

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»**

ФАКУЛЬТЕТ МЕНЕДЖМЕНТУ ТА МАРКЕТИНГУ

КАФЕДРА ЕКОНОМІЧНОЇ КІБЕРНЕТИКИ

«На правах рукопису»

УДК 330.3

ДО ЗАХИСТУ ДОПУЩЕНО:

Завідувач кафедри

_____ Катерина БОЯРИНОВА

«__» грудня 2024 р.

МАГІСТЕРСЬКА ДИСЕРТАЦІЯ

**на здобуття ступеня магістра
за освітньо-професійною програмою
«Економічна аналітика»
зі спеціальності 051 Економіка**

**на тему: «Оптимізаційна аналітика у формуванні інвестиційного портфеля
підприємства»**

Виконав:

студент 2-го курсу, групи УА-31мп

КРИКУН Євген Олександрович

Науковий керівник:

доцент кафедри економічної кібернетики, к.ф.-м.н., доц

ЛАЗАРЕНКО Ірина Сергіївна

Рецензент:

завідувач кафедри економіки і підприємництва, д.е.н., проф.

ТУЛЬЧИНСЬКА Світлана Олександрівна

Засвідчую, що у цій магістерській
дисертації немає запозичень з праць інших
авторів без відповідних посилань

Студент _____
(підпис)

Київ – 2024

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет менеджменту та маркетингу
Кафедра економічної кібернетики
Рівень вищої освіти – другий (магістерський)
Спеціальність – 051 Економіка
Освітньо-професійна програма «Економічна аналітика»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Катерина БОЯРИНОВА

« 20 » червня 2024 р.

ЗАВДАННЯ
на магістерську дисертацію студенту

- 1. Тема дисертації** «Оптимізаційна аналітика у формуванні інвестиційного портфеля підприємства», науковий керівник дисертації Лазаренко Ірина Сергіївна, к.ф.-м.н. доц., затверджені наказом по університету від 08.11.2024 року № 5019-с.
- 2. Термін подання студентом дисертації** 09.12.2024 року.
- 3. Об'єктом дослідження:** є проекти інвестиційної діяльності підприємства, у вигляді формування портфелю акцій.
- 4. Предмет дослідження (Вихідні дані):** є аналітичні методи та практичні аспекти використання сучасних підходів до формування оптимального інвестиційного портфелю, зокрема застосування алгоритмів прогнозування часових рядів і методів аналізу великих даних. Особлива увага приділена інструментам машинного навчання, глибокого навчання (Deep Learning) і Data Science для прогнозування фінансових показників та формування ефективних інвестиційних рішень в умовах цифрової трансформації економіки. В додаток до цього, важливим аспектом є порівняльний аналіз отриманих результатів, який є підґрунтям проведення економічної аналітики отриманих результатів. Вихідними даними є історичні дані цін акцій з Yahoo Finance, аналітичні матеріали у цій галузі.

5. Перелік завдань, які потрібно розробити:

а) теоретична частина:

- розкрити сутність, зміст інвестиційних проєктів та портфельної теорії;
- дослідити методи прогностичної аналітики класичного машинного навчання для роботи із часовими рядами;
- здійснити підбір та обґрунтувати підходи до визначення наповнення оптимального інвестиційного портфелю.

б) аналітична частина:

- провести аналіз та оцінку оптимізаційних методів для формування оптимального портфелю;
- дослідити аналітичний інструментарій прогнозування часових рядів;
- провести аналітичне моделювання для визначення ефективності інвестиційних проєктів.

в) рекомендаційна частина:

- провести прогностичну аналітику реалізації інвестиційного портфелю з використанням методів аналізу часових рядів;
- використання аналітичних методів машинного навчання для аналізу та прогнозування результатів реалізації інвестиційних проєктів;
- провести компаративний аналіз та розробити рекомендації для формування оптимального інвестиційного портфелю на основі економіко-математичного моделювання та аналізу даних.

6. Орієнтовний перелік ілюстративного матеріалу:

- Структура DNN
- Структура RNN шару
- Структура LSTM шару
- Історичні ціни акцій цільових компаній
- Порівняльний аналіз прогнозованих значень моделі ARIMA із фактичними даними
- Щоденні норми прибутку прогнозованих значень
- Параметри нейронної мережі DNN
- Прогнозовані значення за допомогою початкової моделі
- Швидкість навчання моделі
- Зміна втрат у процесі навчання моделі
- Зміна втрат у процесі навчання моделі починаючи з 10 епохи
- Прогностичні результати оптимізованої нейронної мережі
- Порівняльний аналіз DNN прогнозування із реальними даними
- Структура RNN моделі
- Графік зміни швидкості навчання
- Порівняльний аналіз RNN прогнозування
- Порівняльний аналіз RNN прогнозування для всіх обраних цін акцій
- Аналіз швидкості навчання моделей

- Структура комбінованої LSTM-RNN моделі
- Порівняльний аналіз прогнозування LSTM-RNN моделі із фактичними значеннями
- Порівняльний аналіз MSE метрики
- Теплова карта для R2 метрики
- Порівняльний аналіз портфельних метрик
- Порівняльний аналіз вагових коефіцієнтів портфелів

7. Орієнтовний перелік публікацій за напрямом роботи:

- 1) Lazarenko I., Krykun Y. Portfolio management with time series analysis methods Economic bulletin of National technical university of Ukraine «Kyiv polytechnical institute». 2024. No 29. URL: <https://ev.fmm.kpi.ua/article/view/309267/300787> (видання категорії Б);
- 2) Krykun Y., Lazarenko I., Data mining techniques for portfolio building. Моделювання та прогнозування економічних процесів: зб. тез доп. XVIII Міжнар. наук.-практ. конф., м. Київ, 5 груд. 2024 р. Київ: КПІ ім. Ігоря Сікорського, Вид-во «Політехніка», 2024. URL: <https://mperproc.fmm.kpi.ua>.

8. Дата видачі завдання: 19 червня 2024 року.

КАЛЕНДАРНИЙ ПЛАН

<i>№ з/п</i>	<i>Назва етапів виконання магістерської дисертації</i>	<i>Термін виконання етапів магістерської дисертації</i>	<i>Примітка</i>
1.	Вибір напрямку дослідження, узгодження завдання та змісту магістерської дисертації з науковим керівником.	01.06.2024- 20.06.2024	
2.	Збір необхідної інформації, вивчення та аналіз літературних джерел щодо досліджуваної тематики.	21.06.2024- 05.09.2024	
3.	Розгляд теоретичних засад та підходів до оптимізаційної аналітики у формуванні інвестиційного портфелю підприємства. Надання на перевірку першого розділу.	06.09.2024- 23.09.2024	
4.	Оцінювання класичного моделювання оптимального портфелю цінних паперів.	24.09.2024- 30.09.2024	
5.	Застосування аналітично-математичного інструментарію визначення оптимального портфелю з використанням класичних методів аналізу часових рядів	01.10.2024- 14.10.2024	
6.	Моделювання оптимального портфелю з використанням Deep Learning технологій.	15.10.2024- 21.10.2024	
7.	Аналітичне моделювання для визначення ефективності інвестиційних проєктів з використанням методів машинного навчання. Надання на перевірку другого розділу	22.10.2024- 04.11.2024	
8.	Реалізація прогностичної аналітика для інвестиційного портфелю з використанням методів аналізу часових рядів.	05.11.2024- 11.11.2024	
9.	Компаративний аналіз результатів та рекомендації для формування оптимального інвестиційного портфелю. Надання на перевірку третього розділу.	12.11.2024- 18.11.2024	
10.	Оформлення магістерської дисертації другого (магістерського) рівня вищої освіти.	19.11.2024- 28.11.2024	
11.	Подання магістерської дисертації для перевірки на збіг/схожість, отримання відгуку керівника та рецензії.	29.11.2024- 06.12.2024	
12.	Подання магістерської дисертації до захисту.	09.12.2024	

Студент

_____ Євген КРИКУН
(підпис)

Науковий керівник

_____ Ірина ЛАЗАРЕНКО
(підпис)

РЕФЕРАТ

Магістерська дисертація Крикуна Євгена Олександровича на тему «Оптимізаційна аналітика у формуванні інвестиційного портфеля підприємства» зі спеціальності 051 Економіка, освітньо-професійної програми «Економічна аналітика», Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», 2024, Київ.

Магістерська дисертація складається зі вступу, 3 розділів, висновків, списку використаних джерел та додатків. Виконана в обсязі 134 сторінки, містить 24 рисунки, 25 таблиць та 1 додаток.

Актуальність теми. Ефективне управління інвестиційним портфелем є ключовою задачею сучасної економіки, особливо в умовах глобалізації, динамічних змін фінансових ринків та зростаючої невизначеності. Використання інноваційних підходів, таких як Big Data, машинне навчання (Deep Learning) та Data Science, дозволяє підвищити точність прогнозів, знизити ризики й оптимізувати структуру портфеля. Алгоритми прогнозування й автоматизованої оптимізації забезпечують адаптацію до змін у зовнішньому середовищі. Актуальність теми зумовлена потребою в нових методологіях, що поєднують теоретичні основи інвестиційного аналізу з інноваційними рішеннями для управління портфелем в умовах цифрової трансформації.

Зв'язок дослідження з науковими програмами, планами, темами. Магістерська дисертація виконувалась в Національному технічному університеті України «Київський політехнічний інститут імені Ігоря Сікорського» відповідно до планів наукових досліджень кафедри економічної кібернетики за темою «Аналітика та моделювання економічного розвитку в імперативах Trans Tech».(№ 0123U101760). Роль автора полягає у визначенні оптимального портфелю акцій, ґрунтовному аналізі портфельної теорії, оптимізаційних методів та методів прогнозування часових рядів.

Метою магістерською дисертації є ґрунтовний аналіз теоретичних засад портфельної теорії, розробка та застосування оптимізаційних моделей для

формування ефективного інвестиційного портфеля, а також проведення прогностичного аналізу динаміки історичних цін акцій із використанням методів класичного машинного навчання та алгоритмів глибокого навчання. Особливу увагу приділено порівняльному аналізу результатів, отриманих різними підходами, із подальшим проведенням економічної оцінки й формуванням обґрунтованих інвестиційних рішень та рекомендацій для створення оптимального портфеля.

Завдання дослідження: розкрити сутність, зміст інвестиційних проєктів та портфельної теорії; дослідити методи прогностичної аналітики класичного машинного навчання для роботи із часовими рядами; здійснити підбір та обґрунтувати підходи до визначення наповнення оптимального інвестиційного портфелю. Провести аналіз та оцінку оптимізаційних методів для формування оптимального портфелю; дослідити аналітичний інструментарій прогнозування часових рядів; провести аналітичне моделювання для визначення ефективності інвестиційних проєктів. Провести прогностичну аналітику реалізації інвестиційного портфелю з використанням методів аналізу часових рядів; використання аналітичних методів машинного навчання для аналізу та прогнозування результатів реалізації інвестиційних проєктів; провести компаративний аналіз та розробити рекомендації для формування оптимального інвестиційного портфелю на основі економіко-математичного моделювання та аналізу даних.

Об'єктом дослідження: є проєкти інвестиційної діяльності підприємства, у вигляді формування портфелю акцій.

Предметом дослідження – є аналітичні методи та практичні аспекти використання сучасних підходів до формування оптимального інвестиційного портфелю, зокрема застосування алгоритмів прогнозування часових рядів і методів аналізу великих даних. Особлива увага приділена інструментам машинного навчання, глибокого навчання (Deep Learning) і Data Science для прогнозування фінансових показників та формування ефективних інвестиційних рішень в умовах цифрової трансформації економіки. В додаток до цього, важливим аспектом є порівняльний

аналіз отриманих результатів, який є підґрунтям проведення економічної аналітики отриманих результатів.

Методи дослідження. аналітичний, емпіричний, індукція, дедукція, оптимізаційний, прогностичний, компаративний.

Наукова новизна одержаних результатів. Елементами наукової новизни магістерської дисертації є: порівняльний аналіз двох видів прогностично-аналітичного інструментарію: цін акцій та коваріаційної матриці, удосконалено підхід побудови ARIMA моделі шляхом додаткового покрокового навчання.

Практичне значення одержаних результатів. Розроблений аналітично-математичний інструментарій, зокрема RNN, DNN та ARIMA моделі, мають вагоме практичне застосування у формуванні оптимального інвестиційного портфеля. Їх використання, у поєднанні з удосконаленою моделлю оптимізації портфеля, забезпечує досягнення ефективних результатів, що можуть бути успішно впроваджені у практичну діяльність.

Апробація результатів магістерської дисертації: Виступ на конференції з публікацією тез доповіді: Krykun Y., Lazarenko I., Data mining techniques for portfolio building. Моделювання та прогнозування економічних процесів: зб. тез доп. XVIII Міжнар. наук.-практ. конф., м. Київ, 5 груд. 2024 р. Київ: КПІ ім. Ігоря Сікорського, Вид-во «Політехніка», 2024. URL: <https://mperproc.fmm.kpi.ua>

Публікації. Стаття (видання категорії Б): Lazarenko I., Krykun Y. Portfolio management with time series analysis methods Economic bulletin of National technical university of Ukraine «Kyiv polytechnical institute». 2024. No 29. URL: <https://ev.fmm.kpi.ua/article/view/309267/300787>

Ключові слова: економічна аналітика, прогнозування, машинне навчання, часові ряди, оптимізація, інвестиційний проєкт, інвестиційний портфель.

ABSTRACT

Master's thesis by Krykun Yevhen Oleksandrovysh titled "Optimization Analytics in Forming the Investment Portfolio of an Enterprise", in the specialty 051 Economics, educational-professional program "Economic Analytics," National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute," 2024, Kyiv.

The master's thesis consists of an introduction, three chapters, conclusions, a list of references, and appendices. It comprises 134 pages, includes 24 figures, 25 tables, and 1 appendix.

The relevance of the topic. Effective investment portfolio management is a key task in the modern economy, especially in the context of globalization, dynamic changes in financial markets, and growing uncertainty. The use of innovative approaches such as Big Data, machine learning (Deep Learning), and Data Science enables improved forecast accuracy, risk reduction, and portfolio structure optimization. Predictive algorithms and automated optimization ensure adaptability to changes in the external environment. The relevance of the topic is driven by the need for new methodologies that combine the theoretical foundations of investment analysis with innovative solutions for portfolio management in the context of digital transformation.

Connection of work with scientific programs, plans, topics. The master's thesis was carried out at the National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute" in accordance with the research plans of the Department of Economic Cybernetics on the topic "Analytics and Modelling of Economic Development within the Imperatives of Trans Tech" (No. 0123U101760). The author's role involved determining the optimal stock portfolio, conducting a comprehensive analysis of portfolio theory, optimization methods, and time series forecasting techniques.

The purpose of study of the master's thesis is a comprehensive analysis of the theoretical foundations of portfolio theory, the development and application of optimization models for creating an efficient investment portfolio, as well as conducting predictive analysis of stock price dynamics using classical machine learning methods and deep learning

algorithms. Special attention is given to the comparative analysis of results obtained using different approaches, followed by an economic evaluation and the formulation of well-founded investment decisions and recommendations for creating an optimal portfolio.

Research objectives: reveal the essence and content of investment projects and portfolio theory; study predictive analytics methods of classical machine learning for working with time series; select and justify approaches to determining the composition of an optimal investment portfolio. Conduct analysis and evaluation of optimization methods for forming an optimal portfolio; explore analytical tools for time series forecasting; perform analytical modelling to assess the effectiveness of investment projects. Conduct predictive analytics for the implementation of an investment portfolio using time series analysis methods; apply analytical machine learning methods to analyse and forecast the outcomes of investment project implementation; perform a comparative analysis and develop recommendations for forming an optimal investment portfolio based on economic-mathematical modelling and data analysis.

The object of the study: investment activity projects of the enterprise, in the form of forming a stock portfolio.

The subject of the study: analytical methods and practical aspects of using modern approaches to formulating an optimal investment portfolio, particularly the application of time series forecasting algorithms and big data analysis methods. Special attention is given to machine learning, deep learning (Deep Learning), and Data Science tools for forecasting financial indicators and creating effective investment decisions in the context of the digital transformation of the economy. Additionally, an important aspect is the comparative analysis of the obtained results, which forms the foundation for conducting economic analytics of the outcomes.

Research methods: analytical, empirical, induction, deduction, optimization, forecasting, comparative.

Scientific novelty of the results. The elements of scientific novelty in the master's thesis include a comparative analysis of two types of predictive-analytical tools: stock prices

and the covariance matrix. The approach to building ARIMA models was improved through additional step-by-step training.

Practical significance of the results. The developed analytical-mathematical tools, including RNN, DNN, and ARIMA models, have significant practical applications in forming an optimal investment portfolio. Their use, combined with the improved portfolio optimization model, ensures the achievement of effective results that can be successfully implemented in practical activities.

Approbation of work results: Presentation at a conference with the publication of abstract: Krykun Y., Lazarenko I., Data mining techniques for portfolio building. Modelling and forecasting economic processes: collection of abstracts of the XVIII International Scientific and Practical Conference, Kyiv, December 5, 2024. Kyiv: Igor Sikorsky Kyiv Polytechnic Institute, Publishing House "Polytechnica," 2024. URL: <https://mpeproc.fmm.kpi.ua>

Publications: Article (Category B journal): Lazarenko I., Krykun Y., Portfolio management with time series analysis methods. Economic Bulletin of National Technical University of Ukraine "Kyiv Polytechnic Institute." 2024. No. 29. URL: <https://ev.fmm.kpi.ua/article/view/309267/300787>

Keywords: economic analytics, forecasting, machine learning, time series, optimization, investment project, investment portfolio.

ЗМІСТ

ВСТУП	13
1 ТЕОРЕТИКО-МЕТОДИЧНІ ПІДХОДИ ДО ОПТИМІЗАЦІЙНОЇ АНАЛІТИКИ У ФОРМУВАННІ ІНВЕСТИЦІЙНОГО ПОРТФЕЛЮ ПІДПРИЄМСТВА	17
1.1 Сутність, зміст інвестиційних проєктів та портфельної теорії.....	17
1.2 Методи прогностичної аналітики класичного машинного навчання для вдосконалення портфельної теорії	22
1.3 Сучасні технології аналізу та визначення наповнення оптимального інвестиційного портфелю	31
Висновки до першого розділу	39
2 АНАЛІТИЧНО-МАТЕМАТИЧНИЙ ІНСТРУМЕНТАРІЙ ФОРМУВАННЯ ОПТИМАЛЬНОГО ІНВЕСТИЦІЙНОГО ПОРТФЕЛЮ	41
2.1 Аналіз та оцінка класичних методів моделювання оптимального портфелю цінних паперів.....	41
2.2 Аналітично-математичний інструментарій визначення оптимального портфелю з використанням класичних методів аналізу часових рядів	48
2.3 Аналітичне моделювання для визначення ефективності інвестиційних проєктів з використанням методів машинного навчання	56
Висновки до другого розділу	63
3 ПРОГНОСТИЧНА ЕКОНОМІЧНА АНАЛІТИКА ОПТИМАЛЬНИХ ІНВЕСТИЦІЙНИХ ПОРТФЕЛЕЙ	66
3.1 Прогностична аналітика реалізації інвестиційного портфелю з використанням методів аналізу часових рядів.....	66
3.2 Аналітичних методи машинного навчання для прогнозування результатів реалізації інвестиційних проєктів.....	78
3.3 Компаративний аналіз та рекомендації для формування оптимального інвестиційного портфелю	105
Висновки до третього розділу	116
ВИСНОВКИ	118
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	121
ДОДАТКИ	125
ДОДАТОК А	125

ВСТУП

Актуальність. Ефективне управління портфелем інвестицій є однією з центральних задач сучасного економічного розвитку. Умови глобалізації, швидкоплинних змін на фінансових ринках, зростаючого рівня невизначеності та розвитку цифрових технологій створюють нові виклики для інвесторів. Використання інноваційних підходів, таких як аналіз великих даних (Big Data), машинне навчання (Deep Learning) та інструменти Data Science, стає необхідною умовою для підвищення точності прогнозів, зменшення ризиків та оптимізації структури інвестиційного портфеля. Особливе значення мають алгоритми прогнозування та автоматизованої оптимізації, які забезпечують адаптацію до швидких змін у зовнішньому середовищі. Дослідження спрямоване на розробку методологій, здатних поєднувати теоретичні основи інвестиційного аналізу з практичними рішеннями для ефективного управління портфелем в умовах цифрової трансформації економіки.

Питання оптимізації інвестиційного портфеля вперше було ґрунтовно досліджено Гаррі Марковіцем у 1952 році, що стало основою сучасної портфельної теорії. Подальший розвиток цієї галузі забезпечив Вільям Шарп, який розробив модель оцінки капітальних активів (CAPM), закріпивши фундаментальні принципи оцінки ризику та доходності. Що стосується технологій глибокого навчання (Deep Learning), вагомий внесок у їхній розвиток зробили Ендрю Ін та Ян Лекус, чії дослідження стали визначальними у формуванні теоретичних і практичних підходів до цієї галузі.

Магістерська дисертація виконувалась в Національному технічному університеті України «Київський політехнічний інститут імені Ігоря Сікорського» відповідно до планів наукових досліджень кафедри економічної кібернетики за темою «Аналітика та моделювання економічного розвитку в імперативах Trans Tech». (№ 0123U101760). Роль автора полягає у визначенні оптимального портфелю акцій,

ґрунтовному аналізі портфельної теорії, оптимізаційних методів та методів прогнозування часових рядів.

Метою магістерської дисертації є ґрунтовний аналіз теоретичних засад портфельної теорії, розробка та застосування оптимізаційних моделей для формування ефективного інвестиційного портфеля, а також проведення прогнозного аналізу динаміки історичних цін акцій із використанням методів класичного машинного навчання та алгоритмів глибокого навчання. Особливу увагу приділено порівняльному аналізу результатів, отриманих різними підходами, із подальшим проведенням економічної оцінки й формуванням обґрунтованих інвестиційних рішень для створення оптимального портфеля.

Відповідно до поставленої мети було вирішено такі завдання:

- розкрито сутність, зміст інвестиційних проєктів та портфельної теорії;
- досліджено методи прогностичної аналітики класичного машинного навчання для роботи із часовими рядами;
- здійснено підбір та обґрунтувати підходи то визначення наповнення оптимального інвестиційного портфелю.
- проведено аналіз та оцінку оптимізаційних методів для формування оптимального портфелю;
- досліджено аналітичний інструментарій прогнозування часових рядів;
- проведено аналітичне моделювання для визначення ефективності інвестиційних проєктів.
- проведено прогностичну аналітику реалізації інвестиційного портфелю з використанням методів аналізу часових рядів.
- використано аналітичні методи машинного навчання для аналізу та прогнозування результатів реалізації інвестиційних проєктів
- проведено компаративний аналіз та розробити рекомендації для формування оптимального інвестиційного портфелю на основі економіко-математичного моделювання та аналізу даних.

Об'єктом дослідження є проекти інвестиційної діяльності підприємства, у вигляді формування портфелю акцій.

Предметом дослідження є аналітичні методи та практичні аспекти використання сучасних підходів до формування оптимального інвестиційного портфелю, зокрема застосування алгоритмів прогнозування часових рядів і методів аналізу великих даних. Особлива увага приділена інструментам машинного навчання, глибокого навчання (Deep Learning) і Data Science для прогнозування фінансових показників та формування ефективних інвестиційних рішень в умовах цифрової трансформації економіки. В додаток до цього, важливим аспектом є порівняльний аналіз отриманих результатів, який є підґрунтям проведення економічної аналітики отриманих результатів.

База досліджень: базою дослідження є фондовий ринок США, на якому торгуються зазначені портфельній компанії.

Методи дослідження. Для виконання аналітичних завдань використовувалися як теоретичні, так і емпіричні методи. Метод індукції застосовувався для формування загальних висновків на основі аналізу конкретних даних, тоді як метод дедукції використовувався для перевірки гіпотез, сформованих на основі теоретичних засад. Прогностичні та оптимізаційні методи слугували інструментом для побудови аналітично-математичного інструментарію і визначення оптимальних рішень. Емпіричний підхід у поєднанні з компаративним аналізом дозволив визначити альтернативні рішення та обґрунтувати вибір оптимального. Для проведення розрахунків, моделювання, аналітики та візуалізації економічних даних використано інформаційні технології: Python та прикладні бібліотеки, Excel.

Наукова новизна. Елементами наукової новизни магістерської дисертації є:

- порівняльний аналіз двох видів прогностично-аналітичного інструментарію: цін акцій та коваріаційної матриці;
- удосконалено підхід побудови ARIMA моделі шляхом додаткового покрокового навчання.

Практичне значення одержаних результатів. Розроблений аналітично-математичний інструментарій, зокрема RNN, DNN та ARIMA моделі, мають вагоме практичне застосування у формуванні оптимального інвестиційного портфеля. Їх використання, у поєднанні з удосконаленою моделлю оптимізації портфеля, забезпечує досягнення ефективних результатів, що можуть бути успішно впроваджені у практичну діяльність.

Апробація результатів магістерської дисертації:

Крыкун У. Lazarenko I., Data mining techniques for portfolio building. Моделювання та прогнозування економічних процесів: зб. тез доп. XVIII Міжнар. наук.-практ. конф., м. Київ, 5 груд. 2024 р. Київ: КПІ ім. Ігоря Сікорського, Вид-во «Політехніка», 2024. URL: <https://mreproc.fmm.kpi.ua>

Публікації. Стаття (видання категорії Б):

Lazarenko I., Krykun U. Portfolio management with time series analysis methods Economic bulletin of National technical university of Ukraine «Kyiv polytechnical institute». 2024. No 29. URL: <https://ev.fmm.kpi.ua/article/view/309267/300787>

1 ТЕОРЕТИКО-МЕТОДИЧНІ ПІДХОДИ ДО ОПТИМІЗАЦІЙНОЇ АНАЛІТИКИ У ФОРМУВАННІ ІНВЕСТИЦІЙНОГО ПОРТФЕЛЮ ПІДПРИЄМСТВА

1.1 Сутність, зміст інвестиційних проєктів та портфельної теорії

Формування оптимального портфеля акцій є однією з ключових задач для інвесторів, які прагнуть досягти максимальної дохідності за заданого рівня ризику. Оптимальний портфель зазвичай складається з цінних паперів, що належать до різних економічних секторів або географічних регіонів.

Такий портфель дозволяє інвестору максимізувати дохідність своїх інвестицій. Різні акції мають різний потенціал дохідності, а диверсифікація інвестицій між ними підвищує ймовірність отримання високих прибутків. Окрім цього, портфельна диверсифікація сприяє зниженню загального ризику інвестицій. Завдяки розподілу вкладень між активами з різними рівнями ризику, зменшується вплив специфічних ризиків, пов'язаних із окремими акціями чи секторами економіки.

Формування оптимального портфеля також дозволяє скористатися перевагами диверсифікації. Розподіл активів між некорельованими або слабо корельованими акціями сприяє зниженню впливу несприятливих подій на результати інвестування. Одним із підходів до формування такого портфеля є пасивна інвестиційна стратегія, яка базується на реплікації ринкових показників або використанні індексних фондів. Цей підхід виходить із припущення, що ринок загалом забезпечує зростання прибутковості з часом, і завдання інвестора — отримати частку цього зростання, розподіляючи капітал відповідно до вагових коефіцієнтів ринкового індексу.

Важливий внесок у розвиток концепції оптимального портфеля зробив американський економіст Гаррі Марковіц, який у 1952 році представив свою дисертацію на тему «Вибір портфеля». Його Сучасна портфельна теорія (Modern Portfolio Theory, МРТ) і досі залишається однією з найпопулярніших інвестиційних

стратегій, яка є альтернативою традиційному підходу до вибору акцій. МРТ забезпечує науково обґрунтовані інструменти управління портфелем, що, за правильного використання, сприяють створенню диверсифікованого та прибуткового інвестиційного портфеля [1, 2].

Розглянемо ключові аспекти сучасної портфельної теорії (МРТ), її переваги та недоліки. МРТ (Modern Portfolio Theory) — це теорія фінансових інвестицій, яка застосовує статистичні методи для оптимального розподілу ризику портфеля цінних паперів і оцінки його доходності. Її основні складові включають оцінку активів (sekuriti valuation), прийняття інвестиційних рішень (asset allocation decision), оптимізацію портфеля (portfolio optimization) і вимірювання результативності (performance measurement).

Попри абстрактність і нехтування деякими практичними аспектами, такими як податки, операційні витрати, нескінченна подільність активів і рівень поінформованості інвесторів, МРТ та її похідні теорії — CAPM і арбітражна теорія — мають значну практичну цінність. Вони забезпечують базові знання для управління доходністю та ризиками портфеля цінних паперів [2, 3].

На досконало функціонуючому ринку цінних паперів фахівець із управління портфелем має можливість аналізувати ринкові тенденції, прогнозувати майбутню динаміку та оцінювати інвестиційні властивості фінансових інструментів. Теоретичні знання відіграють ключову роль у виявленні фундаментальних закономірностей ринку цінних паперів і дозволяють застосовувати ефективні критерії на практиці. Згідно з припущеннями сучасної портфельної теорії, варто відзначити, що розвинені фінансові ринки характеризуються високим рівнем інформованості фінансових інститутів і відносно низькими операційними витратами у порівнянні з обсягами здійснюваних угод. У зв'язку з цим у певних випадках цими факторами можна знехтувати [2, 3].

Модель Марковіца пропонує підхід до формування портфеля як комбінації доступних інвестиційних активів. Основна ідея полягає в пошуку оптимального

розподілу інвестицій між фінансовими інструментами, щоб досягти мінімального рівня ризику при заданому рівні доходності або, навпаки, максимізувати очікуваний дохід за прийнятного рівня ризику. Ця модель дає змогу визначити ефективний портфель, який забезпечує найнижчий ризик за певної доходності. Водночас слід зазначити, що модель Марковіца представляє спектр ефективних портфелів, а не єдине універсальне рішення.

Застосування цієї теорії має значний практичний сенс для інвесторів, оскільки вона допомагає раціонально підходити до побудови портфеля, враховуючи баланс між ризиком і очікуваним доходом. Оптимізація портфеля дає можливість зменшити ризику та підвищити потенційний дохід. Крім того, інтеграція моделі Марковіца та сучасних принципів портфельного управління сприяє прийняттю зважених інвестиційних рішень і забезпечує ефективний контроль над активами.

Основні припущення, на яких базується модель Марковіца, можна викласти наступним чином:

- у якості показника доходності цінних паперів приймається математичне очікування, яке є усередненим значенням доходності за попередніми періодами. Це припущення відображає ідею, що середнє значення доходу є ключовим параметром для оцінки перспектив інвестицій;
- ризик інвестицій визначається через стандартне (середньоквадратичне) відхилення доходності. Цей підхід дозволяє кількісно оцінити варіативність результатів і, відповідно, потенційні відхилення від очікуваної доходності;
- вважається, що історичні дані, які використовуються для розрахунків доходності і ризику, цілком адекватно відображають тенденції майбутніх періодів. Іншими словами, передбачається стабільність статистичних характеристик доходності цінних паперів у часі;
- ступінь взаємодії між різними цінними паперами описується через коефіцієнт лінійної кореляції. Цей показник дозволяє оцінити, наскільки зміна

дохідності одного активу впливає на дохідність іншого, що є важливим для формування оптимального портфеля [2, 4].

Надійний інвестиційний портфель, згідно з концепцією Марковіца, формується шляхом вибору певної кількості цінних паперів. Кожен з цих фінансових інструментів характеризується власною очікуваною нормою прибутковості, що визначає його потенціал для створення доходу. Такий підхід дозволяє зменшити ризики за рахунок диверсифікації, тобто розподілу капіталу між різними видами активів.

У сфері фінансів дохід визначається як сукупність прибутків, отриманих від інвестиційної діяльності. Це поняття охоплює зміни ринкової вартості активів, а також грошові надходження, які інвестор отримує у процесі володіння цінними паперами або іншими видами активів. До таких надходжень можуть належати відсоткові платежі за борговими зобов'язаннями, купонні виплати за облігаціями, грошові дивіденди за акціями, дивіденди у формі додаткових акцій, а також виплати, пов'язані з похідними фінансовими інструментами або структурованими продуктами. Дохід можна вимірювати у двох основних формах: в абсолютних значеннях, наприклад у грошовому еквіваленті (доларах, євро тощо), або у відсотках від початково вкладеної суми, що є більш універсальним показником. В останньому випадку цей параметр часто називають нормою прибутковості за період володіння активом.

Коли інвестиційна діяльність приносить не прибуток, а збитки, говорять про від'ємну прибутковість. Така ситуація виникає, якщо ринкова вартість активів знижується або отримані грошові потоки не покривають інвестовану суму. При цьому від'ємна прибутковість також може вимірюватися в абсолютному вираженні чи у відсотках.

Для аналізу та порівняння результатів інвестування за періоди часу різної тривалості використовується стандартизація показників дохідності. Цей процес дозволяє звести дохідність кожного активу до еквівалента за певний стандартний

період, найчастіше один рік. У результаті такої трансформації отримується ставка дохідності за стандартний період, яку в разі річного обчислення називають річною нормою прибутковості. Сам процес перерахунку носить назву річницізації, і він є важливим інструментом для прийняття інвестиційних рішень.

Рентабельність інвестицій (ROI) - це прибуток на вкладений долар. Це показник ефективності інвестицій, а не їхнього розміру (наприклад, рентабельність власного капіталу, рентабельність активів, рентабельність задіяного капіталу).

Існує безліч методів для розрахунку норми прибутку, але одним із найпоширеніших є використання логарифмічної норми прибутку [5]:

$$r_t = \ln\left(\frac{P_t}{P_{t-1}}\right), \quad (1.1)$$

де r_t – норма прибутку на момент часу t ;

P_t – ціна активу на момент часу t ;

P_{t-1} – ціна активу на момент часу $t-1$.

Очікувана дохідність фінансового інструменту визначається як середнє значення норм прибутку за попередні часові інтервали [6, с.111]:

$$R = \frac{1}{t} \sum_{i=1}^t r_i, \quad (1.2)$$

де R – середня дохідність за вибраний період.

Дохідність інвестиційного портфеля за моделлю Марковіца обчислюється як сума добутків очікуваної дохідності окремих активів і їх вагових коефіцієнтів у портфелі [7]:

$$R_p = \sum_i R_i w_i, \quad (1.3)$$

де R_p — загальна дохідність портфеля;

R_i — очікувана дохідність кожного активу;

w_i — частка активу в загальному портфелі.

