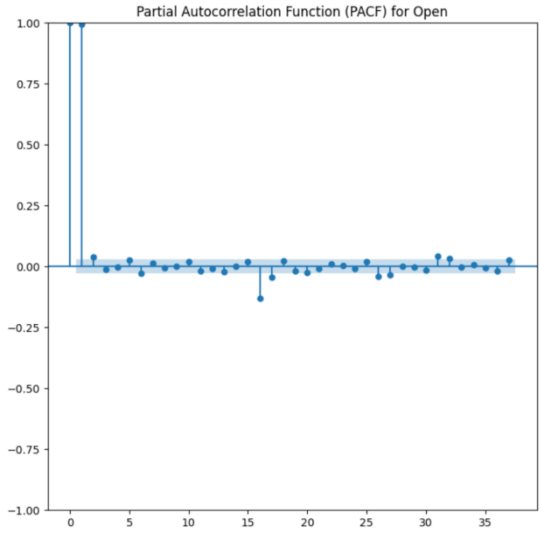
Adj Close - KPSS Test

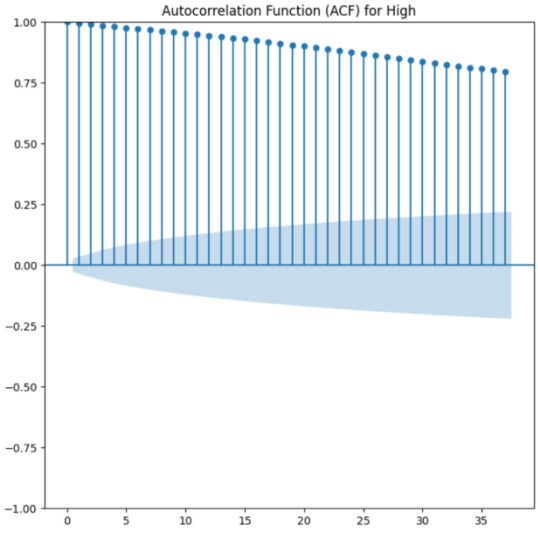
KPSS Statistic: 1.8871367613238275

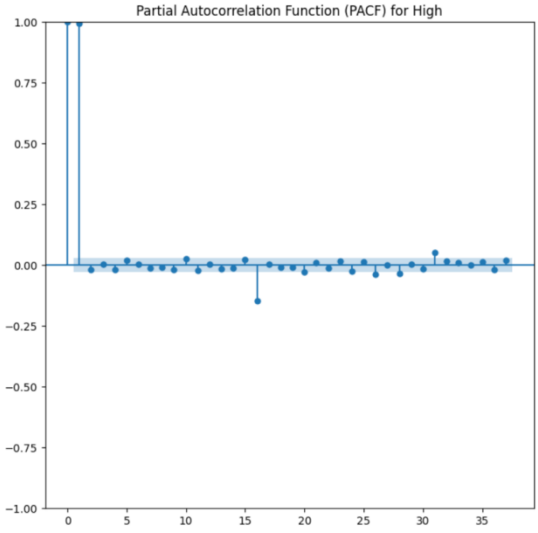
p-value: 0.01

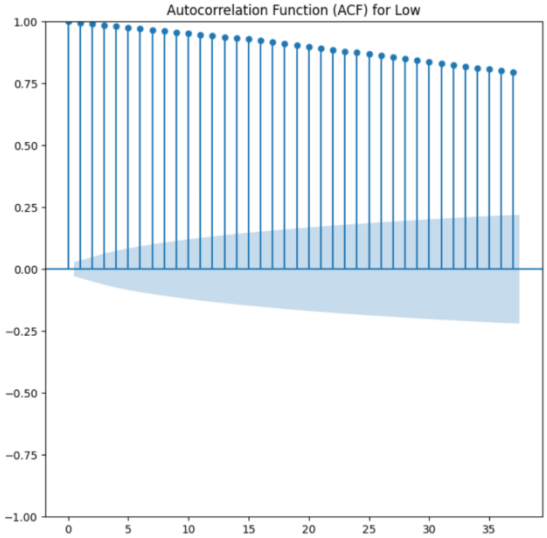
Critical Values: {'10%': 0.347, '5%': 0.463, '2.5%': 0.574, '1%': 0.739}

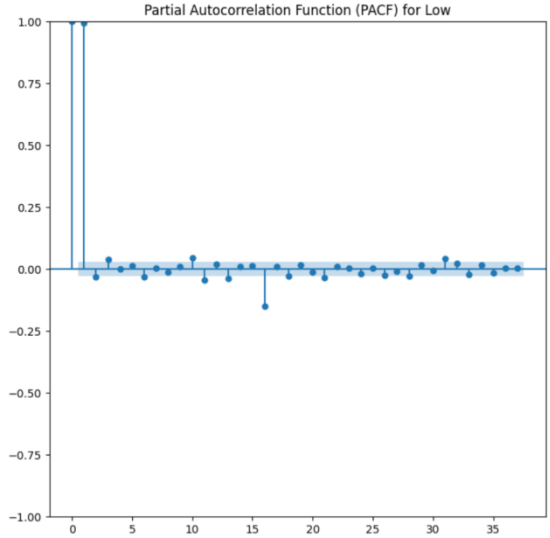

Autocorrelation Function (ACF) for Open
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30
35


Partial Autocorrelation Function (PACF) for Open
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30
35


Autocorrelation Function (ACF) for High
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30
35


Partial Autocorrelation Function (PACF) for High
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30
35


Autocorrelation Function (ACF) for Low
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30
35


Partial Autocorrelation Function (PACF) for Low
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30
35

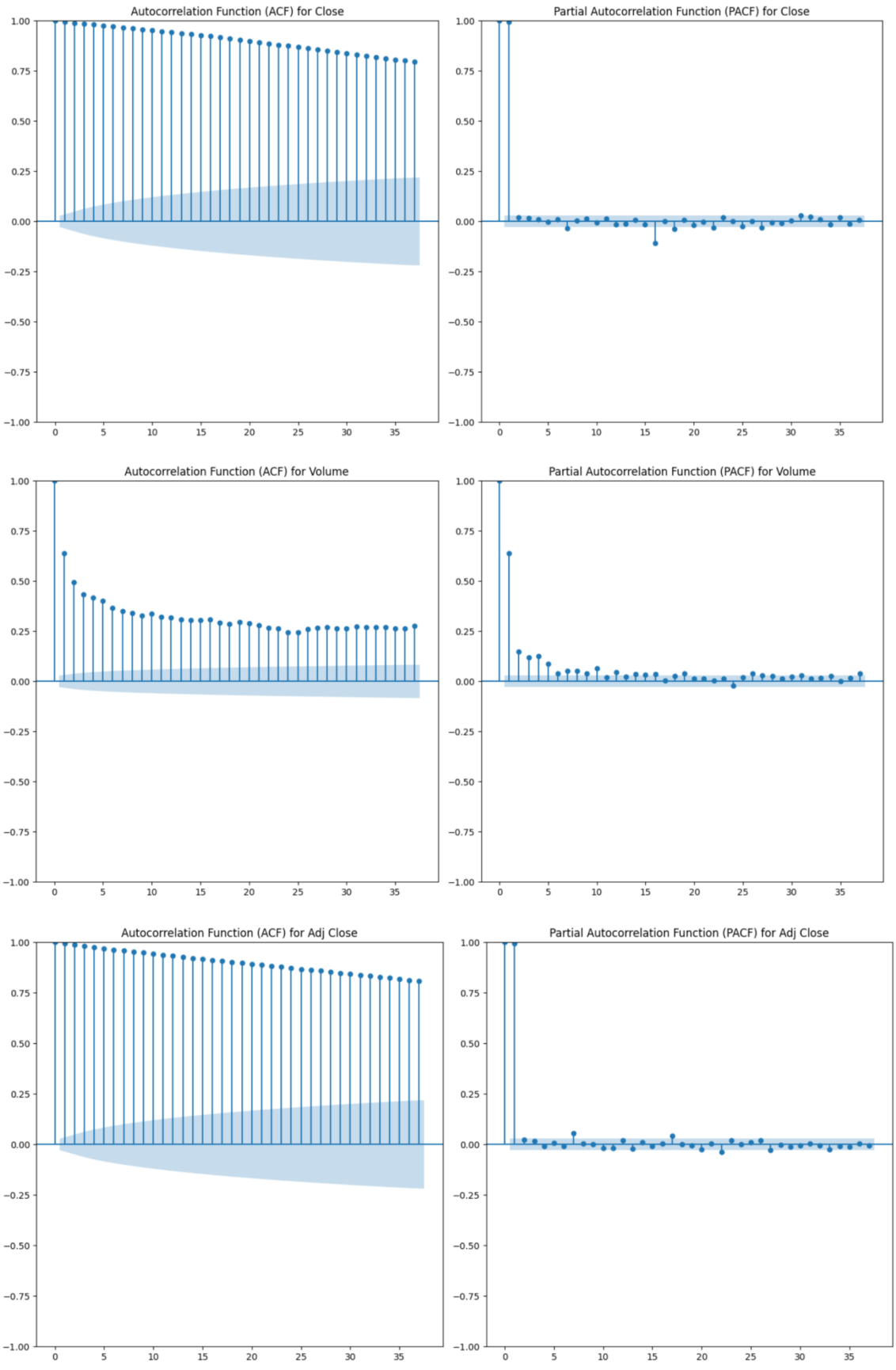


Рисунок 3.6 – Графіки автокореляційної функції (ACF) та часткової автокореляційної функції (PACF)

Опис результату:

– ACF – високі значення на великих лагах вказують на сильну автокореляцію в даних, що може свідчити про тренди;
– PACF – високі значення на початкових лагах і зниження на подальших лагах допомагають визначити кількість лагів, що потрібно враховувати в моделях AR.

3.2.3 Модель ARIMA для прогнозування часового ряду

Першочергово завантажуємо історичні дані акцій компанії Apple за період з 1 січня 2020 року до 1 січня 2021 року - для першого тестування і з 1 січня 2003 року до 1 січня 2013 року - для другого . Ці дані включають щоденні ціни відкриття, закриття, найвищу та найнижчу ціну за день, а також обсяги торгів.

