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

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

До захисту допущено:

Завідувач кафедри

_____ Олександр РОЛІК

«___»_____20__ р.

Дипломний проєкт

на здобуття ступеня бакалавра
за освітньо–професійною програмою «Інформаційні управляючі системи та технології»
спеціальності 126 «Інформаційні системи та технології»
на тему: «Кількісний аналіз фінансових ринків за допомогою нейронних мереж»

Виконав (–ла):
студент (–ка) IV курсу, групи ІС–92
Матвєєнков Костянтин Валентинович _____

Керівник:
Доцент кафедри ІСТ, к.т.н
Шимкович Володимир Миколайович _____

Рецензент:
Доцент кафедри ОТ, к.т.н
Волокита Артем Миколайович _____

Засвідчую, що у цьому дипломному проєкті немає запозичень з праць інших авторів без відповідних посилань.
Студент (–ка)_____

Київ – 2023 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 126 «Інформаційні системи та технології»

Освітньо–професійна програма «Інформаційні управляючі системи та технології»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Олександр РОЛІК
«___»_____20__ р.

ЗАВДАННЯ

на дипломний проєкт студенту

Матвєєнкову Костянтину Валентиновичу

1. Тема проєкту «Кількісний аналіз фінансових ринків за допомогою нейронних мереж», керівник проєкту Шимкович Володимир Миколайович, доцент кафедри ІСТ, к.т.н, затверджені наказом по університету від «31» травня 2023 р. №2101–с.

2. Термін подання студентом проєкту: 12 червня 2023 року.

3. Вихідні дані до проєкту: графіки аналізу та файл з даними.

4. Зміст пояснювальної записки: аналіз предметної області, проектування системи, методи розробки, дослідження та експерименти.

5. Перелік графічного матеріалу(із зазначенням обов’язкових креслеників, плакатів, презентацій тощо): схема нейронної мережі, діаграма потоків даних, діаграма станів, діаграма прецедентів, діаграма компонентів.

6. Дата видачі завдання 22 лютого 2023 року.

Календарний план

№ з/п	Назва етапів виконання проєкту	Термін виконання етапів проєкту	Примітка
1.	Затвердження теми роботи	11.02.23	
2.	Проведення аналізу предметної області	17.04.23 – 23.04.23	
3.	Огляд існуючих рішень у даній галузі, розгляд наявних стратегій аналізу	24.04.23 – 30.04.23	
4.	Проектування програмного забезпечення	01.05.23 – 07.05.23	
5.	Розробка програмного продукту	08.05.23 – 14.05.23	
6.	Конфігурація та налаштування систем для дослідження	15.05.23 – 21.05.23	
7	Оформлення користувацької документації	22.05.23 – 06.06.23	

Студент Костянтин МАТВЄЄНКОВ

Керівник Володимир ШИМКОВИЧ

АНОТАЦІЯ

Матвєєнков К.В. Кількісний аналіз фінансових ринків за допомогою нейронних мереж. КПІ ім. Ігоря Сікорського, Київ, 2023.

Проєкт містить 70 с. тексту, 44 рисунка та посилання на 16 наукових джерел.

Ключові слова: нейронна мережа, машинне навчання, фінансових аналіз, кількісний аналіз, штучний інтелект.

Об’єктом розробки є система аналізу фінансових ринків для передбачення руху та знаходження аномалій.

Метою роботи є підвищення ефективності аналізу фінансових ринків за допомогою розробки програмного забезпечення на базі нейронних мереж. Можливість аналізувати фінансовий інструмент дивлячись на результат який було видано нейронною мережею дає змогу інвестору зосередитись на правильному підході та курсі прийняття рішення.

У дипломному проєкті було розроблено програмне забезпечення, яке буде використовувати розроблену модель та стратегії для аналізу фінансових ринків за допомогою нейронних мереж та підтримки прийняття рішень щодо інвестування.

Отриманні результати мають користь не лише для інвесторів або трейдерів які тільки починають свою кар’єру, а й для інженерів–науковців, які вирішують задачі у галузі аналізу часових рядів за допомогою нейронних мереж.

SUMMARY

Matvieienkov K.V. Quantitative Analysis of Financial Markets Using Neural Networks. Igor Sikorsky Kyiv Polytechnic Institute, Kyiv, 2023.

The project contains 70 pages of text, 44 figures, and references to 16 scientific sources.

Keywords: Neural network, machine learning, financial analysis, quantitative analysis, artificial intelligence.

The object of the project is a system for analysing financial markets to predict movement and detect anomalies.

The goal of the work is to enhance the efficiency of financial market analysis through the development of software based on neural networks. The ability to analyse a financial instrument by looking at the result issued by the neural network allows the investor to focus on the correct approach and decision–making course.

The diploma project has a developed software that will use the developed model and strategies for analysing financial markets using neural networks and supporting investment decision–making.

The results obtained are beneficial not only for investors or traders who are just starting their career, but also for engineer–scientists who solve problems in the field of time series analysis using neural networks.

Номер рядка		Позначення	Найменування	Кільк. аркушів	Номер екзем.	Примітка
1			Документація загальна			
2						
3			Знову розроблена			
4						
5	А4	ІС92.016БАК.005 ПЗ	Пояснювальна записка	70		
6	А3	ІС92.016БАК.005 Д1	Схема нейронної мережі	1		
7						
8	А3	ІС92.016БАК.005 Д2	Діаграма потоків даних	1		
9						
10	А3	ІС92.016БАК.005 Д3	Діаграма станів	1		
11						
12	А3	ІС92.016БАК.005 Д4	Діаграма прецедентів	1		
13						
14	А3	ІС92.016БАК.005 Д5	Діаграма компонентів	1		
15						
16						
17						
18						
19						
20						
21						
22						
23						
24						
25						
26						
27						
28						

Зм.	Аркуш	№ докум.	Підпис	Дата	ІС92.016БАК.004 ТП		
Розроб.		Матвєєнков К.В.			Кількісний аналіз фінансових ринків за допомогою нейронних мереж. Відомість проєкту	Літ.	Аркуш 1 / Аркушів 1
Керівн.		Шимкович В.М				КПІ ім. Ігоря Сікорського Група ІС–92	
Затв.							

Пояснювальна записка
до дипломного проекту
на тему: «Кількісний аналіз фінансових ринків за допомогою нейронних мереж»

Київ – 2023 року

ЗМІСТ

ВСТУП

У сучасному глобалізованому світі, де технології постійно розвиваються та впливають на кожен аспект нашого життя, фінансові ринки зазнають суттєвих змін. Вони стають все більш динамічними, вимогливими та складними для аналізу. Ці зміни накладають певні виклики на традиційні методи інвестування та аналізу ринку. Що було дієвим і надійним у минулому, сьогодні може виявитися неефективним у світлі швидких та непередбачуваних змін на фінансових ринках.

Сучасні фінансові ринки стали надзвичайно складними та динамічними. Традиційні методи аналізу фінансових ринків, які базуються на статистичних моделях та економічно–фундаментальних підходах, зустрічають значні виклики у такому складному ринковому середовищі. Це пояснюється тим, що фінансові ринки підлягають багатьом факторам, таким як геополітичні події, макроекономічні зміни, новини та події, які можуть миттєво впливати на ціни активів. Крім того, великий обсяг доступних даних на фінансових ринках робить традиційні методи аналізу менш ефективними.

Зростання складності фінансових ринків вимагає та дає можливість шукати нові підходи та інструменти для їх аналізу, використання технологій машинного навчання та глибокого навчання стає все більш актуальним. Нейронні мережі, здатні моделювати складні залежності та виявляти нелінійні закономірності у даних, дозволяють отримати більш точні та надійні прогнози на фінансових ринках. Вони можуть адаптуватися до постійно змінних ринкових умов, виявляти складні патерни та незрозумілі залежності, що дозволяє здійснювати більш обґрунтовані та ефективні інвестиційні рішення.

Однією з ключових переваг використання нейронних мереж є їх здатність автоматично виявляти та використовувати складні взаємозв'язки між факторами на фінансових ринках. Такі моделі можуть аналізувати великі обсяги даних, включаючи історичні ціни, новини, соціальні медіа, макроекономічні показники та інші фактори, що впливають на ринок. Вони можуть виявляти складні патерни та тренди, які є важко помітними для людського аналітика, та здійснювати прогнози на основі цих взаємозв'язків.