Ризик окремого фінансового інструменту визначають через стандартне відхилення норм прибутку [6, с.111]:

$$\sigma_i = \sqrt{\frac{1}{t-1} \sum_{i=1}^t (r_i - \bar{r})^2}, \quad (1.4)$$

Ризик усього портфеля визначають за такими рівняннями [8, 9]:

$$\begin{aligned} \sigma_p^2 &= E \left[\left(R_p - E(R_p) \right)^2 \right] = E \left[\left(\sum_i R_i w_i - \sum_i E(R_i) w_i \right)^2 \right] = \\ &= E \left[\left(\sum_i w_i (R_i - E(R_i)) \right)^2 \right] = E \left[\sum_i \sum_j w_i w_j (R_i - E(R_i)) (R_j - E(R_j)) \right] =, \quad (1.5) \\ &= \sum_i \sum_j w_i w_j E \left[(R_i - E(R_i)) (R_j - E(R_j)) \right] = \sum_i \sum_j w_i w_j \text{cov}(R_i, R_j) = \\ &= \sum_i \sum_j w_i w_j \sigma_{ij} \end{aligned}$$

де σ_p^2 — дисперсія доходності портфеля.

$$\sigma_p = \sqrt{\sigma_p^2}, \quad (1.6)$$

В межах класичної теорії Марковіца інвестор може обирати між двома основними стратегіями. З одного боку, можлива максимізація очікуваного прибутку для досягнення найвищого доходу від вкладень. З іншого боку, інвестор може прагнути мінімізувати ризик портфеля, особливо якщо він не схильний до ризикових інвестицій.

1.2 Методи прогностичної аналітики класичного машинного навчання для вдосконалення портфельної теорії

Модель Марковіца (або теорія портфеля Марковіца) є фундаментальним інструментом фінансового аналізу, який дозволяє інвесторам формувати оптимальний

портфель активів, мінімізуючи ризик при заданому рівні очікуваного доходу. Однак ця модель має важливе обмеження: вона розглядає лише поточний період часу, базуючи свої розрахунки на історичних даних, без врахування можливих змін у майбутньому. Це означає, що Модель Марковіца не враховує динаміку ринків і можливість прогнозування майбутньої поведінки активів, що може суттєво знизити її ефективність у реальних умовах.

Для подолання цього недоліку та побудови портфеля з урахуванням перспектив розвитку ринку можна застосовувати класичні статистичні методи прогнозування, такі як ARIMA та GARCH.

Для побудови прогнозованої коваріаційної матриці з використанням моделі ARIMA необхідно провести серію кроків, спрямованих на прогнозування майбутніх цін акцій. Завдяки цьому підходу можна передбачити майбутні значення цін, які є ключовими в оцінці динаміки фінансових інструментів. Прогнозування цін акцій за допомогою ARIMA дозволяє отримати очікувані значення майбутньої дохідності кожного активу. Це є критичним компонентом у процесі побудови ефективного портфеля, оскільки прогнозована дохідність є основою для розрахунку очікуваної норми прибутку. З використанням прогнозованих цін можна оцінити ризик через аналіз волатильності. Це стає можливим завдяки побудові прогнозованої коваріаційної матриці, яка враховує взаємозв'язки між різними активами. Відповідно, прогнозована коваріаційна матриця дозволяє більш точно оцінити систематичний і несистематичний ризик портфеля, що є необхідним для прийняття зважених інвестиційних рішень.

Щоб більш детально оцінити всі можливості ARIMA, перейдемо до її загального визначення та принципу її роботи.

Моделі ARIMA, акронім від AutoRegressive Integrated Moving Average, є фундаментальною статистичною структурою, яка широко використовується для аналізу та прогнозування часових рядів. Ці моделі були вперше представлені Боксом і Дженкінсом у їхній знаковій праці 1970-х років, де вони виклали систематичну

методологію побудови та діагностики моделей, придатних для роботи з даними, залежними від часу. З моменту свого створення моделі ARIMA здобули значну популярність у різних дисциплінах, включаючи економіку, фінанси, науки про довкілля та метеорологію, завдяки своїй універсальності та ефективності у відображенні динамічної поведінки часових рядів.

Структура моделей ARIMA поєднує три основні компоненти: авторегресійний (AR), інтегрований (I) і ковзного середнього (MA). Авторегресійний компонент відображає залежність між поточним спостереженням і кількома попередніми значеннями, фіксуючи стійкість ефектів у часі. Компонент ковзного середнього моделює вплив попередніх помилок прогнозування на поточні значення, дозволяючи враховувати нерегулярні коливання в даних. Інтеграційний компонент забезпечує стаціонарність шляхом різницювання даних, що є критичним для обробки часових рядів із трендами або сезонністю. Разом ці елементи дозволяють моделям ARIMA захоплювати як систематичні, так і стохастичні характеристики часового ряду, забезпечуючи надійну основу для прогнозування та розпізнавання закономірностей.

Особливістю моделей ARIMA є їхня адаптивність до різних типів даних. Шляхом налаштування параметрів, пов'язаних із компонентами AR, I та MA, модель можна підлаштувати під широкий спектр характеристик часових рядів. Ця гнучкість робить ARIMA особливо цінною у таких сферах, як фінанси, де точність прогнозування цін активів або доходності вимагає здатності моделювати складні зв'язки та волатильність [8].

Моделі ARIMA (AutoRegressive Integrated Moving Average) є статистичними інструментами, які використовуються для аналізу та прогнозування часових рядів. Їх основний принцип базується на концепції стаціонарності — ключової характеристики часових рядів, за якої статистичні властивості, такі як середнє значення, дисперсія та структура автокореляції, залишаються стабільними з часом. Стаціонарні часові ряди легше аналізувати та моделювати, оскільки їх властивості не змінюються, що дозволяє виявляти закономірності, які можна використовувати для прогнозування. У випадках,

коли часовий ряд не є стаціонарним, моделі ARIMA застосовують ряд трансформацій для досягнення стаціонарності, що дозволяє ефективно проводити моделювання. [8, 9].

Моделі ARIMA поєднують три ключові компоненти, які дозволяють моделювати та прогнозувати часові ряди. Ці компоненти працюють у взаємодії, виявляючи та моделюючи зв'язки в даних, що в кінцевому підсумку підвищує точність прогнозування:

- Авторегресійний компонент (AR) моделі ARIMA описує залежність між поточним спостереженням і попередніми, так званими лаговими спостереженнями. Цей компонент позначається параметром «р», який визначає кількість попередніх спостережень, що включаються в модель для прогнозування поточного значення. Авторегресійна модель, $AR(p)$, використовує історичні дані для врахування тимчасових залежностей, надаючи уявлення про те, як минулі значення впливають на теперішні. Зокрема, авторегресія є статистичною моделлю, в якій змінна, що змінюється, регресує на свої попередні значення, створюючи основу для розуміння безперервності чи загасання часових закономірностей;

- Компонент ковзного середнього (MA) у моделях ARIMA є важливим елементом, який сприяє підвищенню точності прогнозування часових рядів. Позначений параметром «q», компонент MA розроблено для моделювання залишкових похибок у часовому ряді після врахування авторегресійного компонента. Фактично, він включає лінійну комбінацію попередніх похибок прогнозу для коригування передбачень, що дозволяє моделі точніше враховувати динаміку даних. Процес $MA(q)$ орієнтований на використання ковзного середнього попередніх «q» похибок для опису короткострокових коливань та випадкових викидів у наборі даних. Ці похибки вважаються непередбачуваними порушеннями або шумом, які впливають на часовий ряд. Аналізуючи та враховуючи ці залишкові значення, компонент MA забезпечує механізм згладжування даних і підвищує точність прогнозування майбутніх значень. Цей ефект згладжування є особливо корисним у випадках, коли

часовий ряд демонструє раптові зміни або нерегулярні патерни, які не можуть бути пояснені виключно авторегресійними термінами чи детермінованими компонентами ряду. З математичної точки зору, процес $MA(q)$ передбачає, що поточне спостереження можна виразити як зважену суму попередніх похибок. Це робить його універсальним інструментом для аналізу часових рядів, здатним враховувати стохастичну природу фінансових і економічних даних;

- Інтегрований компонент (I) у моделях ARIMA відіграє важливу роль у підготовці часових рядів до аналізу, усуваючи їх нестійкі характеристики. Цей компонент позначається параметром «d», який вказує кількість операцій диференціювання, застосованих до даних. Диференціювання є математичним перетворенням, що полягає у відніманні поточного спостереження від його попереднього значення, що дозволяє усунути довгострокові тренди або сезонність, які можуть приховувати основні закономірності. Завдяки усуненню цих трендів інтегрований компонент перетворює дані у стаціонарний часовий ряд, що є необхідною умовою для більшості методів моделювання часових рядів, зокрема ARIMA. Правильний вибір параметра диференціювання «d» є критично важливим, оскільки недостатнє диференціювання може залишити у даних залишкові тренди, а надмірне диференціювання може ускладнити модель і знизити її ефективність [8, 9].

Існує кілька основних типів моделей ARIMA, які часто використовуються в аналізі часових рядів. Кожна з цих моделей надає специфічну структуру для розуміння і прогнозування часових даних залежно від їхніх властивостей, таких як стаціонарність, тренди та автокореляція:

- $ARIMA(1,0,0)$, або модель першого порядку авторегресії, є особливо корисною для прогнозування цін акцій, коли часовий ряд є стаціонарним і демонструє автокореляцію. У цьому випадку майбутня ціна акції може бути змодельована як лінійна функція її попереднього значення у поєднанні з константою. Наприклад, якщо завтрашню ціну закриття акції можна передбачити, використовуючи лише сьогоднішню ціну закриття та певну корекцію, то цей процес описується як модель

першого порядку авторегресії. Це передбачає, що минулі значення акцій прямо впливають на їхні майбутні ціни, що дозволяє враховувати короткострокові залежності в даних;

- $ARIMA(0,1,0)$, яка представляє модель випадкового блукання, широко використовується для нестационарних рядів цін акцій. У моделі випадкового блукання кожна наступна ціна акції визначається модифікацією попередньої ціни з додаванням випадкових коливань. Незважаючи на свою простоту, ця модель ефективно відображає випадковий характер рухів цін, який часто спостерігається на фінансових ринках. Наприклад, зміни в цінах акцій часто зумовлені стохастичними процесами, такими як реакція ринку на новини або настрої інвесторів, а не детермінованими трендами. Це робить модель випадкового блукання природним вибором для моделювання диференційованих рядів цін акцій, де минулі зміни впливають на майбутні коливання;

- $ARIMA(1,1,0)$, або модель диференційованої авторегресії першого порядку, розширює можливості моделі випадкового блукання для врахування автокореляції помилок. У прогнозуванні цін акцій ця модель додає лаг диференційованої ціни до рівняння прогнозування, що дозволяє вирішити проблему серійної кореляції та підвищити точність прогнозів. Наприклад, якщо щоденна зміна ціни акції залежить не лише від випадкових ринкових факторів, але й від зміни, зафіксованої попереднього дня, модель $ARIMA(1,1,0)$ ефективно враховує цю залежність;

- $ARIMA(0,1,1)$ без константи відповідає простому експоненціальному згладжуванню. Цей підхід особливо підходить для акцій, ціни яких не демонструють сильної сезонності або трендів. У цій моделі коливання цін згладжуються за допомогою одного параметра згладжування, який контролює вагу історичних спостережень. У контексті прогнозування акцій параметр, близький до 1, приділяє більше уваги нещодавнім рухам цін, тоді як менші значення враховують ширшу

історичну перспективу. Це дозволяє моделі адаптуватися до різних патернів даних, що робить її ефективною для прогнозування стабільних, мало волатильних акцій;

- ARIMA(0,1,1) із константою розширює модель простого експоненціального згладжування, додаючи компонент росту. Константа забезпечує, що прогнозовані ціни акцій відображають поступові зміни з часом, що особливо важливо для компаній, які демонструють стабільне зростання або спад через фундаментальні бізнес-фактори, такі як збільшення частки ринку або фінансові труднощі [10].

Аналіз часових рядів розпочинається з перевірки стаціонарності даних, адже це є ключовою умовою для коректного моделювання та прогнозування. Часовий ряд зазвичай складається з трьох систематичних компонентів — рівня, тренду та сезонності, а також однієї несистематичної компоненти, яку називають шумом.

Стаціонарність означає, що ряд не демонструє трендів, сезонності чи змін у варіабельності протягом часу. Зокрема:

- середнє значення ряду залишається постійним;
- дисперсія не змінюється у часі;
- автоковаріація між двома точками залежить лише від лагу (різниці у часі), а не від конкретних часових моментів [11].

Нестаціонарні ряди, як правило, характеризуються трендами чи сезонними паттернами. Для досягнення стаціонарності необхідно виконати їх коригування за допомогою таких методів, як диференціювання, логарифмічне перетворення або десезоналізація.

Одним із найпоширеніших статистичних методів перевірки стаціонарності є тест Діккі-Фуллера (ADF). Цей тест оцінює наявність одиничного кореня у часовому ряді, що є ознакою нестаціонарності. Гіпотези тесту формулюються таким чином:

- нульова гіпотеза (H_0): Часовий ряд має одиничний корінь (є нестаціонарним);

- альтернативна гіпотеза (H_1): Часовий ряд не має одиничного кореня (є стаціонарним).

Якщо нульова гіпотеза не відхиляється, ряд вважається нестационарним. Навпаки, відхилення нульової гіпотези вказує на стаціонарність ряду [11].

GARCH модель має дещо інший підхід до оптимізації портфелю. Ознайомимося із основними поняттями та історією розробки такого статистичного методу.

Модель узагальненої авторегресивної умовної гетероскедастичності (GARCH) є одним із найвпливовіших інструментів економетрики для аналізу та прогнозування мінливості, що змінюється з часом. Її запропонував Тім Боллерслев у 1986 році як узагальнення раніше розробленої Робертом Ф. Енглем моделі ARCH. GARCH забезпечує структуру для опису та прогнозування кластеризації й стійкості мінливості, що часто спостерігається на фінансових ринках. Адаптивність і точність цієї моделі роблять її незамінним інструментом сучасного фінансового аналізу [12].

Модель GARCH вирішує проблему гетероскедастичності – ситуації, коли дисперсія помилок не є сталою з часом [13]. Традиційні економетричні моделі зазвичай припускають постійну дисперсію (гомоскедастичність), що є нереалістичним для фінансових даних, які характеризуються коливаннями у мінливості. Модель GARCH усуває це обмеження, роблячи умовну дисперсію залежною від попередніх спостережень і дисперсій, що дозволяє враховувати зміну мінливості з часом.

Моделі GARCH широко застосовуються у фінансах для різних цілей. Вони дозволяють прогнозувати майбутню ринкову волатильність, що сприяє розробці торгових стратегій та практик управління ризиками [14]. Крім того, моделі GARCH допомагають оцінювати такі показники ризику, як Value at Risk (VaR), що є важливим для оцінки потенційних втрат в інвестиційних портфелях [15]. Включення змінної волатильності в моделі ціноутворення активів дозволяє точніше відображати динаміку ринку, що веде до більш обґрунтованих інвестиційних рішень [14].

Моделі GARCH мають свої переваги та обмеження. Вони ефективно відображають кластеризацію волатильності, що є поширеною особливістю фінансових часових рядів, і надають структуру для моделювання змінної волатильності, підвищуючи точність фінансових прогнозів. Однак моделі GARCH передбачають певну функціональну форму динаміки волатильності, яка може не повністю відображати всі ринкові поведінки, а оцінка параметрів може бути складною і вимагати великих обсягів даних для надійних результатів.

Для подолання певних обмежень та відображення більш складних ринкових поведінок було розроблено кілька розширень моделі GARCH. Експоненціальна GARCH (EGARCH) дозволяє враховувати асиметричні ефекти позитивних і негативних шоків на волатильність, що враховує «ефект важеля», коли негативні доходи збільшують майбутню волатильність більше, ніж позитивні доходи тієї ж величини [16]. Інтегрована GARCH (IGARCH) вводить одиничний корінь у процес GARCH, дозволяючи стійким шокам впливати на волатильність, що корисно при моделюванні довгострокових залежностей у фінансових часових рядах. Порогова GARCH (TGARCH), також відома як модель GJR, враховує асиметрії, дозволяючи різні реакції волатильності на позитивні та негативні шоки, ефективно моделюючи такі явища, як ефект важеля [16].

Прогнозування коваріаційної матриці за допомогою моделей GARCH є важливим інструментом в управлінні фінансовими ризиками та оптимізації інвестиційних портфелів. Моделі GARCH дозволяють враховувати часову змінність волатильності та кореляцій між активами, що сприяє більш точному оцінюванню ризиків і прийняттю обґрунтованих інвестиційних рішень.

Коваріаційна матриця відображає взаємозв'язки між доходностями різних активів у портфелі. Традиційні методи оцінки коваріаційної матриці, такі як використання історичних даних або експоненційно зваженого середнього, можуть не враховувати динамічні зміни волатильності та кореляцій у часі. Моделі GARCH, зокрема їх багатовимірні варіанти (MGARCH), дозволяють моделювати та

прогнозувати часову змінність коваріаційної матриці, що забезпечує більш точні оцінки ризиків.

У практиці фінансового аналізу моделі GARCH широко використовуються для прогнозування волатильності та коваріаційних матриць. Наприклад, дослідження, проведене на основі даних семи акцій, включених до індексу Української біржі, показало ефективність використання ортогональної моделі GO-GARCH для оцінки варіаційно-коваріаційної матриці дохідностей. Це дозволило розрахувати оптимальні ваги в портфелі та покращити його характеристики [17].

1.3 Сучасні технології аналізу та визначення наповнення оптимального інвестиційного портфелю

Прогнозування часових рядів є ключовим компонентом у різних галузях, що дозволяє передбачати майбутні значення на основі історичних даних. Традиційні статистичні методи, такі як ARIMA та експоненціальне згладжування, широко застосовуються для цього завдання. Однак вони часто не справляються зі складними, нелінійними взаємозв'язками, притаманними багатьом реальним наборам даних. Поява глибокого навчання принесла більш досконалі техніки, здатні захоплювати ці складні закономірності, тим самим підвищуючи точність прогнозування.

Моделі глибокого навчання, зокрема рекурентні нейронні мережі (RNN) та їхні варіанти, такі як мережі довготривалої короткочасної пам'яті (LSTM), добре працюють з послідовними даними завдяки здатності зберігати часові залежності. Ці моделі продемонстрували значні покращення порівняно з традиційними методами у різних завданнях прогнозування. Нещодавно архітектури, такі як трансформери, були застосовані до прогнозування часових рядів, використовуючи механізми уваги для ефективного моделювання довготривалих залежностей [18].

Щільні нейронні мережі (Dense Neural Networks, DNN), відомі також як повнозв'язані мережі, є фундаментальними архітектурними в глибокому навчанні, де

кожен нейрон одного шару з'єднаний з кожним нейроном попереднього шару. Ця структура дозволяє DNN моделювати складні, нелінійні взаємозв'язки в даних, що робить їх придатними для різних завдань, включаючи прогнозування фінансових часових рядів.

DNN працює за наступною архітектурою (рис. 1.1):

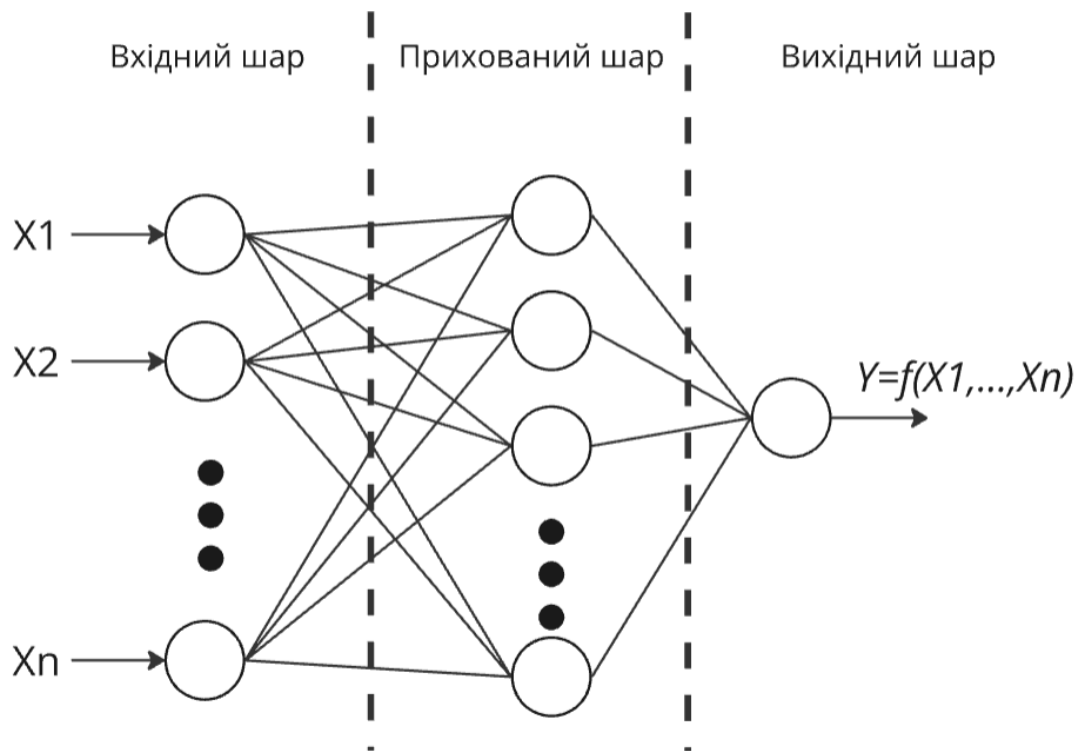


Рисунок 1.1 – Структура DNN

Джерело: складено на основі[19]

У DNN дані проходять через кілька шарів: вхідний шар, один або більше прихованих шарів та вихідний шар. Кожен нейрон застосовує зважену суму своїх вхідних сигналів, пропускає результат через активаційну функцію та передає вихідний сигнал до наступного шару. Ця глибока, повнозв'язана архітектура дозволяє мережі навчатися складним шаблонам шляхом налаштування ваг під час навчання. Щільна зв'язність забезпечує, що інформація від усіх нейронів одного шару впливає на кожен нейрон наступного шару, сприяючи всебічному вивченню ознак [20].

Фінансові часові ряди, такі як ціни акцій, за своєю природою є шумними та демонструють складні часові залежності. DNN здатні захоплювати ці нелінійні взаємозв'язки, сприяючи точному прогнозуванню. Однак стандартні DNN не враховують послідовний характер часових рядів. Для вирішення цього DNN часто поєднують з архітектурами, такими як рекурентні нейронні мережі (RNN) або мережі довготривалої короткочасної пам'яті (LSTM), які призначені для обробки часових залежностей.

Рекурентні нейронні мережі (RNN) - це клас штучних нейронних мереж, які здатні обробляти послідовні дані, зберігаючи певну форму пам'яті завдяки своїм рекурентним зв'язкам. Така архітектура дозволяє RNN фіксувати часові залежності, що робить їх особливо придатними для таких завдань, як прогнозування часових рядів. У контексті прогнозування часових рядів ШНМ використовують свій внутрішній стан для обробки послідовностей даних, ефективно вивчаючи закономірності та залежності з часом. Ця здатність дозволяє їм моделювати складну часову динаміку, притаманну послідовним даним [21].

Продемонструємо роботу RNN на прикладі схеми:

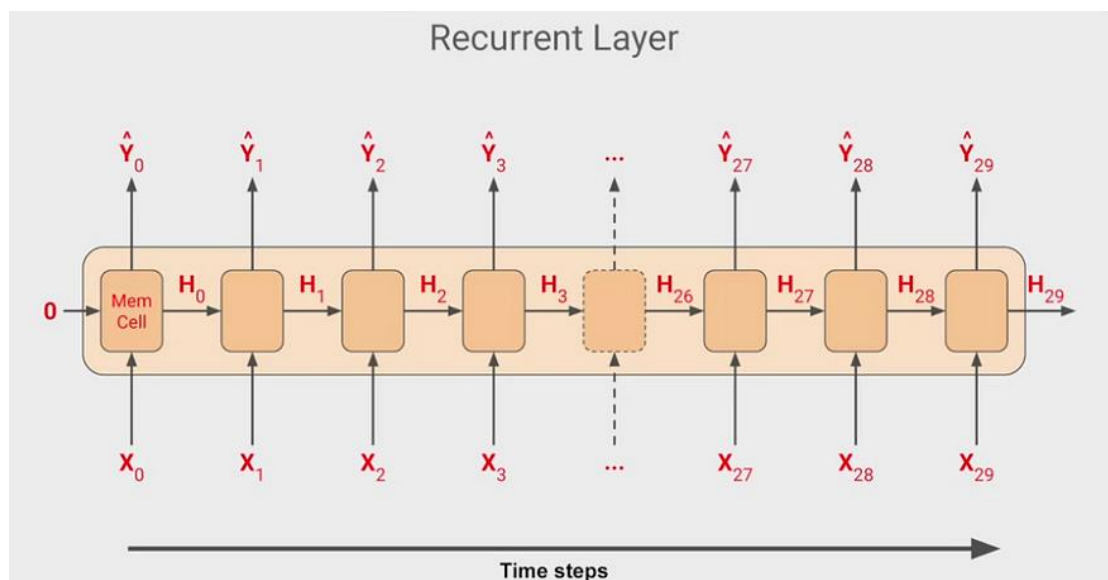


Рисунок 1.2 – Структура RNN шару

Джерело: складено на основі[22]

Зображення (рис. 1.2) ілюструє функціонування рекурентної нейронної мережі (RNN), яка працює з послідовними даними, використовуючи часові кроки. На кожному часовому кроці t мережа отримує вхідний сигнал X_t , що є поточним елементом послідовності. Вхідний сигнал обробляється разом із прихованим станом H_{t-1} , який є результатом обчислень з попереднього кроку. Цей процес дозволяє мережі зберігати контекст і враховувати інформацію з попередніх етапів.

Кожен блок у мережі представляє окремий рекурентний осередок. Осередок приймає два вхідні параметри: поточний вхідний сигнал X_t і прихований стан H_{t-1} , який зберігає «пам'ять» попереднього стану. Осередок обчислює новий прихований стан H_t , комбінуючи ці два параметри через активаційну функцію (зазвичай тангенс або сигмоїда). Новий прихований стан передається на наступний часовий крок, а також може бути використаний для обчислення вихідного сигналу Y_t на поточному етапі.

Часові кроки розташовані послідовно, і мережа обчислює приховані стани H_0, H_1, \dots, H_n до останнього кроку. Це забезпечує передачу інформації через всю послідовність, що дозволяє враховувати залежності між елементами навіть на значній відстані один від одного. У цьому процесі RNN ефективно зберігає «пам'ять» про попередні елементи, водночас адаптуючи її на кожному наступному кроці.

На зображенні видно, що вихідний сигнал Y_t може обчислюватися на кожному часовому кроці. Це залежить від архітектури мережі: вона може видавати вихід лише на останньому кроці або після кожного. Така гнучкість робить RNN придатними для завдань на кшталт обробки тексту, прогнозування часових рядів або аналізу послідовних даних у різних сферах.

Шар SimpleRNN у Keras є базовою архітектурою рекурентних нейронних мереж (RNN), призначеною для обробки послідовних даних шляхом інтеграції часових динамік у моделі нейронних мереж. У цій конфігурації вихід з попереднього кроку часу рекурсивно подається назад у мережу разом із поточним вхідним сигналом. Цей рекурсивний механізм дозволяє мережі зберігати певну форму пам'яті, ефективно

захоплюючи та вивчаючи часові шаблони в даних. Шар SimpleRNN працює, обробляючи вхідні послідовності крок за кроком, оновлюючи свій внутрішній стан на кожному кроці часу, щоб відобразити як минулу, так і поточну інформацію. Цей підхід дозволяє мережі моделювати залежності в часі, що особливо корисно для завдань, таких як прогнозування часових рядів, де розуміння часової контексту є критичним [23, 24].

Однак важливо зазначити, що хоча шари SimpleRNN можуть ефективно захоплювати короткострокові залежності, вони можуть мати труднощі з довгостроковими залежностями через такі проблеми, як проблема зникання градієнта. Для вирішення цього обмеження були розроблені більш просунуті архітектури, такі як мережі довгої короткочасної пам'яті (LSTM) та згорткові рекурентні блоки (GRU), які пропонують покращені можливості моделювання довгострокових часових залежностей.

Наступним типом нейронних мереж є LSTM.

Мережі довготривалої короткочасної пам'яті (Long Short-Term Memory, LSTM) є спеціалізованим типом рекурентних нейронних мереж (RNN), розробленим для ефективного навчання та збереження довготривалих часових залежностей у послідовних даних. Ця архітектура була створена для подолання проблеми зникнення градієнта, яка часто виникає в традиційних RNN і ускладнює навчання на довгих послідовностях. Завдяки своїй здатності зберігати інформацію протягом тривалих періодів, LSTM широко застосовуються в задачах прогнозування часових рядів, де критично важливо враховувати порядок і тривалість подій [25]. Представимо структуру шару LSTM на рисунку (рис. 1.3)

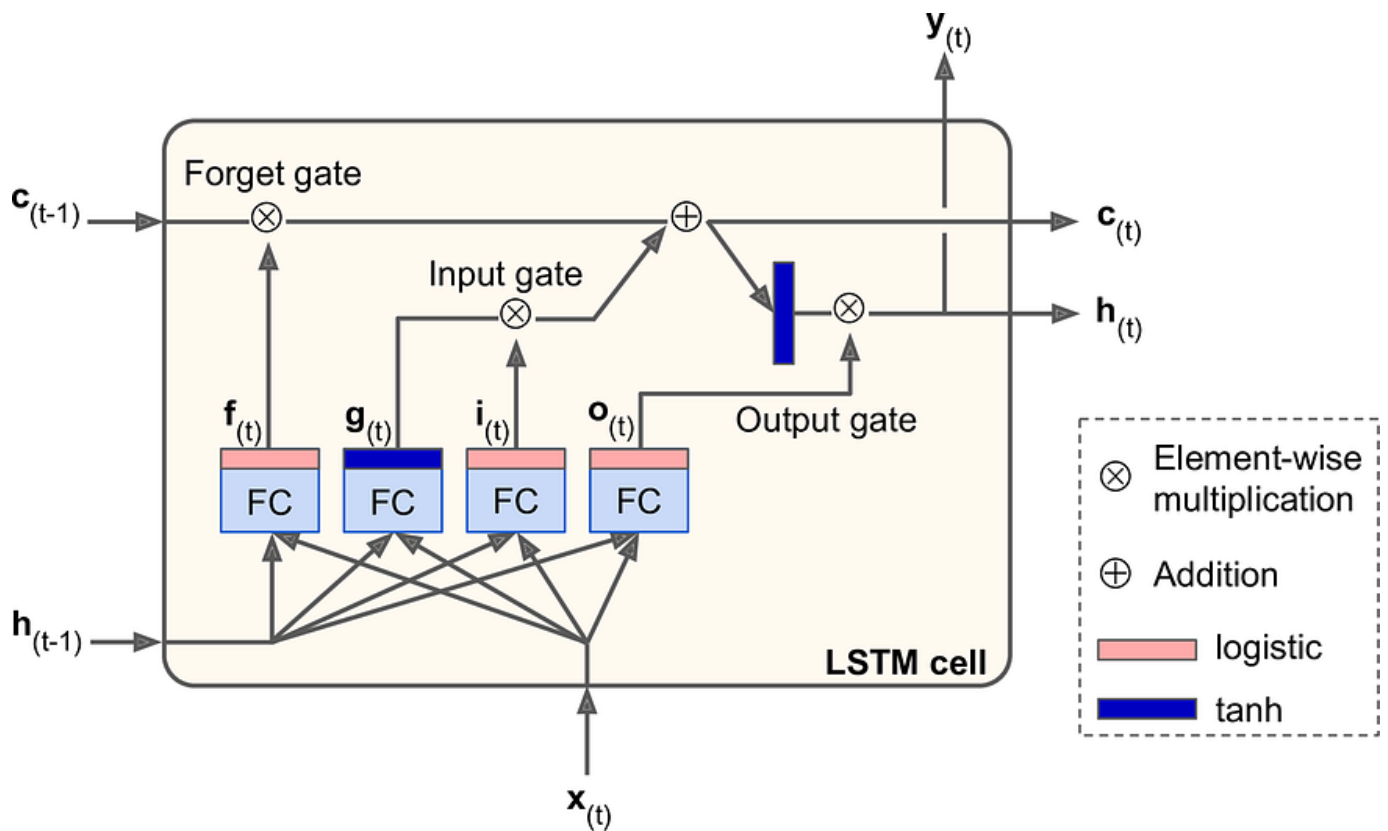


Рисунок 1.3 – Структура LSTM шару

Джерело: складено на основі[26]

Архітектура мережі довготривалої короткочасної пам'яті (Long Short-Term Memory, LSTM) складається з послідовності рекурентно з'єднаних одиниць, відомих як «комірки» або «блоки», що видно із схеми вище (рис. 1.3). Кожна комірка спроектована для управління та збереження інформації протягом тривалих часових інтервалів, що дозволяє LSTM ефективно захоплювати довготривалі залежності в послідовних даних. Опишемо кожен частину архітектури:

- вхідні дані – на кожному кроці LSTM обробляє вхідний вектор x_t , який представляє поточне спостереження, токен або ознаку в послідовності. Цей вхід може надходити з різних джерел, наприклад, слова в реченні для обробки природної мови, точки часових рядів або ознаки, отримані з зображення. Вхідний вектор слугує безпосереднім контекстом або даними, з якими модель працює на конкретному етапі, формуючи основу для подальших обчислень;

- прихований стан – шар підтримує вектор прихованого стану h_t , який втілює поточну «пам'ять» або знання мережі на певному етапі послідовності. Прихований стан динамічно оновлюється на кожному кроці, відображаючи як нові вхідні дані, так і інформацію, збережену з попередніх кроків. Спочатку прихований стан ініціалізується нульовим вектором, забезпечуючи нейтральну відправну точку для обробки послідовності. Цей вектор відіграє ключову роль у виявленні короткострокових залежностей та передачі відповідної інформації на наступні етапи послідовності;

- стан комірки - окрім прихованого стану, LSTM використовує вектор стану комірки c_t , який призначений для збереження довгострокової інформації в межах послідовності. На відміну від прихованого стану, стан комірки є більш стабільним, маючи механізми для вибіркового збереження або видалення інформації. На початку послідовності стан комірки також ініціалізується нульовим вектором. З часом він змінюється залежно від вхідних даних та внутрішніх рішень мережі, забезпечуючи вплив важливих контекстуальних деталей з попередніх етапів на подальші обчислення;

- Gates - для ефективного регулювання потоку інформації LSTM використовує три спеціалізовані Gates, кожен з яких реалізується за допомогою шарів нейронної мережі із сигмоїдною активацією. Ці Gates виступають як фільтри, визначаючи, яку інформацію слід зберегти, оновити чи видалити.

Gates бувають трьох типів:

- Forget Gate – визначають, які частини стану осередку мають бути видалені або «забуті», дозволяючи мережі позбавлятися несуттєвої або застарілої інформації;

- Input Gate – визначають, яку нову інформацію слід додати до стану осередку на основі поточного входу та прихованого стану;

- Output Gate – контролюють, яка частина стану осередку буде використана для формування поточного прихованого стану, який впливає на вихід мережі або розрахунки на наступному етапі [26].