Історичні дані закриття цін розподіляються на навчальний та тестовий набори. 80% даних використовується для тренування моделі, а решта 20% – для тестування.

На основі навчального набору даних тренується модель ARIMA з параметрами (1, 2, 1). Після тренування моделі виводяться значення критеріїв Акаіке (AIC) та Байєсівського інформаційного критерію (BIC), які допомагають оцінити якість моделі.

З використанням тренованої моделі ARIMA здійснюється прогнозування цін акцій на період, який відповідає тестовому набору даних. Порівнюються прогнозовані значення з фактичними цінами закриття.

Для оцінки точності прогнозу обчислюємо корінь середньоквадратичної помилки (RMSE) . RMSE дає змогу оцінити середню відстань між прогнозованими та фактичними значеннями.

Для візуалізації результатів будуємо графік, який показує навчальні дані, фактичні ціни закриття з тестового набору та прогнозовані ціни. Це дозволяє візуально оцінити якість прогнозу та ефективність моделі ARIMA у передбаченні цін акцій.

AIC: 963.3646571093428

BIC: 973.259609208987

RMSE: 4.381005773491716

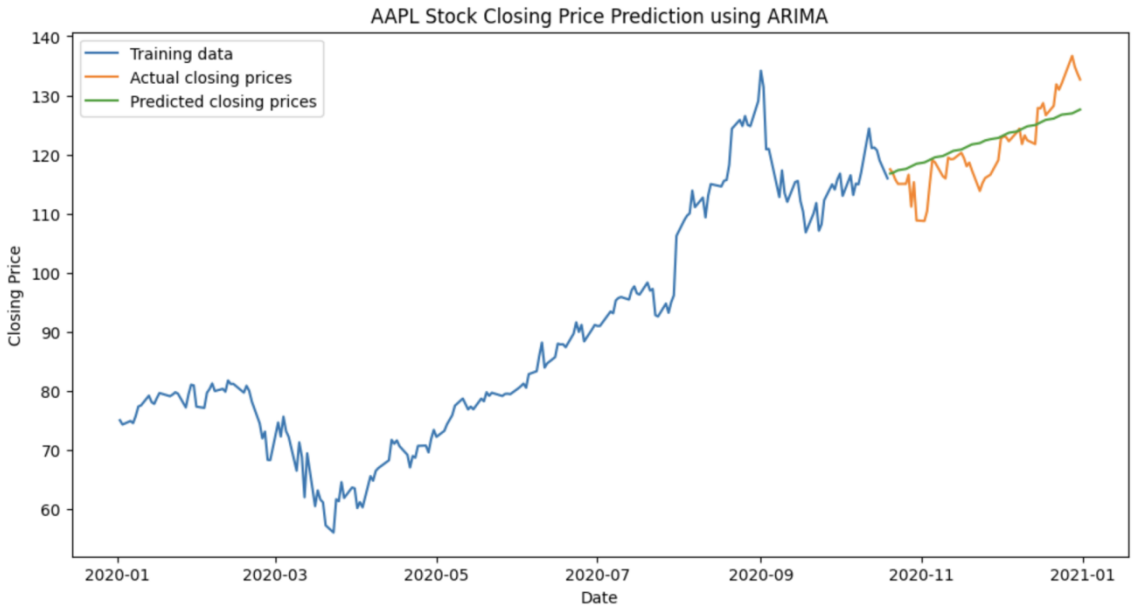


Рисунок 3.7 – Приклад візуалізації прогнозування методом ARIMA

Опис результатів графіка.

1. Період: 2020 – 2021.
2. Синя лінія: навчальні дані (фактичні ціни закриття акцій за цей період).
3. Помаранчева лінія: фактичні ціни закриття у тестовому наборі даних.
4. Зелена лінія: прогнозовані ціни закриття за допомогою моделі ARIMA.

AIC: -3516.938946341991

BIC: -3500.119784172673

RMSE: 3.685301056909807

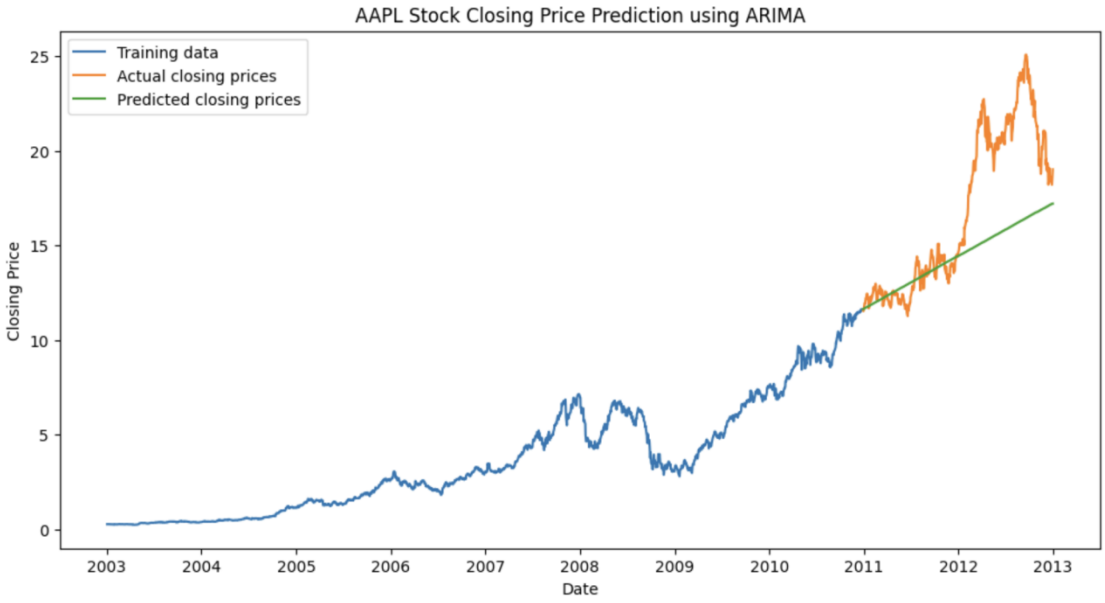


Рисунок 3.8 – Приклад візуалізації прогнозування методом ARIMA

Опис результатів графіка:

1. Період: 2020 – 2021.
2. Синя лінія: навчальні дані (фактичні ціни закриття акцій за цей період).
3. Помаранчева лінія: фактичні ціни закриття у тестовому наборі даних.
4. Зелена лінія: прогнозовані ціни закриття за допомогою моделі ARIMA.

AIC та BIC:

– AIC (Akaike Information Criterion): вказує на якість моделі. Нижчі значення означають кращу модель з урахуванням кількості параметрів;

– BIC (Bayesian Information Criterion): подібно до AIC, але більш суворо штрафує за кількість параметрів. Нижчі значення вказують на кращу модель;

– RMSE (Root Mean Squared Error): RMSE: міра середньої похибки між фактичними та прогнозованими значеннями. Нижчі значення вказують на меншу похибку.

Опис результатів.

1. Більше даних для навчання.
   1.1. Довша вибірка (2003 – 2013) надає більше даних для навчання моделі, що дозволяє краще зрозуміти довгострокові тренди та сезонність.
   1.2. Більша кількість даних допомагає моделі краще вловлювати загальні макроекономічні фактори, що впливають на ціни акцій.
2. Менша чутливість до короткострокових коливань.
   2.1. Модель ARIMA краще передбачає довгострокові тренди, оскільки вона менш чутлива до короткострокових флуктуацій, які можуть бути випадковими або викликаними зовнішніми факторами.
   2.2. На довшому періоді такі коливання згладжуються, що покращує загальний прогноз.
3. Менше даних для навчання.
   3.1. Коротша вибірка (2020 – 2021) надає менше даних для навчання моделі, що обмежує її здатність вловлювати всі можливі тренди та сезонність.
   3.2. Недостатня кількість даних може призвести до недооцінки важливих факторів, що впливають на ціни акцій.
4. Вплив короткострокових подій.
   4.1. Короткострокові дані більше піддаються впливу випадкових подій, таких як новини, корпоративні оголошення або інші несподівані фактори.
   4.2. Модель ARIMA не завжди може врахувати ці події, що призводить до відхилень у прогнозах.

3.2.4 Модель LSTM для прогнозування часового ряду

Дані завантажуються з CSV файлу, після чого відбувається їх масштабування за допомогою MinMaxScaler: масштабування здійснюється для стовпців 1-4, 4 окремо, і 5+. Це робиться для того, щоб всі дані були в одному діапазоні (0-1), що полегшує навчання моделі.

Далі створюються послідовності даних для LSTM.

1. Встановлюється розмір вікна (window_size) в 50.
2. Для кожного кроку створюються вектори X (послідовності з 50 попередніх значень) і Y (цільове значення).
3. Поділ на навчальний і тестовий набори
4. Дані діляться на навчальний та тестовий набори в співвідношенні 90% до 10% без перемішування (shuffle=False).

Побудова і тренування моделі LSTM. Модель створюється за допомогою Sequential і додаються два шари LSTM (128 і 64 нейронів відповідно) – повнозв'язний шар з 16 нейронами з активацією ReLU і вихідний шар з одним нейроном з лінійною активацією.

Модель компілюється з оптимізатором RMSprop і функцією втрат MSE, після чого тренується на навчальних даних протягом 10 епох з розміром батчу 64. Після тренування модель оцінюється на тестових даних.

1. Виводиться кількість зразків в тестовому наборі.
2. Виводиться середньоквадратична похибка (MSE) для тестового набору.

Прогнозування і зворотне масштабування. Модель використовується для прогнозування на тестовому наборі.

1. Прогнозовані значення та цільові значення повертаються до початкового масштабу за допомогою inverse_transform.

2. Обчислюються метрики MSE і MAE для оцінки точності прогнозу.
3. Візуалізація результатів
4. Прогнозовані значення та фактичні ціни акцій відображаються на графіку.