Машинне навчання та нейронні мережі – це вже не просто теоретичні науки, але практичні інструменти, які застосовуються в багатьох сферах нашого життя. Вони вже використовуються у таких областях як медицина, інженерія, маркетинг та, звичайно, фінанси. Використання технологій машинного навчання для аналізу фінансових даних може допомогти у розробці нових стратегій інвестування, які враховують динаміку ринку та використовують складність даних для досягнення більш високих прибутків. При цьому важливим є розробка програмного забезпечення, яке вміє ефективно працювати з нейронними мережами та аналізувати фінансові дані.

Метою цієї роботи є підвищення ефективності аналізу фінансових ринків за допомогою розробки програмного забезпечення на базі нейронних мереж.

Для досягнення мети цієї роботи треба вирішити наступні задачі:

– Розробити методику збору та підготовки фінансових даних для подальшого використання в нейронних мережах. Це включає в себе обробку, масштабування та нормалізацію даних для підвищення якості моделей;

– розробити архітектуру нейронної мережі, яка буде використовуватися для аналізу фінансових даних. Це включає в себе вибір типу мережі, кількості шарів, типів нейронів та оптимізацію гіперпараметрів;

– застосувати розроблену модель до реальних фінансових даних та провести експерименти для оцінки її ефективності та точності;

– розробити програмне забезпечення, яке буде використовувати розроблену модель та стратегії для аналізу фінансових ринків та підтримки прийняття рішень щодо інвестування.

Ці задачі допоможуть досягти поставленої мети – покращення ефективності аналізу фінансових ринків.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Опис предметного середовища

В даному розділі здійснюється аналіз предметного середовища, в рамках якого буде використовуватися розроблена система прогнозування руху цін активів на фінансових ринках. Розглядаються основні процеси та суб'єкти, що беруть участь в діяльності на фінансових ринках, а також взаємодія між ними.

1.1.1 Опис процесу діяльності

Фінансові ринки – це місце, де здійснюється торгівля фінансовими інструментами, такими як акції, облігації, валюта та похідні фінансові інструменти. Основні процеси діяльності на фінансових ринках включають:

– Торгівля активами: Купівля та продаж акцій, облігацій, валют, товарів та інших фінансових інструментів;

– аналіз фінансових ринків: Вивчення динаміки цін на різні активи, аналіз факторів, які впливають на ціну, прогнозування майбутнього розвитку ринку;

– розробка нових фінансових інструментів: Створення нових інвестиційних продуктів, які задовольняють потреби різних інвесторів;

– керування ризиками: Мінімізація ризиків, пов'язаних з інвестуванням, забезпечення фінансової стабільності, зменшення можливості фінансових втрат;

– регулювання ринків: Забезпечення ефективного та справедливого функціонування фінансових ринків, запобігання маніпуляціям та іншим нечесним практикам.

Основні суб'єкти, що беруть участь в діяльності на фінансових ринках, включають інвесторів, брокерів, дилерів, аналітиків, регуляторів.

1.1.2 Опис функціональної моделі

Функціональна модель системи прогнозування руху цін активів на фінансових ринках включає наступні основні елементи:

- Збір і обробка історичних даних про рух цін активів;
- вибір та налаштування алгоритму для знаходження аномалій;
- пошук аномалій у руху цін активів на основі обраного алгоритму;
- відображення результатів прогнозування для користувачів системи;
- вибір та налаштування моделі нейронної мережі для оцінки ризиків для заходження у ринок;
- тренування нейронної мережі на історичних даних;
- оцінка точності прогнозів та їх коригування.

Користувачі системи прогнозування руху цін активів на фінансових ринках включають:

- Інвестори та трейдери – особи, які використовують систему для отримання прогнозів та прийняття інвестиційних рішень;
- аналітики – фахівці, які забезпечують розвиток та підтримку моделей прогнозування.

1.2 Огляд наявних аналогів

Квантитативний аналіз використовує математичні та статистичні методи для вивчення інвестиційних або фінансових тенденцій. Нейронні мережі стали популярними інструментами квантитативного аналізу через їх здатність навчатися та адаптуватися до нових даних. Декілька наявних стратегій використання нейронних мереж для квантитативного аналізу фінансових ринків включають:

- Прогнозування часових рядів: Нейронні мережі можуть бути використані для прогнозування майбутніх значень фінансового часового ряду на основі попередніх значень. Цей підхід використовується у багатьох фінансових аплікаціях, включаючи прогнозування цін на акції та волатильності. На рисунку 1.1

можна побачити стандартний приклад графіку часового ряду. Цей графік вказує на процентну зміну ВВП з 1980 року;

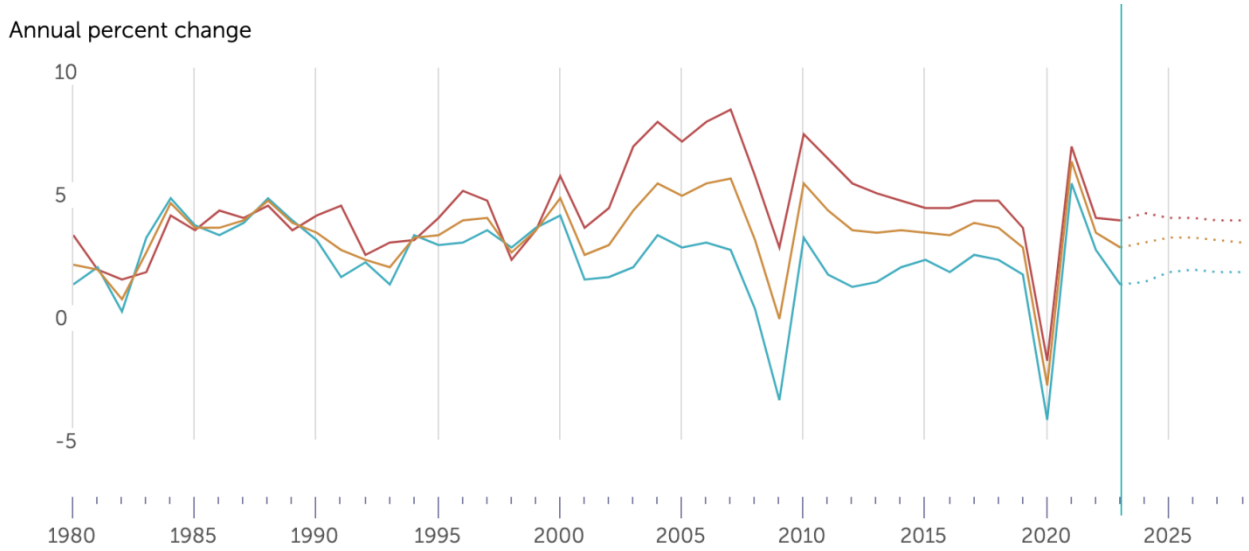

Рисунок 1.1 – Приклад вигляду часового ряду на графіку[1]

– алгоритмічна торгівля: Нейронні мережі можуть бути використані для розробки алгоритмічних торгових стратегій. Вони можуть аналізувати великі обсяги даних та визначати оптимальні моменти для купівлі або продажу активів. Як можна побачити на рисунку 1.2 алгоритмічна торгівля займає дуже великий сектор фінансового ринку. Торгівля акціями майже на 70 відсотків приходиться на алгоритмічну торгівлю;