Прогнозування часових рядів передбачає передбачення майбутніх значень на основі раніше спостережених даних. LSTM особливо підходять для цього завдання, оскільки можуть моделювати складні часові динаміки, захоплюючи як короткострокові коливання, так і довгострокові тренди. Їхня структура дозволяє працювати з різними сценаріями прогнозування, включаючи уніваріантні (одна змінна) та мультіваріантні (декілька змінних) часові ряди, а також одноетапні та багатоетапні прогнози.

Реалізація мережі LSTM для прогнозування часових рядів включає кілька ключових етапів, кожен з яких відіграє важливу роль у забезпеченні точності та надійності прогнозів:

- підготовка даних – на цьому етапі часові ряди трансформуються у формат, придатний для навчання моделі. Це передбачає створення пар «вхід-вихід», де послідовності попередніх спостережень використовуються для передбачення наступних значень. Наприклад, якщо маємо послідовність [10, 20, 30, 40, 50], її можна розбити на вхідні послідовності [10, 20, 30] з відповідним виходом [40], і [20, 30, 40] з виходом [50]. Такий підхід дозволяє моделі навчитися залежностям між попередніми та майбутніми значеннями;

- масштабування даних – нормалізація даних є критично важливою для ефективного навчання нейронної мережі. Масштабування змінних до діапазону [0, 1] або [-1, 1] допомагає уникнути домінування одних ознак над іншими та сприяє швидшій і стабільнішій конвергенції моделі. Це досягається за допомогою методів, таких як мін-макс нормалізація або стандартизація;

- проектування моделі – на цьому етапі визначається архітектура мережі LSTM, включаючи кількість шарів, кількість нейронів у кожному шарі та інші гіперпараметри. Вибір оптимальної архітектури залежить від складності даних та специфіки завдання. Наприклад, для складних часових рядів може бути доцільним використання багатошарової LSTM з великою кількістю нейронів, тоді як для простіших задач достатньо однієї LSTM-шару з меншою кількістю нейронів;

- навчання – під час навчання модель оптимізує свої ваги, мінімізуючи різницю між передбаченими та фактичними значеннями. Цей процес здійснюється за допомогою алгоритмів оптимізації, таких як градієнтний спуск, у поєднанні з методом зворотного поширення помилки через час (Backpropagation Through Time). Важливо контролювати процес навчання, щоб уникнути перенавчання, використовуючи методи регуляризації та техніки, такі як рання зупинка;
- оцінка та прогнозування – після навчання модель оцінюється на тестових даних, які не використовувалися під час навчання, для перевірки її здатності узагальнювати знання. Метрики, такі як середньоквадратична помилка (MSE) або середня абсолютна помилка (MAE), використовуються для кількісної оцінки точності прогнозів. Після успішної оцінки модель може бути використана для прогнозування майбутніх значень часових рядів [27].

Висновки до першого розділу

У першому розділі роботи, присвяченому теоретико-методичним підходам до оптимізаційної аналітики у формуванні інвестиційного портфеля підприємства, розглянуто основи портфельної теорії, а також сучасні інструменти статистичного та математичного аналізу для її вдосконалення. Зокрема, дослідження підкреслює, що формування оптимального портфеля акцій є однією з ключових задач для інвесторів, спрямованих на досягнення балансу між ризиком і доходністю. Ця задача вимагає системного підходу до оцінки активів, диверсифікації портфеля та прогнозування фінансових показників.

Перший аспект розділу акцентує увагу на класичній портфельній теорії Гаррі Марковіца. Його модель стала фундаментом для сучасних підходів до управління портфелем, оскільки пропонує науково обґрунтований метод мінімізації ризику за заданої доходності або, навпаки, максимізації доходу при прийнятному рівні ризику. Основними перевагами цього підходу є можливість створення диверсифікованого

портфеля, який враховує кореляційні взаємозв'язки між активами. Однак, з огляду на статичність базової моделі, її застосування потребує адаптації до змін ринкових умов і впливу зовнішніх факторів, що призводить до необхідності використання сучасних інструментів для вдосконалення цієї методології.

Другий аспект роботи розглядає застосування класичних статистичних методів, таких як ARIMA і GARCH, для прогнозування ринкових тенденцій і побудови ефективних портфелів. Використання ARIMA дозволяє моделювати поведінку часових рядів і передбачати майбутні значення дохідності активів. Цей підхід сприяє розрахунку прогнозованих коваріаційних матриць, які є ключовими для оцінки систематичних і несистематичних ризиків портфеля. У свою чергу, моделі GARCH забезпечують більш точну оцінку волатильності, враховуючи динамічні зміни ризиків у часі. Це дозволяє інвесторам краще розуміти мінливість ринкових умов і відповідно коригувати структуру своїх портфелів.

Третій аспект аналізує використання глибокого навчання для вдосконалення портфельної теорії. Технології глибокого навчання, зокрема рекурентні нейронні мережі (RNN) та їхні варіанти, такі як LSTM, дозволяють моделювати нелінійні та складні взаємозв'язки між фінансовими показниками. Ці інструменти відкривають нові можливості для прогнозування цінових трендів і оптимізації структури портфеля. Наприклад, LSTM-мережі забезпечують збереження як короткострокових, так і довгострокових залежностей у часових рядах, що є критично важливим для ефективного прогнозування в умовах волатильності ринку.

У розділі показано, що інтеграція традиційних фінансових методів із сучасними підходами до аналізу даних сприяє створенню більш адаптивних та ефективних інвестиційних портфелів. Оптимізаційна аналітика на основі інструментів статистики, машинного навчання та нейронних мереж відкриває нові горизонти для прийняття інвестиційних рішень, враховуючи як історичні дані, так і динаміку сучасних фінансових ринків. Це підкреслює важливість комплексного підходу до аналізу, який поєднує теоретичні засади з інноваційними практичними рішеннями.

2 АНАЛІТИЧНО-МАТЕМАТИЧНИЙ ІНСТРУМЕНТАРІЙ ФОРМУВАННЯ ОПТИМАЛЬНОГО ІНВЕСТИЦІОННОГО ПОРТФЕЛЮ

2.1 Аналіз та оцінка класичних методів моделювання оптимального портфелю цінних паперів

Для класичної моделі формування оптимального портфеля цінних паперів застосовується модель Гаррі Марковіца, відома як Сучасна теорія портфеля (Modern Portfolio Theory, MPT). Ця теорія є фундаментальною основою для створення портфелів інвестицій. Головною метою зазначеної моделі є визначення часток, які займають акції окремих компаній у складі загального портфеля.

У рамках сучасної теорії портфеля розглядаються два ключових показники: очікувана дохідність (Expected Return) і очікуваний ризик портфеля. Відповідно до теорії Марковіца, очікувану дохідність портфеля можна обчислити за допомогою спеціальної формули, що враховує різні фактори та їх взаємозв'язки:

$$R_p = \sum_i R_i w_i, \quad (2.1)$$

де R_p — дохідність портфелю;

R_i — дохідність активу;

w_i — частка активу в портфелі [28].

Очікуваний ризик портфеля відображає рівень ризикованості сформованого інвестиційного портфеля, тобто визначає ймовірність втрати вкладених коштів за заданого складу цінних паперів та їх ваг у портфелі. З наукової точки зору, очікуваний ризик портфеля математично визначається як стандартне відхилення його доходності. Перед тим як обчислити стандартне відхилення очікуваної доходності, необхідно визначити дисперсію цієї доходності.

Дисперсія слугує показником варіації або ступеня коливань доходності активів у портфелі. Вона демонструє, наскільки значними можуть бути відхилення фактичної

доходності портфеля від очікуваної. Чим більша дисперсія, тим вищий ризик нестабільності доходу. У рамках теорії Марковіца дисперсія обчислюється за допомогою наступної формули:

$$\begin{aligned}
 \sigma_p^2 &= E \left[\left(R_p - E(R_p) \right)^2 \right] = E \left[\left(\sum_i R_i w_i - \sum_i E(R_i) w_i \right)^2 \right] = \\
 &= E \left[\left(\sum_i w_i (R_i - E(R_i)) \right)^2 \right] = E \left[\sum_i \sum_j w_i w_j (R_i - E(R_i)) (R_j - E(R_j)) \right] = \quad (2.2) \\
 &= \sum_i \sum_j w_i w_j E \left[(R_i - E(R_i)) (R_j - E(R_j)) \right] = \sum_i \sum_j w_i w_j \text{cov}(R_i, R_j) = \\
 &= \sum_i \sum_j w_i w_j \sigma_{ij}
 \end{aligned}$$

де σ_p^2 — дисперсія доходності портфелю.

Очікуваний ризик усього портфеля обчислюється шляхом взяття квадратного кореня з дисперсії доходності, що визначається за формулою:

$$\sigma_p = \sqrt{\sigma_p^2}, \quad (2.3)$$

де σ_p — стандартне відхилення доходності портфелю, або «очікуваний ризик всього портфелю».

У подальшому аналізі підприємство, що займається формуванням інвестиційного портфеля, має змогу самостійно регулювати два ключові критерії та орієнтуватися на досягнення оптимального рішення для поставленої інвестиційної задачі.

Один із головних критеріїв, який зазвичай враховується під час створення портфеля, — це ризик. Інвестор може визначати прийнятні рівні ризику та розподіляти свої інвестиції між різними активами, враховуючи цей параметр. Для мінімізації ризику консервативний інвестор може надати перевагу стабільним активам, збільшуючи їхню частку у своєму портфелі. Натомість більш агресивний інвестор, готовий прийняти вищий рівень ризику задля потенційно вищих доходів, може спрямувати кошти на активи із більшою волатильністю.

Другий важливий критерій — дохідність. Формуючи портфель, інвестор визначає свої фінансові цілі та прагне досягти оптимального рівня доходу. Це передбачає балансування інвестицій між активами з різними рівнями потенційного прибутку залежно від індивідуальних пріоритетів та фінансових завдань. Наприклад, у разі довгострокової інвестиційної стратегії інвестор може зосередитися на активах із вищою дохідністю, які здатні забезпечити значні прибутки в перспективі.

Для забезпечення ефективної оптимізації портфеля, спрямованої одночасно на максимізацію прибутку та мінімізацію ризику, застосуємо підхід багатокритеріальної оптимізації.

Перший критерій – це функція очікуваного прибутку, яка буде оптимізуватися на досягнення максимального значення:

$$f_1 = R_p \rightarrow \max, \quad (2.4)$$

Другий критерій полягає в необхідності оптимізації функції загального ризику портфеля до мінімального рівня:

$$f_2 = \sigma_p \rightarrow \min, \quad (2.5)$$

Визначивши дві цільові функції, стає можливим виконати згортку критеріїв. Оптимізація цільової функції здійснюється шляхом мінімізації. З урахуванням усіх критеріїв отримуємо наступну систему обмежень:

$$\begin{cases} \sum_{i=1}^n w_i = 1 \\ w_i \geq 0,01, \\ R_p > 0 \\ \sigma_p \geq 0 \end{cases} \quad (2.6)$$

Цільова функція, яка буде оптимізуватися:

$$W = \frac{\alpha * \sigma_p}{(1 - \alpha) * R_p} \rightarrow \min, \quad (2.7)$$

Визначивши головний критерій та обмеження, маємо задачу оптимізації для знаходження оптимального портфелю акцій:

$$W = \frac{\alpha * \sigma_p}{(1 - \alpha) * R_p} \rightarrow \min$$

$$\left\{ \begin{array}{l} \sum_{i=1}^n w_i = 1 \\ w_i \geq 0,01 \\ R_p > 0 \\ \sigma_p \geq 0 \end{array} \right. , \quad (2.8)$$

Підставивши рівняння 2.1 та 2.2 маємо задачу оптимізації наступного вигляду:

$$W = \frac{\alpha * \sqrt{w^T * cov * w}}{(1 - \alpha) * \sum_{i=1}^n R_i * w_i} \rightarrow \min$$

$$\left\{ \begin{array}{l} \sum_{i=1}^n w_i = 1 \\ w_i \geq 0,01 \\ \sum_{i=1}^n R_i * w_i > 0 \\ \sqrt{w^T * cov * w} \geq 0 \end{array} \right. , \quad (2.9)$$

Класична модель формування оптимального портфеля цінних паперів нерозривно пов'язана з аналізом сукупності економічних факторів, які визначають стан ринкового середовища та створюють контекст для прийняття інвестиційних рішень. Один із найважливіших макроекономічних індикаторів — рівень інфляції, оскільки він безпосередньо впливає на реальну дохідність інвестицій. Інфляція є показником, що відображає зростання загального рівня цін у економіці, що, своєю чергою, зменшує купівельну спроможність доходу від портфеля. У середовищі високої інфляції інвестору слід враховувати ризик девальвації реального прибутку та потенційної втрати вартості активів, особливо якщо їхня номінальна дохідність не компенсує інфляційного впливу. Це може зумовлювати переорієнтацію інвестицій на активи, які зберігають свою вартість за умов інфляції, такі як захищені інфляцією облігації (TIPS) або інструменти з реальними активами, наприклад, нерухомість чи товари. Іншим фундаментальним макроекономічним чинником є ключова процентна

ставка, яка встановлюється центральним банком. Цей показник є критичним у визначенні вартості капіталу, впливаючи як на прибутковість, так і на ризикованість фінансових інструментів. Коли ключова ставка підвищується, вартість запозичень зростає, що, як правило, знижує інтерес до високоризикових активів, таких як акції, оскільки інвестори віддають перевагу більш стабільним і менш ризиковим інструментам, таким як державні облігації. З іншого боку, зниження ключової ставки стимулює економічну активність, зменшуючи вартість кредитів і водночас сприяючи зростанню цін на акції через підвищення очікуваної дохідності. Інвестори, враховуючи ці чинники, можуть коригувати структуру портфеля, балансуючи між стабільністю і потенційно вищими доходами від більш ризикових активів.

Додатково, ці два фактори — інфляція та ключова ставка — перебувають у взаємозв'язку: для боротьби з інфляцією центральні банки часто вдаються до підвищення ставок, що одночасно впливає на доступність капіталу та зміну поведінки інвесторів. Важливо враховувати і макроекономічну ситуацію загалом, включно зі станом економічного зростання, рівнем зайнятості, торговим балансом і коливаннями валютних курсів, які також суттєво впливають на дохідність та ризик портфеля.

На мікроекономічному рівні формування оптимального портфеля цінних паперів вимагає глибокого аналізу фінансових показників окремих компаній, що становлять його основу.

Одним із ключових індикаторів є рентабельність капіталу (Return on Equity, ROE), яка показує, наскільки ефективно компанія використовує власний капітал для генерування прибутку. Високий ROE свідчить про ефективне управління ресурсами та стабільну здатність до генерування доходу, що є важливим критерієм для включення активів компанії до портфеля.

Іншим важливим показником є рентабельність активів (Return on Assets, ROA), яка відображає ефективність використання всієї сукупності активів компанії для створення прибутку. ROA дозволяє оцінити, наскільки добре компанія керує своїми ресурсами незалежно від структури капіталу. У поєднанні з ROE цей показник дає

змогу інвесторам зрозуміти, чи зростання прибутковості є результатом ефективності операцій чи збільшення боргового фінансування.

Чистий прибуток є базовим показником, що демонструє фінансові результати компанії. Стабільно високий чистий прибуток сигналізує про здатність компанії підтримувати конкурентоспроможність, задовольняти потреби акціонерів і реінвестувати кошти для подальшого зростання. Водночас важливо оцінювати тренд цього показника, аби виявити можливі ризики зниження фінансової ефективності. Рівень боргового навантаження (Debt-to-Equity Ratio) є критичним параметром, який відображає співвідношення між борговими та власними ресурсами компанії. Надмірна частка боргів у структурі капіталу підвищує фінансовий ризик, оскільки вказує на більшу залежність від кредиторів і спроможність виконувати боргові зобов'язання в умовах змінних економічних обставин. З іншого боку, оптимальний рівень використання боргового фінансування може сприяти зростанню доходності на акціонерний капітал за рахунок ефекту фінансового важеля. Стабільність грошових потоків компанії є ще одним ключовим фактором для мікроекономічного аналізу. Постійні й передбачувані грошові потоки свідчать про стійкість бізнес-моделі компанії та її здатність фінансувати операційну діяльність, виплачувати дивіденди і обслуговувати борги. Інвестори зазвичай приділяють особливу увагу цьому аспекту, адже компанії з нестабільними потоками, навіть за наявності високих прибутків, можуть зіштовхнутися з проблемами ліквідності.

Важливим фактором також є глобальні економічні умови. Рівень економічного зростання, який відображає загальну продуктивність економіки, є основним індикатором для оцінки привабливості інвестицій у певному регіоні. Економіки, що демонструють стабільне зростання ВВП, зазвичай пропонують більше інвестиційних можливостей завдяки збільшенню корпоративних доходів і розширенню ринків. Обсяг міжнародної торгівлі також має значний вплив, адже країни з відкритою економікою, високим експортом і імпортом, як правило, створюють сприятливі умови

для інвесторів, забезпечуючи доступ до великих ринків і диверсифікованих можливостей.

Валютні курси визначають вартість інвестицій в іноземні активи, впливаючи як на потенційну дохідність, так і на ризик. Наприклад, девальвація валюти може зменшити дохідність для іноземних інвесторів, тоді як стабільний курс створює сприятливі умови для інвестування. Геополітична стабільність є критичним чинником для зниження невизначеності на ринках. Політичні кризи, торговельні війни або регіональні конфлікти можуть різко зменшити привабливість активів через зростання ризику втрат і зниження ліквідності.

З іншого боку, країни з передбачуваною політичною ситуацією та сприятливими регуляторними умовами приваблюють інвесторів завдяки стабільності та прозорості. Розвиток фінансових ринків у регіонах, де розташовані активи, є ще одним важливим аспектом. Ринки з високою ліквідністю, ефективною інфраструктурою та широким спектром фінансових інструментів пропонують більше можливостей для ефективної диверсифікації портфеля. Наприклад, доступність похідних фінансових інструментів, таких як ф'ючерси та опціони, дозволяє інвесторам ефективно управляти ризиками.

Проте, варто зазначити, що глобалізація економіки призводить до зростання кореляції між активами в різних країнах та регіонах, що ускладнює досягнення справжньої диверсифікації. У таких умовах необхідно приділяти увагу інвестиціям у активи, які мають низьку або негативну кореляцію з іншими складовими портфеля. Це можуть бути, наприклад, інструменти альтернативних інвестицій (недержавні облігації, хедж-фонди, нерухомість) або активи з інших економічних зон, які меншою мірою підпадають під вплив глобальних економічних циклів. Подібний підхід дозволяє зменшити сукупний ризик портфеля, забезпечуючи стабільність доходів навіть у разі глобальних економічних потрясінь.

2.2 Аналітично-математичний інструментарій визначення оптимального портфелю з використанням класичних методів аналізу часових рядів

Класичний підхід Марковіца, який передбачає згортку критеріїв, має суттєвий недолік, що обмежує його ефективність у практичному застосуванні. Головна проблема полягає у тому, що цей метод базується на побудові портфеля, орієнтованого виключно на теперішній момент часу, без врахування існуючих ринкових тенденцій та змін, які можуть вплинути на майбутню динаміку. Такий підхід втрачає актуальність, оскільки він не враховує довгострокові перспективи, що є критично важливими для стратегічного формування інвестиційного портфеля. Крім того, подібна модель не забезпечує гнучкості, необхідної для адаптації до змін у ринковому середовищі, що знижує її практичну цінність для сучасного фінансового аналізу.

Щоб вирішити цю проблему, розглянемо дві статистичні моделі, які допоможуть спрогнозувати коваріаційні матриці (матриці ризиків). Цими моделями є модель GARCH та модель ARIMA.

ARCH(m) процес є математичною моделлю, яка описує залежність дисперсії в певний момент часу від значень спостережень у попередні m моментів часу. Іншими словами, для такого процесу варіативність даних у теперішній момент не є сталою, а визначається історичними значеннями, що дозволяє ефективно моделювати явища зі змінною волатильністю. Математичний вираз цієї залежності можна записати у вигляді співвідношення:

$$\text{Var}(y_t | y_{t-1}, \dots, y_{t-m}) = \sigma_t^2 = \alpha_0 + \alpha_1 y_{t-1}^2 + \dots + \alpha_m y_{t-m}^2, \quad (2.10)$$

де y_t — значення в період t ;

y_{t-1} — значення в період $t-1$;

σ_t^2 — варіація в період t ;

$\alpha_0, \alpha_1, \dots, \alpha_m$ — коефіцієнти моделі [38, 39].

З певними обмеженнями, накладеними на коефіцієнти, ряд y_t у квадраті теоретично матиме вигляд AR(m).

GARCH (узагальнена авторегресійна умовно гетероскедастична) модель є одним із ключових інструментів в аналізі часових рядів, який дозволяє моделювати змінність (волатильність) фінансових або інших динамічних даних у часі. Ця модель ґрунтується на використанні попередніх спостережень квадратів залишків (або помилок прогнозу) та попередніх значень дисперсії для оцінки поточного значення дисперсії в момент часу t . Таким чином, GARCH модель дозволяє враховувати особливості часових рядів, зокрема залежність між значеннями волатильності, що робить її особливо корисною в аналізі даних, де спостерігається кластеризація високої і низької волатильності.

У цій роботі модель GARCH була обрана для побудови відповідного математичного апарату, який дозволяє точно оцінювати дисперсію даних та прогнозувати її динаміку у часі. Її використання забезпечує більш глибоке розуміння поведінки процесів і дозволяє врахувати складну природу умовної гетероскедастичності, яка часто присутня в реальних даних.

Перш ніж перейти до виконання прогнозування, необхідно попередньо визначити умовну дисперсію для періоду $t+1$. Цей розрахунок виконується відповідно до заданої математичної формули, яка враховує попередні статистичні дані та параметри моделі.

$$\sigma_{t+1}^2 = \alpha_0 + \alpha_1 y_t^2 + \beta_1 \sigma_t^2, \quad (2.11)$$

Після цього, знайдемо умовну дисперсію в період $t+s$

$$\widehat{\sigma}_{t+1}^2(GARCH) = \widehat{\alpha}_0 \sum_{i=0}^{s-2} (\widehat{\alpha}_1 + \widehat{\beta}_1)^i + (\widehat{\alpha}_1 + \widehat{\beta}_1)^{s-1} \widehat{\sigma}_{t+1}^2, \quad (2.12)$$

де $s=(1,2, \dots N_t)$ та σ_{t+1}^2 – день, наступний за прогнозом волатильності на перший день кожного місяця отриманий в результаті розв'язання рівняння 2.20 [40].

Маючи умовні дисперсії, розрахуємо стандартні похибки GARCH-моделей для логарифмічних норм прибутковості кожної компанії за допомогою

формули:

$$std_{residual_{it}} = \frac{residual_{it}}{\sigma_{it}^2}, \quad (2.13)$$

де $residual$ – залишок GARCH моделі для i -ої компанії в момент часу t .

Отримавши вектори залишків кожної моделі, необхідно обчислити умовну кореляційну матрицю (Conditional Correlation Matrix). Цю матрицю можна визначити за допомогою наступної формули:

$$CCC = \frac{1}{n_i} (cov_{residuals}_i^T \times cov_{residuals}_i), \quad (2.14)$$

де $cov_{residuals}_i$ – коваріаційна матриця залишків моделі для i -ої компанії;

$cov_{residuals}_i^T$ – транспонована коваріаційна матриця залишків моделі для i -ої компанії;

n_i – кількість залишків моделі для i -ої компанії.

Використовуючи моделі GARCH для кожної компанії, визначимо прогнозовані значення елементів головної діагоналі прогнозованої коваріаційної матриці. Здійснимо прогноз на період $t+1$. Після обчислення прогнозованих значень діагональних елементів, розрахуємо прогнозовану коваріаційну матрицю за формулою:

$$cov_p = D \times CCC, \quad (2.15)$$

де cov_p – прогнозована коваріаційна матриця на період $t+1$.

Отже, побудувавши прогнозовану коваріаційну матрицю, отримуємо задачу оптимізації з динамічною коваріаційною матрицею:

$$W = \frac{\alpha * \sqrt{w^T * cov_p * w}}{(1 - \alpha) * \sum_{i=1}^n R_i * w_i} \rightarrow min$$

$$\left\{ \begin{array}{l} \sum_{i=1}^n w_i = 1 \\ w_i \geq 0,01 \\ \sum_{i=1}^n R_i * w_i > 0 \\ \sqrt{w^T * cov_p * w} \geq 0 \end{array} \right. , \quad (2.16)$$

Економічна інтерпретація моделі GARCH (Generalized Autoregressive Conditional Heteroskedasticity) полягає в її здатності моделювати динаміку волатильності, що є ключовим елементом у фінансовому аналізі. Волатильність відображає рівень ризику та нестабільності, що притаманний фінансовим активам, і часто демонструє властивість кластеризації — періоди високої волатильності змінюються періодами низької. Це явище є важливим для оцінки ризиків, формування інвестиційних стратегій і управління портфелем, адже волатильність безпосередньо впливає на очікувані доходності та ризиковий профіль активів.

Модель GARCH дозволяє описати змінність, враховуючи залежність поточної дисперсії від минулих спостережень, зокрема залишків і дисперсій. Економічно це означає, що ризики, які спостерігалися в минулому, впливають на очікувані ризики в майбутньому. Наприклад, якщо ринок пережив суттєві коливання, модель передбачатиме високий рівень волатильності в наступні періоди.

Такий підхід є значно більш адаптивним, ніж традиційні статичні моделі, оскільки враховує тимчасову структуру ризиків. Для фінансових аналітиків та інвесторів використання GARCH-моделі дозволяє здійснити точнішу оцінку майбутньої динаміки ризиків. Прогнозування умовної дисперсії та кореляційних матриць дозволяє не лише краще оцінювати окремі активи, але й оптимізувати портфелі, знижуючи загальний ризик за рахунок врахування динамічних залежностей між активами. Економічно це дає можливість приймати обґрунтовані інвестиційні

рішення, зменшуючи ймовірність несподіваних втрат через раптове зростання волатильності.

Ще одним статистично простим, але надзвичайно ефективним методом для прогнозування цін акцій, аналізу часових рядів та побудови актуального інвестиційного портфеля є модель ARIMA (Autoregressive Integrated Moving Average). Ця модель базується на комбінації трьох основних компонентів: авторегресії (AR), інтеграції (I) та ковзного середнього (MA). ARIMA є популярним інструментом у фінансовій сфері завдяки своїй універсальності та здатності ефективно моделювати широкий спектр трендів та сезонних змін у часових рядах.

Зробимо декомпозицію моделі ARIMA, щоб зрозуміти принцип її роботи. Авторегресійна частина моделює залежність між поточним спостереженням і певною кількістю попередніх спостережень. Модель AR порядку p , позначена як $AR(p)$, описується формулою:

$$Y_t = c + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \dots + \phi_p Y_{t-p} + \epsilon_t, \quad (2.17)$$

де Y_t - значення у момент часу t ;

c – константа;

ϕ_p – коефіцієнти авторегресії;

ϵ_t – помилка у момент часу t [29].

Диференціююча частина ARIMA представлена параметром d . Вона передбачає перетворення нестационарного часового ряду в стаціонарний шляхом диференціювання послідовних спостережень. Операція диференціювання може бути застосована декілька разів, доки не буде досягнуто стаціонарності. Формула для диференціювання має вигляд:

$$Y_t = Y_t - Y_{t-1}, \quad (2.18)$$

де Y_t – диференційований ряд у момент часу t ;

Y_t – оригінальне значення у момент часу t ;

Y_{t-1} – попереднє значення ряду, а саме, у момент часу $t-1$ [29].

Диференціювання зазвичай виконується кілька разів, доки часовий ряд не стане стаціонарним. Символ $I(d)$ позначає кількість диференціювання, необхідних для досягнення стаціонарності.

Компонент ковзного середнього (MA) у моделі ARIMA відповідає за врахування впливу залишкових помилок, що виникли в попередніх прогнозах. Цей аспект моделі характеризується параметром q , який вказує на кількість попередніх прогнозних помилок, що враховуються для оцінки поточного значення. У практичному застосуванні цей компонент дозволяє моделі адаптуватися до непередбачуваних змін у даних, спричинених випадковими факторами. З математичної точки зору модель ковзного середнього порядку q , або MA(q), можна описати наступним рівнянням:

$$Y_t = c + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \dots + \theta_q \epsilon_{t-q}, \quad (2.19)$$

де θ_q – коефіцієнти ковзного середнього.

Загальна формула для несезонної моделі ARIMA подається у вигляді ARIMA(p , d , q):

$$Y'_t = c + \phi_1 Y'_{t-1} + \phi_2 Y'_{t-2} + \dots + \phi_p Y'_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \dots + \theta_q \epsilon_{t-q}, \quad (2.20)$$

де Y'_t – диференційований та стаціонарний часовий рядом у момент часу t ;

Y_t – оригінальне значення у момент часу t ;

c – константа або середнє значення диференційованого ряду;

$\phi_1, \phi_2, \dots, \phi_p$ – параметри авторегресії, що відображають залежність від попередніх значень;

ϵ_t – це білий шум або випадкова похибка у момент часу t ;

$\theta_1, \theta_2, \dots, \theta_q$ – параметри ковзного середнього, які показують залежність від минулих похибок прогнозу [29].

Маючи розуміння про структуру та ключові складові моделі, перейдемо до розробки математичної послідовності дій за допомогою яких ми можемо провести якісне моделювання.

Маючи часовий ряд, потрібно дослідити його стаціонарність. Дослідити часовий ряд на стаціонарність можна за допомогою тесту Дікі-Фуллера за наступною формулою [30]:

$$t_{\hat{\delta}} = \frac{\hat{\delta}}{SE(\hat{\delta})}, \quad (2.21)$$

де $t_{\hat{\delta}}$ – значення тесту Дікі-Фуллера, яка використовується для перевірки нульової гіпотези ($H_0: \delta = 0$) проти альтернативної гіпотези ($H_1: \delta < 1$);

$\hat{\delta}$ – коефіцієнт при лаговому рівневому члені в регресійній моделі;

$SE(\hat{\delta})$ – стандартна похибка $\hat{\delta}$, яка відображає мінливість оцінки.

Тест Дікі-Фуллера перевіряє чи $\phi=0$ в наступній моделі даних:

$$y_t = \alpha + \beta t + \phi y_{t-1} + \epsilon_t, \quad (2.22)$$

яка записується наступним чином:

$$\Delta y_t = y_t - y_{t-1} = \alpha + \beta t + \gamma y_{t-1} + \epsilon_t, \quad (2.23)$$

де y_t – дані часового ряду.

Таке формулювання дає нам змогу застосувати лінійну регресію до Δy_t відносно t та y_{t-1} і перевірити чи є параметр γ відмінним від 0. Якщо $\gamma = 0$, то в результаті ми отримуємо довільний процес. Проте, у тому випадку, якщо $-1 < 1 + \gamma < 1$, ми отримуємо стаціонарний ряд.

Якщо дослідження на стаціонарність показують, що ряд є нестационарним, потрібно здійснити перетворення Бокса-Кокса. Перетворення часового ряду здійснюється за наступною формулою [31]:

$$y_t^{(\lambda)} = \begin{cases} \frac{y_t^{(\lambda)} - 1}{\lambda}, & \text{якщо } \lambda \neq 0, \\ \log(y_t), & \text{якщо } \lambda = 0 \end{cases}, \quad (2.23)$$

де y_t – значення часового ряду у момент t ;

$y_t^{(\lambda)}$ – перетворене значення;

λ – параметр перетворення, який визначає тип перетворення.

Параметр λ обирається за наступним принципом:

- $\lambda=1$: лінійне перетворення (дані залишаються без змін);
- $\lambda=0$: логарифмічне перетворення;
- $\lambda=-1$: обернене перетворення (інверсія).

Після перетворення значень, можна застосовувати формулу 2.20 для здійснення прогнозування.

Модель ARIMA, має значні інвестиційно-економічні переваги. Завдяки своїй структурі, що поєднує авторегресію, диференціювання та ковзне середнє, вона дозволяє якісно аналізувати динаміку цінових змін, прогнозувати майбутні тенденції та мінімізувати ризики, пов'язані з інвестиціями.

ARIMA здатна ефективно працювати з нестационарними часовими рядами, перетворюючи їх у стаціонарні для подальшого аналізу, що робить її надзвичайно гнучкою в умовах реальних ринкових даних.

Для інвесторів важливою перевагою ARIMA є її здатність враховувати як довгострокові тенденції, так і сезонні коливання, що дозволяє точніше оцінювати потенційну прибутковість інвестиційного портфеля.

Використання моделі сприяє оптимізації прийняття рішень, оскільки вона враховує як минулі дані, так і помилки попередніх прогнозів, адаптуючись до непередбачуваних змін ринку. Це забезпечує створення більш надійних стратегій управління портфелем, підвищуючи його стійкість до волатильності.

Крім того, ARIMA дозволяє розробляти математично обґрунтовані стратегії, спираючись на точні моделі прогнозування, які можна застосувати для оцінки інвестиційних ризиків, визначення оптимальних точок входу та виходу на ринок. Це сприяє раціональному розподілу капіталу та підвищенню ефективності управління фінансовими ресурсами. Завдяки використанню тестів, таких як тест Дікі-Фуллера, ARIMA забезпечує об'єктивність оцінки даних та підвищує довіру до прийнятих рішень.

2.3 Аналітичне моделювання для визначення ефективності інвестиційних проєктів з використанням методів машинного навчання

Такі статистичні підходи як GARCH та ARIMA, не дивлячись на простоту їх інтерпретації мають свої мінуси – вони не є на стільки динамічними та не на стільки можуть розуміти дані, як це роблять моделі на базі Deep Learning технологій.

Для прогностичного моделювання використаємо такі архітектури нейронних мереж як DNN, RNN, LSTM, CNN та змішані нейронні мережі – використання різних типів шарів у одній моделі.

DNN – цільна нейронна мережа, також відома як повнозв'язна або прямопрохідна нейронна мережа, будується шляхом накладання декількох шарів нейронів. У цій структурі кожен шар з'єднаний з усіма нейронами попереднього шару, що дозволяє моделі виявляти та навчатися на патернах даних. У послідовній моделі дані проходять через кожен шар один за одним, рухаючись від вхідного шару до вихідного. Математичне представлення щільного шару має вигляд [32]:

$$y = f(Wx + b), \quad (2.24)$$

де W – вагова матриця;

x – вектор вхідних даних;

b – вектор зсувів;

f – функція активації (у нашому випадку ReLU).

Функція активації ReLU (Rectified Linear Unit) передає вхідні значення без змін, якщо вони додатні; інакше вона повертає нуль. Це дозволяє швидше тренувати модель, вводячи нелінійність та зменшуючи проблему згасання градієнта. Формула функції активації ReLU [33, 34]:

$$ReLU(x) = \max(0, x) = \frac{x + |x|}{2} = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases}, \quad (2.25)$$

де x – вхід нейрона.