Mean Squared Error: 9.240888715386921

Mean Absolute Error: 2.6151085161806926

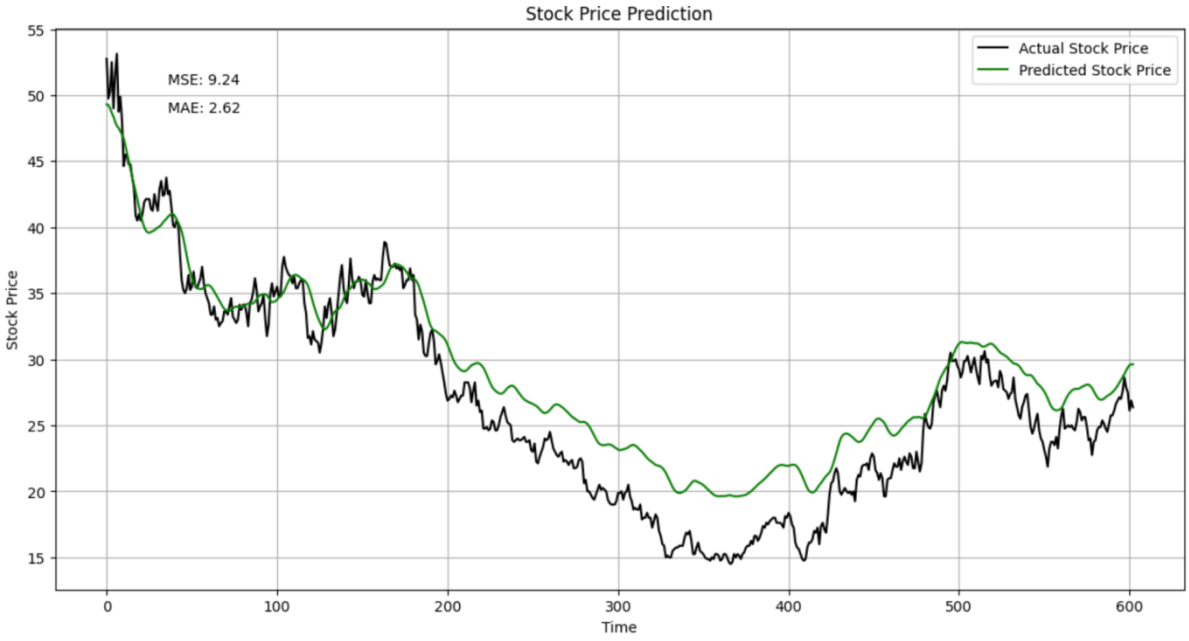


Рисунок 3.9 – Приклад візуалізації прогнозування методом LSTM

Інтерпретація графіку:

– Чорна лінія: відображає фактичні ціни закриття акцій (Actual Stock Price) за певний період часу.
– Зелена лінія: відображає прогнозовані ціни закриття акцій (Predicted Stock Price) за допомогою моделі LSTM.
– Середньоквадратична похибка (MSE) вимірює середнє квадратичне відхилення між фактичними і прогнозованими значеннями. Значення 9.24 вказує на те, що модель має певну похибку, але загалом дає розумні прогнози.
– Середня абсолютна похибка (MAE) вимірює середню абсолютну різницю між фактичними і прогнозованими значеннями. Значення

2.62 вказує на середню похибку у прогнозах, що є досить прийнятним результатом.

Аналіз точності моделі.

1. Відповідність прогнозів фактичним даним:
    1.1. Зелена лінія (прогнозовані ціни) здебільшого слідує за чорною лінією (фактичні ціни), що свідчить про те, що модель вловлює загальні тренди та зміни в цінах акцій.
    1.2. Є певні моменти, де прогнозовані ціни відхиляються від фактичних, особливо в періоди різких змін цін.
2. Короткострокові коливання:
    2.1. Модель LSTM добре вловлює загальні тренди, але може мати труднощі з точним прогнозуванням короткострокових коливань та різких змін цін.
    2.2. Зелена лінія згладжує деякі різкі коливання, що може свідчити про певний рівень згладжування у моделі.
3. Довгострокові тренди:
    3.1. Модель ефективно передбачає довгострокові тренди, як видно з того, як зелена лінія слідує за загальним напрямком чорної лінії. Це свідчить про те, що модель LSTM добре навчається на довгострокових залежностях у часових рядах.

3.2.5 Комбінований метод LSTM та CNN

Починаємо з визначення вікна та періоду: встановлюється розмір вікна (window_size) в 50, що означає, що для кожного прогнозу використовується послідовність з 50 попередніх днів.

Далі створюються вектори X (послідовності з 50 попередніх значень) і Y (цільове значення на наступний день).Дані діляться на навчальний та тестовий набори в співвідношенні 90% до 10% без перемішування (shuffle=False).

Побудова і тренування моделі CNN-LSTM. Модель створюється за допомогою Sequential і додаються наступні шари:

- TimeDistributed Conv1D шари – три шари з конволюційними нейронними мережами для вилучення просторових особливостей з послідовностей;
- MaxPooling1D – шари для зменшення розмірності та уникнення перенавчання;
- Flatten – шар для перетворення 3D даних у 2D;
- Bidirectional LSTM –два двонаправлених LSTM шари для вилучення часових залежностей;
- Dropout – шари для запобігання перенавчанню;
- Dense – вихідний шар з одним нейроном з лінійною активацією.

Модель компілюється з оптимізатором RMSprop і функцією втрат MSE, після чого тренується на навчальних даних протягом 10 епох з розміром батчу 64.

Оцінка моделі і прогнозування. Після тренування модель оцінюється на тестових даних, виводяться метрики середньоквадратичної похибки (MSE) та середньої абсолютної похибки (MAE), виводяться прогнозовані та фактичні значення для тестового набору.

Модель використовується для прогнозування на тестовому наборі — прогнозовані значення та цільові значення повертаються до початкового масштабу, обчислюються метрики MSE та MAE для оцінки точності прогнозу.

Візуалізація і аналіз результатів

Mean Squared Error: 4.215895714378176

Mean Absolute Error: 1.5829427249079713

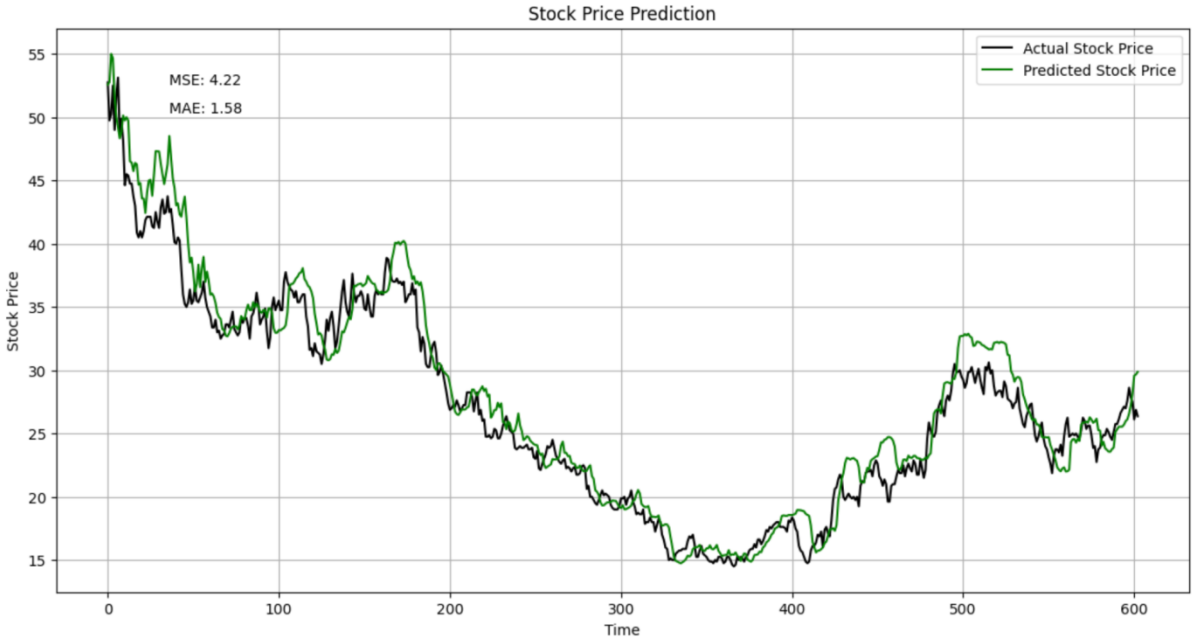


Рисунок 3.10 – Приклад візуалізації прогнозування комбінованим методом LSTM та CNN

Інтерпретація графіку.

1. Чорна лінія відображає фактичні ціни закриття акцій (Actual Stock Price) за певний період часу.
2. Зелена лінія відображає прогнозовані ціни закриття акцій (Predicted Stock Price) за допомогою моделі CNN-LSTM.
3. MSE (Mean Squared Error): середньоквадратична похибка (MSE) вимірює середнє квадратичне відхилення між фактичними і прогнозованими значеннями. Значення 4.22 вказує на те, що модель має відносно низьку похибку, що свідчить про хорошу точність прогнозів.
4. MAE (Mean Absolute Error): середня абсолютна похибка (MAE) вимірює середню абсолютну різницю між фактичними і прогнозованими значеннями. Значення 1.58 вказує на невелику похибку у прогнозах, що є позитивним показником точності моделі.

Аналіз точності моделі:

1. Відповідність прогнозів фактичним даним:
    1.1. Зелена лінія (прогнозовані ціни) досить точно слідує за чорною лінією (фактичні ціни), що свідчить про те, що модель добре вловлює загальні тренди та зміни в цінах акцій.
    1.2. Є певні моменти, де прогнозовані ціни трохи відхиляються від фактичних, але в цілому модель показує високу точність.
2. Короткострокові коливання:
    2.1. Модель LSTM добре вловлює загальні тренди, але може мати труднощі з точним прогнозуванням короткострокових коливань та різких змін цін.
    2.2. Зелена лінія згладжує деякі різкі коливання, що може свідчити про певний рівень згладжування у моделі.
3. Довгострокові тренди:
    3.1. Модель ефективно передбачає довгострокові тренди, як видно з того, як зелена лінія слідує за загальним напрямком чорної лінії.
    3.2. Це свідчить про те, що модель CNN-LSTM добре навчається на довгострокових залежностях у часових рядах.