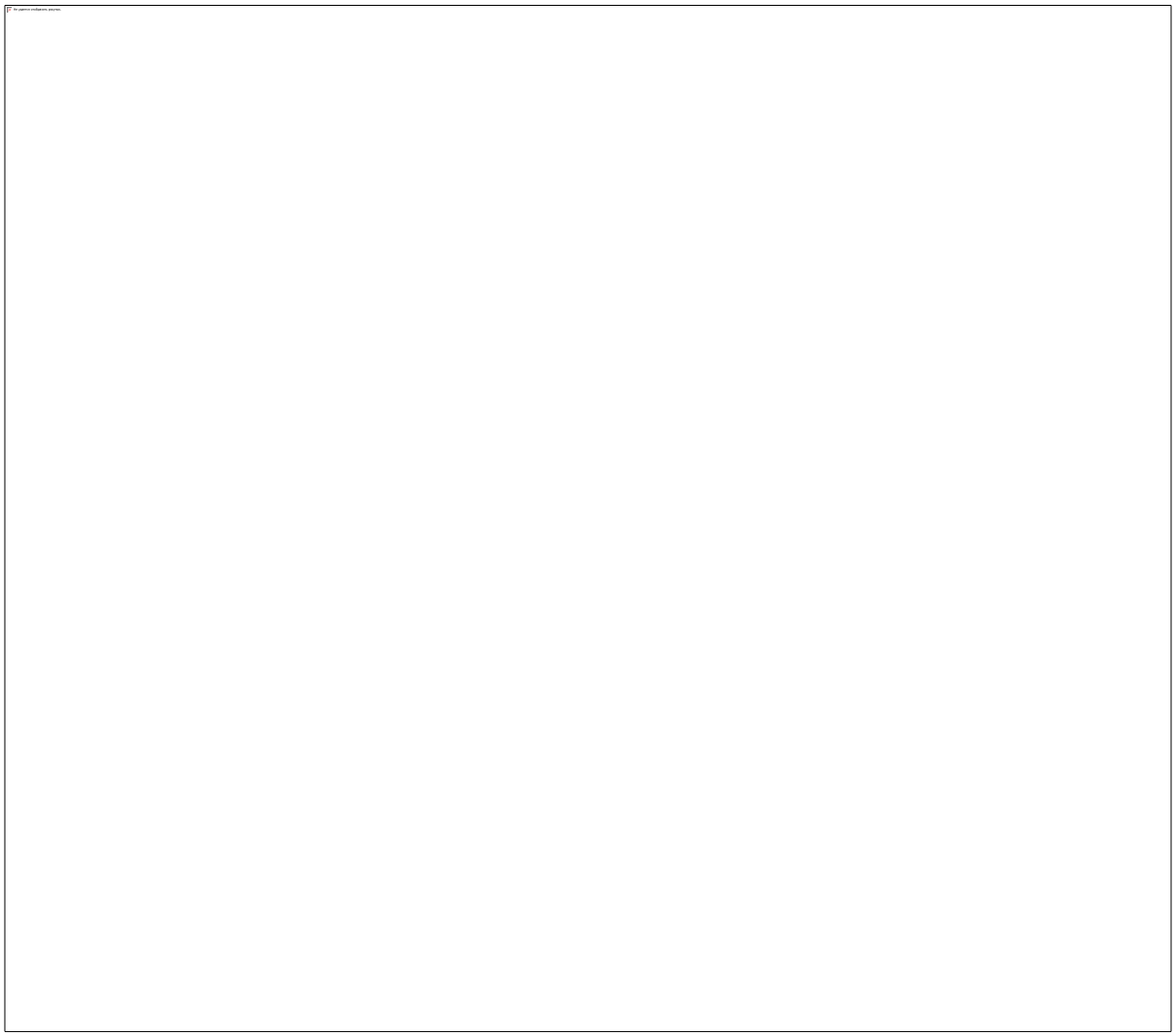

Рисунок 1.2 – Об'єм ринку за фінансовим класом зайнятим алгоритмічною торгівлею[2]

– виявлення шаблонів: Нейронні мережі можуть виявляти складні шаблони в даних, які можуть бути непомітними для людини або традиційних статистичних методів. Це може бути корисним для виявлення тенденцій або аномалій в даних.

Деякі з конкретних підходів та рішень, які використовують нейронні мережі для квантитативного аналізу фінансових ринків, включають:

– TensorTrade: Відкрита бібліотека для алгоритмічної торгівлі, яка використовує машинне навчання та нейронні мережі для оптимізації торгових стратегій. TensorTrade дозволяє створювати та тестувати різні види торгових стратегій з використанням різноманітних даних та індикаторів;

– Prophet: Проект від Facebook, який використовує нейронні мережі для прогнозування часових рядів. Хоча він не спеціально призначений для фінансових ринків, Prophet широко використовується в цій області через свою здатність до точного прогнозування майбутніх значень на основі історичних даних. На рисунку 1.3 зображено один з прикладів використання Prophet для представлення часових рядів;

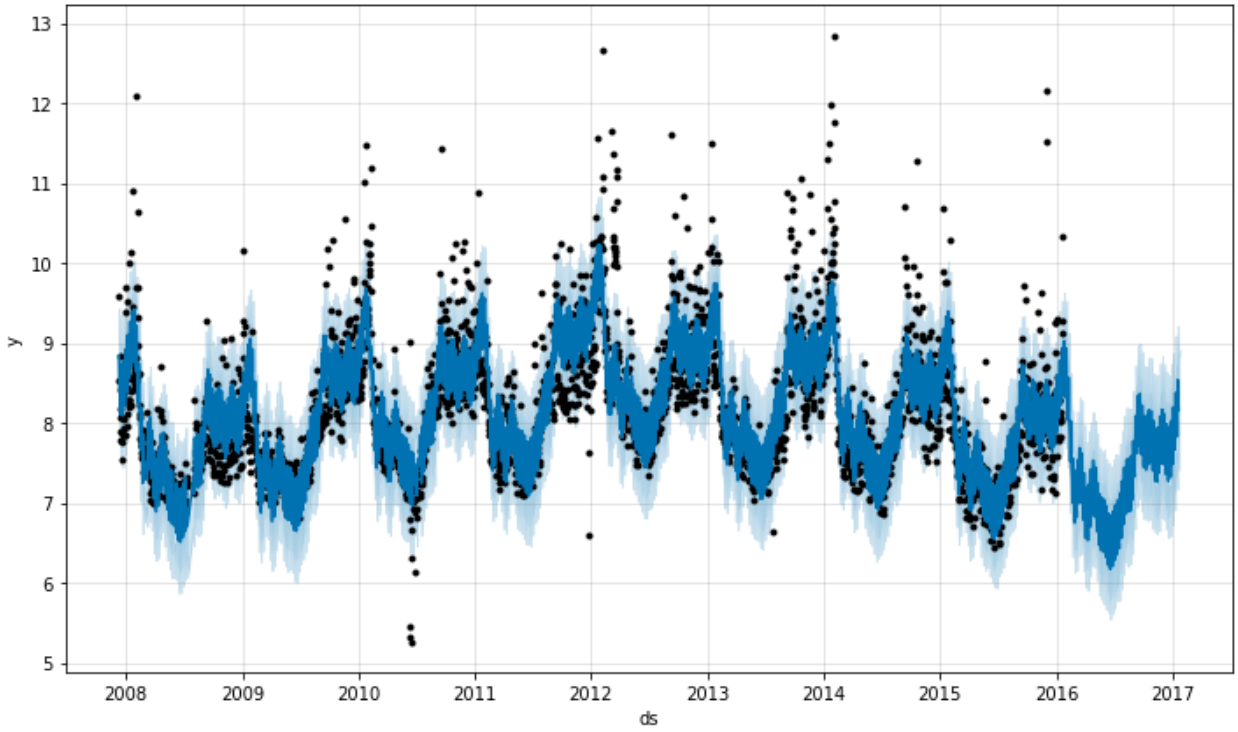

Рисунок 1.3 – Приклад ілюстрації даних за допомогою Prophet[3]

– Alphalens: Інструмент для тестування статистичної значимості прогнозних сигналів. Він використовується для перевірки ефективності сигналів, що генеруються моделями машинного навчання, включаючи нейронні мережі;

– також інші приватні комерційні проекти таких компаній як: Citadel, Jane Street, 2Sigma та інші квантитативні хедж фонди та інвестиційні фірми.

1.3 Огляд стратегій для машинного навчання

Контрольоване навчання – є одним з найпоширеніших видів машинного навчання. В цьому підході, алгоритм машинного навчання "навчається" на основі набору тренувальних даних, який складається з пар вхідних даних та відповідних

їм "правильних" вихідних даних. Навчання ведеться з метою мінімізації помилки прогнозування на тренувальному наборі даних. Прикладами задач, що вирішуються за допомогою контрольованого навчання, є класифікація об'єктів і регресія.

Неконтрольоване навчання – відрізняється від контрольованого тим, що в цьому випадку тренувальний набір даних складається тільки з вхідних даних, і "правильні" вихідні дані не задаються. В цьому випадку, алгоритм машинного навчання намагається знайти в даних структуру або взаємозв'язки між даними. Прикладами задач, що вирішуються за допомогою неконтрольованого навчання, є кластеризація даних і зниження розмірності даних.