Вихідний шар представлений одним нейроном, який надає передбачену ціну акцій на наступному часовому кроці. Для регресійних задач вихідний нейрон зазвичай використовує лінійну функцію активації:

$$\hat{y} = z^{(L)} = \sum_{i=1}^{N_{L-1}} w_i^{(L)} z_i^{(L-1)} + b^{(L)}, \quad (2.26)$$

де $w_i^{(L)}$ – ваги вихідного шару;

$b^{(L)}$ – зсув вихідного шару;

\hat{y} – передбачувана ціна акцій на наступному часовому кроці.

Щоб натренувати нейронну мережу, потрібно застосувати оптимайзер, який буде коригувати параметри нейронної мережі – ваги та зсуви. У даній роботі застосовується SGD функція оптимізації. Щоб застосувати оптимайзер, потрібно визначити loss функцію. У цій роботі застосовується MSE або Хубера (Hubber) як основні функції втрат. Формула MSE має вигляд [35]:

$$L_{MSE} = \frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2, \quad (2.27)$$

де n – кількість точок даних.

Щодо функції Хубера, то ця функція має дещо інший принцип роботи. Функція втрат Хубера працює за наступною формулою [36]:

$$L_{\delta}(a) = \begin{cases} \frac{1}{2} a^2, \text{ якщо } |a| \leq \delta \\ \delta \left(|a| - \frac{1}{2} \delta \right), \text{ якщо } |a| > \delta \end{cases}, \quad (2.28)$$

де a – залишок або $y - \hat{y}$,

δ – пороговий параметр, що визначає точку переходу функції втрат від квадратичної до лінійної.

Наступним чином відбувається обчислення градієнта відносно функції втрат. У випадку для L_{MSE} , функція оптимізації матиме наступний вигляд [37]:

$$w_{t+1} = w_t - \eta * \nabla_w L_{MSE}(w_t), \quad (2.29)$$

де w_t – параметри на ітерації t ;

η – швидкість навчання;

$\nabla_w L_{MSE}(w_t)$ – градієнт втрат MSE відносно w на ітерації t .

У випадку використання функції втрат Хубера, ми маємо наступний вираз для функції оптимізації [37]:

$$w_{t+1} = w_t - \eta * \nabla_w L_{\delta}(w_t), \quad (2.30)$$

де w_t – параметри на ітерації t ;

η – швидкість навчання;

$\nabla_w L_{\delta}(w_t)$ – градієнт втрат функції Хубера відносно w на ітерації t .

Оцінка якості моделі відбувається за допомогою ключових метрик, які будуть описані вкінці підрозділу 2.3.

DNN за своєю архітектурою не є найкращим вибором для роботи з часовими рядами. Саме тому, для часових рядів часто застосовують рекурентні нейронні мережі – RNN.

Рекурентна нейронна мережа (RNN) — це тип нейронної мережі, призначений для обробки послідовних даних шляхом підтримки прихованого стану, який зберігає інформацію про попередні вхідні дані. Ця архітектура дозволяє моделювати часові залежності, що робить її придатною для завдань, таких як прогнозування часових рядів, обробка природної мови та розпізнавання мови.

На кожному часовому кроці t модель обчислює прихований стан h_t та вихід y_t . Оновлення прихованого стану розраховується за наступною формулою [32]:

$$h_t = f(W_x h \cdot x_t + W_h h \cdot h(t-1) + b_h), \quad (2.31)$$

де h_t – прихований шар у час t ;

x_t – вхідний вектор на часовому кроці t ;

$W_x h$ - матриця ваг для зв'язків «вхід-прихований»;

$W_h h$ - матриця ваг для рекурентних зв'язків «прихований-прихований»;

b_h - вектор зміщення;

f – активаційна функція (у нашому випадку ReLU визначена за формулою 2.25).

Щоб обрахувати вихідні значення, потрібно скористатися наступною формулою:

$$y_t = \phi(W_{hy}h_t + b_y), \quad (2.32)$$

де W_{hy} – вагова матриця для зв'язків «прихований-вихід»;

b_y – вектор зміщення для вихідного шару;

ϕ – активаційна функція для вихідного шару - у нашому випадку для Dense(1) шару із функцією активації ReLU.

Побудова моделі, функції оптимізації та Loss визначаються за формулами 2.27 – 2.30.

Мережі довготривалої короткочасної пам'яті (LSTM) є спеціалізованим типом рекурентних нейронних мереж (RNN), розроблених для ефективного захоплення довгострокових залежностей у послідовних даних, що робить їх особливо придатними для прогнозування часових рядів.

Вони мають дещо складнішу архітектуру, ніж звичайні RNN, що можна продемонструвати наступною декомпозицією всієї архітектури. Першим компонентом є забувальний шлюз, який визначає, яку частину попереднього стану комірки C_{t-1} слід зберегти. Він описується за такою формулою [32, 38]:

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f), \quad (2.33)$$

де σ – сигмоїдна активаційна функція;

W_f та U_f – вагові матриці;

h_{t-1} – попередній прихований стан;

x_t – вхідні дані у момент часу t ;

b_f – зміщення.

В свою чергу, сигмоїдна активаційна функція має наступний вигляд [34]:

$$f(x) = \sigma = \frac{1}{1 + e^x}, \quad (2.34)$$

Наступним елементом є вхідний шлюз. Він визначає, яку нову інформацію додати до стану комірки:

$$i_t = \sigma(W_i \cdot x_t + U_i \cdot h_{t-1} + b_i), \quad (2.35)$$

Після вхідного шлюзу ми знаходимо кандидата на стан комірки, а саме пропонуємо нові значення для оновлення стану комірки. Це можна зробити за наступною формулою:

$$\tilde{C}_t = \tanh(W_c \cdot x_t + U_c \cdot h_{t-1} + b_c), \quad (2.36)$$

де \tanh – гіперболічна тангенсова функція активації.

Після знаходження кандидата відбувається оновлення стану комірки шляхом поєднання попереднього стану та нових значень, модульованих забувальним та вхідним шлюзами (формули 2.33-2.35). Оновлюється стан комірки за такою формулою:

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t, \quad (2.37)$$

де \odot - поелементне множення.

Після цього, ми знаходимо вихідний шлюз, який визначає, яку частину стану комірки вивести. Він визначається такою рівністю:

$$o_t = \sigma(W_o \cdot x_t + U_o \cdot h_{t-1} + b_o), \quad (2.38)$$

Вкінці відбувається перехід до прихованого стану. Ми отримуємо кінцевий вихід комірки LSTM для поточного кроку часу [32, 38]:

$$h_t = o_t \odot \tanh(C_t), \quad (2.39)$$

Використання LSTM є сильним рішенням, яке можна посилити використавши додатково таку архітектуру як CNN.

Згорткова нейронна мережа (CNN) для прогнозування часових рядів — це тип моделі глибокого навчання, що використовує згорткові шари для автоматичного виділення важливих ознак із часових даних. Вона обробляє вхідні дані шляхом застосування згорткових операцій, що дозволяє мережі виявляти шаблони, тренди та залежності в часових рядах, роблячи її ефективною для прогнозування.

Першою є згорткова операція, яка відбувається за наступним рівнянням [39]:

$$y(i) = \sum_{j=1}^k x(i+j-1) \cdot w(j) + b, \quad (2.40)$$

де $y(i)$ – вихід згортки в момент i ;

x – вхідні дані;

w – ваги ядра;

b – зміщення;

k – розмір ядра;

j – індекс в середині ядра.

Після згортки застосовується функція активації. У цьому випадку використовується функція активації ReLU, описана у формулі 2.25.

Наступним етапом відбувається операція підвибірки (Pooling) - шари підвибірки зменшують просторовий розмір представлення. Існує декілька видів такого процесу, проте в цій роботі було обрано Max Pooling. Він описується наступною формулою:

$$y(i) = \max(x(i:i+k)), \quad (2.41)$$

де k – розмір вікна підвибірки.

На виході отримується повнозв'язний шар, який описується формулою:

$$y = Wx + b, \quad (2.42)$$

де W – вагова матриця;

x – вирівняний вхідний вектор;

b – вектор зміщення.

Маючи різні варіації побудованих моделей, потрібно виміряти їх ключові показники. В аналізі часових рядів зазвичай використовуються чотири основні метрики для оцінки якості моделі: середня абсолютна похибка (Mean Absolute Error, MAE), середня квадратична похибка (Mean Squared Error, MSE), середня абсолютна відносна похибка (Mean Absolute Percentage Error, MAPE) та коефіцієнт детермінації (R^2).

MAE вимірює середню величину помилок між передбаченими та спостережуваними значеннями без урахування напрямку, забезпечуючи просту інтерпретацію типової помилки передбачення [35]:

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|, \quad (2.43)$$

де y_i – спостережуване значення;

\hat{y}_i – прогнозоване значення;

n – загальна кількість спостережень у вибірці.

MSE обчислює середнє значення квадратів різниць між передбаченими та спостережуваними значеннями, сильніше караючи за більші помилки та підкреслюючи значні розбіжності [35]:

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2, \quad (2.44)$$

MAPE виражає середню абсолютну похибку як відсоток від спостережуваних значень, що робить її незалежною від масштабу і корисною для порівняння моделей на різних наборах даних [35]:

$$MAPE = \frac{1}{n} \sum_{i=1}^n \left| \frac{y_i - \hat{y}_i}{y_i} \right|, \quad (2.45)$$

R^2 визначає частку варіації в спостережуваних даних, яку пояснює модель, демонструючи загальну якість відповідності моделі. Значення, ближчі до 1, свідчать про кращу здатність моделі пояснювати дані [35]:

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}, \quad (2.46)$$

Кожна з цих метрик надає унікальну інформацію про точність моделі та надійність передбачень.

Моделювання оптимального портфеля з використанням технологій Deep Learning дозволяє не лише досягти високої точності прогнозів, але й забезпечити ефективне управління інвестиціями, що має значний економічний ефект. Глибокі нейронні мережі (DNN, RNN, LSTM, CNN), завдяки здатності до аналізу складних

взаємозв'язків та нелінійних залежностей у фінансових даних, значно перевершують традиційні методи, такі як GARCH чи ARIMA, у адаптації до динамічних ринкових умов. Це дозволяє інвесторам і фінансовим інституціям зменшити ризики та максимізувати дохідність портфеля. Впровадження таких методів дозволяє не лише підвищити точність прогнозів, але й оптимізувати витрати, пов'язані з прийняттям інвестиційних рішень.

Ефективніше прогнозування цін активів на основі нейронних мереж зменшує необхідність утримання великих резервів капіталу для покриття можливих втрат, що сприяє більш раціональному використанню ресурсів. Це також забезпечує зниження транзакційних витрат, оскільки передбачувана динаміка ринку дозволяє скоротити кількість непотрібних операцій.

Висновки до другого розділу

Аналіз другого розділу, присвяченого аналітиці та математичному моделюванню оптимальних портфелів, демонструє різноманітність підходів до вирішення задачі оптимізації інвестиційного портфеля. У межах цього дослідження розглянуто класичні методи моделювання, статистичні моделі для аналізу часових рядів, а також сучасні технології на основі глибокого навчання.

Спершу, обговорюється класична модель оптимізації портфеля цінних паперів, заснована на теорії Гаррі Марковіца. Цей підхід передбачає оптимізацію двох ключових критеріїв: очікуваної дохідності та ризику. Модель дозволяє формалізувати процес створення портфеля, використовуючи математичні вирази для обчислення очікуваної дохідності, дисперсії та стандартного відхилення. Незважаючи на фундаментальність та універсальність, модель має певні обмеження, зокрема, фокусування на поточний стан ринку без врахування його динаміки. Це робить її менш придатною для довгострокових стратегій, де важливі передбачення майбутніх змін ринку.

Було представлено розширення класичного підходу, де розглянуто статистичні методи аналізу часових рядів, такі як GARCH та ARIMA. Модель GARCH виявляється особливо корисною для оцінки волатильності, оскільки вона враховує історичні дані та умовну гетероскедастичність. Завдяки цьому можна прогнозувати динаміку ризиків, що забезпечує адаптацію до змін у ринковому середовищі. ARIMA, у свою чергу, є потужним інструментом для моделювання трендів та сезонності, що дозволяє враховувати нестабільність часових рядів та покращувати прогнози майбутніх доходностей. Ці моделі надають гнучкість та здатність до більш глибокого аналізу даних, однак їхнє використання потребує значних обчислювальних ресурсів та високої кваліфікації аналітика.

Особливий акцент у дослідженні зроблено на сучасних підходах, базованих на технологіях глибокого навчання. Нейронні мережі, такі як DNN, RNN, LSTM та CNN, пропонують широкі можливості для обробки складних фінансових даних та моделювання залежностей у часових рядах. Наприклад, рекурентні мережі (RNN) і мережі довготривалої короткочасної пам'яті (LSTM) є ефективними у врахуванні часових залежностей, що важливо для прогнозування майбутніх показників. Глибокі нейронні мережі забезпечують адаптивність і високу точність прогнозування завдяки здатності до навчання на великих масивах даних. Для оптимізації навчання використано функції втрат, такі як MSE та функція Хубера, що забезпечують ефективну корекцію параметрів моделі. Метрики оцінки, зокрема MAE, MSE, MAPE та R^2 , дозволяють об'єктивно оцінити якість моделей, забезпечуючи інструменти для їхнього порівняння та вибору найкращої. Однією з переваг підходів, заснованих на глибокому навчанні, є можливість інтеграції різних архітектур для побудови змішаних моделей, які поєднують сильні сторони кожної з них. Це особливо важливо для задач, що потребують гнучкості та високого рівня адаптивності до змін ринкових умов. Наприклад, використання LSTM у комбінації зі згортковими шарами дозволяє отримати моделі, що не лише враховують часові залежності, а й виділяють ключові особливості вхідних даних.

Усі розглянуті підходи, незважаючи на свої відмінності, спрямовані на досягнення головної мети — формування оптимального портфеля, який забезпечує баланс між дохідністю та ризиком. Класичні методи надають чіткі математичні алгоритми для досягнення цієї мети, статистичні моделі розширюють можливості аналізу, а сучасні технології на базі нейронних мереж дозволяють врахувати складні залежності та великі обсяги даних.

Використання сучасних технологій та моделей дозволяє отримати більш точні прогнози та знижує ризики, водночас забезпечуючи максимальну дохідність у довгостроковій перспективі.

3 ПРОГНОСТИЧНА ЕКОНОМІЧНА АНАЛІТИКА ОПТИМАЛЬНИХ ІНВЕСТИЦІЙНИХ ПОРТФЕЛЕЙ

3.1 Прогностична аналітика реалізації інвестиційного портфелю з використанням методів аналізу часових рядів

Припустимо, що підприємство володіє вільними фінансовими ресурсами, проте не готове інвестувати у розширення своєї діяльності. Воно обирає шлях інвестувати у цінні папери, що торгуються на фондовому ринку США. Підприємство обрало для себе 6 компаній в які готове інвестувати: Apple, Microsoft, Google, Amazon, Visa, Nvidia. Компанії мають наступні символи акцій, за допомогою яких можна відстежувати їх інформацію:

Таблиця 3.1 – Компанії та їх стокові символи, які обрала компанія для інвестування

Компанії	Символ
Apple	AAPL
Google	GOOGL
Nvidia	NVDA
Visa	V
Amazon	AMZN
Microsoft	MSFT

Джерело: [40]

Розпочнемо побудову оптимального портфелю застосувавши модель ARIMA. Для застосування моделі ARIMA та проведення аналітики за допомогою Big Data технологій, необхідно інсталювати прикладі бібліотеки в середовищі Python. Для моделювання ARIMA застосовуються наступні бібліотеки:

- Pandas — бібліотека для зручної роботи з великими обсягами даних. Забезпечує функціонал для зберігання, обробки та аналізу даних у форматі таблиць (DataFrame), підтримує сортування, фільтрацію, об'єднання та групування даних [41];

- NumPy — інструмент для роботи з числовими масивами та багатовимірними структурами даних. Бібліотека надає потужний набір функцій для обчислення лінійної алгебри, роботи з випадковими числами та інших числових операцій [42];

- SciPy — математична бібліотека, яка розширює можливості NumPy. Використовується для розв'язання задач оптимізації, інтегрування, обчислення статистики та роботи з диференціальними рівняннями [43];

- Matplotlib — популярна бібліотека для візуалізації даних. Дозволяє створювати діаграми, графіки, гистограми, візуалізувати часові ряди та будь-які інші залежності між даними;

- YFinance — бібліотека для завантаження фінансових даних, таких як історичні ціни акцій, фінансові показники та інша інформація про компанії, з платформи Yahoo Finance. Вона значно спрощує доступ до фінансової інформації для аналізу та моделювання;

- Arch — спеціалізована бібліотека для аналізу часових рядів та роботи з моделями умовної гетероскедастичності, зокрема GARCH. Використовується для прогнозування волатильності, що має важливе значення в фінансових дослідженнях [44];

- Auto ARIMA — модуль для автоматичного підбору параметрів моделей ARIMA (AutoRegressive Integrated Moving Average). Використовується для моделювання часових рядів та прогнозування, дозволяючи зменшити час на ручний вибір параметрів;

- Statsmodels — бібліотека для статистичного аналізу даних. Підтримує широкий набір інструментів для регресійного аналізу, тестування гіпотез, моделювання часових рядів і створення статистичних моделей [45].

Для роботи із прогностичною аналітикою, потрібно завантажити дані з сайту Yahoo Finance за допомогою Yahoo Finance API [46]. За допомогою API вивантажимо дані за період 2020-01-01 - 2024-10-20. Такий період взятий, оскільки він не враховує

кризу 2007 року, проте враховує кризу спричинену Covid19 та кризу в наслідок війни Росії проти України.

Маючи завантажені дані, проглянемо загальні тенденції та динаміку (рис. 3.1)

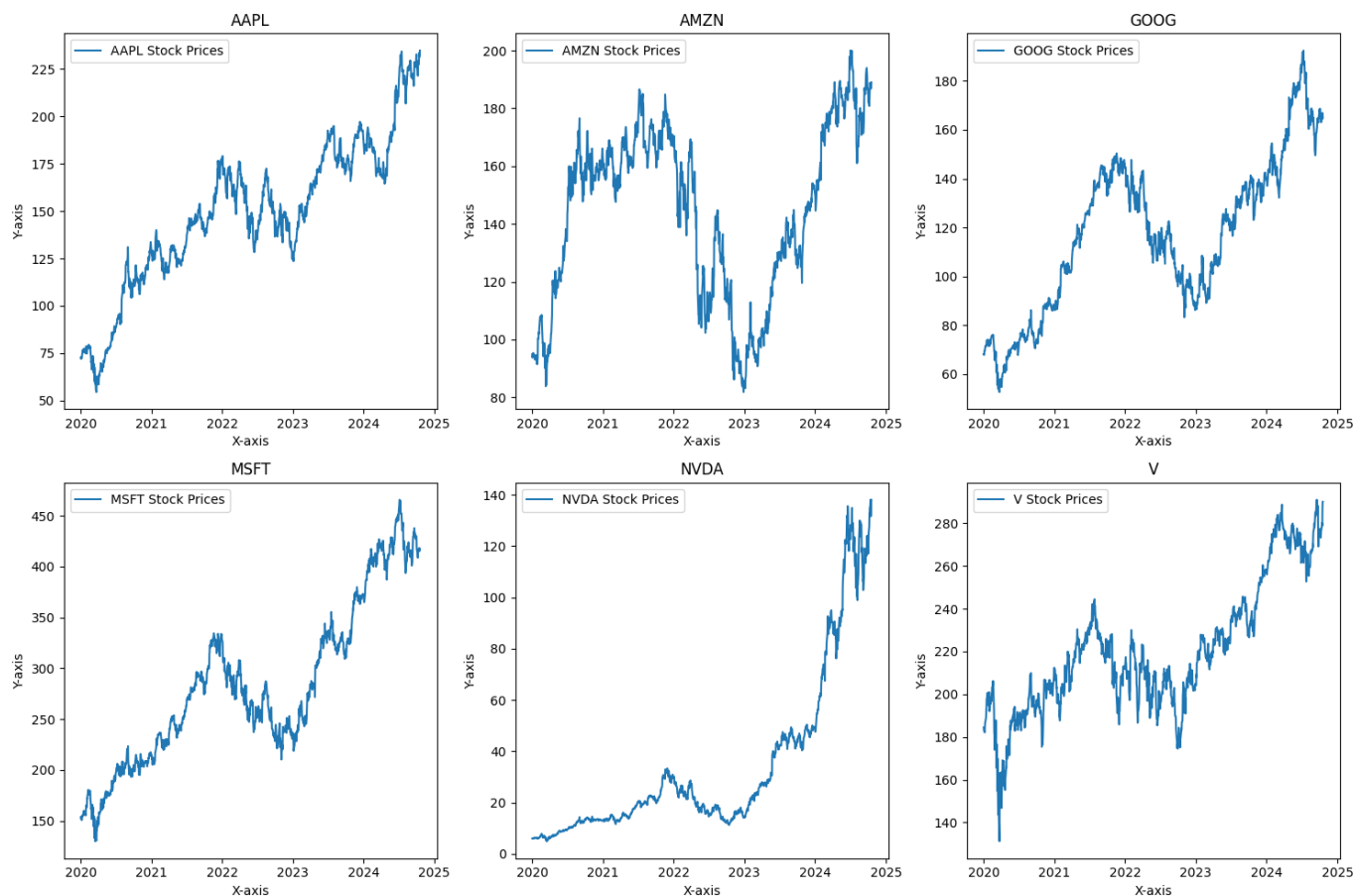


Рисунок 3.1 – Історичні ціни акцій цільових компаній

Джерело: розраховано за даними [40]

Акції AAPL демонструють стійке зростання з регулярними короткостроковими корекціями. Це свідчить про надійність позицій компанії на ринку, що підкріплюється інноваційними продуктами та стабільною клієнтською базою. Зростання вартості акцій вказує на ефективну бізнес-стратегію компанії, що фокусується на екосистемному підході.

Графік AMZN відображає значну волатильність із різкими підйомами та спадами. Після піку в 2021 році спостерігалось зниження у 2022, ймовірно, через постпандемічну перебудову ринку та скорочення обсягів електронної комерції. Проте

у 2023-2024 роках акції почали відновлюватися, що може бути пов'язано з розвитком хмарних сервісів та новими джерелами доходу.

Динаміка акцій GOOG демонструє схожу тенденцію з AMZN: волатильність, підйоми та спади. Зростання у 2024 році може свідчити про інноваційні досягнення компанії, зокрема у сфері штучного інтелекту та цифрової реклами. Незважаючи на це, значні коливання вказують на виклики регуляторного контролю та конкуренції.

Акції MSFT відображають стійке і стабільне зростання. Цей тренд підтверджує ефективність хмарних сервісів Azure та стратегічний фокус на штучному інтелекті. Пік акцій у 2024 році свідчить про високу довіру інвесторів до компанії як провідного гравця в технологічній галузі.

NVDA демонструє вибухове зростання з 2022 року, що збігається з підвищеним попитом на графічні процесори для застосування у штучному інтелекті та обробці великих даних. Таке стрімке зростання робить компанію однією з найбільш привабливих для інвестицій, хоча існує ризик перенасичення ринку та залежності від одного сектора.

Акції V характеризуються стабільним зростанням із помірними коливаннями. Це вказує на високу довіру до компанії як до ключового гравця у сфері фінансових послуг. Збільшення популярності цифрових платежів підтримує зростання, хоча ризики можуть бути пов'язані з регуляторними змінами у фінансовому секторі.

Загальний аналіз показує, що технологічні компанії, такі як AAPL, MSFT та NVDA, демонструють високий потенціал зростання завдяки інноваціям, тоді як AMZN і GOOG залишаються під впливом волатильності. V стабільна завдяки глобальному попиту на фінансові послуги. Така різниця у динаміці акцій свідчить про доцільність диверсифікації портфеля для мінімізації ризиків.

Перед початком моделювання проведемо додатково статистичний аналіз даних та аналіз пропущених значень. Аналіз пропущених значень виконаємо за допомогою команди `stock_data.isna().sum()`.

Таблиця 3.2 – Кількість пропущених значень в щоденних цінах акцій

Символ	К-сть пропущених значень
AAPL	0
GOOGL	0
NVDA	0
V	0
AMZN	0
MSFT	0

Джерело: розраховано за даними [40]

Виконуючи прогнозування за допомогою моделі ARIMA, потрібно поділити дані на тренувальну та тестову вибірки. Поділимо дані так, щоб тренувальна вибірка була 75%, а тестувальна – 25%.

Маючи розподілену вибірку, натренуємо модель. Спочатку створюється порожній словник `arima_predictions`, у який будуть зберігатися прогнози для кожного стовпця даних. Потім для кожного стовпця в даних (який представляє часовий ряд) виконується цикл, що розбиває ряд на навчальну та валідаційну частини. Цей розподіл виконується за допомогою змінної `split_time`, яка встановлює межу, що становить 75% від загальної довжини ряду. Далі використовується бібліотека `rmдаріма` для автоматичного визначення оптимальних гіперпараметрів (p , d , q) моделі ARIMA для навчальної частини. Це виконується за допомогою функції `auto_arima`, яка повертає об'єкт із визначеним порядком моделі. Функція `arima_forecast` відповідає за створення та тренування моделі ARIMA, а також за генерацію одного прогнозу на основі історичних даних. Вона приймає як вхідні параметри поточний історичний ряд і вже підготовлену модель.

Далі реалізується «покрокова» валідація (`walk-forward validation`). Для цього початкові історичні дані копіюються у список `history`, на основі якого формується модель. Прогнози генеруються по одному кроку вперед за допомогою `arima_forecast`,

після чого реальні валідаційні значення додаються до історії для використання в наступних прогнозах.

Натренувавши модель та здійснивши прогнозування, порівняємо прогнозовані значення із тестовими даними. Проведемо графічний аналіз прогнозування (рис. 3.2):

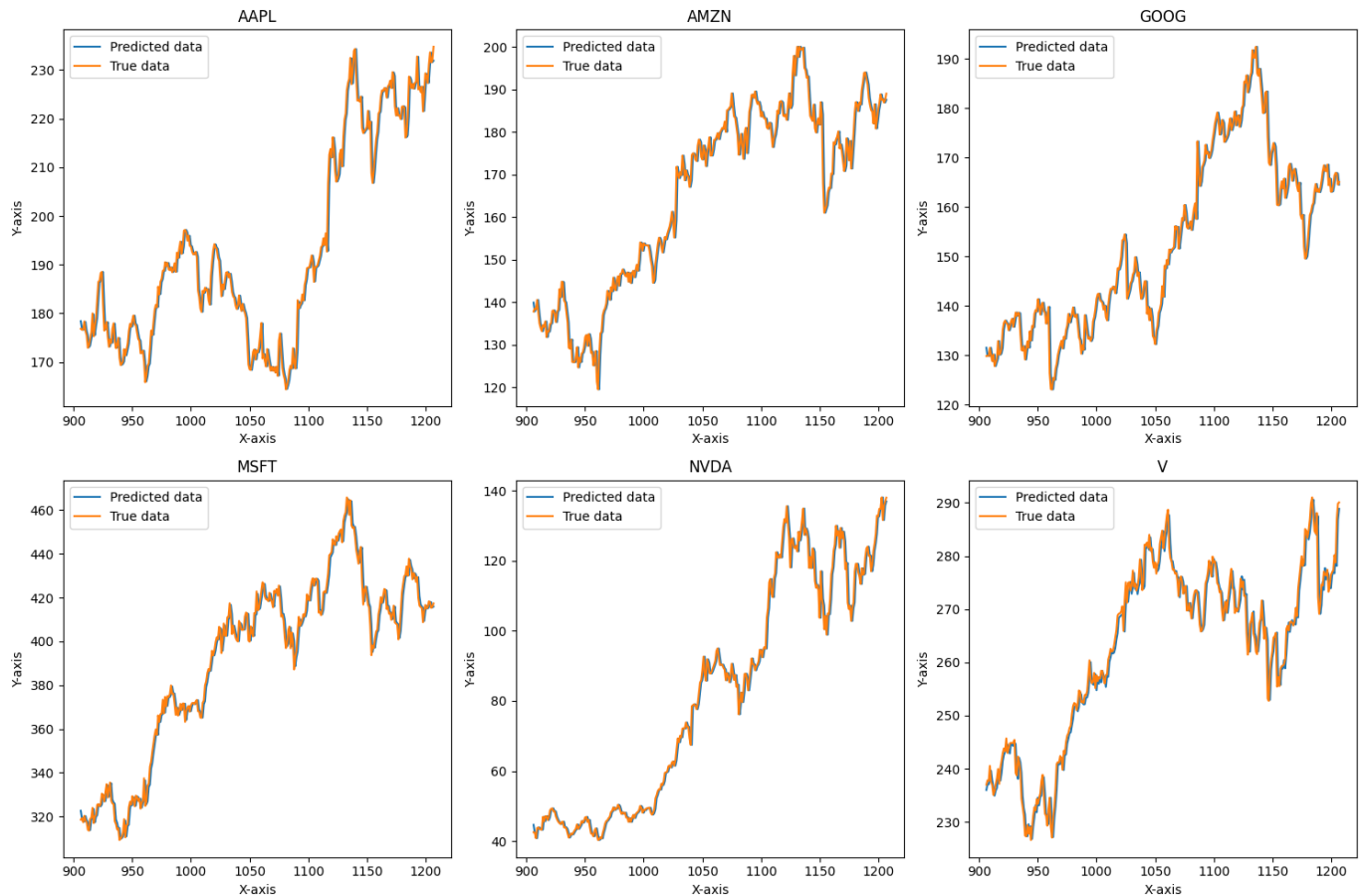


Рисунок 3.2 – Порівняльний аналіз прогнозованих значень моделі ARIMA із фактичними даними

Джерело: складено автором на основі власних розрахунків

Як видно з графіку, модель ARIMA показує високу точність у прогнозуванні даних. Більш точно це можна побачити, застосувавши ключові метрики для оцінки точності прогнозу такі як MSE, MAE, MAPE та R^2 . Результати продемонструємо у таблиці 3.3:

Таблиця 3.3 – Ключові метрики прогнозування ARIMA

	AAPL	AMZN	GOOG	MSFT	NVDA	V
MSE	7,548	8,054	6,586	22,775	8,996	7,321
MAE	2,015	2,099	1,812	3,707	2,050	2,028
MAPE	1,041	1,289	1,196	0,951	2,357	0,773
R ²	0,982	0,983	0,979	0,986	0,991	0,973

Джерело: складено автором на основі власних розрахунків

Модель показала високу точність прогнозування, що підтверджується значенням коефіцієнта детермінації (R²), яке знаходиться в межах від 0.973 до 0.991 для всіх компаній. Це свідчить про те, що майже весь розподіл даних пояснюється моделлю. Найвищий R² спостерігається для NVIDIA (0.991), що вказує на сильний зв'язок між фактичними та прогнозованими значеннями. Це може бути пов'язано з чіткою тенденцією зростання її акцій у часі, яку модель змогла легко уловити. Найнижче значення R² (0.973) спостерігається для Visa, що, хоча й залишається високим, може вказувати на трохи більший рівень випадковості в її часовому ряді.

Показники середньоквадратичної помилки (MSE) та середньої абсолютної помилки (MAE) є досить низькими для всіх компаній, що підтверджує здатність моделі точно відтворювати короткострокові коливання. Високе значення MSE для Microsoft (22.775) може свідчити про більшу волатильність її даних, але це не позначається суттєво на загальній ефективності моделі, оскільки R² все одно становить 0.986.

Середня абсолютна відсоткова помилка (MAPE) для всіх компаній знаходиться на низькому рівні, що вказує на те, що помилки у прогнозах є мінімальними у відносному вираженні. Найнижча MAPE для Visa (0.773) свідчить про відносно стабільну природу її часових рядів, а найвища для NVIDIA (2.357) підкреслює, що навіть за точного прогнозу відносні відхилення можуть зростати через стрімке зростання цін на її акції.

Економічно, такі результати підтверджують, що акції компаній з чіткими довгостроковими трендами (наприклад, NVIDIA) легко піддаються моделюванню,

тоді як компанії з більшою волатильністю (Microsoft, Visa) можуть створювати більше труднощів для передбачення короткострокових змін.

Маючи прогнозовані значення, проведемо побудову портфелю на основі прогнозованих значень за допомогою моделі ARIMA. Побудову портфелю розпочнемо із пошуку очікуваних норм прибутку та очікуваних ризиків для кожної компанії. Відобразимо прогнозовані щоденні норми прибутку графічно (рис 3.3)

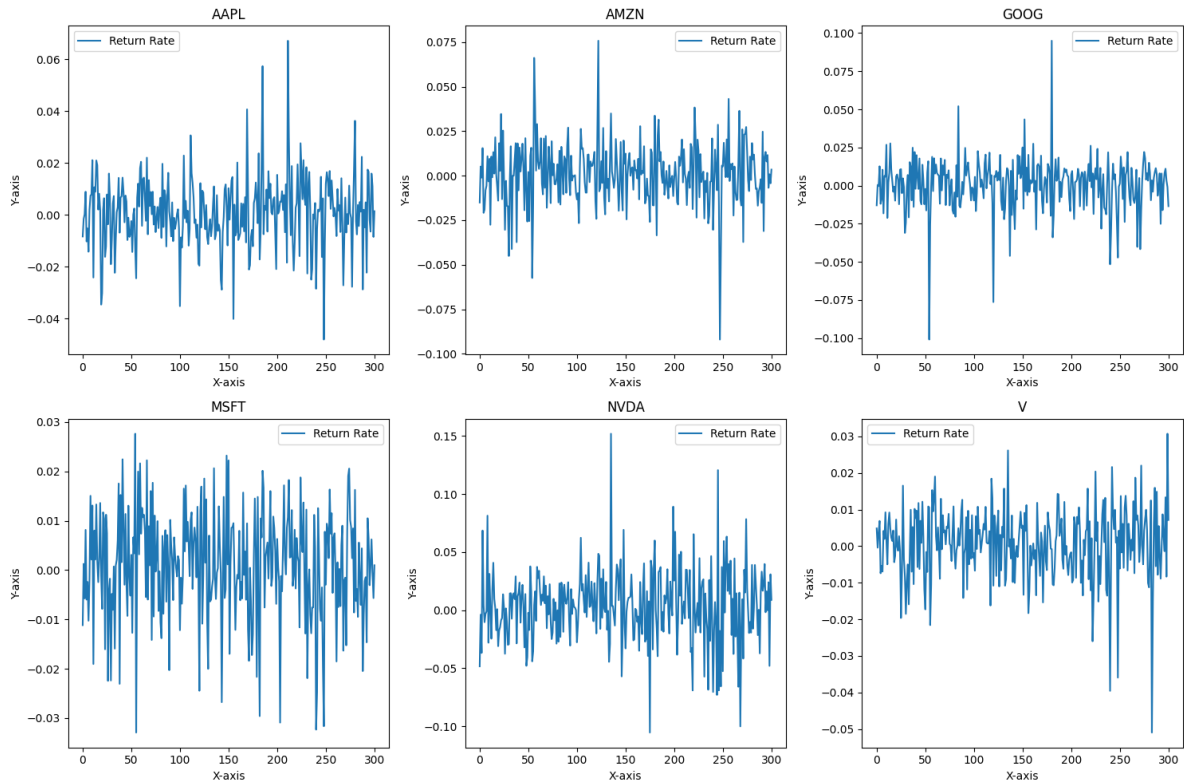


Рисунок 3.3 – Щоденні норми прибутку прогнозованих значень

Джерело: складено автором на основі власних розрахунків

Маючи щоденні норми прибутку, розрахуємо очікувано норму прибутку та очікуваний ризик для акцій кожної компанії. Норму очікуваного прибутку розрахуємо за наступною формулою:

$$R = \frac{\sum_{i=1}^6 r_i}{6}, \quad (3.1)$$

Очікуваний ризик розраховується як стандарте відхилення норм прибутку:

$$\sigma = \sqrt{\frac{\sum_{i=1}^6 (R_i - \bar{R})^2}{6 - 1}}, \quad (3.2)$$

Розраховані показники представимо у вигляді таблиці:

Таблиця 3.4 – Очікувана норма прибутку та очікуваний ризик для обраних компаній на основі прогнозованих даних

Символ	Очікувана норма прибутку	Очікуваний ризик
AAPL	0,000871	0,013638
AMZN	0,000973	0,017511
GOOG	0,000744	0,016920
MSFT	0,000844	0,011012
NVDA	0,003723	0,031535
V	0,000671	0,009682

Джерело: складено автором на основі власних розрахунків

Маючи очікувану норму прибутку та очікуваний ризик кожної компанії, побудуємо оптимальний портфель акцій. Для цього на основі прогнозованих ARIMA значеннях побудуємо коваріаційну матрицю – матрицю ризиків. Коваріаційна матриця має вигляд:

Таблиця 3.5 – Коваріаційна матриця на основі прогнозованих ARIMA значень

Символ	AAPL	AMZN	GOOG	MSFT	NVDA	V
AAPL	0,000186	0,000094	0,000098	0,000076	0,000139	0,000028
AMZN	0,000094	0,000307	0,000167	0,000120	0,000257	0,000043
GOOG	0,000098	0,000167	0,000286	0,000100	0,000203	0,000035
MSFT	0,000076	0,000120	0,000100	0,000121	0,000165	0,000038
NVDA	0,000139	0,000257	0,000203	0,000165	0,000994	0,000047
V	0,000028	0,000043	0,000035	0,000038	0,000047	0,000094

Джерело: складено автором на основі власних розрахунків

Після побудови коваріаційної матриці можна перейти до наступного етапу моделювання, що включає визначення цільових функцій та оптимізаційного процесу. Відповідно до моделі, розробленої в розділі 2, передбачено дві цільові функції f_1 та f_2 , які розраховуються за формулами (2.4) і (2.5).