3.2.6 Комбінований метод LSTM та CNN з з технічними індикаторами.

Додаткові маніпуляції з даними:

- Додавання технічних індикаторів: додаються індикатори Bollinger Bands, MACD, RSI та Ichimoku:
- BollingerBands для середньої, верхньої та нижньої меж.
- MACD для виявлення трендів.
- RSI для оцінки моментуму.
- IchimokuIndicator для аналізу підтримки та опору.
- Пропущені значення заповнюються нулями (fillna(0)).

Вибір характеристик:

- використовується лінійна регресія для визначення важливості характеристик;
- відображається важливість кожної характеристики у вигляді бар-чарту.

Підготовка даних для моделі LSTM.

1. Визначення розміру вікна: встановлюється розмір вікна (window_size) в 50.
2. Створення послідовностей: створюються вектори X (послідовності з 50 попередніх значень) і Y (цільове значення "Close" на наступний день).
3. Поділ на навчальний і тестовий набори: дані діляться на навчальний та тестовий набори в співвідношенні 90% до 10% без перемішування (shuffle=False).
4. Ресайзинг даних: змінюється розмірність даних для відповідності вимогам моделі LSTM.

Побудова і тренування моделі. Модель створюється за допомогою Sequential і додаються наступні шари:

- TimeDistributed Conv1D шари: шари з конволюційними нейронними мережами для вилучення просторових особливостей з послідовностей.
- MaxPooling1D: шари для зменшення розмірності.
- Flatten: шар для перетворення 3D даних у 2D.
- Bidirectional LSTM: два двонаправлених LSTM шари для вилучення часових залежностей.
- Dropout: шари для запобігання перенавчанню.
- Dense: вихідний шар з одним нейроном з лінійною активацією.

Модель компілюється з оптимізатором Adam і функцією втрат MSE, після чого тренується на навчальних даних протягом 5 епох.

Оцінка моделі і прогнозування. Після тренування модель оцінюється на тестових даних:

- виводяться метрики середньоквадратичної похибки (MSE) та середньої абсолютної похибки (MAE);
- виводяться прогнозовані та фактичні значення для тестового набору.

Модель використовується для прогнозування на тестовому наборі:

- прогнозовані значення та цільові значення повертаються до початкового масштабу.
- обчислюються метрики MSE та MAE для оцінки точності прогнозу.

Візуалізуємо і аналізуємо результати.

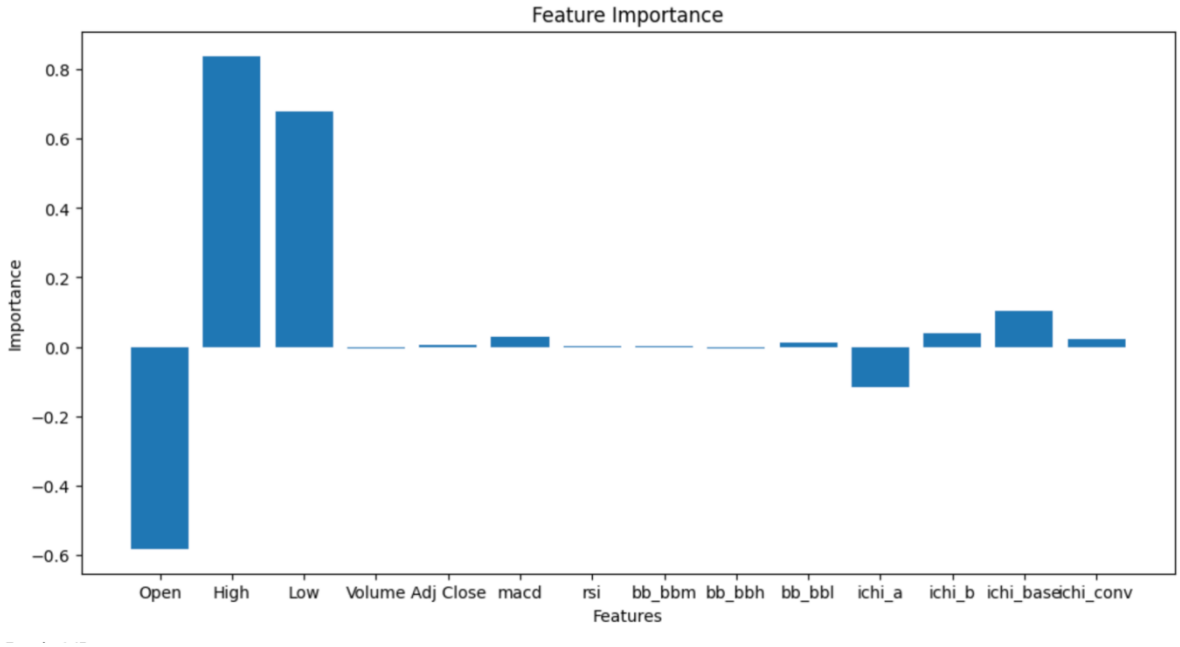


Рисунок 3.11 – Графік важливості характеристик

Перший графік показує важливість різних характеристик (features) для прогнозування цін акцій:

– "High" і "Low" мають найбільшу позитивну важливість;

– такі індикатори, як "MACD", "RSI", "Bollinger Bands" (bb_bbm, bb_bbh, bb_bbl) та компоненти Ichimoku ("ichi_a", "ichi_b", "ichi_base", "ichi_conv") мають незначну або негативну важливість;

– результати лінійної регресії показують, що найважливішими характеристиками є "High" і "Low". Це означає, що ціни відкриття та закриття менш впливають на прогноз, ніж максимальні та мінімальні ціни.

Layer (type)	Output Shape	Param #
time_distributed_10 (TimeDistributed)	(None, 50, 7, 64)	128
time_distributed_11 (TimeDistributed)	(None, 50, 3, 64)	0
time_distributed_12 (TimeDistributed)	(None, 50, 192)	0
bidirectional_4 (Bidirectional)	(None, 50, 400)	628,800
dropout_4 (Dropout)	(None, 50, 400)	0
bidirectional_5 (Bidirectional)	(None, 400)	961,600
dropout_5 (Dropout)	(None, 400)	0
dense_2 (Dense)	(None, 1)	401

Total params: 4,772,789 (18.21 MB)
Trainable params: 1,590,929 (6.07 MB)
Non-trainable params: 0 (0.00 B)
Optimizer params: 3,181,860 (12.14 MB)

Рисунок 3.12 – Структура моделі нейронної мережі

TimeDistributed Convolutional Layers – ці шари вилучають просторові особливості з послідовності даних.

Bidirectional LSTM Layers – захоплюють часові залежності з обох напрямків (вперед і назад).

Dropout Layers – використовуються для запобігання перенавчанню моделі.

Dense Layer – вихідний шар, який передбачає фінальну цінову точку.

Використання TimeDistributed Convolutional Layers допомагає виявити локальні особливості в кожному часовому вікні, а Bidirectional LSTM захоплює як попередні, так і наступні залежності в даних.

Mean Squared Error: 3.5905928137464933

Mean Absolute Error: 1.6354809090154094

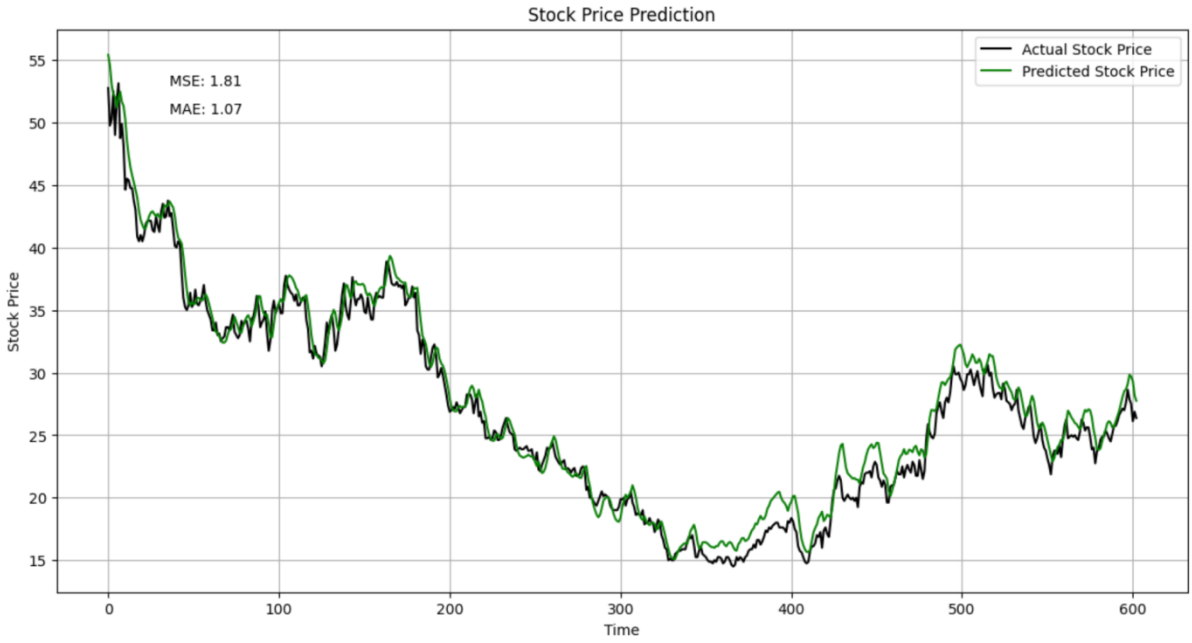


Рисунок 3.13 – Приклад візуалізації прогнозування комбінованим методом LSTM та CNN з технічними індикаторами

Третій графік показує прогнозовані ціни акцій (зеленим) порівняно з фактичними цінами (чорним).

Точність прогнозу: Модель демонструє хорошу точність з низькими значеннями MSE (1.81) та MAE (1.07), що підтверджується як числовими метриками, так і візуальною оцінкою на графіку.