Напівавтоматичне навчання – використовує комбінацію маркованих і немаркованих даних для тренування. Воно призначене для ситуацій, коли марковані дані важко або дорого отримати, але доступні великі обсяги немаркованих даних. Завданнями напівавтоматичного навчання можуть бути як класифікація об'єктів, так і регресія.

Контрольоване навчання є ідеальним підходом для вирішення поставленої задачі, тобто задачі аналізу часових рядів, і ось чому:

– Часові ряди є структурованими даними, де порядок спостережень важливий, а це підходить для контрольованого навчання. Ці дані можуть бути використані для тренування моделі на відповідних виходах, тобто на майбутніх значеннях часового ряду;

– контрольоване навчання вимагає набору даних для тренування, де кожному вхідному датасету відповідає вихідний результат. У випадку часових рядів, можна використовувати попередні точки даних як вхід, а наступну точку як вихід;

– часові ряди часто мають важливі темпоральні властивості, такі як тренди та сезонність. Підходи контрольованого навчання, такі як LSTM, ефективно використовують ці властивості, навчаючись на тренувальних даних і передбачаючи майбутні значення;

– використання контрольованого навчання також дозволяє нам використовувати розмаїті техніки перевірки якості моделі, такі як крос–валідація та розділення на тренувальні та тестові датасети;

– нарешті, контрольоване навчання забезпечує конкретними метриками, які допомагають оцінити процес навчання та ефективність моделі, такими як втрата MSE (Mean Squared Error) або MAE (Mean Absolute Error), що є особливо корисними для задач прогнозування часових рядів.

1.3 Огляд алгоритмів для машинного навчання

1.3.1 Регресивні алгоритми

Алгоритми регресії є одними з найпростіших і найпопулярніших алгоритмів для задач прогнозування. Вони працюють за принципом найкращого припасування лінії або кривої до даних. Це означає, що вони використовують статистичний аналіз для прогнозування вихідного значення, заснованого на вхідних значеннях.

Плюси:

– Вони прості для розуміння і використання;

– вони можуть працювати добре з невеликими наборами даних;

– вони використовуються для прогнозування числових значень, що робить їх дуже корисними в багатьох різних областях.

Мінуси:

– Вони можуть бути недостатньо гнучкими для роботи з складними наборами даних;

– вони можуть мати високу помилку прогнозування, якщо відносини між вхідними і вихідними даними не є лінійними.

1.3.2 Алгоритм навчання на прикладах

Навчання на прикладах, такі як метод найближчих k–сусідів (k–NN), працюють за принципом знаходження подібних прикладів (або "інстансій") у тренувальних даних.

Плюси:

– Вони дуже прості у використанні;

– вони не вимагають ніякої тренувальної фази, тому можуть працювати дуже швидко.

Мінуси:

– Вони можуть бути дуже повільними при великому обсязі даних, оскільки кожний новий приклад має бути порівняний з усіма тренувальними прикладами;

– вони можуть мати високу помилку прогнозування, якщо немає чітко визначених схожостей між прикладами.

1.3.3 Алгоритми регуляризації

Регуляризаційні алгоритми, такі як Lasso та Ridge, використовуються для запобігання перенавчанню моделей шляхом штрафування великих вагових коефіцієнтів. Це робить модель більш гнучкою та здатною до загальної адаптації.

Плюси:

– Вони допомагають зменшити перенавчання, що може бути важливим у багатьох задачах машинного навчання;

– вони можуть допомогти відібрати найважливіші ознаки в даних, роблячи модель більш прозорою та зрозумілою.

Мінуси:

– Вони можуть бути складними для налаштування, оскільки вони вимагають вибору відповідного гіперпараметру регуляризації;

– якщо параметр регуляризації встановлено надто високим, модель може стати занадто простою і мати велику помилку прогнозування.

1.3.4 Алгоритми дерева рішень

Алгоритми дерева рішень працюють за принципом поділу даних на менші підмножини на основі різних властивостей. Вони використовуються в задачах класифікації та регресії.

Плюси:

– Вони прості для розуміння та використання;

– вони можуть працювати добре з невеликими наборами даних;

– вони можуть працювати з числовими та категоріальними даними.

Мінуси:

– Вони можуть легко перенавчатися, особливо якщо дані мають багато ознак;

– вони можуть бути неефективними при роботі з великими наборами даних.

1.3.5 Алгоритми за Баєсом

Алгоритми Баєса використовуються в задачах класифікації та регресії. Вони працюють на основі Теореми Баєса [5], яка дозволяє оновлювати ймовірність гіпотези на основі нових даних.

Плюси:

– Вони дуже гнучкі і можуть працювати з великою кількістю ознак;

– вони враховують невпевненість, що робить їх корисними в задачах з невизначеними або неповними даними.

Мінуси:

– Вони можуть бути складними для налаштування і використання;

– вони можуть бути повільними при великих обсягах даних.

1.3.6 Алгоритми кластеризації

Алгоритми кластеризації використовуються для групування подібних прикладів в наборах даних. Вони часто використовуються в задачах безконтрольного навчання, наприклад, для аналізу шаблонів або для розпізнавання образів.

Плюси:

- Вони можуть виявити приховані шаблони в даних;
- вони можуть працювати з великими наборами даних.

Мінуси:

- Вони можуть бути складними для налаштування, особливо якщо дані мають багато ознак або якщо не є відомою кількість кластерів;
- вони можуть мати проблеми з визначенням форми та розміру кластерів.

1.3.7 Алгоритм навчання асоціативних правил

Алгоритми навчання асоціативних правил використовуються для виявлення зв'язків або асоціацій між ознаками в наборах даних. Вони часто використовуються в системах рекомендацій для виявлення товарів, які часто купуються разом.

Плюси:

- Вони можуть виявити цікаві шаблони та асоціації в даних;
- вони можуть працювати з великими наборами даних.

Мінуси:

- Вони можуть бути повільними при великих обсягах даних;
- вони можуть виявити багато незначущих асоціацій, якщо параметри не налаштовані правильно.

1.3.8 Алгоритми штучних нейронних мереж

Алгоритми штучних нейронних мереж (ANN) використовуються в складних задачах, таких як класифікація зображень, розпізнавання та обробка природної мови.

Плюси:

– Вони дуже потужні та можуть працювати з великою кількістю ознак та класів;

– вони можуть вивчити нелінійні відносини та складні шаблони в даних.

Мінуси:

– Вони можуть бути складними для налаштування та використання;

– вони можуть потребувати великих обсягів даних для навчання.

1.3.9 Алгоритми глибокого навчання

Алгоритми глибокого навчання – це продвинута версія ANN, яка може використовувати багато шарів прихованих нейронів для вивчення ще більш складних шаблонів. Вони використовуються в задачах, які вимагають високого рівня абстракції, таких як розпізнавання образів, генерація тексту та обробка природної мови.

Плюси:

– Вони дуже потужні та можуть працювати з великою кількістю ознак та класів;

– вони можуть вивчити нелінійні відносини та складні шаблони в даних.

Мінуси:

– Вони можуть бути складними для налаштування та використання;

– вони можуть потребувати великих обсягів даних для навчання.

1.3.10 Алгоритми зниження розмірності

Алгоритми зниження розмірності використовуються для зменшення кількості ознак в даних без значної втрати інформації. Вони використовуються для покращення ефективності інших алгоритмів машинного навчання, усунення шуму та удосконалення візуалізації даних.

Плюси:

- Вони можуть працювати з великими наборами даних;
- вони можуть допомогти зменшити шум і відібрати найважливіші ознаки в даних.

Мінуси:

- Вони можуть втратити деяку інформацію при зниженні розмірності даних;
- вони можуть бути складними для налаштування та використання.

1.3.11 Вибір алгоритму

Ми обираємо алгоритм глибокого навчання з наступних причин:

- Глибоке навчання дозволяє моделям автоматично вивчати та витягувати особливості з часових рядів, що веде до кращих прогнозів;
- Алгоритми глибокого навчання, зокрема LSTM, мають змогу "запам'ятовувати" або зберігати інформацію з минулого, що робить їх особливо корисними для аналізу часових рядів;
- Алгоритми глибокого навчання володіють здатністю впоратися з великими об'ємами даних, що часто зустрічаються в задачах прогнозування часових рядів;
- вони можуть використовувати різні типи даних – структуровані та неструктуровані – для прогнозування, що робить їх вкрай гнучкими та універсальними.