Після визначення цільових функцій виконується згортка критеріїв, а оптимізація цільової функції спрямована на знаходження її мінімуму. З урахуванням властивостей критеріїв формується система обмежень, яка представлена у формулі (2.6).

$$\begin{cases} \sum_{i=1}^6 w_i = 1 \\ w_i \geq 0,01 \\ R_p > 0 \\ \sigma_p \geq 0 \end{cases}$$

Для того, щоб провести оптимізацію одночасно двох критеріїв, застосуємо формулу (2.7). Маючи обмеження та цільову функцію, сформуємо повноцінну задачу оптимізації. Врахуємо, що інвестор є нейтральним до вищого доходу та низького ризику, то ж параметр α у згортці критеріїв буде рівний 0.5. Для моделювання застосуємо формулу (2.9)

$$W = \frac{0.5 * \sqrt{w^T * cov * w}}{(1 - 0.5) * \sum_{i=1}^6 R_i * w_i} \rightarrow \min$$

$$\begin{cases} \sum_{i=1}^6 w_i = 1 \\ w_i \geq 0,01 \\ \sum_{i=1}^6 R_i * w_i > 0 \\ \sqrt{w^T * cov * w} \geq 0 \end{cases}$$

Підставивши всі змінні, отримуємо наступні результати:

- Цільова оптимізаційна функція $W = 7,699$;
- Очікуваний ризик $\sigma_{\text{портфелю}} = 1,319\%$;
- Очікуваний дохід $I = 0,171\%$.

У таблиці нижче будуть наведені вагові коефіцієнти, що характеризують частки кожного активу в складі сформованого інвестиційного портфеля:

Таблиця 3.6 – Ваги портфелю сформованого на основі прогнозованих ARIMA значеннях

Символ компанії	W
MSFT	0,010
AMZN	0,010
GOOG	0,010
AAPL	0,134
NVDA	0,331
V	0,505

Джерело: складено автором на основі власних розрахунків

Ваги портфеля, сформованого на основі прогнозованих значень ARIMA, демонструють нерівномірний розподіл інвестицій серед компаній. Найбільша частка портфеля припадає на компанію Visa (V) з вагою 0,505, що вказує на її стабільність або високу привабливість для інвесторів. NVIDIA (NVDA) також має значну частку в портфелі – 0,331, що може бути результатом високих темпів зростання її акцій. Акції Apple (AAPL) займають помірну частку в портфелі з вагою 0,134. Інші компанії, такі як Microsoft (MSFT), Amazon (AMZN) та Google (GOOG), мають найменші ваги (по 0,010 кожна), що свідчить про менший вплив цих активів у сформованому портфелі.

Наступним кроком є побудова оптимального портфелю за допомогою моделі GARCH. GARCH має дещо інший підхід, оскільки він не прогнозує значення цін акцій, а одразу надає прогнозовану волатильність на заданий період, що в свою чергу формує матрицю ризиків. Для калькуляції прогнозованої коваріаційної матриці, знайдемо матрицю умовних кореляцій за формулою (2.14):

Таблиця 3.7 – Матриця умовних кореляцій CCC

Символ	AAPL	AMZN	GOOG	MSFT	NVDA	V
AAPL	1,029534	0,579240	0,595685	0,663052	0,547900	0,469716
AMZN	0,579240	1,050238	0,650976	0,667890	0,567508	0,384740
GOOG	0,595685	0,650976	0,987008	0,688008	0,549556	0,441593
MSFT	0,663052	0,667890	0,688008	0,999732	0,635715	0,472105
NVDA	0,547900	0,567508	0,549556	0,635715	0,990377	0,392265
V	0,469716	0,384740	0,441593	0,472105	0,392265	0,994757

Джерело: складено автором на основі власних розрахунків

Після побудови матриці умовних кореляцій, перейдемо до обрахунку прогнозованої коваріаційної матриці за формулою (2.15). Обрахована матриця має наступний вигляд:

Таблиця 3.8 – Прогнозована коваріаційна матриця

Символ	AAPL	AMZN	GOOG	MSFT	NVDA	V
AAPL	1,029534	0,579240	0,595685	0,663052	0,547900	0,469716
AMZN	0,579240	1,050238	0,650976	0,667890	0,567508	0,384740
GOOG	0,595685	0,650976	0,987008	0,688008	0,549556	0,441593
MSFT	0,663052	0,667890	0,688008	0,999732	0,635715	0,472105
NVDA	0,547900	0,567508	0,549556	0,635715	0,990377	0,392265
V	0,469716	0,384740	0,441593	0,472105	0,392265	0,994757

Джерело: складено автором на основі власних розрахунків

Маючи прогнозовану коваріаційну матрицю, перейдемо до побудови оптимального портфелю та знаходження ключових показників. Процес побудови портфелю аналогічний до того як це відбувалося в моделі ARIMA: застосування формул (2.4) – (2.9).

Вирішивши задачу оптимізації, маємо наступні результати:

- Цільова оптимізаційна функція $W = 10,627$;
- Очікуваний ризик $\sigma_{\text{портфелю}} = 1,604\%$;
- Очікуваний дохід $I = 0,150\%$.

У таблиці наведемо ваги нового обрахованого портфелю

Таблиця 3.9 – Ваги портфелю сформованого на основі GARCH підходу

Символ компанії	W
AMZN	0,010
GOOG	0,010
V	0,010
AAPL	0,181
NVDA	0,374
MSFT	0,415

Джерело: складено автором на основі власних розрахунків

Аналіз показує, що портфель, сформований на основі GARCH-підходу, значною мірою орієнтований на акції Microsoft (MSFT) та NVIDIA (NVDA), які отримали найбільші ваги – 0,415 та 0,374 відповідно. Це свідчить про високу ефективність цих

активів у контексті зниження ризиків та максимізації очікуваної дохідності. Apple (AAPL) займає середню позицію в портфелі з вагою 0,181, тоді як ваги акцій Amazon (AMZN), Google (GOOG) та Visa (V) значно нижчі – по 0,010 для кожної компанії. Така структура портфеля вказує на сильну концентрацію капіталу в технологічному секторі, особливо в акціях лідерів галузі, що може бути виправдано високими очікуваннями щодо їхнього потенціалу зростання. Низькі ваги інших активів можуть бути обумовлені їхньою меншою стійкістю до волатильності або меншою ефективністю у зниженні загального ризику портфеля.

3.2 Аналітичних методи машинного навчання для прогнозування результатів реалізації інвестиційних проєктів

Маючи всі ті ж самі умови економічної задачі, здійснимо прогностичне та оптимізаційне моделювання за допомогою Deep Learning технологій для часових рядів. Вхідними даними будуть історичні ціни акцій компаній з попереднього підрозділу (табл. 3.1).

Розпочнемо наше моделювання із завантаження необхідних бібліотек у середовище Python:

- TensorFlow – це одна з найбільш популярних бібліотек для машинного навчання. Вона використовується для побудови, тренування та оцінювання моделей машинного навчання та глибокого навчання. TensorFlow забезпечує зручний інтерфейс для роботи з нейронними мережами, дозволяючи виконувати чисельні обчислення з використанням графів обчислень. У даному коді бібліотека може використовуватись для реалізації метрик оцінки або створення складніших моделей;
- Keras – це високорівневий API для створення нейронних мереж, який працює поверх TensorFlow [47];

Розпочнемо прогностичне моделювання із побудови нейронної мережі, яка буде побудована на архітектурі щільних шарів (Dense) [48].

Процес моделювання нейронних мереж завжди починається з ретельної підготовки навчальних і тестових даних. Цей етап є критично важливим, оскільки якісна підготовка даних безпосередньо впливає на точність та узагальнюючу здатність моделі. Для того, щоб навчання нейронної мережі було ефективним, дані мають бути перетворені в спеціальний формат, який називається віконним представленням (sliding window format). У цьому форматі дані не подаються на вхід мережі у вигляді суцільної вибірки, а розбиваються на менші послідовності або «вікна». Кожне «вікно» представляє собою підмножину даних, що використовується як вхід для моделі, а цільове значення визначається на основі майбутніх значень у послідовності. Такий підхід є особливо важливим у роботі з часовими рядами, оскільки дозволяє нейронній мережі вчитися враховувати залежності між послідовними значеннями. Наприклад, якщо маємо часовий ряд, віконний вигляд дозволяє створювати підмножини даних, де кожне вікно містить кілька попередніх значень (вхід) і одне наступне значення (вихід). Це допомагає мережі краще розуміти структуру даних і робити точні прогнози. Вибір розміру вікна (кількості попередніх значень у кожному фрагменті) є ключовим параметром, який визначає здатність моделі вловлювати коротко- чи довгострокові залежності.

Напишемо функцію `windowed_dataset`, яка приведе дані у потрібний формат. Функція приймає часовий ряд (послідовність значень) і розбиває його на сегменти, які потім використовуються для побудови моделей прогнозування.

Функція отримує чотири аргументи: `series`, що представляє часовий ряд у вигляді масиву чисел; `window_size`, який задає розмір кожного вікна (кількість часових кроків, що входять у кожне вікно); `batch_size`, який визначає розмір пакету даних для навчання; та `shuffle_buffer`, що вказує розмір буфера для перемішування набору даних.

У середині функції використовується API `TensorFlow Dataset` для перетворення даних. Спершу з послідовності значень створюється об'єкт `tf.data.Dataset` за

допомогою методу `from_tensor_slices`. Це дозволяє працювати з масивом даних у вигляді структурованого набору, оптимізованого для навчання моделей.

Потім застосовується метод `window`, який розбиває дані на послідовності фіксованого розміру (`window_size + 1`). Тут додається 1, щоб мати можливість розділити кожне вікно на вхідні значення (фічі) та вихідне значення (мітку). Параметр `shift=1` задає зміщення між послідовними вікнами, тобто вікна накладаються одне на одне. Аргумент `drop_remainder=True` гарантує, що всі вікна матимуть однаковий розмір.

Метод `flat_map` використовується для згортання вкладених структур, що створюються під час розбиття на вікна, у плоский список. У цьому випадку кожне вікно перетворюється в окрему партію за допомогою функції `lambda window: window.batch(window_size + 1)`.

Далі використовується метод `map` для перетворення кожного вікна. Він розділяє кожне вікно на дві частини: усі значення, окрім останнього (вхідні значення, `window[:-1]`), та останнє значення (вихідна мітка, `window[-1]`).

Щоб уникнути залежності моделі від порядку даних, використовується метод `shuffle`, який перемішує вікна. Розмір буфера для перемішування задається параметром `shuffle_buffer`, що дозволяє контролювати ефективність перемішування.

Після цього метод `batch` об'єднує кілька вікон у пакети (`batch_size`), що спрощує навчання моделей, які оптимізовані для роботи з пакетами даних. Останній етап включає використання методів `cache` та `prefetch`. `cache` дозволяє зберігати дані в оперативній пам'яті, що прискорює повторювані виклики, тоді як `prefetch(1)` асинхронно завантажує наступний пакет, зменшуючи затримки під час навчання. Функція повертає підготовлений об'єкт `dataset`, який можна безпосередньо використовувати для навчання моделей у TensorFlow.

Передамо для функції формування набору даних часовий ряд AAPL. Маючи сформований набір даних, побудуємо модель. Модель буде мати вигляд (рис. 3.4)

Model: "sequential_25"

Layer (type)	Output Shape	Param #
dense_70 (Dense)	(None, 150)	3,150
dense_71 (Dense)	(None, 100)	15,100
dense_72 (Dense)	(None, 50)	5,050
dense_73 (Dense)	(None, 1)	51

Total params: 23,351 (91.21 KB)

Trainable params: 23,351 (91.21 KB)

Non-trainable params: 0 (0.00 B)

Рисунок 3.4 – Параметри нейронної мережі DNN

Джерело: складено автором на основі власних розрахунків

Ми маємо структуру моделі нейронної мережі типу Sequential, що містить чотири повнозв'язні (Dense) шари:

- перший шар має 150 нейронів з 3150 параметрами,
- другий – 100 нейронів із 15100 параметрами,
- третій – 50 нейронів із 5050 параметрами,
- вихідний шар складається з одного нейрона з 51 параметром.

Загальна кількість параметрів моделі становить 23351, всі вони є тренуваними. Немає нетренованих параметрів. Це свідчить про те, що всі параметри використовуються для оптимізації в процесі навчання.

Наступним етапом скопілюємо модель за допомогою команди «compile». Передамо моделі SGD оптимайзер, швидкість навчання визначимо на рівні 10^{-7} . Скопілювавши модель, навчимо її на тренувальному наборі даних протягом 100 епох.

Переглянемо прогностичну здатність навченої моделі на тестувальних даних (рис. 3.5)

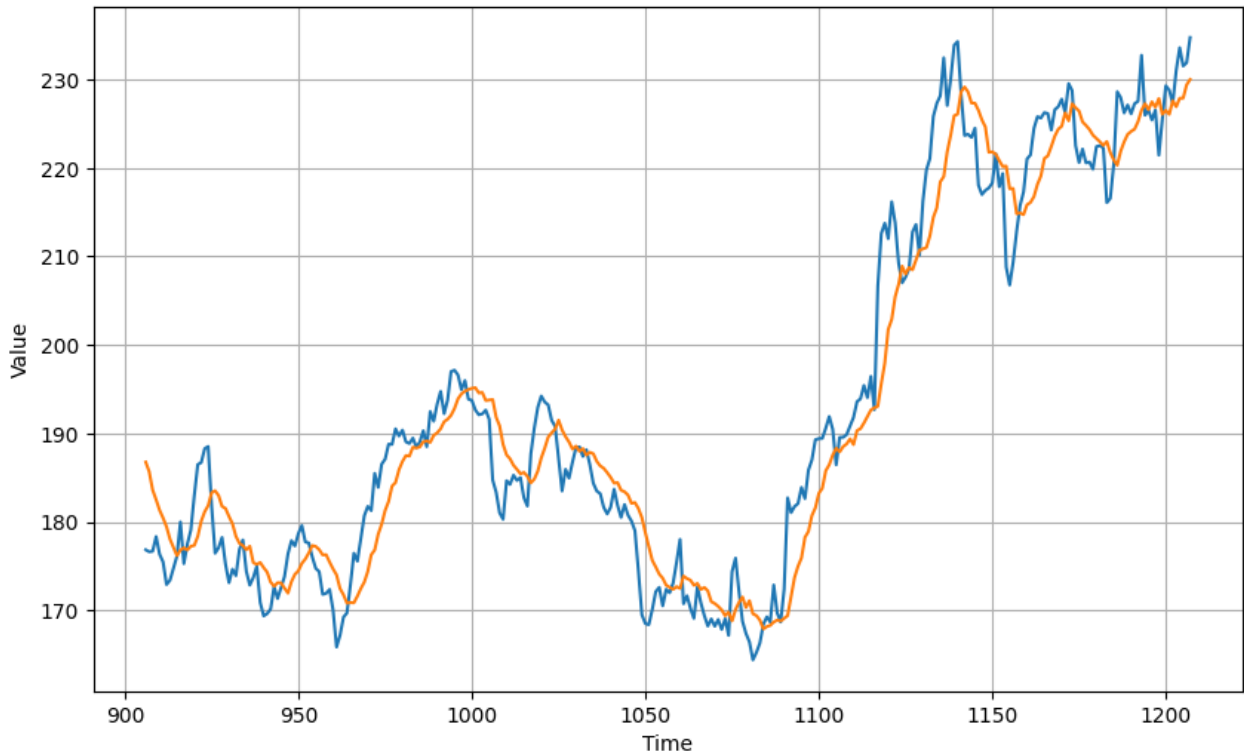


Рисунок 3.5 – Прогнозовані значення за допомогою початкової моделі

Джерело: складено автором на основі власних розрахунків

З аналізу видно, що модель добре захоплює загальну тенденцію в даних. Під час спаду значень, приблизно у проміжку між 1000 і 1050, модель досить точно слідує за фактичними значеннями, хоча місцями згладжує деякі дрібні коливання. Це свідчить про те, що модель ефективно працює зі середньостроковими трендами, але не завжди враховує короткострокову волатильність. У період різкого зростання після 1100 модель також досить точно відображає тренд, хоча є невелике відставання у прогнозуванні піків. Це може вказувати на те, що модель не повністю адаптується до швидких змін у ряді, але загалом показує високу узгодженість із загальними закономірностями. На останніх етапах, ближче до 1200, модель демонструє ще кращу точність, наближаючись до фактичних значень. Це говорить про її здатність навчатися на довгострокових залежностях, що дозволяє ефективно прогнозувати навіть на віддалених часових горизонтах. Однак згладжування різких коливань залишається одним із ключових обмежень моделі.

Проаналізуємо ключові метрики моделі такі як MSE, MAE та R^2 (табл 3.10)

Таблиця 3.10 – Ключві метрики початкової моделі DNN

Метрика	Показник
MSE	25,683
MAE	4,050
R^2	0,940

Джерело: складено автором на основі власних розрахунків

Вище зазначені метрики підтверджують аналітику описану із графічного методу. Модель добре описує та розуміє загальні тенденції, проте згладжує короткострокову волатильність.

Для того, щоб покращити модель, оптимізуємо значення швидкості навчання. Це можна зробити за допомогою аналізу графіку навчання. Натренувавши модель, проаналізуємо графік навчання моделі та те як змінювалася швидкість навчання (рис. 3.6)

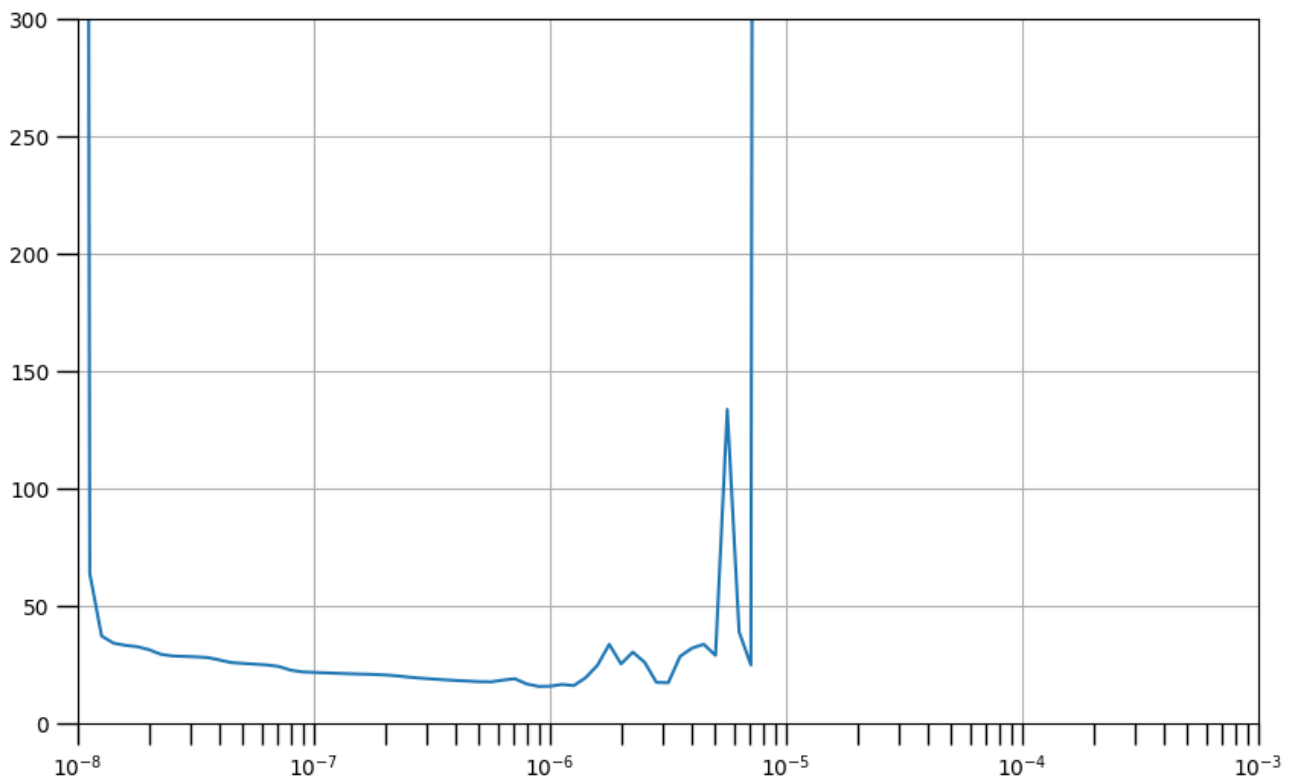


Рисунок 3.6 – Швидкість навчання моделі

Джерело: складено автором на основі власних розрахунків

Графік демонструє, що на початку (при дуже малих значеннях швидкості навчання, близько 10^{-8}) модель навчається дуже повільно, і значення метрики залишається високим. Потім значення метрики починає знижуватись, досягаючи мінімуму при значеннях швидкості навчання в діапазоні близько 10^{-6} . Після цього спостерігається різке збільшення значення метрики, що свідчить про дестабілізацію навчання через занадто велику швидкість навчання.

Оптимальне значення швидкості навчання знаходиться в районі $4 \cdot 10^{-6}$, де метрика досягає мінімального стабільного значення. Це значення забезпечує баланс між швидкістю збіжності і стабільністю навчання, дозволяючи моделі ефективно оновлювати ваги без ризику дивергенції. Для подальшого покращення моделі, візьмемо це значення та перевіримо його в навчальному процесі.

Навчивши модель, перевіримо графік втрат (рис. 3.7)

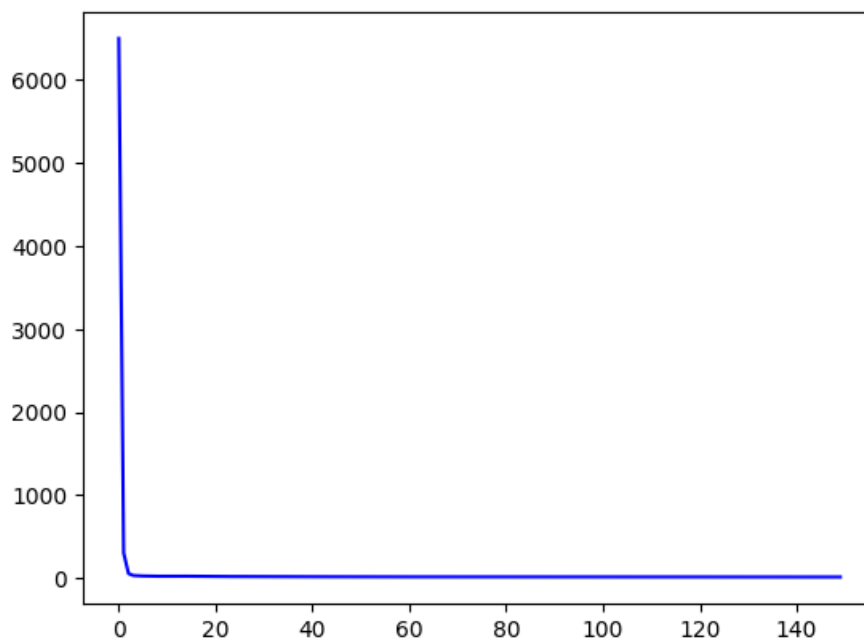


Рисунок 3.7 – Зміна втрат у процесі навчання моделі

Джерело: складено автором на основі власних розрахунків

З графіку важко помітити зміну навчання, оскільки після першої епохи рівень втрат впав досить сильно. Проаналізуємо графік почавши з 10 епохи (рис. 3.8)

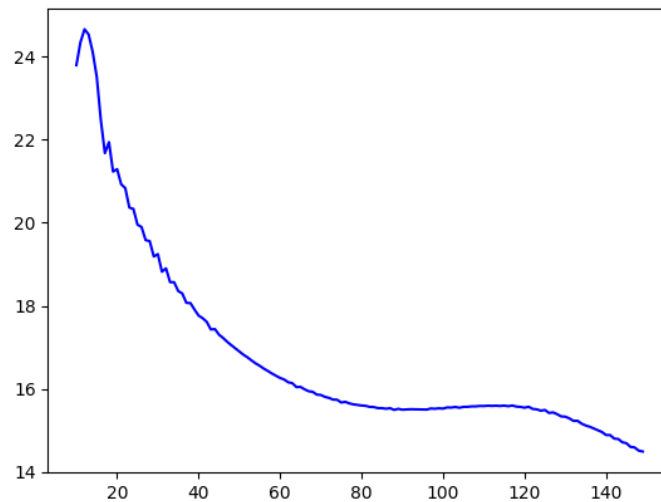


Рисунок 3.8 – Зміна втрат у процесі навчання моделі починаючи з 10 епохи
Джерело: складено автором на основі власних розрахунків

Аналізуючи графік бачимо, що процес навчання є дещо нестабільний у певних епохах, проте ми бачимо, що з кожною наступною епохою втрати падають, що свідчить про кращу натренованість моделі.

Маючи готову модель, протестуємо її на тестувальних даних та порівняємо із фактичними даними (рис. 3.9)

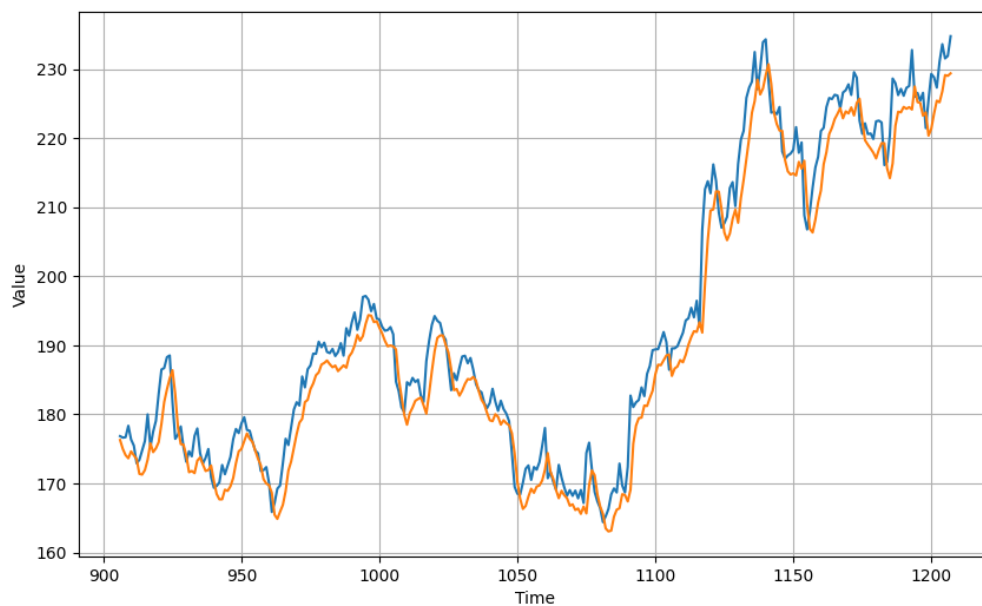


Рисунок 3.9 – Прогностичні результати оптимізованої нейронної мережі
Джерело: складено автором на основі власних розрахунків

Відмінності між фактичними даними (синя лінія) та прогнозом (помаранчева лінія) є мінімальними, що свідчить про високу точність моделі. Однак у деяких місцях помітні невеликі розбіжності, особливо у фазах швидких змін значень. Це може означати, що модель трохи запізнюється або недооцінює амплітуду таких змін.

Загалом графік демонструє, що модель ефективно прогнозує загальний тренд і більшість короткострокових коливань.

Розуміючи як працює розробка нейронної мережі, розробимо DNN нейронні мережі для прогнозування цін акцій, які обрала компанія. Для цього розробимо код для автоматичного навчання та зберігання даних. Маючи моделі, прогнозуємо дані та порівняємо із реальними даними (рис 3.10)



Рисунок 3.10 – Порівняльний аналіз DNN прогнозування із реальними даними

Джерело: складено автором на основі власних розрахунків

Загалом модель демонструє високу точність прогнозування, оскільки сині лінії дуже близько прилягають до помаранчевих, відображаючи подібну динаміку.

На графіках спостерігаються моменти, коли модель трохи запізнюється за трендом або недооцінює чи переоцінює екстремальні значення. Проте в більшості випадків основні тенденції, зокрема підйоми, спади та коливання, моделюються коректно. Наприклад, для таких компаній, як AAPL та NVDA, модель точно відтворює сильні коливання, тоді як у GOOG та V є деякі помітніші відхилення на пікових точках.

Модель ефективно справляється з передбаченням як плавних змін, так і різких стрибків, проте в деяких випадках прогнози можуть бути менш точними при різких коливаннях, як це помітно на графіку для AMZN. Для MSFT прогнозовані дані майже ідеально відповідають реальним, що свідчить про відмінне узгодження.

Проаналізуємо ключові метрики DNN прогнозування (табл. 3.11)

Таблиця 3.11 – Ключові метрики прогнозування DNN

	AAPL	AMZN	GOOG	MSFT	NVDA	V
MSE	15,166	15,230	9,431	25,770	23,868	9,338
MAE	3,087	3,038	2,367	4,024	3,547	2,341
MAPE	1,583	1,862	1,557	1,030	4,240	0,892
R ²	0,965	0,967	0,969	0,984	0,977	0,966

Джерело: складено автором на основі власних розрахунків

Аналіз ключових метрик прогнозування моделі DNN свідчить про її високу точність, що підтверджується значеннями R^2 , які знаходяться в межах від 0,965 до 0,984. Ця метрика демонструє, наскільки добре модель пояснює варіацію реальних даних.

Найкраще значення R^2 спостерігається для MSFT (0,984), що вказує на майже ідеальну відповідність між прогнозованими та фактичними значеннями для цієї компанії. Це підтверджується також відносно низькими значеннями інших помилок: MSE становить 25,770, MAE – 4,024, а MAPE – 1,030%. GOOG має досить високе значення R^2 (0,969), що теж свідчить про високу відповідність. MSE для цієї компанії

є найнижчим серед усіх (9,431), як і MAE (2,367) та MAPE (1,557%), що говорить про ефективне передбачення навіть для найменших відхилень.

AAPL, AMZN і V демонструють дуже схожі значення R^2 (0,965–0,967), що вказує на стабільну продуктивність моделі для цих компаній. Хоча їх MSE і MAE не є найкращими, MAPE для V (0,892%) є найменшим серед усіх, що свідчить про найточніше прогнозування у відносному вираженні.

Для NVDA, незважаючи на високе значення R^2 (0,977), помітно дещо гірші значення MAPE (4,240%) і MAE (3,547), що вказує на те, що модель справляється із загальними трендами, але може недооцінювати чи переоцінювати окремі коливання.

Маючи прогнозовані значення, перейдемо до побудови оптимального портфелю. Щоб побудувати портфель акцій, використаємо алгоритм описаний у підрозділі 3.1.

Прогнозована коваріаційна матриця має вигляд (табл. 3.12):

Таблиця 3.12 – Матриця ризиків на основі прогнозованих значень DNN

	AAPL	AMZN	GOOG	MSFT	NVDA	V
AAPL	0,000081	0,000036	0,000040	0,000033	0,000030	0,000012
AMZN	0,000036	0,000136	0,000055	0,000058	0,000044	0,000020
GOOG	0,000040	0,000055	0,000112	0,000042	0,000039	0,000015
MSFT	0,000033	0,000058	0,000042	0,000065	0,000038	0,000017
NVDA	0,000030	0,000044	0,000039	0,000038	0,000178	0,000008
V	0,000012	0,000020	0,000015	0,000017	0,000008	0,000037

Джерело: складено автором на основі власних розрахунків

Маючи прогнозовану коваріаційну матрицю, перейдемо до безпосереднього вирішення задачі оптимізації.

Провівши оптимізацію, маємо наступні результати:

- Цільова оптимізаційна функція $W = 3,544$;
- Очікуваний ризик $\sigma_{\text{портфелю}} = 0,852\%$;
- Очікуваний дохід $I = 0,240\%$.

У таблиці наведемо ваги сформованого портфелю:

Таблиця 3.13 – Ваги портфелю сформованого на основі прогнозованих DNN значеннях

Символ компанії	W
MSFT	0,010
AMZN	0,010
GOOG	0,010
AAPL	0,028
V	0,355
NVDA	0,587

Джерело: складено автором на основі власних розрахунків

Розподіл ваг у портфелі, сформованому на основі прогнозованих значень моделі DNN, показує значну концентрацію на NVDA та V, яким присвоєно ваги 0,587 та 0,355 відповідно. Це свідчить про те, що саме ці компанії забезпечують основний внесок у дохідність портфеля за прийняттого рівня ризику. Ваги для інших компаній значно менші: AAPL має вагу 0,028, тоді як MSFT, AMZN та GOOG отримали ваги по 0,010 кожна. Такий розподіл вказує на те, що модель виявила найбільший потенціал у NVDA та V, тоді як інші компанії використовуються лише для диверсифікації ризиків.