Висновки до розділу 3

У третьому розділі було проведено огляд програмного забезпечення та аналіз отриманих результатів. Розділ 3 продемонстрував ефективність сучасних методів прогнозування часових рядів на основі машинного навчання, підкресливши важливість підготовки даних та вибору відповідних алгоритмів

для досягнення високої точності прогнозів. Основні результати можна підсумувати наступним чином.

1. Обґрунтування вибору програмного забезпечення. Було вибрано Python через його простоту, читабельність та багату екосистему бібліотек, таких як pandas, numpy, та scikit-learn. Ці бібліотеки забезпечують потужні інструменти для очищення та маніпуляції даними, що є критично важливим для підготовки даних до моделей машинного навчання.
2. Опис використаних даних та алгоритму роботи програми. Використані дані містили історичні ціни акцій Apple Inc. з різними показниками, такими як ціна відкриття, закриття, найвища та найнижча ціна за день, а також обсяги торгів. Дані були перевірені на пропуски та підготовлені до аналізу через нормалізацію та фільтрацію .
3. Аналіз та побудова моделей прогнозування:
    - 3.1. Було розглянуто кілька методів прогнозування, включаючи моделі ARIMA, LSTM та комбіновані методи LSTM з CNN.
    - 3.2. Модель ARIMA. Використана для аналізу та прогнозування часових рядів. Результати показали, що модель добре передбачає довгострокові тренди, хоча менш ефективна для короткострокових флуктуацій .
    - 3.3. Модель LSTM. Використана для прогнозування з урахуванням довгих залежностей у часових рядах. Результати продемонстрували високу точність у довгостроковому прогнозуванні, хоча деякі короткострокові коливання залишалися проблемними .
    - 3.4. Комбіновані методи LSTM та CNN. Додаткові технічні індикатори та комбіновані моделі показали ще кращі результати, зокрема завдяки здатності вилучати просторові

особливості та часові залежності одночасно. Це дозволило значно підвищити точність прогнозів .

4. Висновки щодо точності моделей. Всі розглянуті моделі показали добрі результати у довгостроковому прогнозуванні цін акцій. Найкращі результати були досягнуті при використанні комбінованих методів LSTM та CNN з технічними індикаторами, які дозволили значно знизити похибку прогнозів та краще вловлювати як короткострокові, так і довгострокові тренди .

РОЗДІЛ 4 ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

У цьому розділі здійснюється оцінка ключових параметрів програмного забезпечення, створеного для прогнозування фінансових часових рядів із застосуванням алгоритмів машинного навчання, таких як моделі ARIMA, LSTM та їхніх комбінацій. Далі представлено порівняння різноманітних варіантів реалізації модуля з ціллю обрання найбільш оптимального, враховуючи економічні чинники та характеристики продукту, які впливають на ефективність функціонування та сумісність з апаратним забезпеченням. Для досягнення цієї мети було застосовано метод функціонально-вартісного аналізу.

Функціонально-вартісний аналіз (ФВА) – це підхід, який дає змогу визначити фактичну вартість продукту чи послуги незалежно від організаційної структури компанії. ФВА здійснюється для ідентифікації можливостей зменшення витрат шляхом застосування ефективніших методів виробництва та досягнення оптимального співвідношення між споживчою цінністю продукту та витратами на його створення. Для виконання аналізу використовуються економічні, технічні та конструкторські дані.

Процес функціонально-вартісного аналізу передбачає визначення послідовності стадій розробки продукту, обчислення загальних (річних) витрат та кількості робочого часу, ідентифікацію джерел витрат та фінальний розрахунок вартості програмного забезпечення.

4.1 Постановка задачі проектування

У даній роботі використовується підхід функціонально-вартісного аналізу (ФВА) для здійснення техніко-економічної оцінки процесу розробки програмного забезпечення, призначеного для прогнозування фінансових часових рядів із застосуванням моделей інтелектуального аналізу даних та їхніх комбінацій. Оскільки рішення, прийняті під час проектування та реалізації компонентів, мають вплив на всю систему, кожна окрема підсистема повинна задовольняти загальним вимогам. Тому фактичний аналіз полягає в аналізі функцій програмного продукту, створеного для збору, обробки та прогнозування фінансових часових рядів.

Програмне забезпечення має відповідати таким технічним вимогам:

- можливість роботи на персональних комп'ютерах зі стандартною конфігурацією;
- інтуїтивно зрозумілий та зручний у користуванні інтерфейс;
- оперативна обробка даних та доступ до інформації в режимі реального часу;
- простота масштабування та обслуговування;
- мінімізація витрат на впровадження та експлуатацію.

4.2 Обґрунтування функцій програмного продукту

Головна функція F_0 – розробка програмного продукту, який вирішує 0 задачу порівняння методів штучного інтелекуту та будує їхні моделі. Беручи за основу цю функцію, можна виділити наступні:

- F_1 – вибір мови програмування;

– F_2 – вибір методу машинного навчання;

– F_3 – вибір середовища розробки.

Кожна з цих функцій має декілька варіантів реалізації.

1. Функція F_1: а – Python або б – R.
2. Функція F_2 : а – LSTM або б – ARIMA.
3. Функція F_3 : а – Jupyter Notebook або б – Google Colab.

Варіанти реалізації основних функцій наведені у морфологічній карті системи (рис. 4.1).

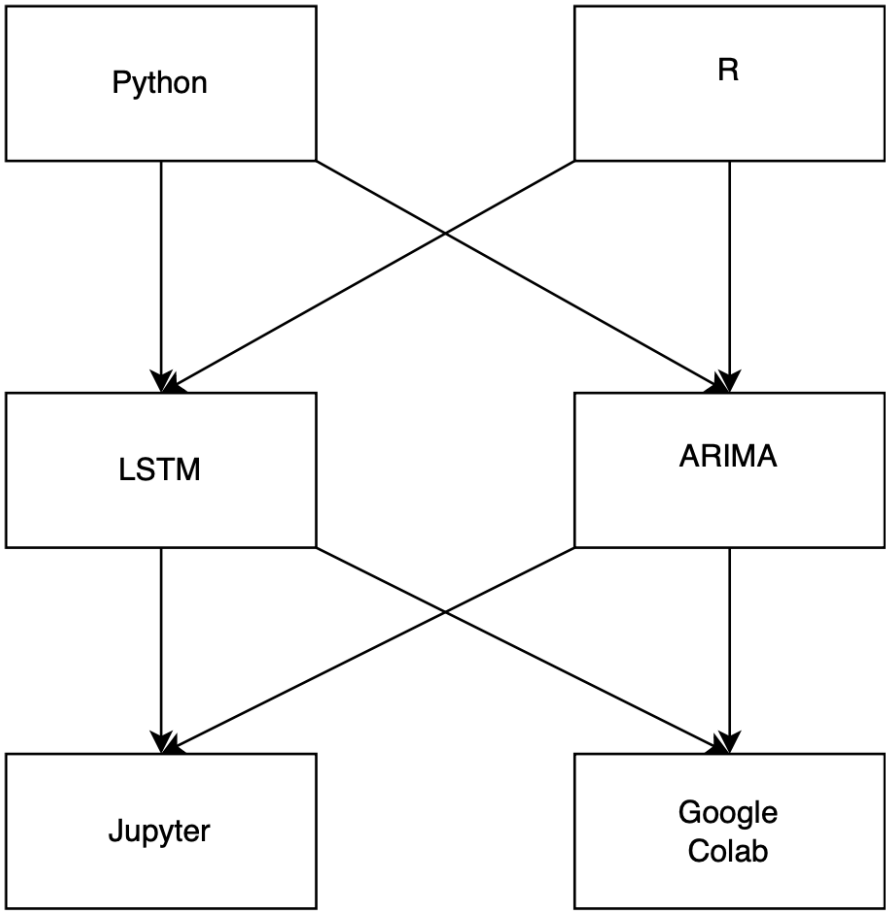


Рисунок 4.1 – Морфологічна карта

Морфологічна карта представляє сукупність усіх потенційних варіантів реалізації основних функцій. Спираючись на цю карту, створюємо матрицю переваг та недоліків різних варіантів основних функцій (табл. 4.1). На основі

проведеного аналізу можна зробити висновок, що під час розробки програмного забезпечення деякі варіанти реалізації функцій слід виключити, оскільки вони не відповідають вимогам, поставленим перед програмним продуктом. Ці варіанти позначені у морфологічній карті.

Таблиця 4.1 – Позитивно-негативна матриця

Функції	Варіанти реалізації	Переваги	Недоліки
F_1	А	Інтуїтивний синтаксис, широкий спектр бібліотек та інструментів, кросплатформеність, швидка розробка	Відносно низька швидкість роботи з великими даними
	Б	популярний для статистичного аналізу, багатий набір пакетів	Складніша розробка, обмежена підтримка великих проектів
F_2	А	Висока точність для складних часових рядів	Потребує більше обчислювальних ресурсів
	Б	Добре підходить для стаціонарних даних, гарно задокументована, широко використовується	Менш ефективний для складних нелінійних залежностей
F_3	А	Зручний інтерфейс, безкоштовне використання	Відсутня можливість роботи без інтернету
	Б	Доступність з будь-якого пристрою, інтеграція з Google Drive	Залежність від інтернет-з’єднання, обмеження на використання ресурсів

Функція F_1. Перевагу віддаємо швидкості розробки та наявності стандартних бібліотек для обчислення. Для спрощення роботи по написанню коду варіант б має бути відкинутий. Функція F_2. Обидва варіанти можна

використовувати в розробці. Функція F_3. Віддаємо перевагу варіанту *а* в разі вибору мови програмування Python.

Таким чином, будемо розглядати такі варіанти реалізації ПП:

$$F_1\text{а} - F_2\text{а} - F_3\text{а},$$

$$F_1\text{а} - F_2 b - F_3\text{а}.$$

1. F_1 : а (Python);
2. F_2 : а (LSTM) або б (ARIMA);
3. F_3 : а (Jupyter Notebook).

Для оцінювання якості розглянутих функцій обрана система параметрів, описана нижче.