1.4 Постановка задачі

Цей проект створюється з метою розробки системи для кількісного аналізу фінансових ринків за допомогою нейронних мереж. Ця система буде використовувати передові методи аналізу даних, машинного навчання та штучного інтелекту для створення прогнозів руху цін активів на фінансових ринках.

1.4.1 Призначення розробки

Розробка призначена для трейдерів, інвестиційних менеджерів та фінансових аналітиків, які хочуть використовувати передові технології для покращення своїх торгових рішень. Система допоможе їм виконувати точний аналіз ринку та робити більш обґрунтовані прогнози щодо майбутніх рухів цін.

1.4.2 Цілі та задачі розробки

Метою цієї розробки є покращення точності прогнозування рухів цін на фінансових ринках за допомогою нейронних мереж.

Для досягнення цієї мети, поставлено такі завдання:

– Провести аналіз наявних стратегій для кількісного аналізу фінансових ринків;

– розробити систему для збору та обробки фінансових даних, яка буде використовуватися для тренування та перевірки моделі нейронної мережі;

– обґрунтувати вибір конкретної нейронної мережі для використання в системі;

– розробити та оптимізувати модель нейронної мережі для прогнозування рухів цін активів на фінансових ринках;

– провести аналіз ефективності системи, використовуючи історичні дані та порівнюючи результати прогнозування моделі з реальними рухами цін на ринку;

– обґрунтувати можливість впровадження розробленої системи на реальному фінансовому ринку, враховуючи її ефективність, надійність та вартість;

– розробити документацію для користувачів системи, включаючи інструкції з використання та обслуговування системи.

Ці завдання відображають основні етапи розробки проекту і допоможуть досягти поставленої мети. Крім того, вони допоможуть забезпечити, що система буде відповідати потребам користувачів і допоможе їм виконувати більш точний аналіз фінансових ринків і робити кращі інвестиційні рішення.

Висновок до розділу

Під час аналізу предметного середовища та діяльності, було виявлено, що робота з фінансовими ринками вимагає високої точності та ефективності аналізу та прогнозування. Це підтверджує необхідність використання квантитативних методів аналізу, зокрема, штучних нейронних мереж.

В ході огляду наявних аналогів було виявлено декілька стратегій, що використовують нейронні мережі для аналізу фінансових ринків. Вони використовують різні типи нейронних мереж та стратегій трейдінгу, але жодна з них не враховує всі потреби та особливості праці трейдерів на фінансових ринках.

Тому, було визначено мету та завдання проекту: створити систему, що використовує нейронні мережі для аналізу фінансових ринків, зокрема для прогнозування цін на активи. Система повинна бути простою у використанні, надійною та ефективною.

Завдання, сформульовані в цьому розділі, відображають основні етапи розробки проекту і допоможуть досягти поставленої мети. Вони допоможуть забезпечити, що система буде відповідати потребам користувачів і допоможе їм виконувати більш точний аналіз фінансових ринків.

2 ПРОЕКТУВАННЯ СИСТЕМИ

2.1 Структура системи

В основу розробленої системи покладено нейронну мережу, яка виконує аналіз даних і прогнозування потенційних результатів інвестицій. Ця система була спеціально розроблена для використання інвесторами, які шукають досконалого засобу для аналізу і моніторингу їх поточних і потенційних інвестицій, забезпечуючи максимальну ефективність і впевненість в своїх фінансових рішеннях.

Структура системи складається з кількох ключових компонентів, кожен з яких відіграє важливу роль у загальному циклі обробки і аналізу даних. Спільно ці компоненти створюють потужний інструмент, який допомагає користувачам робити обґрунтовані прогнози щодо майбутніх фінансових результатів.

Нейронна мережа є серцевиною системи. Це потужний інструмент для обробки даних, який може приймати широкий спектр вхідних даних, обробляти їх і використовувати для створення точних прогнозів. Використовуючи передові алгоритми машинного навчання та глибокого навчання, нейронна мережа здатна виявляти складні шаблони і тенденції в даних, які можуть виявитися непомітними для людського ока. Вона не лише виконує прогнозування, але і навчається з кожним новим набором даних, постійно покращуючи свою точність і надійність.

Модуль введення даних створений для забезпечення зручного та ефективного способу вводу даних користувачем. Користувач може вводити інформацію про зазначені нами фінансові активи, які він хоче аналізувати. Користувач має вводити лише відповідні до випадку історичні дані за період який бере за основу для інвестування.

Модуль виведення даних відповідає за відображення результатів аналізу та прогнозування. Він конвертує складні числові дані, що генеруються нейронною мережею, у зрозумілому та зручному для аналізу форматі. Це може бути графік, таблиця з даними або інший формат візуалізації, який дозволить користувачеві швидко оцінити результати прогнозу.

Також користувач зможе самостійно натренувати свою мережу для того щоб оцінити результати, для цього йому просто буде потрібно завантажити історичні дані одного з активів які підходять до категорії індексів. Після цього користувач буде мати можливість використати загальну модель по цьому датасету для подальшого аналізу.

В результаті ця структура робить систему гнучкою і адаптивною, дозволяючи їй працювати з різними видами фінансових активів і враховувати широкий спектр факторів, які можуть вплинути на результати інвестицій. Це не лише полегшує процес інвестицій для користувача, але й дозволяє системі постійно вдосконалюватися, навчаючись з кожного нового набору даних, що вводиться.

2.2 Вибір мови програмування системі

Обрання мови програмування для конкретного проекту часто відбувається за основою багатьох факторів, таких як специфіка задачі, вимоги до проекту, наявність бібліотек та інструментів, що підтримуються, а також особистий досвід та знання розробника. У контексті цього проекту, було розглянуто наступні мови програмування: Python, JavaScript, Java та C++.

2.2.1 Мова програмування Python

Python – це високорівнева, динамічна мова програмування, яка відзначається своєю простотою та читабельністю. Вона має чистий, простий синтаксис, який зробив її вибором №1 для багатьох розробників, особливо для тих, хто працює з машинним навчанням та науковими обчисленнями. Бібліотеки, такі як NumPy, Pandas, Matplotlib, Scikit–learn та TensorFlow, роблять Python особливо корисним для обробки та аналізу даних, а також для моделювання і візуалізації. В Python є потужна підтримка спільноти, що включає багато навчальних ресурсів та модулів. Python також підтримує різні стилі програмування, включаючи процедурне, об'єктно–орієнтоване та функціональне програмування. Python має вбудовану підтримку тестування, що полегшує виявлення та виправлення помилок. Крім того,

Python працює на багатьох платформах, включаючи Windows, Linux, macOS, що робить його універсальним вибором для розробників.

2.2.2 Мова програмування Javascript

JavaScript – це мова програмування, яка використовується переважно на стороні клієнта для створення інтерактивних веб–сайтів. Вона може використовуватися також для серверної розробки за допомогою Node.js. JavaScript дозволяє розробникам створювати анімації, обробляти події користувачів та виконувати асинхронні запити до сервера. JavaScript – це мова з першокласною підтримкою функцій, що сприяє функціональному стилю програмування. JavaScript може працювати з JSON, форматом даних, що часто використовується для веб–API. Останнім часом JavaScript став більш важливим на серверній стороні через розвиток Node.js. Однак, він не володіє такою ж кількістю аналітичних бібліотек, як Python, що робить його менш підходящим для задач аналізу даних та машинного навчання.

2.2.3 Мова програмування Java

Java – це сильно типізована, об'єктно–орієнтована мова програмування, яка була розроблена з метою надання "напиши один раз, запусти будь–де" здатності. Це означає, що Java може працювати на будь–якій платформі, яка підтримує Java Virtual Machine (JVM). Java використовується в широкому спектрі застосувань, від веб–серверів до мобільних застосунків Android. Java має сильну підтримку спільноти та велику кількість відкритого програмного забезпечення, включаючи потужні засоби для машинного навчання, як Apache Spark і Deeplearning4j. Однак, Java має складніший синтаксис порівняно з Python, і навчання Java може зайняти більше часу. Крім того, для навчання моделей машинного навчання часто використовуються бібліотеки, написані на Python.