Моделі DNN – не є основним та найкращим інструментом для роботи із часовими рядами. Значно краще себе показують моделі RNN, архітектура яких дозволяє їм краще запам'ятовувати часові структури та залежності. Наступним етапом, побудуємо RNN модель, яка буде складатися з простих шарів SimpleRNN.

Для початку роботи, видозмінимо функцію для створення набору даних. У DNN вхідні тренувальні дані мали розмірність:

- (32, 20) – для незалежних змінних;
- (32,) – для залежної змінної.

Для RNN та всіх рекурентних нейронних мереж вид набору даних змінюється.

Тепер ми маємо:

- (32, 20, 1) – для незалежних змінних;
- (32, 1) – для залежної змінної.

RNN обробляють дані послідовно, враховуючи залежності між кроками в часі.

Тому потрібен тривимірний вхідний формат, який описує:

- Кількість послідовностей в одному батчі;
- Довжину кожної послідовності
- Кількість ознак для кожного тимчасового кроку.

Створимо багатошарову нейронну мережу RNN. Її архітектура має вигляд (рис. 3.11)

Model: "sequential_16"

Layer (type)	Output Shape	Param #
simple_rnn_12 (SimpleRNN)	(None, 20, 40)	1,680
simple_rnn_13 (SimpleRNN)	(None, 40)	3,240
dense_46 (Dense)	(None, 1)	41

Total params: 4,961 (19.38 KB)
 Trainable params: 4,961 (19.38 KB)
 Non-trainable params: 0 (0.00 B)

Рисунок 3.11 – Структура RNN моделі

Джерело: складено автором на основі власних розрахунків

Маючи структуру моделі, проведемо попереднє навчання та оцінимо оптимальне значення швидкості навчання (рис. 3.12)

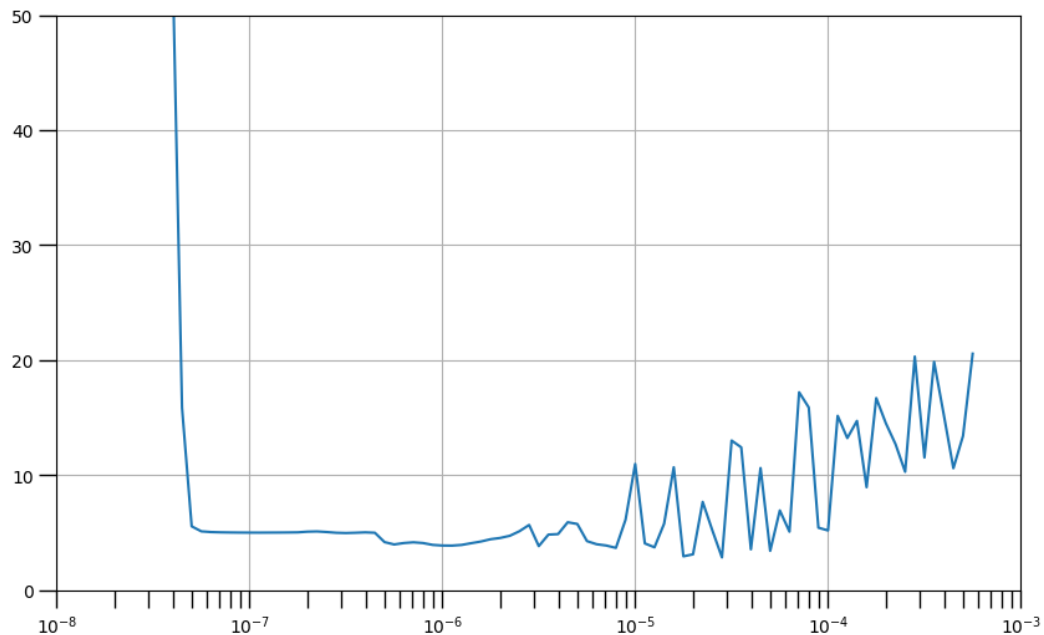


Рисунок 3.12 – Графік зміни швидкості навчання

Джерело: складено автором на основі власних розрахунків

Як бачимо з графіку, найбільш оптимальною швидкістю навчання є $5 \cdot 10^{-6}$, оскільки така швидкість показує найбільш стабільні значення.

Визначивши оптимальне значення, навчимо модель на 100 епохах та спрогнозуємо дані. Прогнозовані дані у порівнянні із реальними даними проаналізуємо графічно (рис. 3.13)

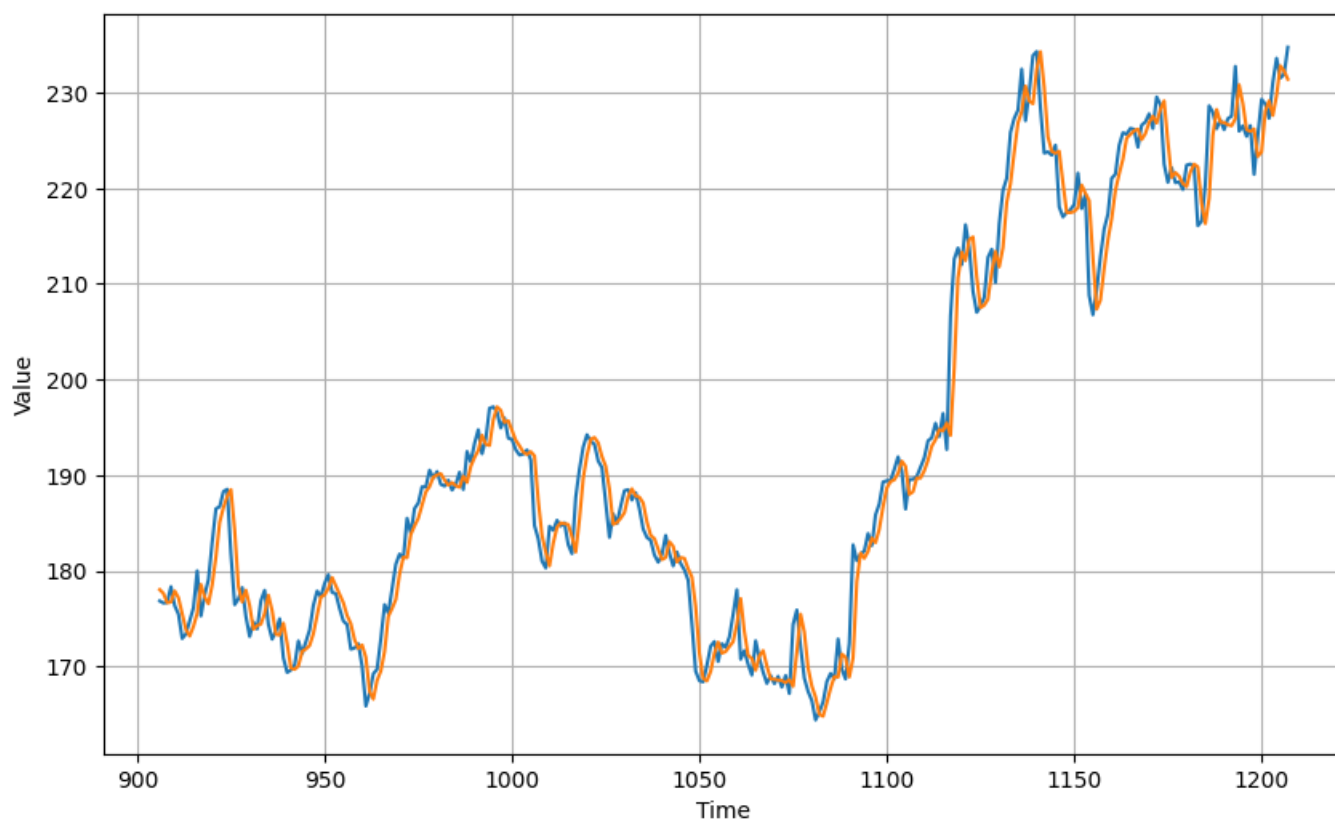


Рисунок 3.13 – Порівняльний аналіз RNN прогнозування

Джерело: складено автором на основі власних розрахунків

Загальна картина показує, що модель ефективно захоплює динаміку реальних даних, відтворюючи як короткострокові коливання, так і довгострокові тренди.

У періодах, де відбуваються різкі зміни значень, наприклад, під час стрімких зростань або спадів, модель демонструє високу точність, майже збігаючись із фактичними значеннями. Це свідчить про те, що RNN добре адаптується до нелінійних змін і складних залежностей у даних. Однак у деяких випадках можна помітити незначне запізнення прогнозу, особливо у фазах різких зламів тренду, що є

типовою поведінкою для рекурентних нейронних мереж, коли модель намагається врахувати історичні залежності.

Модель RNN демонструє точне передбачення в періодах стабільних трендів, що підтверджується практично ідентичними лініями на графіку. Це говорить про здатність моделі враховувати попередню інформацію для побудови якісних прогнозів.

Оскільки, ми маємо оптимально побудовану модель та оптимально розраховану швидкість навчання, побудуємо один великий код, який буде працювати як DNN, проте із RNN моделлю, набором даних її виду та із її оптимальною швидкістю навчання. Продемонструємо результати навчання моделі графічно порівнявши прогнозовані значення кожного часового ряду із фактичними даними (рис. 3.14)

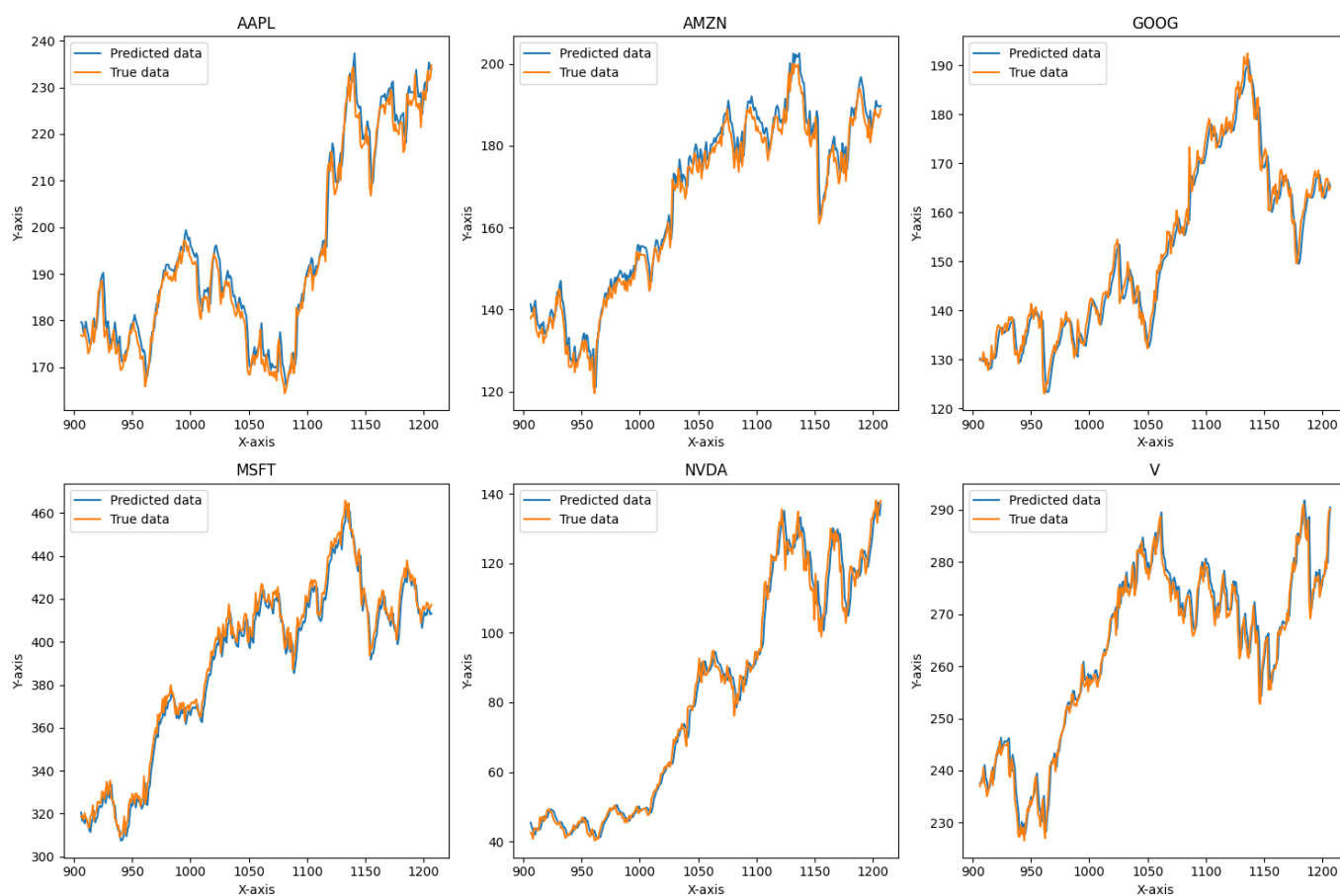


Рисунок 3.14 – Порівняльний аналіз RNN прогнозування для всіх обраних цін акцій

Джерело: складено автором на основі власних розрахунків

Загалом результати прогнозування демонструють високу точність, що має важливі економічні наслідки, особливо в контексті прийняття інвестиційних рішень.

На всіх графіках модель досить точно відтворює тренди реальних даних. Вона добре справляється з короткостроковими коливаннями та довгостроковими трендами, що робить її корисною для прогнозування змін на ринку акцій. Наприклад, для компаній, таких як AAPL і NVDA, модель точно слідує за різкими стрибками та спаданнями, що вказує на здатність адаптуватися до динаміки швидких змін у ринковій ситуації.

З економічної точки зору така точність прогнозування дає інвесторам змогу ефективно оцінювати майбутні ризики та доходи. Наприклад, для компанії V, модель відображає точну відповідність між прогнозованими та реальними значеннями, що дозволяє використовувати цей прогноз для мінімізації ризиків та оптимізації портфеля. Для NVDA спостерігається хороша відповідність тренду, особливо у фазах зростання, що може свідчити про перспективність акцій компанії та високий потенціал доходів для інвесторів.

Точність прогнозування акцій GOOG і AMZN є важливою для оцінки стабільності їхньої динаміки. Модель показує, що ці акції мають виражені циклічні коливання, які враховані в прогнозах. Це може допомогти підприємству передбачати періоди можливих корекцій ринку або пікових значень.

Однак у деяких випадках спостерігаються невеликі відхилення між реальними та прогнозованими значеннями, особливо у фазах різкої зміни трендів. Для компаній MSFT та AAPL, хоча загальна відповідність є високою, деякі локальні піки залишаються недооціненими. Це може вказувати на те, що модель менш точна в екстремальних умовах ринку, де вплив зовнішніх факторів (економічних новин, ринкових шоків) може бути сильнішим.

Економічний аналіз демонструє, що така модель прогнозування може використовуватися для стратегічного управління активами, формування портфеля та прийняття обґрунтованих рішень щодо інвестування. Висока точність прогнозів

зменшує невизначеність і допомагає ефективніше розподіляти ресурси, орієнтуючись на компанії, які мають високий потенціал доходності з прийнятним рівнем ризику.

Проаналізуємо ключові метрики RNN прогнозування (табл. 3.14)

Таблиця 3.14 – Ключові метрики прогнозування RNN

	AAPL	AMZN	GOOG	MSFT	NVDA	V
MSE	11,156	12,123	8,692	30,419	12,958	7,619
MAE	2,537	2,682	2,242	4,490	2,553	1,975
MAPE	1,319	1,641	1,478	1,150	2,978	0,754
R2	0,974	0,974	0,972	0,982	0,988	0,972

Джерело: складено автором на основі власних розрахунків

Найвищий R^2 має прогноз для акцій NVIDIA (NVDA) — 0,988, що свідчить про найкраще відображення фактичної динаміки цін цієї акції моделлю. Дещо нижчий, але все ще високий R^2 спостерігається для Microsoft (MSFT) — 0,982, що також свідчить про точність прогнозу. Для Apple (AAPL) і Amazon (AMZN) коефіцієнт R^2 становить по 0,974, що вказує на задовільну, хоча й трохи менш точну відповідність, ніж у попередніх випадках. Google (GOOG) і Visa (V) мають найнижчий R^2 серед усіх — 0,972, хоча різниця між ними та іншими акціями є незначною.

Крім R^2 , розглянуто й інші метрики, які допомагають глибше оцінити якість прогнозу. Середньоквадратична помилка (MSE) найменша для акцій Visa (V) — 7,619, що свідчить про найменше середнє відхилення передбачених цін від фактичних. Середня абсолютна помилка (MAE) теж найнижча для Visa (1,975), що підтверджує точність прогнозу цієї акції. Натомість найбільші значення MSE і MAE спостерігаються для Microsoft (MSFT) — 30,419 і 4,490 відповідно, що вказує на дещо гіршу передбачуваність цієї акції за цими метриками.

Показник середньої абсолютної відносної помилки (MAPE) свідчить про найточніший прогноз для Visa (0,754), тоді як для NVIDIA (2,978) він є найвищим, що може вказувати на чутливість моделі до великих відхилень у даних для цієї акції.

Маючи прогнозовані значення, перейдемо до моделювання оптимального портфелю акцій. Побудуємо прогностичну коваріаційну матрицю (табл. 3.15)

Таблиця 3.15 – Матриця ризиків на основі прогнозованих значень RNN

	AAPL	AMZN	GOOG	MSFT	NVDA	V
AAPL	0,000140	0,000070	0,000052	0,000062	0,000057	0,000023
AMZN	0,000070	0,000275	0,000067	0,000114	0,000079	0,000042
GOOG	0,000052	0,000067	0,000106	0,000052	0,000068	0,000018
MSFT	0,000062	0,000114	0,000052	0,000121	0,000059	0,000038
NVDA	0,000057	0,000079	0,000068	0,000059	0,000468	0,000003
V	0,000023	0,000042	0,000018	0,000038	0,000003	0,000081

Джерело: складено автором на основі власних розрахунків

Побудувавши прогнозовану матрицю ризиків, проведемо оптимізацію цільової функції та розрахуємо ключові показники. Провівши оптимізацію, маємо наступні результати:

- Цільова оптимізаційна функція $W = 5,393$;
- Очікуваний ризик $\sigma_{\text{портфелю}} = 1,079\%$;
- Очікуваний дохід $I = 0,200\%$.

У таблиці наведемо ваги сформованого портфелю:

Таблиця 3.16 – Ваги портфелю сформованого на основі RNN значеннях

Символ компанії	W
AMZN	0,010
MSFT	0,010
GOOG	0,023
AAPL	0,101
V	0,421
NVDA	0,435

Джерело: складено автором на основі власних розрахунків

Цільова оптимізаційна функція W , що дорівнює 5,393, відображає ефективність портфелю за обраним критерієм, враховуючи співвідношення доходності та ризику. Портфель характеризується низьким рівнем ризику ($\sigma = 1,079\%$), що свідчить про високу стійкість до коливань ринку. Очікуваний дохід портфелю становить 0,200%, що є помірним показником і може відповідати стратегії збереження капіталу за умов низького ризику.

Ваги активів, наведені в табл. 3.16, демонструють сильну концентрацію портфелю на двох компаніях — NVIDIA (43,5%) і Visa (42,1%). Такий розподіл

свідчить про те, що саме ці активи були визначені як найперспективніші з точки зору очікуваної доходності та стабільності. Інші компанії отримали значно меншу частку в портфелі, зокрема Amazon і Microsoft — лише по 1%, а Google і Apple — 2,3% та 10,1% відповідно. Це може бути обумовлено нижчими прогнозованими показниками доходності або підвищеним ризиком для цих активів у контексті оптимізаційної функції.

Такий підхід до формування портфелю може мати певні інвестиційні наслідки. Висока концентрація на двох компаніях збільшує залежність портфелю від їхньої ринкової динаміки, що може бути ризикованим у разі негативних змін для цих активів. Однак обидві компанії, обрані для основної ваги, ймовірно, мають високі показники надійності та потенціал зростання, що компенсує цей ризик.

В цілому, портфель орієнтований на стабільність і низький ризик, що робить його привабливим для інвесторів з помірною схильністю до ризику або для стратегій збереження капіталу. Однак потенціал зростання доходності може бути обмеженим через високу концентрацію активів, що залишає відкритим питання щодо його ефективності за умов зростаючого ринку.

Наступним етапом перейдемо до розробки LSTM-RNN моделі для прогнозування акцій. Це буде міксована модель, яка буде обробляти дані за допомогою LSTM [49] шарів та RNN шарів, щоб отримати найкращі властивості кожної з архітектур.

Такий підхід є складнішим, ніж застосування лише DNN або RNN шарів. У цьому випадку, потрібно індивідуально підібрати швидкість навчання для кожної моделі. Перед визначенням моделі очищуються існуючі сесії Keras, щоб запобігти конфліктам із попередніми обчисленнями. Нейронна мережа визначається за допомогою Keras Sequential API. Архітектура моделі складається з вхідного шару, який відповідає зазначеному розміру вікна і одного розміру ознаки, що вказує, що кожен зразок входу матиме певну кількість часових кроків з однією ознакою.

Модель включає шар LSTM (Long Short-Term Memory) з 40 одиницями та функцією активації ReLU, налаштований на повернення послідовностей. Шар LSTM добре підходить для захоплення часових залежностей у послідовних даних, що робить його відповідним для аналізу часових рядів. Після шару LSTM розташований шар SimpleRNN, також з 40 одиницями і функцією активації ReLU. Цей шар додатково обробляє послідовні дані, можливо, виявляючи додаткові шаблони чи тенденції. Щільний шар з 20 одиницями та функцією активації ReLU, який зв'язує всі попередні дані. Завершальний шар — це щільний (повністю підключений) шар з однією одиницею, який видає прогнозоване значення для наступного часової позначки. Щоб підготувати дані до навчання, використовується функція `lstm_windowed_dataset`, яка створює набір даних, придатний для моделі LSTM. Встановлюється планувальник швидкості навчання, який динамічно змінює швидкість навчання під час тренування. Планувальник експоненціально збільшує швидкість навчання протягом епох, починаючи з дуже малого значення ($1e-8$) і збільшуючи її на коефіцієнт, що залежить від номера епохи. Це експоненціальне збільшення дозволяє моделі досліджувати широкий діапазон швидкостей навчання, допомагаючи визначити швидкість, яка забезпечує найбільш ефективне навчання. Оптимізатор ініціалізується за допомогою стохастичного градієнтного спуску (SGD) з моментом 0,9. моментум допомагає прискорити оптимізатор у відповідному напрямку та зменшити коливання, що потенційно призводить до швидшої збіжності. Модель компілюється з функцією втрат Huber, яка менш чутлива до викидів, ніж середньоквадратична похибка, і поєднує переваги середньої абсолютної похибки та середньоквадратичної похибки.

Навчання проводиться протягом 100 епох, протягом яких планувальник швидкості навчання регулює швидкість навчання відповідно до визначеного графіка. Записується історія навчання, включно зі значеннями втрат на кожній епосі. Після навчання обчислюється масив швидкостей навчання, які використовувалися під час епох. Цей масив відповідає швидкостям навчання, застосованим на кожній епосі, відповідно до раніше визначеного експоненціального збільшення.

Генерується графік для візуалізації зв'язку між швидкістю навчання і втратою. Вісь x графіка представляє швидкість навчання в логарифмічному масштабі, що дозволяє ефективно відобразити широкий діапазон швидкостей. Вісь y показує значення втрат, зафіксовані під час навчання. До графіка додаються лінії сітки для підвищення читабельності, а параметри міток коригуються для зручності. Межі осей встановлюються, щоб зосередити увагу на діапазоні, що цікавить, як для швидкостей навчання, так і для значень втрат.

Аналізуючи цей графік, можна визначити швидкість навчання, при якій втрата починає значно зменшуватися, не роблячи модель нестабільною. Ця оптимальна швидкість навчання є критичною для ефективного навчання моделі, оскільки вона врівноважує швидкість збіжності зі стабільністю процесу навчання.

Проаналізуємо швидкості навчання для кожної моделі (рис. 3.15):

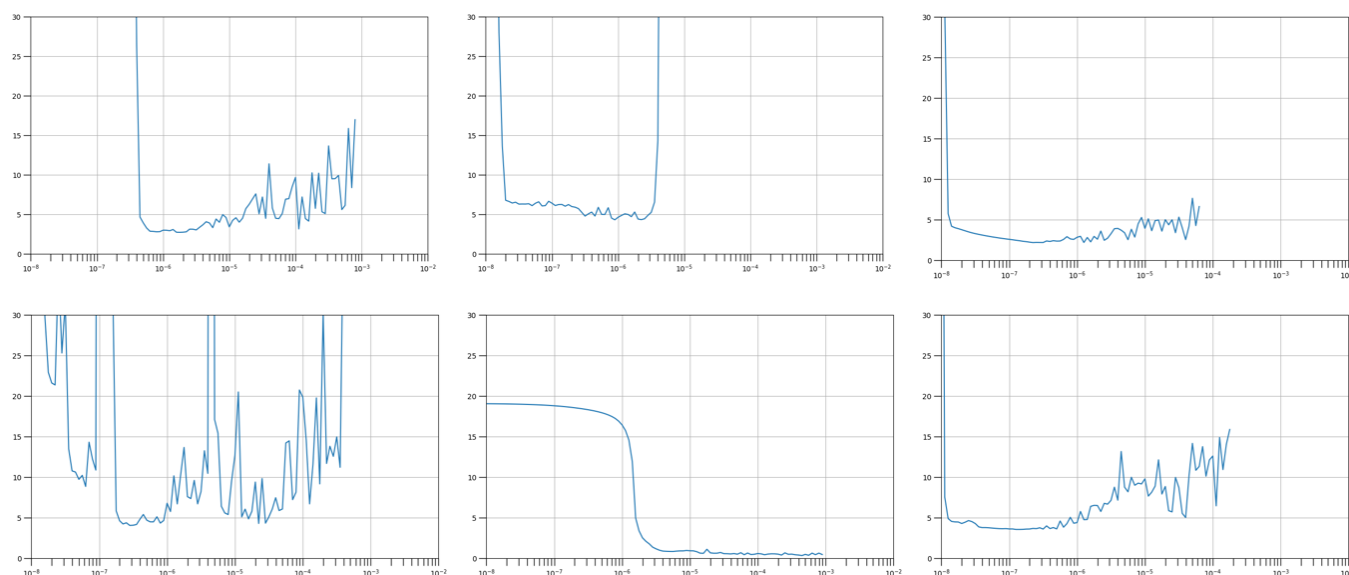


Рисунок 3.15 – Аналіз швидкості навчання моделей

Джерело: складено автором на основі власних розрахунків

Як бачимо, для кожної моделі швидкість навчання є різною. Це зумовлено тим, що моделі LSTM є більш складними та нестабільними, особливо, якщо активувати модель активатором ReLU.

На першому графіку втрати спочатку коливаються на високих значеннях, потім різко знижуються, а згодом знову зростають у міру збільшення швидкості навчання. Оптимальна швидкість навчання на рівні $1e-6$, де втрати починають різко зменшуватися, перш ніж стати нестабільними. На другому графіку спостерігається схожий характер. Спочатку втрати залишаються на високому рівні, є відносно стабільними, а потім різко падають. Після певного порогу втрати починають зростати, що свідчить про нестабільність на високих швидкостях навчання. Оптимальна швидкість навчання дорівнює $2e-7$, де втрати вперше стабільно зменшуються. Третій графік демонструє значні коливання втрат на менших швидкостях навчання, що вказує на труднощі зі збіжністю моделі. Проте є діапазон, де втрати короткочасно стабілізуються перед різким зростанням. Оптимальна швидкість навчання приблизно становить $2e-7$. Четвертий графік показує значну нестабільність втрат у більшій частині діапазону швидкостей навчання. Незважаючи на коливання, є короткий проміжок, де втрати зменшуються стабільно перед тим, як знову стати нестабільними. У цьому діапазоні знаходиться оптимальна швидкість навчання для цієї моделі - $2e-7$. На п'ятому графіку втрати залишаються стабільно високими для малих швидкостей навчання. Потім вони різко падають у певній точці, стабілізуються на короткому діапазоні швидкостей, а потім знову зростають. Оптимальна швидкість навчання становить $1e-5$. Шостий графік демонструє початкові коливання на малих швидкостях навчання. Потім спостерігається стрімке зниження втрат у міру зростання швидкості, після чого втрати різко зростають і стають нестабільними на вищих значеннях швидкості. Оптимальна швидкість навчання знаходиться на рівні $1e-7$, де втрати є низькими та стабільними одразу після зниження.

Маючи оптимальні швидкості навчання опишемо загальну архітектуру моделі. Вона має наступний вигляд (рис. 3.16):

Model: "sequential"

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 20, 40)	6,720
simple_rnn (SimpleRNN)	(None, 40)	3,240
dense (Dense)	(None, 20)	820
dense_1 (Dense)	(None, 1)	21

Total params: 10,801 (42.19 KB)

Trainable params: 10,801 (42.19 KB)

Non-trainable params: 0 (0.00 B)

Рисунок 3.16 – Структура комбінованої LSTM-RNN моделі

Джерело: складено автором на основі власних розрахунків

Після навчання, перейдемо до оцінки кожної моделі. Проведемо графічний аналіз прогнозованих даних (рис. 3.17)

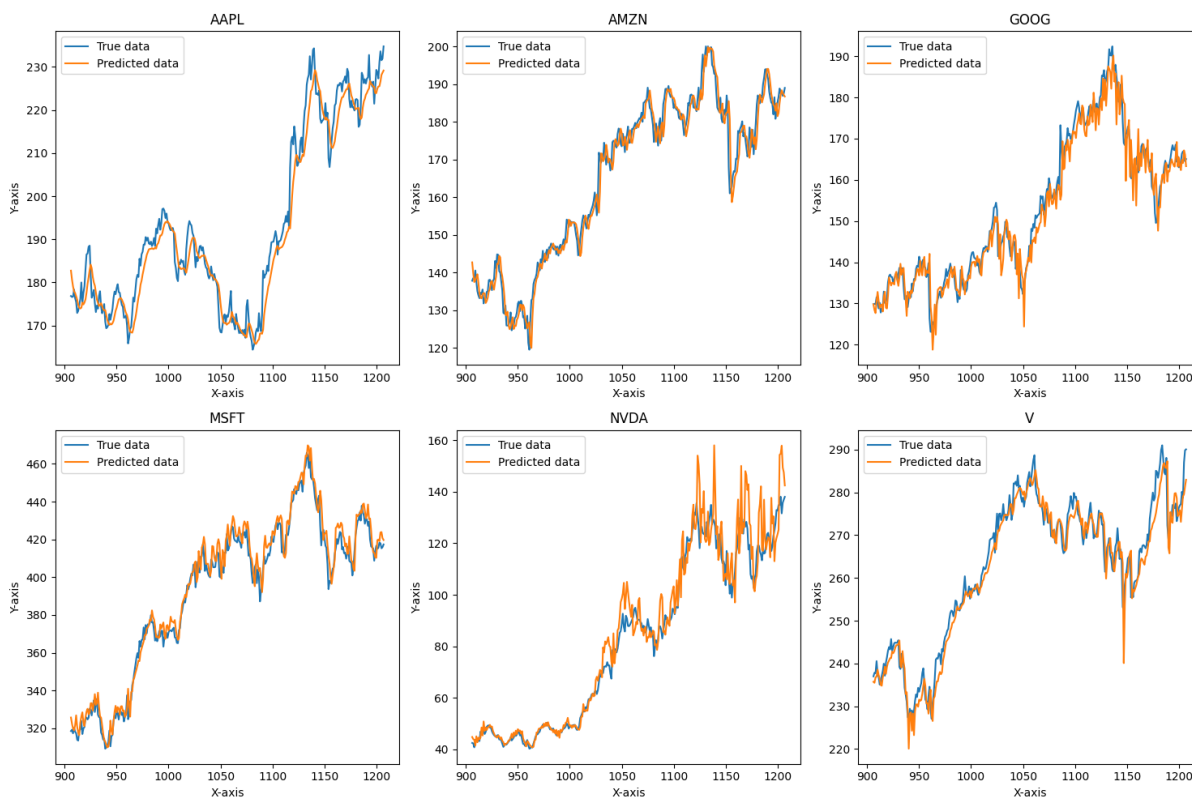


Рисунок 3.17 – Порівняльний аналіз прогнозування LSTM-RNN моделі із фактичними значеннями

Джерело: складено автором на основі власних розрахунків

Для більшості акцій прогнозовані дані добре співпадають із фактичними, що свідчить про те, що модель ефективно захоплює загальні тенденції та шаблони часових рядів. Узгодженість між двома кривими демонструє здатність моделі передбачати загальний напрямок і поведінку вартості акцій. Однак точність прогнозів відрізняється для кожного графіку. У випадку з AAPL і AMZN прогнозовані дані особливо добре співпадають із фактичними, що вказує на високий рівень точності. Модель успішно відображає як зростання, так і зниження цін акцій, що свідчить про те, що вона засвоїла основні динаміки цих часових рядів. Для GOOG прогнозовані значення також здебільшого відповідають фактичним, але подекуди трапляються відхилення, коли прогнозовані значення трохи відстають або перевищують фактичні. Ці відхилення свідчать про те, що, хоча модель загалом працює добре, вона може стикатися з труднощами у відтворенні раптових змін у цінах акцій. MSFT демонструє схожий тренд, де прогнози в основному відповідають фактичним значенням. Проте помітні незначні розбіжності, особливо під час різких змін або переходів, коли прогнози моделі можуть бути трохи запізнілими або менш точними. На підграфіку NVDA спостерігається помітне збільшення похибок прогнозування у порівнянні з іншими акціями. Хоча прогнозовані дані відображають загальні тенденції, модель має труднощі з точним відтворенням високої волатильності та різких стрибків у фактичних даних. Це свідчить про те, що модель може відчувати складнощі з адаптацією до більш волатильних акцій або тих, що мають швидкі зміни у вартості. Для V прогнозовані дані знову добре відповідають фактичним, що свідчить про сильну здатність моделі до прогнозування. Проте, як і на інших підграфіках, незначні розбіжності трапляються під час швидких змін ціни, де прогнози моделі трохи відхиляються від фактичних значень.