4.3 Обґрунтування системи параметрів ПП

На підставі інформації, розглянутої вище, визначаються ключові параметри вибору, які будуть використані для обчислення коефіцієнта технічного рівня. Для характеристики програмного продукту будемо використовувати такі параметри:

- X1 – швидкість виконання коду мовою програмування;
- X2 – обсяг пам'яті, необхідний для обчислень та зберігання даних;
- X3 – тривалість навчання на наборі даних;
- X4 – потенційний розмір програмного коду.

Гірші, середні та кращі значення параметрів обираються на основі вимог замовника та умов, що характеризують експлуатацію програмного продукту, як показано у таблиці 4.2.

Таблиця 4.2 – Основні параметри ПП

Назва параметра	Умовні позначення	Одиниці виміру	Значення параметра		
			гірші	середні	кращі
оперативність мови програмування	X1	оп/мс	12000	16000	21000
об’єм пам’яті	X2	Мб	410	140	75
час навчання даних	X3	мс	6	4	3
потенційний об’єм програмного коду	X4	кількість рядків коду	3200	1900	800

За даними таблиці 4.3 будуються графічні характеристики параметрів – рис. 4.2 – рис. 4.5.

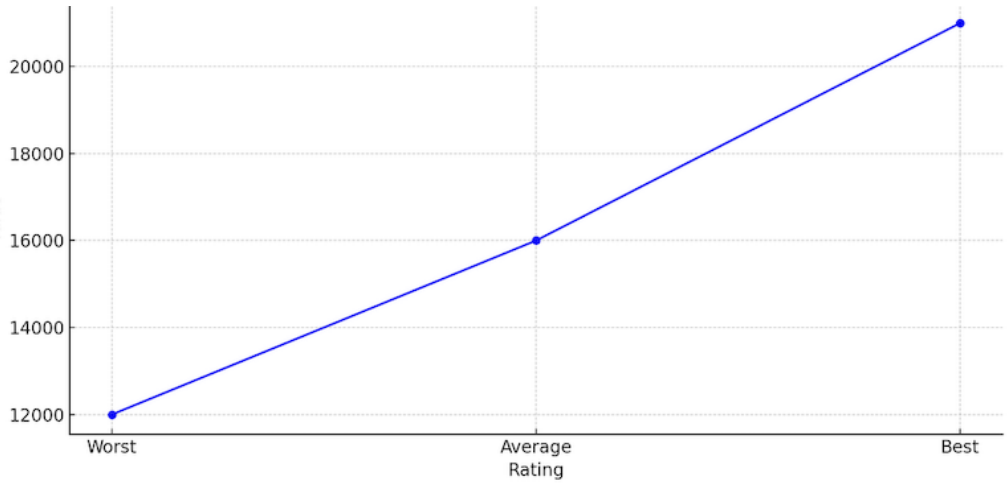


Рисунок 4.2 – X1, оперативність мови програмування

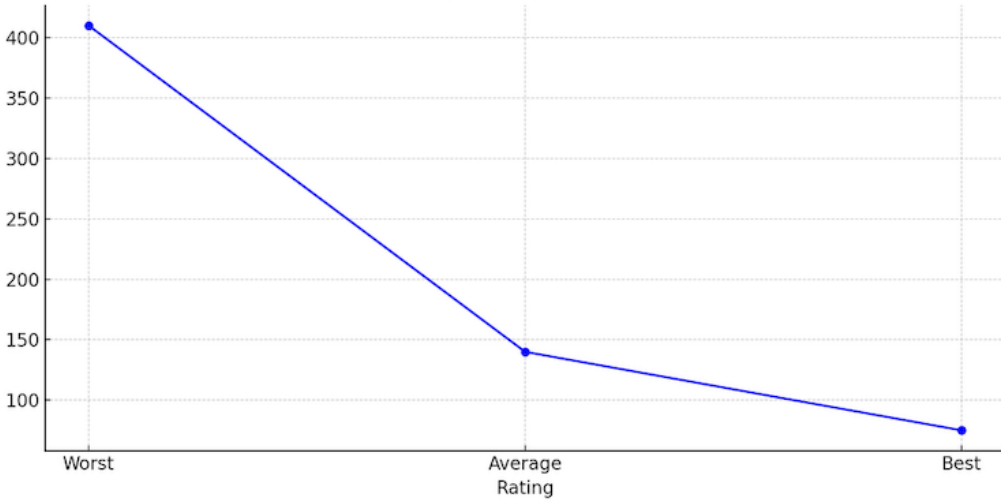


Рисунок 4.3 – X2, об’єм пам’яті

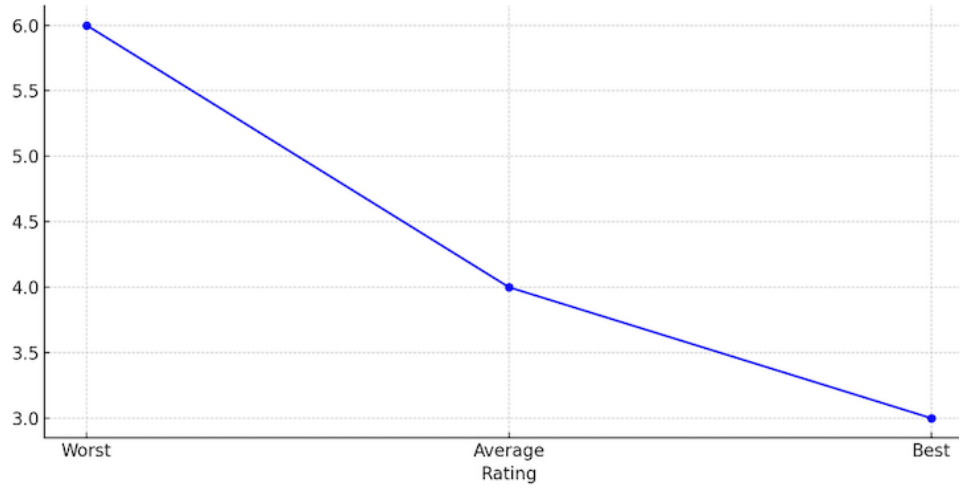


Рисунок 4.4 – X3, час навчання даних

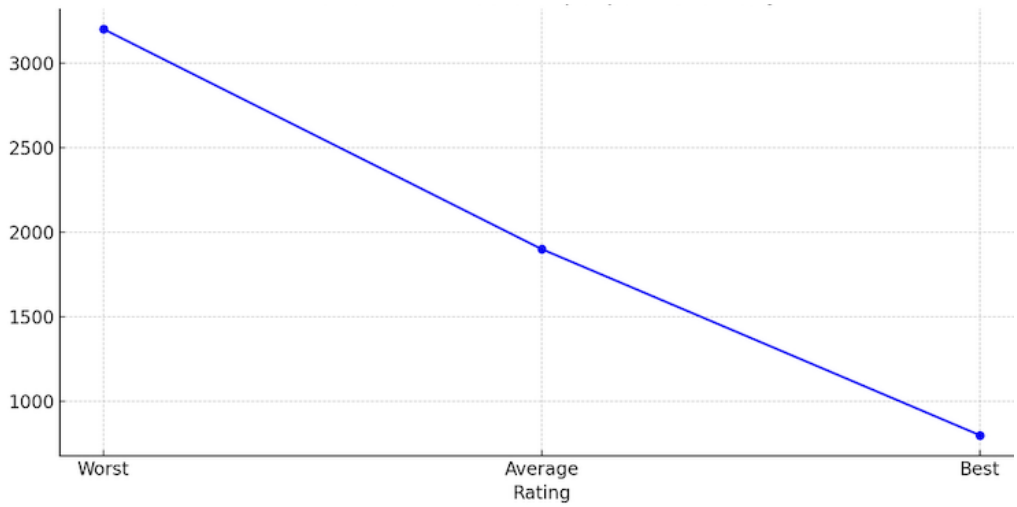


Рисунок 4.5 – X4, потенційний об’єм програмного коду

4.4 Аналіз експертного оцінювання параметрів

Після детального обговорення й аналізу кожний експерт оцінює ступінь важливості кожного параметру для конкретно поставленої цілі – розробка програмного продукту, який дає найбільш точні результати при знаходженні параметрів моделей адаптивного прогнозування і обчислення прогнозних значень. Значимість кожного параметра визначається методом попарного порівняння. Оцінку проводить експертна комісія із 7 людей. Визначення коефіцієнтів значущості передбачає:

- визначення рівня значимості параметра шляхом присвоєння різних рангів;
- перевірку придатності експертних оцінок для подальшого використання;
- визначення оцінки попарного пріоритету параметрів;
- обробку результатів та визначення коефіцієнту значимості.

Результати експертного ранжування наведені у таблиці 4.3

Таблиця 4.3 – Попарне порівняння параметрів.

Позначення параметра	Назва параметра	Одиниці виміру	Ранг параметра за оцінкою експерта							Сума рангів Ri	Відхилення Δi	Δi_2
			1	2	3	4	5	6	7			
X1	оперативність мови програмування	Оп/мс	5	3	1	2	4	2	1	18	-6.5	42.25
X2	об’єм пам’яті	Mb	3	4	6	6	5	4	6	34	9.5	90.25
X3	час навчання даних	мс	1	3	4	1	1	3	1	14	-10.5	110.25
X4	потенційний об’єм програмного коду	к-ть рядків коду	5	4	3	5	4	5	6	32	7.5	56.25
	Разом		14	14	14	14	14	14	14	98	0	299

Для перевірки степені достовірності експертних оцінок, визначимо наступні параметри:

– сума рангів кожного з параметрів і загальна сума рангів:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 98,$$

де N – число експертів;

n – кількість параметрів.

– середня сума рангів:

$$T = \frac{1}{n} R_{ij} = 24{,}5.$$

– відхилення суми рангів кожного параметра від середньої суми рангів:

$$\Delta_i = R_i - T.$$

Сума відхилень по всім параметрам повинна дорівнювати 0;

– загальна сума квадратів відхилення:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 299.$$

Порахуємо коефіцієнт узгодженості:

$$W = \frac{12S}{N^2(n^3-n)} = \frac{12 \cdot 197}{7^2(4^3-4)} = 1.22 > W_k = 0{,}67.$$

Ранжування можна вважати достовірним, тому що знайдений коефіцієнт узгодженості перевищує нормативний, котрий дорівнює 0,67.