2.2.4 Мова програмування C++

C++ – це мова програмування загального призначення, яка підтримує процедурне, об'єктно–орієнтоване та рівень більш низький відносно до інших мов, програмування. C++ широко використовується в розробці системного та ігрового програмного забезпечення, де потрібна висока продуктивність та контроль над апаратними ресурсами. C++ має складніший синтаксис і меншу стандартну бібліотеку порівняно з Python. Він також має меншу підтримку для високорівневих операцій обробки даних та машинного навчання. C++ може використовуватися для розробки високопродуктивних модулів для Python, але це вимагає глибокого розуміння обох мов.

Отже, у контексті обраної системи, Python є найкращим вибором з кількох причин. По–перше, Python має широкий набір бібліотек і інструментів для машинного навчання, що дозволяє нам легко розробити та налаштувати нейронну мережу для поставленої задачі. Бібліотеки, такі як TensorFlow та Keras, надають високорівневий інтерфейс для розробки та тренування моделей глибокого навчання.

По–друге, Python є простою та зручною мовою програмування, що сприяє швидкій прототипній розробці та експериментуванню. Його синтаксис легко зрозуміти, а код легко підтримувати.

По–третє, Python добре підтримується спільнотою та має велику кількість навчальних ресурсів, що дозволяє нам швидко вирішувати проблеми та знаходити відповіді на питання.

Нарешті, Python відмінно підходить для аналізу даних. Бібліотеки, такі як Pandas і NumPy, забезпечують зручні інструменти для обробки, аналізу та візуалізації даних.

Враховуючи все вище сказане, можемо впевнено заявити, що Python є найкращим вибором мови програмування для цієї системи.

2.3 Аналіз та використання наявних бібліотек

2.3.1 Бібліотека Pandas

Pandas — це високорівнева бібліотека Python, призначена для зручної роботи зі структурованими даними. Вона надає широкий спектр високорівневих інструментів для аналізу даних, включаючи об'єкти для маніпуляції даними (DataFrame і Series), функції для завантаження/запису даних у різні формати (CSV, Excel, SQL, JSON та ін.), засоби для очищення та трансформації даних, обробки пропущених значень і т.д. Бібліотека pandas є незамінним інструментом при роботі з даними в Python.

2.3.2 Бібліотека Numpy

Numpy – це фундаментальна бібліотека для наукових обчислень в Python. Вона надає високопродуктивні багатовимірні масиви і набір інструментів для роботи з ними. Numpy є основою для більшості наукових бібліотек Python, включаючи Pandas, Matplotlib, SciPy, Scikit–Learn. Бібліотека numpy є основою для проведення числових обчислень в Python.

2.3.3 Бібліотека Tensorflow

TensorFlow – це відкрита бібліотека для машинного навчання і глибокого навчання, розроблена командою Google Brain. Вона надає комплексний набір інструментів для розробки, навчання та використання моделей машинного навчання на різних платформах. TensorFlow підтримує широкий спектр алгоритмів машинного навчання, включаючи нейронні мережі, і дозволяє використовувати графічні процесори для прискорення обчислень.

2.3.4 Бібліотека keras

Keras – є високорівневим інтерфейсом для розробки та навчання нейронних мереж і спроможна працювати поверх бібліотеки TensorFlow. Вона була розроблена з метою забезпечення простого і швидкого прототипування (модульність, гнучкість і спрощеність). Keras дозволяє користувачам легко і швидко створювати моделі глибокого навчання, роблячи її ідеальним інструментом для даної системи.

2.3.5 Бібліотека sklearn

Scikit–Learn – це бібліотека для машинного навчання в Python. Вона містить ряд інструментів для створення моделей машинного навчання, включаючи класифікацію, регресію, кластеризацію, і зниження розмірності. Scikit–Learn також містить ряд інструментів для обробки даних, включаючи кодування категорійних змінних, масштабування, і обробку пропущених значень.

2.3.6 Бібліотека seaborn

Seaborn – це бібліотека візуалізації даних Python, базована на matplotlib. Вона надає високорівневий інтерфейс для створення чітких та інформативних графіків. Завдяки вбудованим темам оформлення і палітрам кольорів, seaborn дозволяє легко і швидко створювати кольорові інформативні графіки.

2.3.7 Бібліотека matplotlib

Matplotlib – це бібліотека для створення статичних, анімованих і інтерактивних візуалізацій в Python. Вона може бути використана в скриптах Python, оболонках Python і Jupyter, веб–додатках і може генерувати PNG, SVG, JPG, EPS, PDF і PGF файли. Matplotlib є основним інструментом для візуалізації даних в Python.

2.4 Вхідні дані

Метою даного проекту є розробка системи для аналізу та прогнозування рухів на фінансових ринках, з особливим акцентом на великі акційні індекси, такі як S&P 500, Dow Jones, NASDAQ та FTSE. Ці великі індекси обрані через їх глобальну значимість та високу ліквідність, що робить їх ідеальними кандидатами для даного дослідження.

Вхідні дані для системи включатимуть цілу ряду інформації, спрямованої на забезпечення всебічного вигляду на стан ринку. Зокрема, система буде аналізувати історичні дані про ціни, волатильність та обсяг торгів, що дозволить виявити тенденції, паттерни та аномалії в рухах цін.

Крім того, система включатиме важливі макроекономічні показники, такі як ставки Федерального резерву, дані про безробіття, інфляційні дані, дані про ВВП та інші. Ці показники важливі, оскільки вони можуть впливати на загальний економічний клімат і, отже, на настрої на фінансових ринках.

Нарешті, система буде використовувати дані про товари та облігації. Дані про товари важливі, оскільки ціни на ключові товари, такі як нафта або золото, можуть впливати на рухи акційних ринків. Облігації та процентні ставки також важливі, оскільки вони можуть впливати на вартість грошей та інвестиційні стратегії.

2.4.1 Збір даних

Збір даних проводився на основі 4 класів даних в кожному по декілька представників. У ці 4 класи входять: дані на ціну актива для аналізу, економічні показники, дані на різні продукти, дані про процентні ставки та облігації. Насамперед було використано 6 датасетів, які було проаналізовано, нормалізовано та скомбіновано в один.

Перелік та опис використаних датасетів:

– S&P500 – історичні дані індексу 500 найпотужніших компаній США, насамперед у світі. Як можна побачимо на рисунку 2.1 дані в себе включають

індикатори про ціну на початку дня, у кіні дня також об’єм та найбільшу з найменшими цінами за день;

Date	open_sp500	high_sp500	low_sp500	close_sp500	adj_close_sp500	volume_sp500
2000-01-01	1466.063314	1474.280029	1451.579956	1464.573324	1464.573324	5.599667e+08
2000-01-02	1467.656657	1476.140015	1444.969970	1459.896647	1459.896647	7.458833e+08
2000-01-03	1469.250000	1478.000000	1438.359985	1455.219971	1455.219971	9.318000e+08
2000-01-04	1455.219971	1455.219971	1397.430054	1399.420044	1399.420044	1.009000e+09
2000-01-05	1399.420044	1413.270020	1377.680054	1402.109985	1402.109985	1.085500e+09

Рисунок 2.1 – Приклад ілюстрації даних датасету S&P500

На рисунку 2.2 зображено графік ціни на відкриття. Це ціна, за якою відкривається торгівля на S&P 500 в кожен конкретний торговий день. Цей показник є важливим, оскільки він відображає початковий баланс між попитом та пропозицією, а також показує настрій інвесторів на початок дня. Трейдери та інвестори часто звертають увагу на цей показник, оскільки він може допомогти в прогнозуванні майбутніх цінових рухів.

Рисунок 2.2 – Частота зміни ціни відкриття S&P500

На рисунку 2.3 можна побачити графік найвищої ціни за день. Цей показник вказує на найвищу ціну, до якої дійшов індекс S&P 500 протягом торгового дня. Це важливий показник, який відображає силу попиту в торговий день. Він може бути

корисним для трейдерів та інвесторів, оскільки він може допомогти виявити потенційні рівні опору в майбутніх торгових сесіях.

Рисунок 2.3 – Частота зміни найвищої ціни S&P500

На рисунку 2.4 можна побачити графік найнижчої ціни індексу. Це найнижча ціна, до якої індекс S&P 500 опустився протягом торгового дня. Він відображає силу пропозиції в торговий день та може слугувати вказівкою для потенційних рівнів підтримки. Трейдери та інвестори використовують цю інформацію, щоб розуміти, як могла б змінитися динаміка ринку відносно минулих днів.