Більш точно можна оцінити якість кожного прогнозованого часового ряду, замірявши ключові метрики. Ключові метрики для оцінки є такими ж як і для попередніх моделей. Метрики та оцінки мають наступний вигляд (табл. 3.17):

Таблиця 3.17 – Ключові метрики прогнозування LSTM-RNN моделей

	AAPL	AMZN	GOOG	MSFT	NVDA	V
MSE	23,086	25,191	14,642	37,084	41,933	20,173
MAE	3,855	3,883	2,714	4,907	4,017	3,597
MAPE	1,968	2,380	1,780	1,253	4,348	1,372
R2	0,946	0,946	0,952	0,978	0,960	0,927

Джерело: складено автором на основі власних розрахунків

Для моделі MSFT спостерігається найвищий показник R^2 (0,978), що свідчить про найкращу відповідність моделі фактичним даним. Незважаючи на відносно високе значення MSE (37,084) і MAE (4,907), низький MAPE (1,253%) вказує на те, що модель забезпечує точне прогнозування з точки зору відносних помилок, навіть якщо абсолютні похибки є більшими.

GOOG демонструє другий за якістю результат із R^2 на рівні 0,952. Модель має найнижчі значення MSE (14,642), MAE (2,714), і MAPE (1,780%), що вказує на високу точність прогнозування. Незважаючи на трохи нижчий R^2 порівняно з MSFT, ця модель може вважатися більш стабільною через низькі абсолютні та відносні помилки.

NVDA займає третє місце за якістю прогнозування з R^2 на рівні 0,960. Хоча значення MSE (41,933) і MAE (4,017) є вищими, ніж у моделей MSFT та GOOG, модель демонструє задовільний рівень точності. Однак високий MAPE (4,348%) свідчить про те, що модель може бути менш ефективною для прогнозування у випадках значної волатильності акцій.

Моделі AAPL і AMZN мають однакові значення R^2 (0,946), що свідчить про схожу якість прогнозування. Модель AAPL характеризується трохи нижчим MSE (23,086) і MAE (3,855), у той час як AMZN має вищий MSE (25,191) і MAE (3,883). З огляду на ці показники, модель AAPL трохи перевершує AMZN за точністю. Обидві моделі демонструють помірно низький MAPE (1,968% для AAPL та 2,380% для AMZN), що свідчить про їх ефективність у прогнозуванні з точки зору відносних похибок.

Модель V має найнижче значення R^2 (0,927), що свідчить про найгіршу якість прогнозування серед усіх моделей. Незважаючи на це, модель демонструє низьке значення MSE (20,173), MAE (3,597), і MAPE (1,372%), що означає відносно високий рівень точності у прогнозах. Проте її R^2 нижче, ніж у інших моделей, що вказує на слабшу відповідність фактичним даним.

Маючи прогнози, перейдемо до процесу оптимізації та побудови оптимального портфелю акцій. Підхід такий ж як і в попередніх моделях – побудова прогнозованої коваріаційної матриці, пошук очікуваних норм та ризиків на основі прогнозованих значень. Маємо наступну матрицю прогнозованих ризиків (табл. 3.18)

Таблиця 3.18 – Прогнозована матриця ризиків LSTM-RNN моделями

	AAPL	AMZN	GOOG	MSFT	NVDA	V
AAPL	0,000034	0,000023	0,000031	0,000025	0,000051	0,000009
AMZN	0,000023	0,000214	0,000044	0,000073	-0,000037	0,000007
GOOG	0,000031	0,000044	0,000770	0,000103	0,000177	0,000004
MSFT	0,000025	0,000073	0,000103	0,000171	0,000163	0,000025
NVDA	0,000051	-0,000037	0,000177	0,000163	0,004195	0,000004
V	0,000009	0,000007	0,000004	0,000025	0,000004	0,000133

Джерело: складено автором на основі власних розрахунків

Отримавши прогнозовану матрицю даних, можемо перейти до формування оптимального інвестиційного портфеля, використовуючи раніше описані функції та алгоритми оптимізації.

На цьому етапі враховуються ключові показники прогнозів, такі як очікувані доходності, ризики та кореляції між активами, що дозволяє забезпечити максимально ефективне розподілення капіталу. Оптимізувавши, маємо наступні результати:

- Цільова оптимізаційна функція $W = 6,970$;
- Очікуваний ризик $\sigma_{\text{портфелю}} = 0,579\%$;
- Очікуваний дохід $I = 0,083\%$.

У таблиці наведемо ваги сформованого портфелю:

Таблиця 3.19 – Ваги портфелю сформованого на основі LSTM-RNN значення

Символ компанії	W
MSFT	0,010
GOOG	0,010
NVDA	0,028
AMZN	0,082
V	0,126
AAPL	0,744

Джерело: складено автором на основі власних розрахунків

Оптимізована цільова функція досягла значення $W = 6,970$, що відображає ефективність обраного розподілу активів у контексті заданих обмежень та умов моделі. Очікуваний ризик портфеля (σ портфелю = 0,579%) вказує на високу стабільність портфеля з низькою волатильністю. Це свідчить про орієнтацію на захищеність інвестицій від суттєвих коливань на ринку. У свою чергу, очікуваний дохід ($I = 0,083\%$) є помірним і відповідає стратегії консервативного підходу до формування портфеля.

Аналіз ваг активів у портфелі демонструє значну концентрацію на акціях компанії Apple (AAPL) з вагою $w = 0,744$, що становить 74,4% від загального обсягу інвестицій. Такий розподіл вказує на високу залежність портфеля від динаміки цього активу, що може бути як перевагою, так і потенційним ризиком, враховуючи специфіку ринку Apple. Інші активи, зокрема акції компаній Visa (V) і Amazon (AMZN), отримали помірні ваги у 12,6% і 8,2% відповідно, що дозволяє забезпечити певний рівень диверсифікації. Менші ваги для акцій Nvidia (NVDA), Google (GOOG) і Microsoft (MSFT), які разом становлять лише 4,8% портфеля, свідчать про обмежений внесок цих активів у загальну структуру портфеля. Це може свідчити про те, що їхня волатильність або низький рівень прогнозованої доходності не дозволили їм отримати вищу пріоритетність в оптимізації. Загалом, сформований портфель відповідає стратегії мінімізації ризиків із помірним рівнем доходності. Висока концентрація в одному активі (AAPL) може створювати певні ризики, особливо в умовах можливих несприятливих змін на ринку.

3.3 Компаративний аналіз та рекомендації для формування оптимального інвестиційного портфелю

Прогностична та оптимізаційна аналітика проведена успішно. Порівняємо ключові метрики кожної моделі в порівнянні один з одним та зробимо висновки, яка з моделей показує найкращу прогностичну здатність (не враховуємо модель GARCH, оскільки цей процес прогнозував волатильність, а не ціни акцій). Результати покажемо у порівняльній таблиці:

Таблиця 3.20 – Ваги портфелю сформованого на основі реальних даних

Модель	Метрика	Історичні ціни акцій					
		AAPL	AMZN	GOOG	MSFT	NVDA	V
ARIMA	MSE	7,548	8,054	6,586	22,775	8,996	7,321
	MAE	2,015	2,099	1,812	3,707	2,050	2,028
	MAPE	1,041	1,289	1,196	0,951	2,357	0,773
	R ²	0,982	0,983	0,979	0,986	0,991	0,973
DNN	MSE	15,166	15,230	9,431	25,770	23,868	9,338
	MAE	3,087	3,038	2,367	4,024	3,547	2,341
	MAPE	1,583	1,862	1,557	1,030	4,240	0,892
	R ²	0,965	0,967	0,969	0,984	0,977	0,966
RNN	MSE	11,156	12,123	8,692	30,419	12,958	7,619
	MAE	2,537	2,682	2,242	4,490	2,553	1,975
	MAPE	1,319	1,641	1,478	1,150	2,978	0,754
	R ²	0,974	0,974	0,972	0,982	0,988	0,972
LSTM-RNN	MSE	23,086	25,191	14,642	37,084	41,933	20,173
	MAE	3,855	3,883	2,714	4,907	4,017	3,597
	MAPE	1,968	2,380	1,780	1,253	4,348	1,372
	R ²	0,946	0,946	0,952	0,978	0,960	0,927

Джерело: складено автором на основі власних розрахунків

Виходячи з цієї таблиці, проведемо графічний аналіз та проаналізуємо ключові параметри. Порівняємо MSE метрику в межах різних моделей (рис. 3.18)

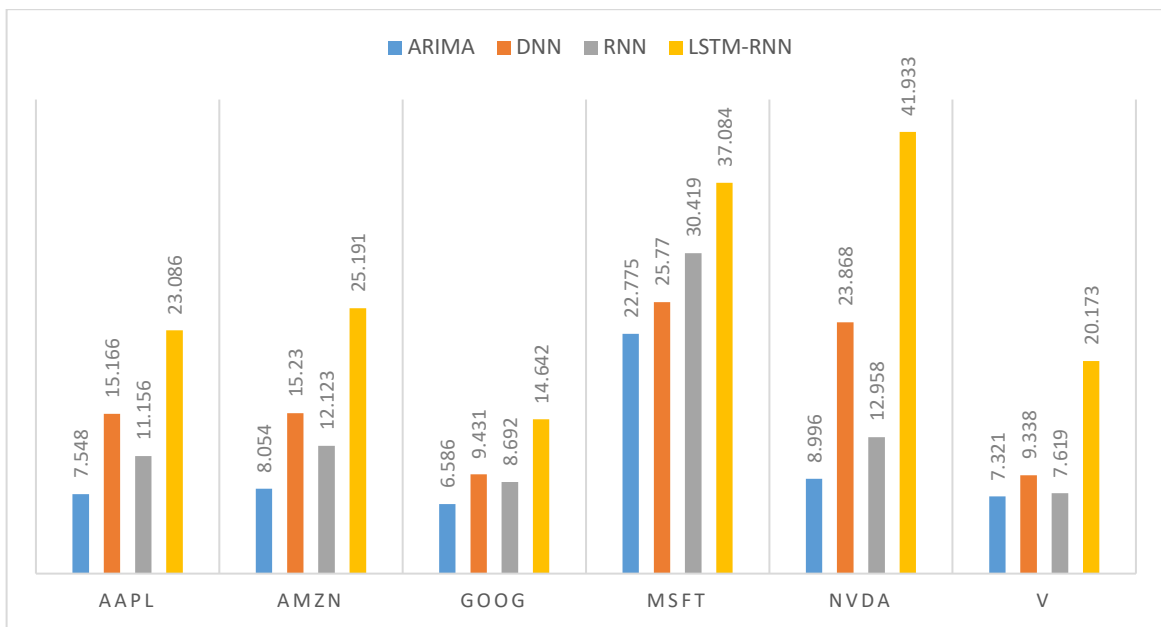


Рисунок 3.18 – Порівняльний аналіз MSE метрики

Джерело: складено автором на основі власних розрахунків

ARIMA демонструє найкращу продуктивність для всіх розглянутих компаній, оскільки її MSE є найнижчим у порівнянні з іншими моделями. Наприклад, для компаній Apple (AAPL) та Microsoft (MSFT) її результати значно випереджають навіть найближчі за точністю моделі. Це свідчить про те, що ARIMA є особливо надійним інструментом для короткострокового прогнозування цін акцій у стабільних умовах ринку.

У випадку з DNN та RNN, ці моделі показують середній рівень точності. Наприклад, для акцій Amazon (AMZN) та Google (GOOG) їхні результати знаходяться близько до ARIMA, що вказує на те, що ці моделі можуть бути ефективними в умовах більш складних залежностей у даних. Однак, у випадках таких компаній, як Microsoft (MSFT) та NVIDIA (NVDA), їхні помилки значно перевищують результати ARIMA, що знижує їхню практичну корисність у цих випадках.

LSTM-RNN демонструє найвищий рівень MSE майже для всіх компаній, особливо для NVIDIA та Microsoft. Це може свідчити про недостатню оптимізацію моделі або її слабку адаптацію до обсягу доступних даних. Хоча LSTM-RNN потенційно краще підходить для роботи з часовими рядами через свою здатність

вловлювати довгострокові залежності, у цьому аналізі вона виявилася найменш ефективною.

Оцінимо ще один ключовий показник, який показує варіацію залежної змінної від варіації незалежної змінної. Для більш кращої оцінки R^2 застосуємо теплову карту (рис. 3.19)

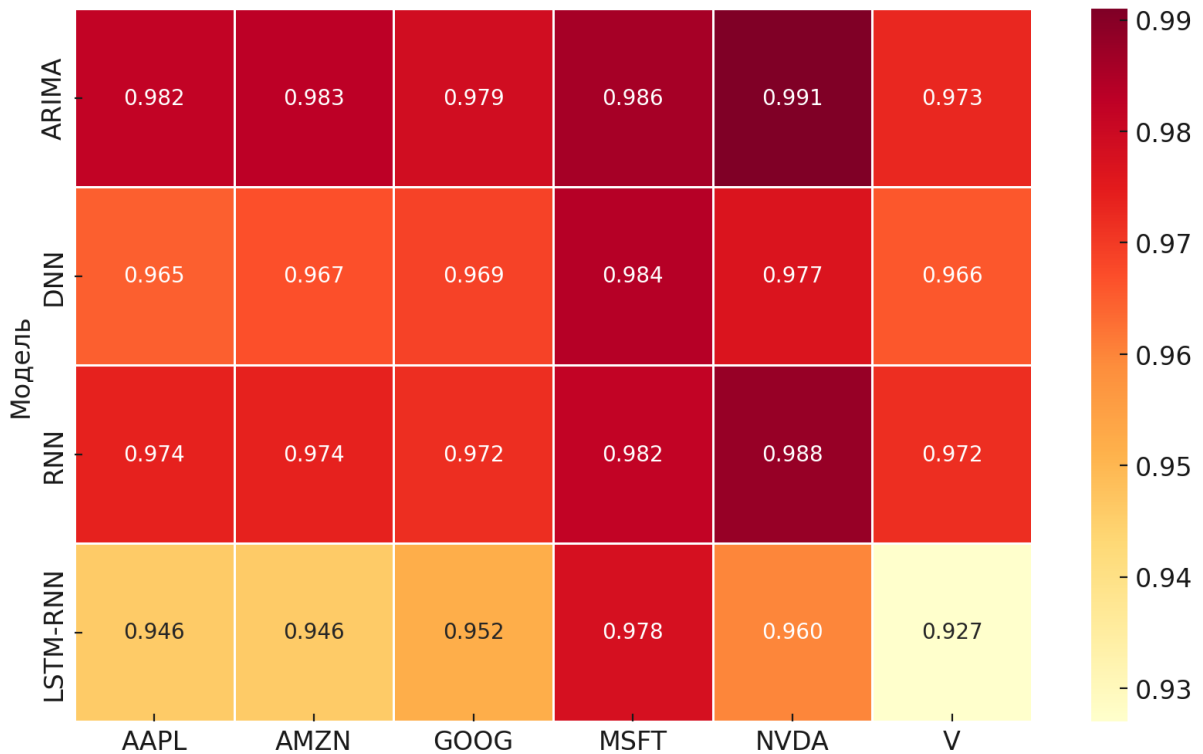


Рисунок 3.19 – Теплова карта для R^2 метрики

Джерело: складено автором на основі власних розрахунків

Модель ARIMA показала найвищу ефективність серед усіх, із показниками, що перевищують 0,97 для всіх компаній, особливо для NVDA (0,991), що свідчить про її сильну здатність уловлювати тенденції в часі. Це означає, що ARIMA може бути більш ефективною у стабільних умовах або для даних із чіткими трендами. DNN також демонструє досить високі результати, особливо для Microsoft (0,984), що вказує на потенціал нейронних мереж у складних багатофакторних завданнях. Однак її показники трохи нижчі порівняно з ARIMA, що може свідчити про недостатню адаптацію до специфіки даних або необхідність покращення навчання. RNN показала

сильні результати, які наближаються до ARIMA, особливо для Microsoft (0,982) і NVIDIA (0,988), що вказує на її перевагу в обробці послідовних даних і можливість уловлювати залежності в часі. Проте відносно невисокі результати для інших компаній свідчать про ймовірні обмеження в загальній узагальненості цієї моделі. LSTM-RNN показала найнижчі результати серед представлених моделей, з особливо помітним падінням для компанії V (0,927). Хоча ця модель теоретично добре підходить для обробки довготривалих залежностей у даних, її слабші показники можуть вказувати на проблеми в оптимізації чи адаптації до специфіки фінансових часових рядів.

Якщо повертатися в загальному до таблиці з метриками (табл. 3.20), то варто зазначити, що модель ARIMA демонструє найкращі результати у більшості метрик, включаючи середню абсолютну похибку (MAE), середню абсолютну процентну похибку (MAPE) та коефіцієнт детермінації. Це свідчить про її високу точність і надійність для прогнозування показників акцій. Завдяки цим перевагам вона є найкращим вибором серед розглянутих моделей. Модель RNN займає друге місце, оскільки показує високу точність, яка наближається до ARIMA. Вона демонструє конкурентоспроможні результати за метриками MAE, MAPE та коефіцієнтом детермінації, що свідчить про її надійність та ефективність у прогнозуванні. На третьому місці розташовується модель DNN. Хоча вона показує задовільні результати, її точність поступається як ARIMA, так і RNN. Метрики MAE і MAPE є вищими, що свідчить про більшу похибку у прогнозах порівняно з лідерами. Модель LSTM-RNN займає останнє, четверте місце. Вона демонструє найгірші результати за всіма основними метриками, зокрема, має найвищі значення середньоквадратичної похибки (MSE) та найменшу точність прогнозів. Це свідчить про те, що вона є найменш ефективною у цьому аналізі.

Економічно, ці результати свідчать, що ARIMA залишається надійним інструментом для прогнозування фінансових даних, особливо для великих стабільних компаній. DNN і RNN також можуть бути використані, але їх ефективність залежить

від характеру даних та налаштування моделей. LSTM-RNN у своєму поточному стані може не бути оптимальним вибором для задач прогнозування в цьому контексті.

Перед тим як перейти до порівняльного аналізу отриманих результатів, побудуємо оптимальний портфель на основі реальних даних. Це дозволить порівняти прогнозовані результати із фактичними та визначитись, яка з моделей є найбільш оптимальною.

Провівши оптимізаційну аналітику, маємо такі результати на реальних даних:

- Цільова оптимізаційна функція $W = 7,9452$;
- Очікуваний ризик $\sigma_{\text{портфелю}} = 1,429\%$;
- Очікуваний дохід $I = 0,191\%$.

Ваги сформованого портфелю:

Таблиця 3.21 – Ваги портфелю сформованого на основі реальних даних

Символ компанії	W
AMZN	0,010
GOOG	0,010
MSFT	0,010
AAPL	0,148
NVDA	0,370
V	0,452

Джерело: складено автором на основі власних розрахунків

Економічний аналіз сформованого портфеля, побудованого на основі реальних даних, свідчить про збалансованість вибраних активів, що дозволяє досягти оптимального співвідношення між ризиком та доходністю. Цільова оптимізаційна функція, яка досягла значення 7,945, демонструє ефективність математичної моделі оптимізації для цього портфеля. Очікуваний ризик портфеля становить 1,429%, що вказує на його низький рівень волатильності. Це є позитивним сигналом для інвесторів, які прагнуть мінімізувати ризики в умовах нестабільних ринків. Очікуваний дохід портфеля, рівний 0,191%, показує помірний рівень прибутковості, що є характерним для портфелів із консервативною стратегією. Така доходність може задовольнити інвесторів, орієнтованих на стабільність капіталу та мінімізацію втрат.

Структура портфеля базується на ваговій оптимізації активів, яка забезпечує збалансоване розподілення інвестицій між компаніями різних секторів. Найбільшу частку в портфелі займають акції компанії Visa (45,2%) та NVIDIA (37,0%), що свідчить про високу довіру до їхнього потенціалу генерувати стабільні прибутки. Значна вага NVIDIA може бути обумовлена її лідерськими позиціями на ринку високотехнологічних продуктів, зокрема в галузі штучного інтелекту та графічних процесорів. Акції Apple також займають вагому частку (14,8%), що вказує на їхню стабільність і здатність генерувати дохід завдяки їхньому широкому споживчому ринку. Акції таких технологічних гігантів, як Amazon, Google і Microsoft, мають найменші ваги в портфелі (по 1,0% кожна). Це може бути зумовлено високим ступенем диверсифікації портфеля та прагненням уникати надмірної залежності від одного сегмента ринку.

Проаналізуємо портфелі побудовані за допомогою класичних та Deep Learning моделей. Для цього, побудуємо таблицю із загальними показниками кожного портфелю (табл. 3.22):

Таблиця 3.22 – Порівняльний аналіз ключових параметрів портфелів

	Фактичний портфель	ARIMA	GARCH	DNN	RNN	LSTM-RNN
Цільова функція	7,945	7,699	10,627	3,544	5,393	6,97
Ризик	1,43%	1,32%	1,60%	0,85%	1,08%	0,58%
Дохід	0,19%	0,17%	0,15%	0,24%	0,20%	0,08%

Джерело: складено автором на основі власних розрахунків

На основі порівняльного аналізу ключових параметрів портфелів можна зробити висновок, що моделі ARIMA та RNN показали найкращі результати, близькі до фактичного портфеля. Модель ARIMA забезпечує високу точність прогнозування з мінімальним рівнем ризику (1,32%) та дохідністю (0,17%), що дуже близькі до параметрів фактичного портфеля. Це свідчить про її надійність і відповідність до консервативної стратегії інвестування. Модель RNN, у свою чергу, демонструє вищий рівень дохідності (0,20%), хоча її ризик (1,08%) залишається дещо більшим, ніж у ARIMA, але меншим за фактичний портфель. Це робить RNN перспективним вибором

для інвесторів, орієнтованих на вищу прибутковість. Моделі GARCH, DNN та LSTM-RNN, хоч і показують певні переваги, поступаються ARIMA та RNN за ключовими метриками. GARCH має найвищий рівень ризику (1,60%), що робить її менш привабливою для інвесторів із низькою толерантністю до волатильності. DNN забезпечує найнижчий рівень ризику (0,85%), але її дохідність (0,24%) значно перевищує фактичні показники, що може свідчити про її оптимізацію для більш агресивного сценарію. LSTM-RNN, хоча і демонструє найнижчий ризик (0,58%), забезпечує мінімальний дохід (0,08%), що знижує її ефективність у порівнянні з іншими моделями.

Це також можна побачити із графіку нижче (рис. 3.20)

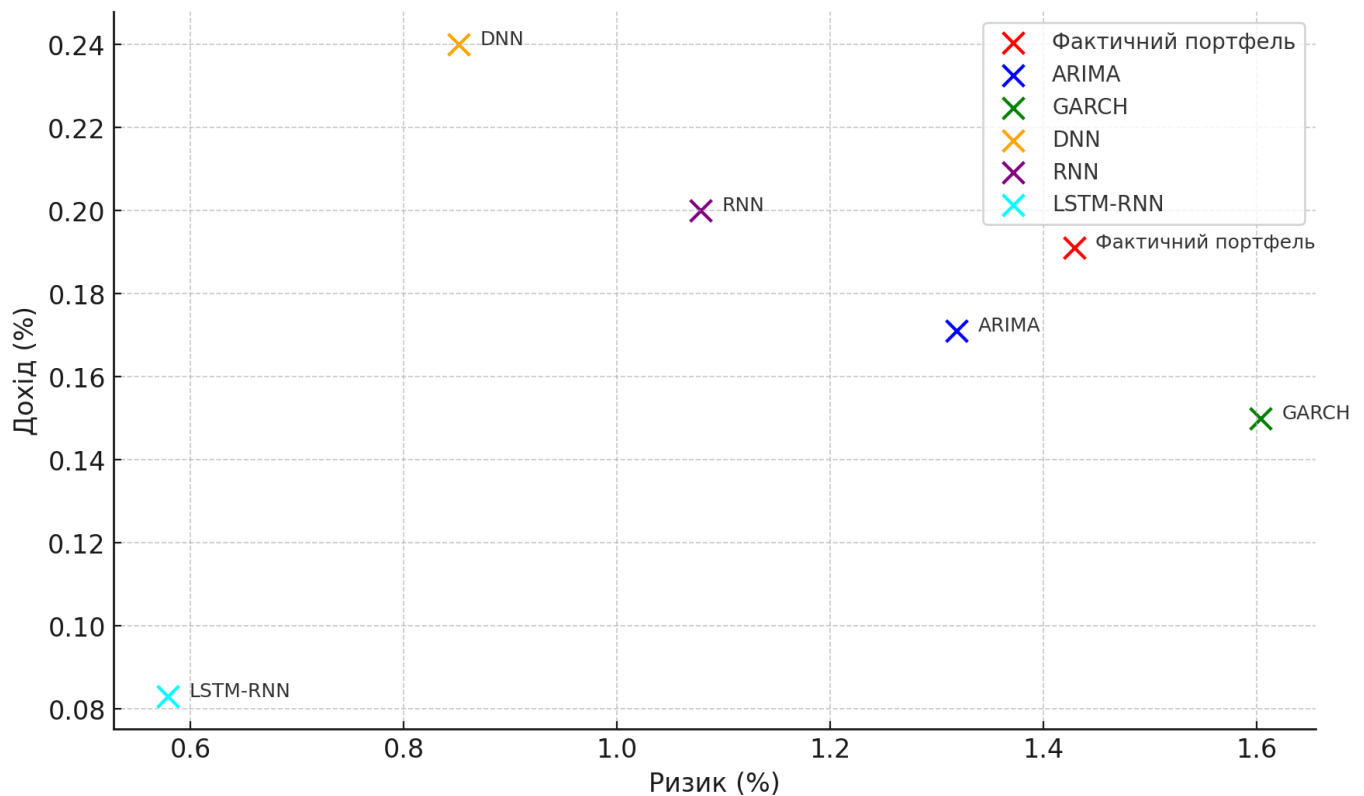


Рисунок 3.20 – Порівняльний аналіз портфельних метрик

Джерело: складено автором на основі власних розрахунків

Як бачимо з діаграми, ARIMA та RNN є найближчими результатами до Фактичних значень.

Подальший аналіз проведемо по структурі кожного портфелю у порівнянні із фактичними результатами (табл. 3.23)

Таблиця 3.23 – Порівняльний аналіз вагових коефіцієнтів портфелів

Ваги	Фактичний портфель	ARIMA	GARCH	DNN	RNN	LSTM-RNN
MSFT	0,01	0,01	0,415	0,01	0,01	0,01
AMZN	0,01	0,01	0,01	0,01	0,01	0,082
GOOG	0,01	0,01	0,01	0,01	0,023	0,01
AAPL	0,148	0,134	0,181	0,028	0,101	0,744
NVDA	0,37	0,331	0,374	0,587	0,435	0,028
V	0,452	0,505	0,01	0,355	0,421	0,126

Джерело: складено автором на основі власних розрахунків

Для наочного відображення, побудуємо секторні діаграми та проаналізуємо кожен портфель окремо (рис. 3.21). Використаємо пакет Matplotlib [50].

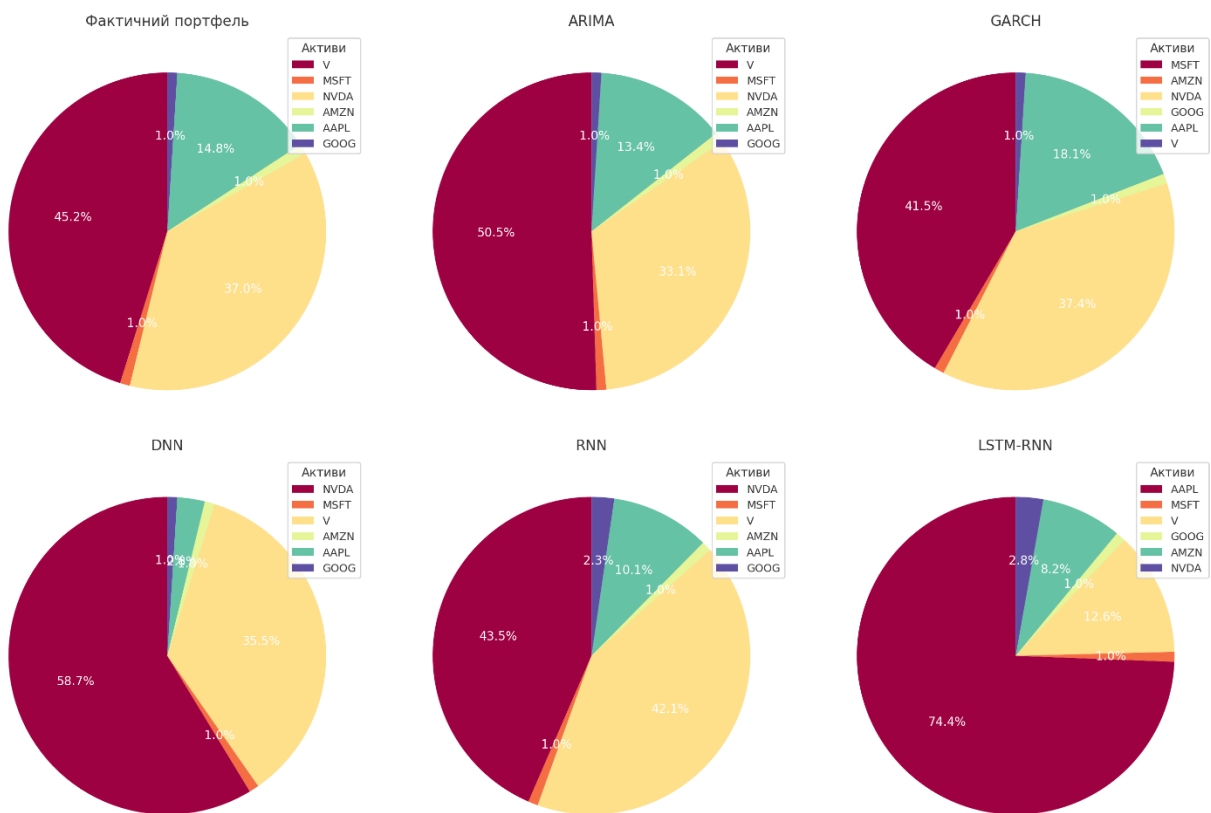


Рисунок 3.21 – Порівняльний аналіз вагових коефіцієнтів портфелів

Джерело: складено автором на основі власних розрахунків

Фактичний портфель демонструє чітку структуру, у якій найбільші ваги припадають на Visa (45,2%) і NVIDIA (37%). Помірну частку займає Apple (14,8%), а решта активів, такі як Microsoft (MSFT), Amazon (AMZN) і Google (GOOG), мають рівні мінімальні ваги по 1%. Такий розподіл свідчить про диверсифікований і зважений підхід до інвестування, який орієнтується на стабільність ключових активів і мінімізацію ризиків.

Серед представлених моделей портфельів найближчими до фактичного є ARIMA та RNN. Портфель, побудований за моделлю ARIMA, відображає структуру, яка найбільш схожа на реальний портфель. У ньому акції Visa мають вагу 50,5%, що дещо перевищує їх частку у фактичному портфелі, але підтверджує акцент на цьому стабільному активі. NVIDIA, із вагою 33,1%, також займає вагоме місце, подібно до реального розподілу. Apple у портфелі ARIMA представлена вагою 13,4%, що дуже близько до фактичних 14,8%. Водночас Microsoft, Amazon і Google мають однакові мінімальні ваги по 1%, що ідентично фактичному портфелю. Така структура портфеля ARIMA демонструє вдалий баланс між диверсифікацією та концентрацією на ключових активах, роблячи його найближчим до реального.

Портфель RNN також має значну подібність до фактичного портфеля. Вага Visa у цьому портфелі становить 42,1%, що дуже близько до фактичної. NVIDIA, із вагою 43,5%, також зберігає важливу роль у портфелі, навіть перевищуючи її частку у реальному портфелі. Водночас Apple представлена з вагою 10,1%, що трохи нижче за фактичну, але все ще відображає значущість цього активу. Щодо мінімальних активів, таких як Microsoft, Amazon і Google, RNN, подібно до ARIMA, підтримує низькі ваги (1% для Microsoft і Amazon, 2,3% для Google), що узгоджується з реальним розподілом. Ці характеристики роблять модель RNN ще одним потужним кандидатом для моделювання реального портфеля.

Інші моделі демонструють значно більше відхилення від фактичного портфеля. Наприклад, модель GARCH надає надмірно високу вагу Microsoft (41,5%) і суттєво зменшує вагу Visa до 1%, що значно змінює загальну структуру портфеля. Модель

DNN робить акцент на NVIDIA (58,7%) і Visa (35,5%), залишаючи мінімальні ваги для інших активів, що вказує на агресивний підхід із низькою диверсифікацією. Найбільші відхилення демонструє модель LSTM-RNN, у якій вагу Apple збільшено до 74,4%, а інші активи представлені майже незначними частками, що суттєво знижує збалансованість портфеля.

Щоб оцінити кожен портфель неупереджено та, що важливо, з економічної точки зору, змодельуємо ситуацію, в якій, маючи початкову суму інвестицій у розмірі 100000 грн, оцінимо економічну доцільність кожного портфеля шляхом розрахунку грошових еквівалентів ключових метрик та прогнозування очікуваних втрат. Для початку розрахуємо очікуваний дохід та очікуваний ризик кожного побудованого портфелю у фактичних цінах:

Таблиця 3.24 – Ефективність портфелів у грошовому вимірі

Портфель	Очікуваний дохід (грн)	Очікувані втрати (грн)
Фактичний портфель	190	1430
ARIMA	170	1320
GARCH	150	1600
DNN	240	850
RNN	200	1080
LSTM-RNN	80	580

Джерело: складено автором на основі власних розрахунків

Порівняння результатів показує, що найближчим до фактичного портфеля за показником очікуваного доходу є портфель RNN, який має дохід у 200 грн, що перевищує фактичний показник лише на 10 грн. Очікувані втрати RNN становлять 1080 грн, що також ближче до фактичних втрат (1430 грн), ніж у більшості інших моделей.

ARIMA демонструє дохід 170 грн, що на 20 грн нижче фактичного значення, і втрати 1320 грн, які є ближчими до фактичних, ніж у інших моделей, окрім RNN. Це робить ARIMA також відносно близьким до фактичного портфеля. GARCH має дохід 150 грн, що значно нижче фактичного (190 грн), і втрати 1600 грн, які перевищують фактичні на 170 грн, що робить цю модель менш схожою на фактичний портфель.

DNN, хоч і демонструє найвищий дохід у 240 грн, має суттєве відхилення від фактичного за показником очікуваних втрат – 850 грн, що значно нижче, ніж 1430 грн фактичного портфеля. Це свідчить про те, що модель краще мінімізує втрати, але її характеристики помітно відрізняються від фактичного портфеля. LSTM-RNN має найнижчі показники доходу (80 грн) і втрат (580 грн), що свідчить про суттєве відхилення від фактичного портфеля. Хоча модель демонструє мінімальні ризики, її результати далекі від фактичних значень. Таким чином, найближчими до фактичного портфеля є RNN і ARIMA, тоді як DNN пропонує іншу динаміку з акцентом на мінімізацію втрат. GARCH і LSTM-RNN суттєво відрізняються як за дохідністю, так і за ризиком.