Скориставшись результатами ранжирування, проведемо попарне порівняння всіх параметрів і результати занесемо у таблицю 4.4.

Таблиця 4.4 – Попарне порівняння параметрів.

Параметри	Експерти							Кінцева оцінка	Числове значення
	1	2	3	4	5	6	7		
X1 і X2	>	<	<	<	<	<	<	<	0.5
X1 і X3	>	=	<	>	>	<	=	>	1.5
X1 і X4	=	<	<	<	=	<	<	<	0.5
X2 і X3	>	>	>	>	>	>	>	>	1.5
X2 і X4	<	=	>	>	>	<	=	>	1.5
X3 і X4	<	<	>	<	<	<	<	<	0.5

Числове значення, що визначає ступінь переваги і–го параметра над j–тим, аіj визначається по формулі:

$$a_{ij} = \begin{cases} 1{,}5 \text{ при } X_i > X_j; \\ 1{,}0 \text{ при } X_i = X_j; \\ 0{,}5 \text{ при } X_i < X_j. \end{cases}$$

З отриманих числових оцінок переваги складемо матрицю $A=\| a_{ij} \|$. Для кожного параметра зробимо розрахунок вагомості K_{Bi} за наступними формулами:

$$K_{Bi} = \frac{b_i}{\sum_{i=1}^{n} b_i},$$

$$b_i = \sum_{i=1}^{N} a_{ij}.$$

Відносні оцінки розраховуються декілька разів доти, поки наступні значення не будуть незначно відрізнятися від попередніх (менше 2%). На

другому і наступних кроках відносні оцінки розраховуються за наступними формулами:

$$K_{\text{вi}} = \frac{b_i'}{\sum_{i=1}^{n} b_i'},$$

$$b_i' = \sum_{i=1}^{N} a_{ij} b_j.$$

Як видно з таблиці 4.5, різниця значень коефіцієнтів вагомості не перевищує 2%, тому більшої кількості ітерацій не потрібно.

Таблиця 4.5 – Розрахунок вагомості параметрів

Параметри x_i	Параметри x_j				Перша ітер.		Друга ітер.		Третя ітер	
	X1	X2	X3	X4	b_i	K_{Bi}	b_i^1	K_{Bi}^1	b_i^2	K_{Bi}^2
X1	1	0.5	1.5	0.5	3.5	0.22	12.25	0.21	44.875	0.21
X2	1.5	1	1.5	1.5	5.5	0.34	21.25	0.36	77.875	0.36
X3	0.5	0.5	1	0.5	2.5	0.16	9.25	0.16	34.125	0.16
X4	1.5	0.5	1.5	1	4.5	0.28	16.25	0.275	59.125	0.27
					16	1	59	1	216	1

4.5 Аналіз рівня якості варіантів реалізації функцій

Визначаємо рівень якості кожного варіанту виконання основних функцій окремо.

Абсолютні значення параметрів X2 (Об’єм пам’яті), X3 (час попередньої обробки даних) та X4 (потенційний об’єм програмного коду) відповідають

технічним вимогам умов функціонування даного ПП. Абсолютне значення параметра X1 (швидкість роботи мови програмування) обрано не найгіршим. Коефіцієнт технічного рівня для кожного варіанта реалізації ПП розраховується так (табл. 4.7):

$$K_K(j) = \sum_{i=1}^{n} K_{\text{в}i,j} B_{i,j},$$

де n – кількість параметрів;

$K_{\text{в}i}$ – коефіцієнт вагомості *i*–го параметра;

B_i – оцінка *i*–го параметра в балах.

Таблиця 4.6 – Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основні функції	Варіант реалізації функції	Параметри	Абсолютне значення параметра	Бальна оцінка параметра	Коефіцієнт вагомості параметра	Коефіцієнт рівня якості
F1	А	X1	12000	14	0.21	2.9
F2	А	X2	140	10	0.36	3.6
F3	Б	X3	75	11	0.16	1.76
	А	X4	800	8	0.27	2.16

За даними з таблиці 4.6 за формулою:

$$K_K = K_{\text{ТУ}}[F_{1k}] + K_{\text{ТУ}}[F_{2k}] + \ldots + K_{\text{ТУ}}[F_{zk}].$$

визначаємо рівень якості кожного з варіантів:

K_{K1}= 2.9 + 3.6 + 2.16 = 8.66;

K_{K2}= 2.9 + 1.8 + 3.6 = 8.3.

Як видно з розрахунків, кращим є перший варіант, для якого коефіцієнт технічного рівня має найбільше значення.

4.6 Економічний аналіз варіантів розробки ПП

Економічний аналіз варіантів розробки програмного продукту. Для визначення вартості розробки програмного продукту (ПП) спочатку розрахуємо трудомісткість. Усі варіанти включають два окремих завдання.

1. Розробка проекту програмного продукту;
2. Розробка програмної оболонки.

Завдання 1 за ступенем новизни належить до групи А, а завдання 2 – до групи Б. За складністю алгоритми, які використовуються в завданні 1, відносяться до групи 1, а в завданні 2 – до групи 3. Для реалізації завдання 1 використовується довідкова інформація, а завдання 2 використовує інформацію у вигляді даних.

Розрахуємо норми часу на розробку та програмування для кожного завдання.

Для завдання 1, виходячи з норм часу для завдань розрахункового характеру зі ступенем новизни А та групою складності алгоритму 1, трудомісткість становить: ТР = 100 людино-днів. Поправочний коефіцієнт, який враховує вид нормативно-довідкової інформації для завдання 1: K_n = 1,6. Поправочний коефіцієнт, який враховує складність контролю вхідної та вихідної інформації для всіх семи завдань: $K_{ск}$ = 0,9. Оскільки при розробці завдання 1 використовуються стандартні модулі, врахуємо це за допомогою коефіцієнта $K_{ст}$ = 0,9.

Загальна трудомісткість програмування завдання 1:

$$T_1 = 100 \cdot 1{,}6 \cdot 0{,}9 = 144 \text{ людино-днів.}$$

Для завдання 2 (використовується алгоритм третьої групи складності, ступінь новизни Б): $T_p = 40$ людино-днів, $K_п = 0{,}8$, $K_{ск} = 1$, $K_{ст} = 0{,}8$.

Загальна трудомісткість програмування завдання 2:

$$T_2 = 40 \cdot 1{,}0 \cdot 0{,}8 = 32 \text{ людино-днів.}$$

Складаємо трудомісткість відповідних завдань для кожного з обраних варіантів реалізації програми, щоб отримати їх трудомісткість:

$$T_1 = (144 + 32 + 4.8 + 32) \cdot 8 = 1702{,}4 \text{ людино-годин.}$$
$$T_2 = (144 + 32 + 6{,}91 + 32) \cdot 8 = 1719{,}28 \text{ людино-годин.}$$

Варіант II має найвищу трудомісткість. У розробці беруть участь два програмісти з окладом 15000 грн та один аналітик даних з окладом 20000 грн. Середня погодинна зарплата розраховується за формулою:

$$СЧ = \frac{15000+15000+20000}{3 \cdot 21 \cdot 8} = 99.21$$

Зарплата розробників за варіантами становить:

$$C_{зп1} = 99{,}21 \cdot 1702{,}4 \cdot 1.2 = 202674{,}125 \text{ грн,}$$
$$C_{зп2} = 99{,}21 \cdot 1719{,}28 \cdot 1.2 = 204683{,}723 \text{ грн.}$$

Відрахування на єдиний соціальний внесок (22%):

$$C_{від1} = C_{зп1} \cdot 0.22 = 202674{,}125 \cdot 0.22 = 44588.3075 \text{ грн.,}$$
$$C_{від2} = C_{зп2} \cdot 0.22 = 204683{,}723 \cdot 0.22 = 45040.4191 \text{ грн.}$$

Витрати на оплату однієї машино-години (СМ): одна ЕОМ обслуговує одного програміста з окладом 15000 грн та коефіцієнтом зайнятості 0,2.

Для однієї машини:

$$C_г = 12 \cdot 15000 \cdot 0,2 = 36000 \text{ грн.}$$

З урахуванням додаткової заробітної плати:

$$C_{зп} = C_г \cdot (1 + 0.2) = 36000 \cdot 1.2 = 43200 \text{ грн.}$$

З відрахуванням на соціальний внесок:

$$C_{від} = C_{зп} \cdot 0.22 = 43200 \cdot 0,22 = 9504 \text{ грн.}$$

Амортизаційні відрахування (амортизація 25%, вартість ЕОМ – 20000 грн):

$$C_a = 1.15 \cdot 0.25 \cdot 20000 = 5750 \text{ грн.}$$

Витрати на ремонт та профілактику:

$$C_p = 1.15 \cdot 20000 \cdot 0.05 = 1150 \text{ грн.}$$

Ефективний годинний фонд часу ПК за рік:

$$T_{еф} = (365 – 104 – 12 – 16) \cdot 8 \cdot 0.9 = 1678 \text{ годин.}$$

Витрати на оплату електроенергії:

$$C_{ел} = 1677.6 \cdot 0.9 \cdot 5.23 \cdot 0.8 = 6317 \text{ грн .}$$

Накладні витрати:

$$C_{н} = 20000 \cdot 0{,}67 = 13400 \text{ грн.}$$

Річні експлуатаційні витрати:

$$C_{екс} = 43200 + 9504 + 10590 + 5750 + 1150 + 6317 + 13400 = 89911 \text{ грн.}$$

Собівартість однієї машино-години ЕОМ:

$$C_{м-г} = 89911 / 1678 = 53.6 \text{ грн/год .}$$

Оскільки всі роботи, пов'язані з розробкою програмного продукту, виконуються на ЕОМ, витрати на оплату машинного часу, залежно від обраного варіанта реалізації, становлять:

$$C_{м} = 53.6 \cdot 1702{,}4 = 91248{,}64 \text{ грн.}$$
$$C_{м} = 53.6 \cdot 1719{,}28 = 92153,\ 408 \text{ грн.}$$