Рисунок 2.4 – Частота зміни найнижчої ціни S&P500

На рисунку 2.5 можемо бачити графік зміни ціни на закриття. Це ціна, за якою закривається торгівля на S&P 500 кожного торгового дня. Це можливо найбільш важливий показник для багатьох інвесторів, оскільки він відображає консенсус між покупцями та продавцями на кінець дня. Ціна на закриття використовується для розрахунку відносної сили індексу та інших технічних показників, а також для створення графіків.

Рисунок 2.5 – Частота зміни ціни закриття S&P500

Нарешті на рисунку 2.6, що знаходиться нижче, можна спостерігати графік, який демонструє зміни обсягу торгів індексу. Цей показник вказує на кількість акцій, що були куплені та продані протягом торгового дня. Обсяг торгів є надзвичайно важливим, оскільки він відображає інтенсивність торговельної активності на ринку. Високий обсяг торгів може свідчити про високу ступінь участі трейдерів та інвесторів, а також про значну міру впевненості в поточних цінових рівнях. Трейдери та інвестори широко використовують цей показник для оцінки ліквідності ринку і виявлення потенційних цінових трендів. Інформація, яку надає графік обсягу торгів, може бути надзвичайно корисною при прийнятті рішень щодо торгівлі та інвестування. Крім того, аналіз обсягу торгів може допомогти виявити ключові моменти, такі як повільність руху цін або збільшення волатильності, що може бути зв'язано зі змінами в інвестиційних стратегіях. Дані про обсяг торгів

також можуть бути використані для порівняння з попередніми періодами та встановлення трендів, що можуть вказувати на майбутні перспективи ринку.

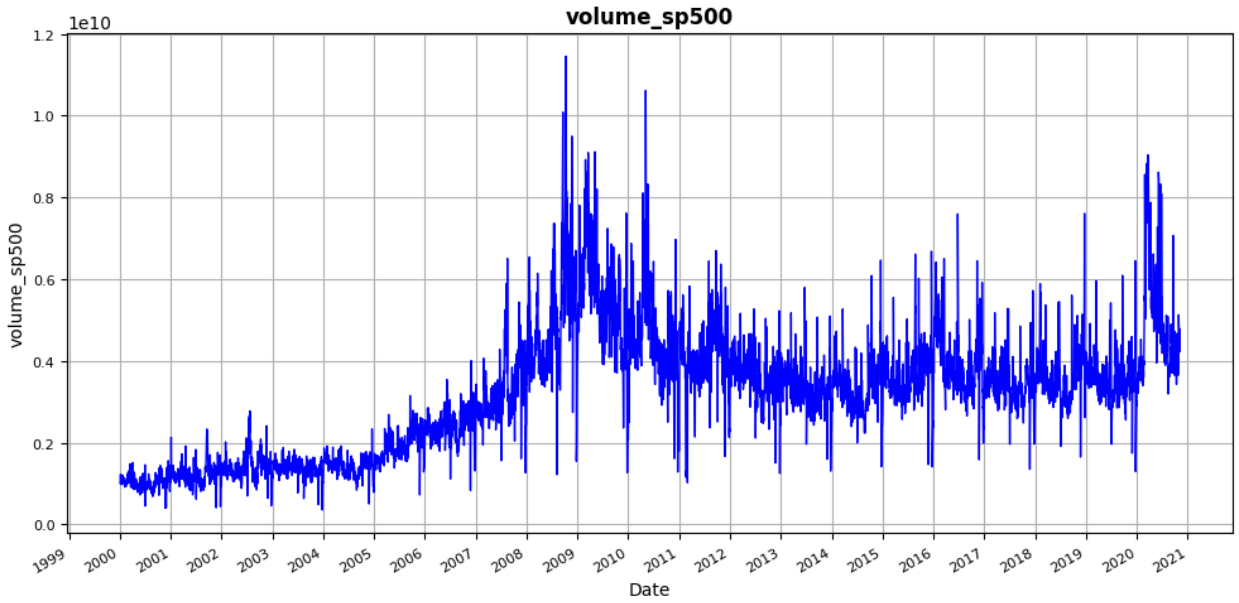

Рисунок 2.6 – Частота зміни обсягу торгів S&P500

– GDP US DAILY – нормалізований датасет який у собі містить приблизне значення ВВП США на кожен день. Спочатку сам датасет мав тільки значення за роки, але за допомогою пакету Pandas та функціям інтрополяції я зміг нормалізувати його до стану на кожен день, нормалізацію можна побачити на рисунку 2.8. На рисунку 2.7 показано як датасет містить інформацію про дату та приблизне значення ВВП у мільярдах;

DATE	GDP
2000-01-01	10002.179000
2000-01-02	10004.877253
2000-01-03	10007.575505
2000-01-04	10010.273758
2000-01-05	10012.972011

Рисунок 2.7 – Приклад ілюстрації даних датасету GDP US DAILY

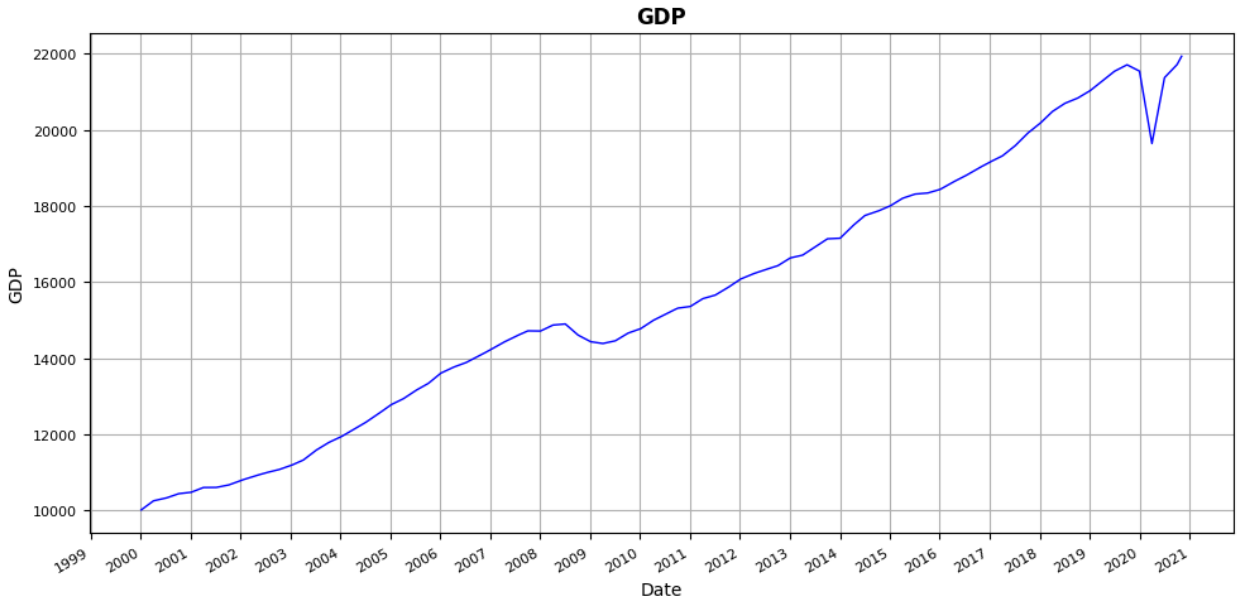

Рисунок 2.8 – Зміна ВВП у мільярдах протягом часу

– INFLATION US – нормалізований датасет який у собі містить відсоток інфляції у США на стан кожного дня. Спочатку сам датасет мав тільки значення за деякі проміжки часу, але після його перетворення, я зміг залучити інформацію за кожен день, це можна побачити на рисунку 2.10 на якому проілюстровано ця зміна. На рисунку 2.9 показано як виглядає нормалізований датасет з приблизним значенням інфляції;

DATE	inflation
2000-01-01	3.376857
2000-01-02	3.375353
2000-01-03	3.373848
2000-01-04	3.372343
2000-01-05	3.370839