Незважаючи на результати, слід звернути увагу на те, що модель ARIMA, незважаючи на гарні результати, навчалася не класичним шляхом. Вона навчалася шляхом постійного донавчання в процесі прогнозування. В той ж час, модель RNN навчилася всього один раз і змогла здійснити точне прогнозування без додаткового донавчання, що потребує ресурси та час. Зважаючи на це та на отримані результати, побудуємо таблицю ранжування моделей (де 1 – найкраща модель, 5 - найгірша), щоб оцінити отримані результати та обрати одну єдину модель (табл. 3.24)

Таблиця 3.25 – Ранжування моделей

Модель	Ранжування
RNN	1
ARIMA	2
DNN	3
LSTM-RNN	4
GARCH	5

Джерело: складено автором на основі власних розрахунків

Таким чином, маємо два найкращі кандидати: RNN та ARIMA.

Висновки до третього розділу

Було проведено детальний аналіз результатів прогнозування та економічного моделювання на основі використання різних підходів до обробки часових рядів і побудови оптимального інвестиційного портфеля. Аналіз розпочався з навчання класичних моделей ARIMA та GARCH, а також сучасних методів глибинного навчання, таких як DNN, RNN і LSTM-RNN. Основну увагу було приділено порівнянню їхньої точності прогнозування та здатності моделювати структуру фінансового портфеля.

Результати прогнозування чітко продемонстрували переваги моделі ARIMA, яка показала найнижчі значення середньоквадратичної похибки (MSE), середньої абсолютної похибки (MAE) та середньої абсолютної процентної похибки (MAPE) для всіх розглянутих компаній. Високі показники коефіцієнта детермінації (R^2), що перевищували 0,97 для більшості випадків, свідчать про її здатність точно відображати тенденції часових рядів. Однак варто зазначити, що ARIMA є ефективною переважно в стабільних ринкових умовах, що може обмежувати її застосування в умовах високої волатильності або непередбачуваних змін.

Моделі на основі нейронних мереж, зокрема RNN, також продемонстрували конкурентоспроможні результати. RNN, попри трохи вищі значення похибок у порівнянні з ARIMA, досягла значної точності за метрикою R^2 (до 0,988 для NVIDIA), що свідчить про її спроможність адаптуватися до складних залежностей у часових рядах. Важливо, що модель RNN забезпечила високу точність прогнозів без необхідності додаткового донавчання, що робить її більш ефективною з погляду витрат часу та обчислювальних ресурсів. Модель DNN, хоча і показала середні результати в загальному рейтингу, виявилася корисною для певних специфічних умов. Її точність поступалася ARIMA та RNN, особливо для акцій із високою волатильністю, таких як NVIDIA, проте результати є вагомими.

Модель LSTM-RNN, хоча і не є лідером за точністю, демонструє дещо кращі результати, ніж GARCH. Її здатність уловлювати довготривалі залежності в даних теоретично робить її перспективною для фінансових часових рядів. Проте в цьому аналізі результати вказують на потребу в додатковій оптимізації та адаптації моделі для конкретних задач.

Окрему увагу було приділено порівнянню вагових структур портфель, сформованих за допомогою прогнозів різних моделей. Найбільш схожими до фактичного портфеля виявилися структури, побудовані за результатами моделей ARIMA та RNN. Портфель ARIMA демонстрував добре збалансовану структуру з акцентом на стабільних активах, таких як Visa і NVIDIA, що забезпечило оптимальне співвідношення між ризиком і прибутковістю. RNN, зі свого боку, продемонструвала конкурентоспроможні результати, зберігаючи високу частку ключових активів і при цьому трохи перевищуючи показники прибутковості фактичного портфеля.

На основі проведеного ранжування моделей було визначено, що RNN та ARIMA є найкращими кандидатами для задач прогнозування цін акцій і побудови оптимального портфеля. RNN отримала найвищу оцінку завдяки здатності до ефективного навчання та високої точності без потреби в постійному донавчанні. ARIMA, у свою чергу, є ідеальним вибором для завдань, що потребують стабільності та мінімізації ризиків.

ВИСНОВКИ

Узагальнюючи результати проведеного дослідження, слід зазначити, що магістерська робота спрямована на поєднання класичних теоретичних засад управління інвестиційним портфелем із сучасними методами аналітики, прогнозування та математичного моделювання, що дозволяє створювати ефективні інвестиційні рішення в умовах цифрової трансформації економіки. Глобалізація, динамічні зміни на фінансових ринках і розвиток цифрових технологій зумовлюють необхідність адаптації традиційних підходів до формування інвестиційного портфеля, що стало основним фокусом цього дослідження.

Ключовою теоретичною основою роботи є класична портфельна теорія Гаррі Марковіца, що стала базисом для оцінки дохідності та ризику активів, а також їхньої диверсифікації. Ця теорія надає інструменти для побудови збалансованого портфеля, який враховує кореляційні взаємозв'язки між активами. Проте в умовах сучасних ринкових викликів, таких як висока волатильність, зростання обсягів даних і непередбачувані зовнішні фактори, класичні підходи потребують вдосконалення. Саме тому дослідження зосереджувалося на інтеграції цих фундаментальних принципів із методами сучасної аналітики.

Особливе місце у дослідженні посідає прогнозування часових рядів із використанням методів машинного навчання та глибокого навчання. Статистичні моделі, такі як ARIMA та GARCH, зарекомендували себе як ефективні інструменти для аналізу ринкових трендів, оцінки волатильності та прогнозування дохідності. ARIMA демонструє високу точність у стабільних ринкових умовах, забезпечуючи точне передбачення тенденцій і сезонних коливань. Модель GARCH, у свою чергу, дозволяє оцінювати волатильність, враховуючи динамічні зміни ризиків, що важливо для адаптації до нестабільних умов.

Методи глибокого навчання, такі як DNN, RNN, та LSTM, відкривають нові горизонти для аналізу складних фінансових даних. Їхня здатність обробляти великі

масиви даних і виявляти нелінійні залежності робить ці моделі надзвичайно перспективними для прогнозування ринкових тенденцій. Наприклад, RNN та LSTM забезпечують ефективне моделювання часових залежностей, що є критично важливим для фінансових часових рядів із високою волатильністю. Особливе значення має використання змішаних моделей, які поєднують сильні сторони різних архітектур, забезпечуючи більш гнучке й точне прогнозування.

У роботі проведено порівняльний аналіз результатів, отриманих із застосуванням різних моделей, що дозволило визначити їхні переваги та недоліки. ARIMA показала найвищу точність прогнозів за стабільних умов ринку, тоді як моделі на основі глибокого навчання, зокрема RNN та LSTM, виявилися більш адаптивними до змінних умов. Це свідчить про необхідність комплексного підходу, що включає використання як класичних статистичних методів, так і сучасних технологій.

Економічне моделювання структури інвестиційного портфеля, проведене на основі прогнозів, підтвердило, що інтеграція різних підходів дозволяє досягти оптимального балансу між дохідністю та ризиком. Зокрема, портфелі, сформовані за прогнозами ARIMA та RNN, продемонстрували високу стабільність і конкурентоспроможність. Вони враховують як стабільні, так і ризиковані активи, що дозволяє ефективно адаптуватися до змін ринкових умов.

Наукова новизна роботи полягає в розробці методологій, які поєднують теоретичні засади інвестиційного аналізу з сучасними інструментами прогнозування й математичного моделювання. Зокрема, вдосконалено підхід до побудови ARIMA моделі через покрокове донавчання, а також проведено порівняльний аналіз прогнозування цін акцій та коваріаційної матриці. Результати дослідження підтверджують, що використання технологій глибокого навчання значно розширює можливості класичних підходів до управління портфелем, дозволяючи створювати більш ефективні інвестиційні стратегії.

Практичне значення отриманих результатів полягає в можливості їхнього застосування для формування інвестиційних рішень, що базуються на точних

прогнозах і враховують динаміку сучасних ринків. Інструменти прогнозування, розроблені в межах роботи, можуть бути використані фінансовими аналітиками та інвесторами для зниження ризиків і підвищення прибутковості портфелів. Крім того, запропоновані підходи можуть бути інтегровані в автоматизовані системи управління портфелем, що є актуальним у контексті розвитку фінансових технологій (FinTech).

Загалом, проведене дослідження демонструє, що інтеграція теоретичних засад із сучасними інструментами Data Science дозволяє створювати ефективні рішення для управління інвестиційним портфелем. У сучасних умовах, коли ринки стають дедалі більш динамічними та непередбачуваними, такі підходи є необхідною умовою для успішної адаптації інвестиційних стратегій. Це підкреслює важливість подальшого розвитку досліджень у цій сфері та їхнього практичного застосування.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Мажара Г. А., Крикун Є. О. Моделювання оптимального інвестиційного портфеля орієнтованого на мінімізацію ризику. *Modern Economics*. 2023. № 38(2023). С. 69-75. URL: [https://doi.org/10.31521/modecon.V38\(2023\)-11](https://doi.org/10.31521/modecon.V38(2023)-11)
2. Крикун Є. О. Оптимізаційне моделювання портфелю цінних паперів в умовах соціально-поведінкових змін: Дипломна робота першого (бакалаврського) рівня вищої освіти / за кер. Лазаренко І. С. Київ: КПІ ім. Ігоря Сікорського, кафедра економічної кібернетики. 2023. URL: <https://ela.kpi.ua/server/api/core/bitstreams/cba8e8aa-d692-4c11-af18-19c429f4a50b/content>
3. 9.4. Використання сучасної портфельної теорії у портфельному менеджменті. Модель оцінювання капітальних активів (CAPM). Необхідна ставка дохідності. Арбітражна теорія. Оцінювання ефективності управління портфелем цінних паперів. Критерії ефективності. URL: <https://buklib.net/books/26654/>
4. Моделювання дохідності та ризику портфеля. URL: https://pidru4niki.com/15660721/investuvannya/modelyuvannya_dohidnosti_riziku_portfelya
5. Rate of return. URL: https://en.wikipedia.org/wiki/Rate_of_return
6. Юхименко Г.К., Лазаренко І.С., Моделювання інвестиційного фонду акцій із застосуванням стратегій керування фінансовими деривативами та хеджування. *Економічний вісник НТУУ «КПІ»*. 2022. №24. С. 110-119. URL: <http://ev.fmm.kpi.ua/article/view/274836/269981>
7. Modern portfolio theory. URL: https://en.wikipedia.org/wiki/Modern_portfolio_theory
8. Stock Forecasting with ARIMA Models: A Comprehensive Guide. URL: <https://www.cameyeam.com/post/stock-forecasting-with-arima-models-a-comprehensive-guide>

9. Autoregressive Integrated Moving Average (ARIMA) Prediction Model. URL: <https://www.investopedia.com/terms/a/autoregressive-integrated-moving-average-arima.asp>
10. What are ARIMA models? URL: <https://www.ibm.com/topics/arima-model>
11. Stock Market Forecasting using Time Series Analysis with ARIMA Model. URL: <https://www.analyticsvidhya.com/blog/2021/07/stock-market-forecasting-using-time-series-analysis-with-arima-model/>
12. What Is the GARCH Process? How It's Used in Different Forms. URL: <https://www.investopedia.com/terms/g/generalizedautoregressiveconditionalheteroskedasticity.asp>
13. Гетероскедастичність. URL: <https://studfile.net/preview/8857901/page:21/>
14. GARCH Model: Definition and Uses in Statistics. URL: <https://www.investopedia.com/terms/g/garch.asp>
15. Використання ARCH / GARCH-моделі для прогнозування волатильності. URL: <https://caseware.com.ua/arch-garch-model/>
16. Autoregressive conditional heteroskedasticity. URL: https://en.wikipedia.org/wiki/Autoregressive_conditional_heteroskedasticity
17. Portfolio optimization using the GO-GARCH model: evidence from Ukrainian Stock Exchange. URL: <https://ea21journal.world/wp-content/uploads/2022/02/ea-V160-23.pdf>
18. A Survey of Deep Learning and Foundation Models for Time Series Forecasting. URL: <https://arxiv.org/pdf/2401.13912>
19. Chapter 8 Dense neural networks. URL: <https://smltar.com/dldnn>
20. Dense Neural Networks: Understanding Their Structure and Function. URL: <https://datascientest.com/en/dense-neural-networks-understanding-their-structure-and-function>
21. Recurrent Neural Networks for Time Series Forecasting: Current Status and Future Directions. URL: <https://arxiv.org/pdf/1909.00590>

22. Recurrent Neural Networks for Time Series. URL: <https://sabankara.medium.com/recurrent-neural-networks-for-time-series-b3132a6afb6a>
23. SimpleRNN layer. URL: https://keras.io/api/layers/recurrent_layers/simple_rnn/
24. Working with RNNs. URL: https://www.tensorflow.org/guide/keras/working_with_rnns
25. LSTM for Time Series Prediction in PyTorch. URL: <https://machinelearningmastery.com/lstm-for-time-series-prediction-in-pytorch/>
26. Understanding LSTM: Architecture, Pros and Cons, and Implementation. URL: <https://medium.com/@anishnama20/understanding-lstm-architecture-pros-and-cons-and-implementation-3e0cca194094>
27. How to Develop LSTM Models for Time Series Forecasting. URL: <https://machinelearningmastery.com/how-to-develop-lstm-models-for-time-series-forecasting/>
28. Lazarenko I., Krykun Y. POTRFOLIO MANAGEMENT WITH TIME SERIES ANALYSIS METHODS. URL: <https://ev.fmm.kpi.ua/article/view/309267/300787>
29. Model Selection for ARIMA. URL: <https://www.geeksforgeeks.org/model-selection-for-arima/>
30. Tests for Stationarity in Time Series — Dickey Fuller Test & Augmented Dickey Fuller(ADF) Test. URL: <https://medium.com/@ritusantra/tests-for-stationarity-in-time-series-dickey-fuller-test-augmented-dickey-fuller-adf-test-d2e92e214360>
31. ПРОГНОЗУВАННЯ ДИНАМІКИ РОЗВИТКУ ТВАРИННИЦТВА ЗА ДОПОМОГОЮ ЧАСОВИХ РЯДІВ. URL: https://www.business-inform.net/export_pdf/business-inform-2024-1_0-pages-110_117.pdf
32. Neural Network Layers: All You Need Is Inside Comprehensive Overview. URL: <https://hackernoon.com/neural-network-layers-all-you-need-is-an-inside-comprehensive-overview>

33. Rectifier (neural networks). URL: [https://en.wikipedia.org/wiki/Rectifier_\(neural_networks\)](https://en.wikipedia.org/wiki/Rectifier_(neural_networks))
34. Activation Functions in Neural Networks [12 Types & Use Cases]. URL: <https://www.v7labs.com/blog/neural-networks-activation-functions>
35. Deep Learning Models for Time Series Forecasting: A Review. URL: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=10583885>
36. Understanding Huber Loss function: Insights from Applications. URL: <https://medium.com/@devcharlie2698619/understanding-huber-loss-function-insights-from-applications-5c1c5145d2c4>
37. 1.5. Stochastic Gradient Descent. URL: <https://scikit-learn.org/stable/modules/sgd.html>
38. Long short-term memory. URL: https://en.wikipedia.org/wiki/Long_short-term_memory?
39. CS231n Convolutional Neural Networks for Visual Recognition. URL: <https://cs231n.github.io/convolutional-networks/#conv>
40. Yahoo Finance. URL: <https://finance.yahoo.com/>
41. pandas documentation. URL: <https://pandas.pydata.org/docs/>
42. NumPy documentation. URL: <https://numpy.org/devdocs/>
43. Optimization and root finding. URL: <https://docs.scipy.org/doc/scipy/reference/optimize.html>
44. arch 7.2.0. URL: <https://pypi.org/project/arch/>
45. statsmodels 0.14.4. URL: <https://www.statsmodels.org/stable/index.html>
46. yfinance 0.2.50. URL: <https://pypi.org/project/yfinance/>
47. Keras 3 API documentation .URL: <https://keras.io/api/>
48. Dense layer. URL: https://keras.io/api/layers/core_layers/dense/
49. LSTM layer. URL: https://keras.io/api/layers/recurrent_layers/lstm/
50. Matplotlib 3.9.3 documentation URL: <https://matplotlib.org/stable/>

ДОДАТКИ

ДОДАТОК А

Лістинг програмного коду

```

!pip install yfinance
!pip install pmdarima
!pip install arch
import pandas as pd
import numpy as np
import yfinance as yf
import pmdarima as pm
import statsmodels.api as sm
import matplotlib.pyplot as plt
from arch import arch_model
tickers = ['AAPL', 'MSFT', 'GOOG', 'AMZN', 'V', 'NVDA']
stock_data = yf.download(tickers, start='2020-01-01', end='2024-10-20')['Adj Close']
stock_data.shape
x = list(stock_data.index)
fig, axs = plt.subplots(2, 3, figsize=(15, 10))
for i, ax in enumerate(axs.flat, 0):
    column = list(stock_data.columns)[i]
    ax.plot(x, stock_data.iloc[:,i], label=f'{column} Stock Prices')
    ax.legend()
    ax.set_title(f'{column}')
    ax.set_xlabel('X-axis')
    ax.set_ylabel('Y-axis')# Adjust layout to prevent overlap
plt.tight_layout()
plt.show()
stock_data.isna().sum()
import pandas as pd
import numpy as np
import statsmodels.api as sm
from statsmodels.tsa.arima.model import ARIMA
import matplotlib.pyplot as plt
from scipy import stats
import yfinance as yf# import pmdarima as pm
import tensorflow as tf
from sklearn.metrics import r2_score
from keras.models import Sequential

```

```

from keras.layers import LSTM, Dense
from tensorflow.keras.utils import plot_model
def plot_series(time, series, format="-", start=0, end=None):
    """
    Visualizes time series data    Args:
        time (array of int) - contains the time steps
        series (array of int) - contains the measurements for each time step
        format - line style when plotting the graph
        label - tag for the line
        start - first time step to plot
        end - last time step to plot
    """    # Setup dimensions of the graph figure
    plt.figure(figsize=(10, 6))    if type(series) is tuple:        for series_num in
series:
        # Plot the time series data
        plt.plot(time[start:end], series_num[start:end], format)    else:
        # Plot the time series data
        plt.plot(time[start:end], series[start:end], format)    # Label the x-axis
    plt.xlabel("Time")    # Label the y-axis
    plt.ylabel("Value")    # Overlay a grid on the graph
    plt.grid(True)    # Draw the graph on screen
    plt.show()def windowed_dataset(series, window_size, batch_size, shuffle_buffer):
    """Generates dataset windows    Args:
        series (array of float) - contains the values of the time series
        window_size (int) - the number of time steps to average
        batch_size (int) - the batch size
        shuffle_buffer(int) - buffer size to use for the shuffle method    Returns:
        dataset (TF Dataset) - TF Dataset containing time windows
    """    # Generate a TF Dataset from the series values
    dataset = tf.data.Dataset.from_tensor_slices(series)    # Window the data but only
take those with the specified size
    dataset = dataset.window(window_size + 1, shift=1, drop_remainder=True)    # Flatten
the windows by putting its elements in a single batch
    dataset = dataset.flat_map(lambda window: window.batch(window_size + 1))    # Create
tuples with features and labels
    dataset = dataset.map(lambda window: (window[:-1], window[-1]))    # Shuffle the
windows
    dataset = dataset.shuffle(shuffle_buffer)    # Create batches of windows

```

```

dataset = dataset.batch(batch_size) # Optimize the dataset for training
dataset = dataset.cache().prefetch(1) return datasetdef
rnn_windowed_dataset(series, window_size, batch_size, shuffle_buffer):
    """Generates dataset windows Args:
        series (array of float) - contains the values of the time series
        window_size (int) - the number of time steps to include in the feature
        batch_size (int) - the batch size
        shuffle_buffer(int) - buffer size to use for the shuffle method Returns:
        dataset (TF Dataset) - TF Dataset containing time windows
    """ # Add an axis for the feature dimension of RNN layers
    series = tf.expand_dims(series, axis=-1) # Generate a TF Dataset from the series
values
    dataset = tf.data.Dataset.from_tensor_slices(series) # Window the data but only
take those with the specified size
    dataset = dataset.window(window_size + 1, shift=1, drop_remainder=True) # Flatten
the windows by putting its elements in a single batch
    dataset = dataset.flat_map(lambda window: window.batch(window_size + 1)) # Create
tuples with features and labels
    dataset = dataset.map(lambda window: (window[:-1], window[-1])) # Shuffle the
windows
    dataset = dataset.shuffle(shuffle_buffer) # Create batches of windows
    dataset = dataset.batch(batch_size) # Optimize the dataset for training
    dataset = dataset.cache().prefetch(1) return dataset
def lstm_windowed_dataset(series, window_size, batch_size, shuffle_buffer):
    """Generates dataset windows Args:
        series (array of float) - contains the values of the time series
        window_size (int) - the number of time steps to include in the feature
        batch_size (int) - the batch size
        shuffle_buffer(int) - buffer size to use for the shuffle method Returns:
        dataset (TF Dataset) - TF Dataset containing time windows
    """ # Add an axis for the feature dimension of RNN layers
    series = tf.expand_dims(series, axis=-1) # Generate a TF Dataset from the series
values
    dataset = tf.data.Dataset.from_tensor_slices(series) # Window the data but only
take those with the specified size
    dataset = dataset.window(window_size + 1, shift=1, drop_remainder=True) # Flatten
the windows by putting its elements in a single batch

```

```

        dataset = dataset.flat_map(lambda window: window.batch(window_size + 1)) # Create
tuples with features and labels
        dataset = dataset.map(lambda window: (window[:-1], window[-1])) # Shuffle the
windows

        dataset = dataset.shuffle(shuffle_buffer) # Create batches of windows
        dataset = dataset.batch(batch_size) # Optimize the dataset for training
        dataset = dataset.cache().prefetch(1) return dataset
arima_predictions = {}
for column in list(stock_data.columns): series = stock_data[column]
    time = np.arange(len(series)) split_time = int(len(series)/4*3) # Get the train set
    time_train = time[:split_time]
    x_train = series[:split_time] # Get the validation set
    time_valid = time[split_time:]
    x_valid = series[split_time:] auto_model = pm.auto_arima(x_train, seasonal=False,
trace=False)
        p, d, q = auto_model.order def arima_forecast(history, model_fit): model =
ARIMA(history, order=(p, d, q))
            model_fit = model.fit() output = model_fit.forecast()
            yhat = output[0]
            return yhat # Walk-forward validation
    history = [x for x in x_train]
    model = ARIMA(history, order=(p, d, q))
    model_fit = model.fit() pred = list()
    for t in range(len(x_valid)): yhat = arima_forecast(history, model_fit)
        pred.append(yhat)
        # Add the predicted value to the training set
        obs = x_valid[t]
        history.append(obs) print('-----')
    print("MSE of {}".format(column), tf.keras.metrics.mse(x_valid, pred).numpy())
    print("MAE of {}".format(column), tf.keras.metrics.mae(x_valid, pred).numpy())
    print("MAPE of {}".format(column), tf.keras.metrics.mape(x_valid, pred).numpy())
    print("R2 Score of {}".format(column), r2_score(x_valid, pred))
arima_predictions[column] = pred
    garch_reurt_rate_data = np.log(stock_data / stock_data.shift(1))
    garch_reurt_rate_data = garch_reurt_rate_data[1:]
    returnes_garch = garch_reurt_rate_data.mean()
    risks_garch = garch_reurt_rate_data.std()
    coeffs = []

```

```

cond_vol = []
std_resids = []
models = []for column in garch_reurt_rate_data.columns:
    model = arch_model(garch_reurt_rate_data[column], p = 1, o = 0, q = 1).fit()
    coeffs.append(model.params)
    cond_vol.append(model.conditional_volatility)
    std_resids.append(model.resid / model.conditional_volatility)
    models.append(model)coeffs_df = pd.DataFrame(coeffs,
index=garch_reurt_rate_data.columns)
    cond_vol_df = pd.DataFrame(cond_vol).transpose().set_axis(garch_reurt_rate_data.columns,
axis = 'columns')
    std_resids_df = pd.DataFrame(std_resids).transpose().set_axis(garch_reurt_rate_data.columns,
axis = 'columns')CCC =
std_resids_df.transpose().dot(std_resids_df).div(len(std_resids_df))
N = len(tickers)
diag = []
D = np.zeros((N, N))for model in models:
    diag.append(model.forecast(horizon = 302).variance.values[-1][0])diag =
np.sqrt(np.array(diag))
    np.fill_diagonal(D, diag)H = np.matmul(np.matmul(D, CCC.values),
D)garch_predict_cov_matrix = pd.DataFrame(H, index=list(stock_data.columns),
columns=list(stock_data.columns))
    predictions = {}for column in list(stock_data.columns): stock_name = column
    series = stock_data[stock_name]
    time = np.arange(len(series)) split_time = int(len(series)/4*3) time_train =
time[:split_time]
    x_train = series[:split_time] time_valid = time[split_time:]
    x_valid = series[split_time:] window_size = 20
    batch_size = 32
    shuffle_buffer_size = 1000 dataset = windowed_dataset(x_train, window_size,
batch_size, shuffle_buffer_size) model_tune = tf.keras.models.Sequential([
    tf.keras.Input(shape=(window_size,)),
    tf.keras.layers.Dense(150, activation="relu"),
    tf.keras.layers.Dense(100, activation="relu"),
    tf.keras.layers.Dense(50, activation="relu"),
    tf.keras.layers.Dense(1)

```

```

    ]) optimizer = tf.keras.optimizers.SGD(learning_rate=1e-6, momentum=0.9)
model_tune.compile(loss="mse", optimizer=optimizer) history = model_tune.fit(dataset,
epochs=150, verbose=0) # Initialize a list
forecast = [] forecast_series = series[split_time - window_size:]
for time in range(len(forecast_series) - window_size):
    forecast.append(model_tune.predict(np.array(forecast_series[time:time +
window_size])[np.newaxis], verbose=0)) results = np.array(forecast).squeeze() print("MSE of
{}".format(column), tf.keras.metrics.mse(x_valid, results).numpy())
    print("MAE of {}".format(column), tf.keras.metrics.mae(x_valid, results).numpy())
    print("MAPE of {}".format(column), tf.keras.metrics.mape(x_valid, results).numpy())
    print("R2 Score of {}".format(column), r2_score(x_valid, results))
predictions[column] = results
predictions_rnn = {}for column in list(stock_data.columns): stock_name = column
series = stock_data[stock_name]
time = np.arange(len(series)) split_time = int(len(series)/4*3) time_train =
time[:split_time]
x_train = series[:split_time] time_valid = time[split_time:]
x_valid = series[split_time:] window_size = 20
batch_size = 32
shuffle_buffer_size = 1000 dataset = rnn_windowed_dataset(x_train, window_size,
batch_size, shuffle_buffer_size) model = tf.keras.models.Sequential([
    tf.keras.Input(shape=(window_size, 1)),
    tf.keras.layers.SimpleRNN(40, return_sequences=True, activation='relu'),
    tf.keras.layers.SimpleRNN(40, activation='relu'),
    tf.keras.layers.Dense(1)
]) optimizer = tf.keras.optimizers.SGD(learning_rate=1e-5, momentum=0.9)
model.compile(loss=tf.keras.losses.Huber(),
optimizer=optimizer,
metrics=["mae"]) history = model.fit(dataset, epochs=100, verbose=0) #
Initialize a list
forecast = [] forecast_series = series[split_time - window_size:]
for time in range(len(forecast_series) - window_size):
    forecast.append(model.predict(np.array(forecast_series[time:time +
window_size])[np.newaxis], verbose=0)) results = np.array(forecast).squeeze() print("MSE of
{}".format(column), tf.keras.metrics.mse(x_valid, results).numpy())
    print("MAE of {}".format(column), tf.keras.metrics.mae(x_valid, results).numpy())
    print("MAPE of {}".format(column), tf.keras.metrics.mape(x_valid, results).numpy())

```

```

        print("R2 Score of {}".format(column), r2_score(x_valid, results))
predictions_rnn[column] = results
    predictions_lstm = {}for column in list(stock_data.columns):
tf.keras.backend.clear_session() stock_name = column
    series = stock_data[stock_name]
    time = np.arange(len(series)) split_time = int(len(series)/4*3) time_train =
time[:split_time]
    x_train = series[:split_time] time_valid = time[split_time:]
    x_valid = series[split_time:] window_size = 20
    batch_size = 32
    shuffle_buffer_size = 1000 dataset = lstm_windowed_dataset(x_train, window_size,
batch_size, shuffle_buffer_size)
    # Build the model
model = tf.keras.models.Sequential([
    tf.keras.Input(shape=(window_size, 1)),
    tf.keras.layers.SimpleRNN(40, return_sequences=True, activation='relu'),
    tf.keras.layers.LSTM(40, activation='relu'),
    tf.keras.layers.Dense(20, activation='relu'),
    tf.keras.layers.Dense(1)
]) # Set the learning rate
learning_rate = 3e-4 # Set the optimizer
optimizer = tf.keras.optimizers.Adam(learning_rate=learning_rate) # Set the training
parameters
model.compile(loss=tf.keras.losses.Huber(),
              optimizer=optimizer,
              metrics=["mae"]) # Train the model
history = model.fit(dataset, epochs=100) # Initialize a list
forecast = [] forecast_series = series[split_time - window_size:]
for time in range(len(forecast_series) - window_size):
    forecast.append(model.predict(np.array(forecast_series[time:time
+
window_size])[np.newaxis], verbose=0)) results = np.array(forecast).squeeze() del model
print("MSE of {}".format(column), tf.keras.metrics.mse(x_valid, results).numpy())
    print("MAE of {}".format(column), tf.keras.metrics.mae(x_valid, results).numpy())
    print("MAPE of {}".format(column), tf.keras.metrics.mape(x_valid, results).numpy())
    print("R2 Score of {}".format(column), r2_score(x_valid, results))
predictions_lstm[column] = results
    from scipy.optimize import minimize
dnn_predictions = pd.DataFrame(predictions)

```

```

returnt_rate_data = np.log(dnn_predictions / dnn_predictions.shift(1))
returnt_rate_data = returnt_rate_data[1:]
returnes = returnt_rate_data.mean()
risks = returnt_rate_data.std()
predict_cov_matrix = returnt_rate_data.cov()
W = np.ones(len(returnes)) / len(returnes)
def finale_function(W):
    risk = np.sqrt((W.transpose()).dot(predict_cov_matrix).dot(W))
    income = np.sum(np.array(returnes) * np.array(W))
    return (0.5*risk) / (0.5*income)
def constraint1(W):
    sum = 1
    sum = sum - np.sum(W)
    return sum
def constraint2(W):
    return np.sum(np.array(returnes) * np.array(W)) - 0.0
def constraint3(W):
    return np.sqrt((W.transpose()).dot(predict_cov_matrix).dot(W)) - 0.0
b = (0.01, 1)
bnds = [b]*6
con1 = {'type':'eq', 'fun': constraint1}
con2 = {'type':'ineq', 'fun': constraint2}
con3 = {'type':'ineq', 'fun': constraint3}
cons = [con1, con2, con3]
portfolio_dnn = minimize(finale_function, W, method='SLSQP', constraints=cons,
bounds=bnds)
portfolio_dnn
print('Risk',
np.sqrt((portfolio_dnn.x.transpose()).dot(predict_cov_matrix).dot(portfolio_dnn.x)) * 100)
print('Income', np.sum(np.array(returnes) * np.array(portfolio_dnn.x)) * 100)
print()
print(pd.DataFrame(portfolio_dnn.x, index = stock_data.columns,
columns=['Weights']).sort_values(by='Weights'))
rnn_predictions = pd.DataFrame(predictions_rnn)
returnt_rate_data = np.log(rnn_predictions / rnn_predictions.shift(1))
returnt_rate_data = returnt_rate_data[1:]
returnes = returnt_rate_data.mean()
risks = returnt_rate_data.std()
predict_cov_matrix = returnt_rate_data.cov()
W = np.ones(len(returnes)) / len(returnes)
portfolio_rnn = minimize(finale_function, W,
method='SLSQP', constraints=cons, bounds=bnds)
portfolio_rnn
print('Fun', portfolio_rnn.fun)

```

```

    print('Risk',
np.sqrt((portfolio_rnn.x.transpose()).dot(predict_cov_matrix).dot(portfolio_rnn.x)) * 100)
    print('Income', np.sum(np.array(returns) * np.array(portfolio_rnn.x)) * 100)
    print()
    print(pd.DataFrame(portfolio_rnn.x,          index          =          stock_data.columns,
columns=['Weights']).sort_values(by='Weights'))
    true_data = stock_data.iloc[split_time:,:]
    returt_rate_data = np.log(true_data /
true_data.shift(1))
    returt_rate_data = returt_rate_data[1:]
    returns = returt_rate_data.mean()
    risks = returt_rate_data.std()
    predict_cov_matrix = returt_rate_data.cov()
    W = np.ones(len(returns)) / len(returns)
    portfolio_true = minimize(finale_function, W,
method='SLSQP', constraints=cons, bounds=bnds)
    portfolio_true
    print('Risk',
np.sqrt((portfolio_true.x.transpose()).dot(predict_cov_matrix).dot(portfolio_true.x)) * 100)
    print('Income', np.sum(np.array(returns) * np.array(portfolio_true.x)) * 100)
    print()
    print(pd.DataFrame(portfolio_true.x,          index          =          stock_data.columns,
columns=['Weights']).sort_values(by='Weights'))
    arima_predictions_df = pd.DataFrame(arima_predictions)
    returt_rate_data = np.log(arima_predictions_df / arima_predictions_df.shift(1))
    returt_rate_data = returt_rate_data[1:]
    returns = returt_rate_data.mean()
    risks = returt_rate_data.std()
    predict_cov_matrix = returt_rate_data.cov()
    W = np.ones(len(returns)) / len(returns)
    portfolio_arima = minimize(finale_function,
W, method='SLSQP', constraints=cons, bounds=bnds)
    portfolio_arima
    print('W', portfolio_arima.fun )
    print('Risk',
np.sqrt((portfolio_arima.x.transpose()).dot(predict_cov_matrix).dot(portfolio_arima.x)) *
100)
    print('Income', np.sum(np.array(returns) * np.array(portfolio_arima.x)) * 100)
    print()
    print(pd.DataFrame(portfolio_arima.x,          index          =          stock_data.columns,
columns=['Weights']).sort_values(by='Weights'))
    #GARCH Portfolio

```

```

    garch_predict_cov_matrixgarch_reurt_rate_data = np.log(stock_data /
stock_data.shift(1))
    garch_reurt_rate_data = garch_reurt_rate_data[1:]returnes = returnes_garch
    risks = risks_garch
    predict_cov_matrix = garch_predict_cov_matrix
    W = np.ones(len(returnes_garch)) / len(returnes_garch)portfolio_garch =
minimize(finale_function, W, method='SLSQP', constraints=cons, bounds=bnds)
    portfolio_garchprint('W', portfolio_garch.fun )
    print('Risk',
np.sqrt((portfolio_garch.x.transpose()).dot(predict_cov_matrix).dot(portfolio_garch.x)) *
100)
    print('Income', np.sum(np.array(returnes) * np.array(portfolio_garch.x)) * 100)
    print()
    print(pd.DataFrame(portfolio_garch.x, index = stock_data.columns,
columns=['Weights']).sort_values(by='Weights'))

```