Накладні витрати складають 60% від заробітної плати:

$$C_{н} = 202674{,}125 \cdot 0{,}6 = 121604{,}475 \text{ грн.}$$
$$C_{н} = 204683{,}723 \cdot 0{,}6 = 122810{,}6234 \text{ грн.}$$

Отже, вартість розробки ПП за варіантами становить:

$$C_{пп} = 202674{,}125+44588.3075+91248{,}64+121604{,}475= 460115{,}5475 \text{ грн.}$$
$$C_{пп} = 204683{,}723+45040.4191+92153,\ 408+122810{,}6234 = 464688{,}1735 \text{ грн.}$$

4.7 Вибір кращого варіанту ПП техніко-економічного рівня

Розрахуємо коефіцієнт техніко-економічного рівня за формулою:

$$K_{тер1} = 8{,}66/460115{,}5475 = 1{,}882 \cdot 10^{-5}$$

$$K_{тер2} = 8{,}3/464688{,}1735 = 1.786 \cdot 10^{-5}$$

Як бачимо, найбільш ефективним є перший варіант реалізації програми з коефіцієнтом техніко-економічного рівня $K_{тер2} = 1.786 \cdot 10^{-5}$. Після проведення функціонально-вартісного аналізу розроблюваного програмного комплексу, можна зробити висновок, що серед альтернатив, відібраних на першому етапі, найкращим варіантом реалізації програмного продукту є перший варіант.

Висновки до розділу 4

Проведено повний функціонально-вартісний аналіз програмного продукту. Виконано оцінку основних функцій та визначено параметри, що характеризують продукт. Проведено експертне оцінювання параметрів і аналіз якості реалізації функцій. Здійснено економічний аналіз варіантів розробки, включаючи трудомісткість, витрати на заробітну плату та інші витрати. На основі цього аналізу обрано оптимальний варіант реалізації програмного продукту.

ВИСНОВКИ

Ця дипломна робота зосереджена на створенні програмного рішення для передбачення фінансових часових рядів, використовуючи передові техніки машинного навчання. Зокрема, досліджуються моделі авторегресійного інтегрованого ковзного середнього (ARIMA), довгої короткочасної пам'яті (LSTM) та їх поєднання.

Перший розділ роботи висвітлює значущість цієї теми в контексті сучасних реалій. В умовах дедалі більшої складності, динамічності та взаємопов'язаності фінансових ринків, виникає гостра необхідність у потужних інструментах для ефективного аналізу та прогнозування фінансових даних. Точні прогнози фінансових показників відіграють ключову роль у прийнятті зважених інвестиційних рішень та ефективному управлінні ризиками. У цьому контексті, методи інтелектуального аналізу даних, зокрема машинне навчання та штучні нейронні мережі, виявляють значні перспективи у вирішенні задач прогнозування фінансових даних.

Аналіз існуючих рішень показав, що Python є найбільш популярною мовою програмування для аналізу фінансових часових рядів завдяки своїй широкій екосистемі бібліотек. Серед методів прогнозування виділяються моделі ARIMA, LSTM, експоненціальне згладжування та Facebook Prophet. На практиці найбільш точні прогнози часто виходять із поєднання кількох методів.

У другому розділі розглянуто теоретичні основи аналізу часових рядів, включаючи поняття часового ряду, адитивної та мультиплікативної моделей, компонентів часових рядів. Описано міри точності прогнозів, роботу з пропущеними даними, нормалізацію та масштабування даних. Розглянуто статистичні тести для аналізу часових рядів, такі як тест Дікі-Фуллера, тест KPSS, автокореляційна та часткова автокореляційна функції. Детально описано моделі ARIMA та LSTM для прогнозування часових рядів.

У третьому розділі представлено результати практичної реалізації програмного забезпечення. Обґрунтовано вибір мови програмування Python та використаних бібліотек. Описано набір даних, що використовувався для прогнозування – історичні дані про ціни акцій Apple Inc. Проведено фільтрацію даних, перевірку на пропуски, нормалізацію та масштабування. Виконано статистичні тести для перевірки стаціонарності ряду. Реалізовано моделі ARIMA, LSTM та їх комбінації для прогнозування цін акцій. Проведено оцінку точності моделей за допомогою метрик MSE та MAE. Візуалізовано результати прогнозування та проаналізовано ефективність моделей.

Результати показали, що комбінована модель LSTM та CNN з технічними індикаторами дала найбільш точні прогнози з найнижчими значеннями MSE та MAE. Ця модель здатна ефективно захоплювати довгострокові залежності та локальні особливості часових рядів, що призводить до покращення точності прогнозування порівняно з окремими моделями ARIMA та LSTM.

Загалом, дипломна робота демонструє успішну розробку програмного забезпечення для прогнозування фінансових часових рядів з використанням сучасних методів машинного навчання. Результати підкреслюють потенціал комбінованих моделей, таких як LSTM+CNN, для підвищення точності прогнозування. Робота робить внесок у область фінансового аналізу та прогнозування, пропонуючи ефективні інструменти для прийняття обґрунтованих інвестиційних рішень та управління ризиками на фінансових ринках.

ПЕРЕЛІК ПОСИЛАНЬ

1. П. І. Бідюк, В. Д. Романенко, О. Л. Тимощук «Аналіз часових рядів: навчальний посібник».
2. “What is LSTM? Introduction to Long Short-Term Memory”, Rebeen Hamad. URL:https://medium.com/@rebeen.jaff/what-is-lstm-introduction-to-long-short-term-memory-66bd3855b9ce
3. “Everything you can do with a time series”, Siddhart Yadav. URL:https://www.kaggle.com/code/thebrownviking20/everything-you-can-do-with-a-time-series#Aim
4. “Time Series Analysis: The Basics”, Australian Bureau of Statistics. URL:https://www.abs.gov.au/websitedbs/d3310114.nsf/home/time+series+analysis:+the+basics
5. Wang, B., Huang, H., & Wang, X. (2012). A novel text mining approach to financial time series forecasting. *Neurocomputing*, 83, 136-145. URL:https://doi.org/10.1016/j.neucom.2011.12.013
6. Ivanyuk, V., Abdikeev, N., & Tsvirkun, A. (2020). Forecasting the dynamics of financial time series based on neural networks. *Journal of Physics: Conference Series*, 1703.URL: https://doi.org/10.1088/1742-6596/1703/1/012030.
7. Bou-Hamad, I., & Jamali, I. (2020). Forecasting financial time-series using data mining models: A simulation study. *Research in International Business and Finance*, 51, 101072.URL: https://doi.org/10.1016/J.RIBAF.2019.101072.
8. P., Dagar, A., Bala, R., & Singh, R. (2018). Financial Time Series Forecasting Using Deep Learning Network. , 23-33. URL:https://doi.org/10.1007/978-981-13-2035-4_3.
9. Sun, H., & Xu, J. (2018). Improved approaches for financial market forecasting based on stationary time series analysis. *2018 IEEE 3rd*

International Conference on Big Data Analysis (ICBDA), 334-339. URL:https://doi.org/10.1109/ICBDA.2018.8367703.

10. Sezer, O., Gudelek, M., & Ozbayoglu, A. (2019). Financial Time Series Forecasting with Deep Learning : A Systematic Literature Review: 2005-2019. *ArXiv*, abs/1911.13288.URL:https://doi.org/10.1016/j.asoc.2020.106181.
11. Bousbaa, Z., Sanchez-Medina, J., & Bencharef, O. (2023). Financial Time Series Forecasting: A Data Stream Mining-Based System. *Electronics*. URL:https://doi.org/10.3390/electronics12092039.
12. Alonso, A., Sipols, A., & Blas, C. (2018). Forecasting financial short time series. *Electronic Journal of Applied Statistical Analysis*, 11, 42-57. URL:https://doi.org/10.1285/i20705948v11n1p42.
13. Bodyanskiy, Y., & Popov, S. (2006). Neural network approach to forecasting of quasiperiodic financial time series. *Eur. J. Oper. Res.*, 175, 1357-1366. URL:https://doi.org/10.1016/J.EJOR.2005.02.012.
14. Sako, K., Mpinda, B., & Rodrigues, P. (2022). Neural Networks for Financial Time Series Forecasting. *Entropy*, 24. URL:https://doi.org/10.3390/e24050657.
15. Tran, D., Kanniainen, J., Gabbouj, M., & Iosifidis, A. (2019). Data-driven Neural Architecture Learning For Financial Time-series Forecasting. *ArXiv*, abs/1903.06751.
16. Derbentsev, V., Matviychuk, A., Datsenko, N., Bezkorovainyi, V., & Azaryan, A. (2020). Machine learning approaches for financial time series forecasting. , 434-450.URL: https://doi.org/10.31812/123456789/4478.
17. Zinenko, A. (2023). Forecasting financial time series using singular spectrum analysis. *Business Informatics*. URL:https://doi.org/10.17323/2587-814x.2023.3.87.100.
18. Rafiuzaman, M. (2014). Forecasting Chaotic Stock Market Data using Time Series Data Mining. *International Journal of Computer Applications*, 101, 27-34.URL: https://doi.org/10.5120/17725-8169.

19. Okasha, M. (2014). Using Support Vector Machines in Financial Time Series Forecasting. *International journal of statistics and applications*, 4, 28-39.

20. Barra, S., Carta, S., Corriga, A., Podda, A., & Recupero, D. (2020). Deep learning and time series-to-image encoding for financial forecasting. *IEEE/CAA Journal of Automatica Sinica*, 7, 683-692. URL:https://doi.org/10.1109/JAS.2020.1003132.

21. Freeborough, W., & Zyl, T. (2022). Investigating Explainability Methods in Recurrent Neural Network Architectures for Financial Time Series Data. *Applied Sciences*.URL: https://doi.org/10.3390/app12031427.

22. He, Q., Pang, P., & Si, Y. (2019). Transfer Learning for Financial Time Series Forecasting. , 24-36.URL: https://doi.org/10.1007/978-3-030-29911-8_3.