Рисунок 2.9 – Приклад ілюстрації даних датасету INFLATION US

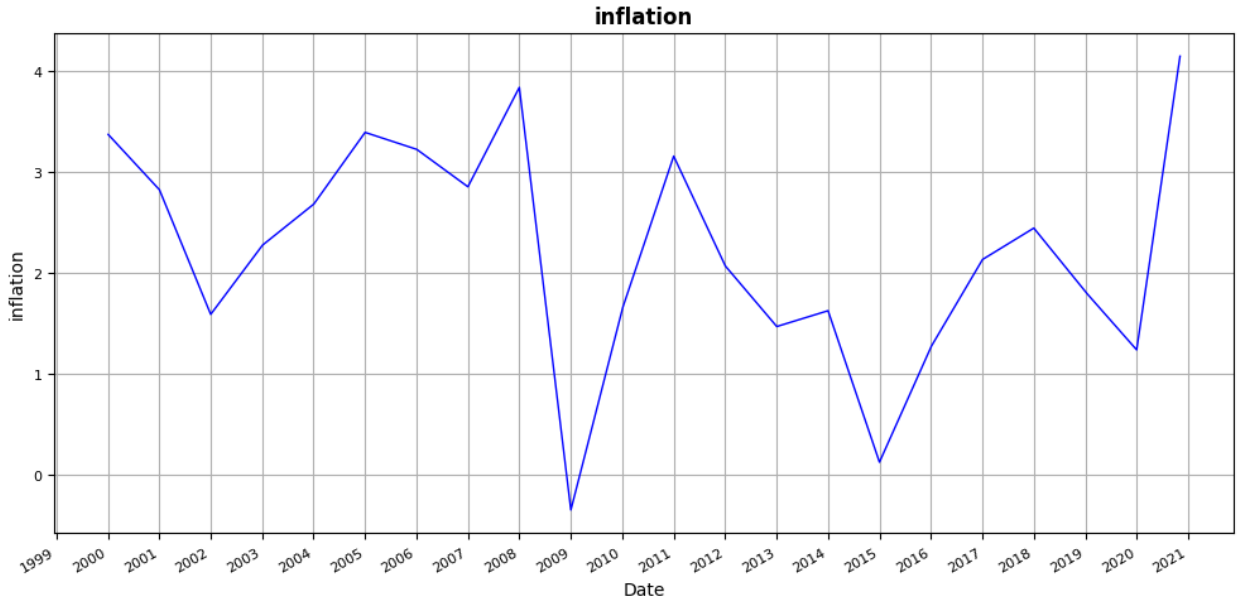

Рисунок 2.10 – Зміна рейтингу інфляції протягом часу

– UNEMPLOYEMENT US – нормалізований датасет який у собі містить відсоток безробіття у США на стан кожного дня. Спочатку сам датасет мав тільки значення за місяці, але після його перетворення, я зміг залучити інформацію за кожен день, це можна побачити на рисунку 2.12 на якому проілюстровано ця зміна. Як можна побачити на рисунку 2.11 вона також є приблизною, але не має відхил від точності менше ніж 1–2 відсотка;

Date	unemployement
2000-01-01	4.000000
2000-01-02	4.003226
2000-01-03	4.006452
2000-01-04	4.009677
2000-01-05	4.012903

Рисунок 2.11 – Приклад ілюстрації даних датасету UNEMPLOYEMENT US

Рисунок 2.12 – Зміна рейтингу безробіття протягом часу

– PRODUCTS – об’єднаний датасет який складається з історичних даних цін на золото, газ та нафту. Він має всі поля як і S&P500 датасет, це можна побачити на рисунку 2.13. Кожен з датасетів на золото, газ та нафту має періодичність в 1 день тому їх було дуже легко скомбінувати, це було зображено на рисунках 2.14, 2.15 та 2.16;

```
            open_gold  high_gold    low_gold  close_gold   volume_gold
Date
2000-01-04  289.500000      289.5  280.000000  283.700000  21621.000000  \
2000-01-05  283.700000      285.0  281.000000  282.100000  25448.000000
2000-01-06  281.600000      282.8  280.200000  282.400000  19055.000000
2000-01-07  282.500000      284.5  282.000000  282.900000  11266.000000
2000-01-08  282.466667      284.3  281.933333  282.833333  17711.666667

            open_gas  high_gas   low_gas  close_gas    volume_gas   open_oil
Date
2000-01-04     2.130  2.200000  2.130000   2.176000  30152.000000  23.900000  \
2000-01-05     2.180  2.200000  2.125000   2.168000  27946.000000  24.250000
2000-01-06     2.165  2.220000  2.135000   2.196000  29071.000000  23.550000
2000-01-07     2.195  2.230000  2.155000   2.173000  28455.000000  23.570000
2000-01-08     2.190  2.238333  2.158333   2.187333  28608.666667  23.393333

             high_oil    low_oil  close_oil    volume_oil
Date
2000-01-04  24.700000  23.890000  24.390000  32509.000000
2000-01-05  24.370000  23.700000  23.730000  30310.000000
2000-01-06  24.220000  23.350000  23.620000  44662.000000
2000-01-07  23.980000  23.050000  23.090000  34826.000000
2000-01-08  23.913333  23.046667  23.303333  32013.333333
```

Рисунок 2.13 – Приклад ілюстрації даних датасету PRODUCTS

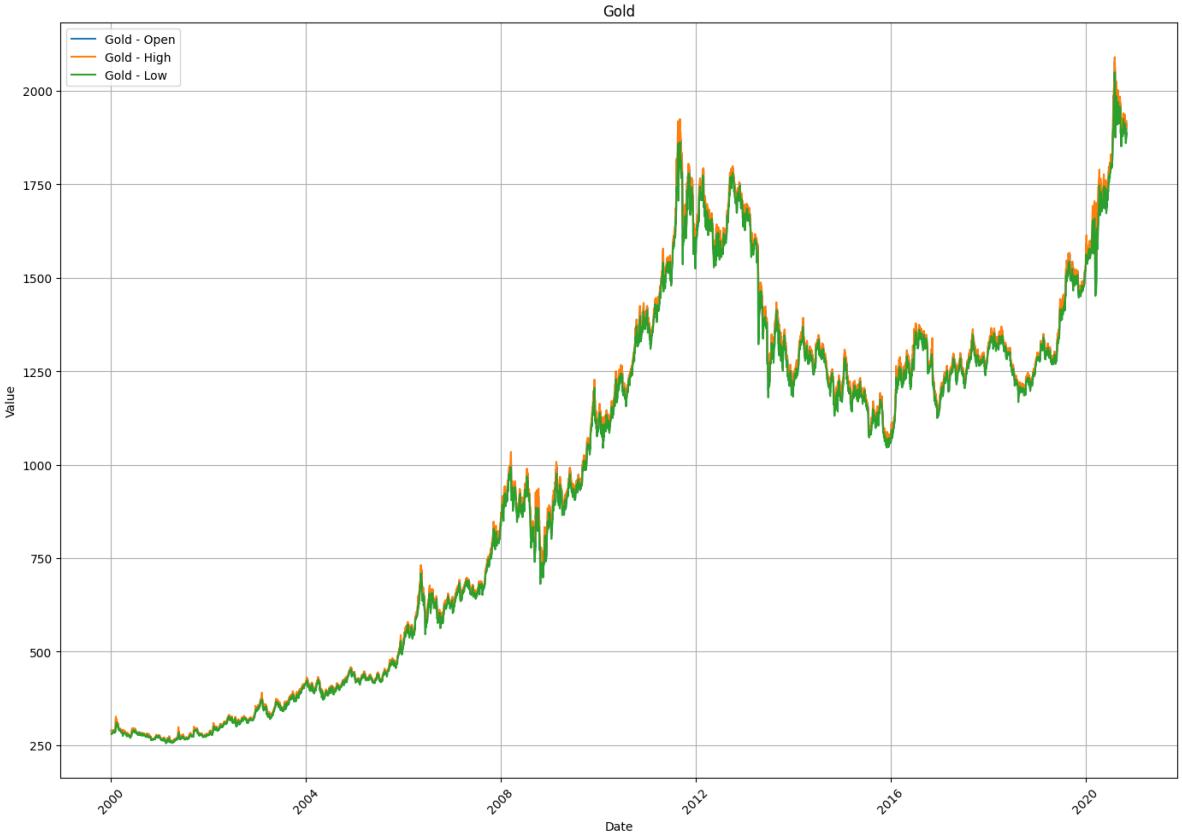

Рисунок 2.14 – Зміна показників датасету GOLD(PRODUCTS)

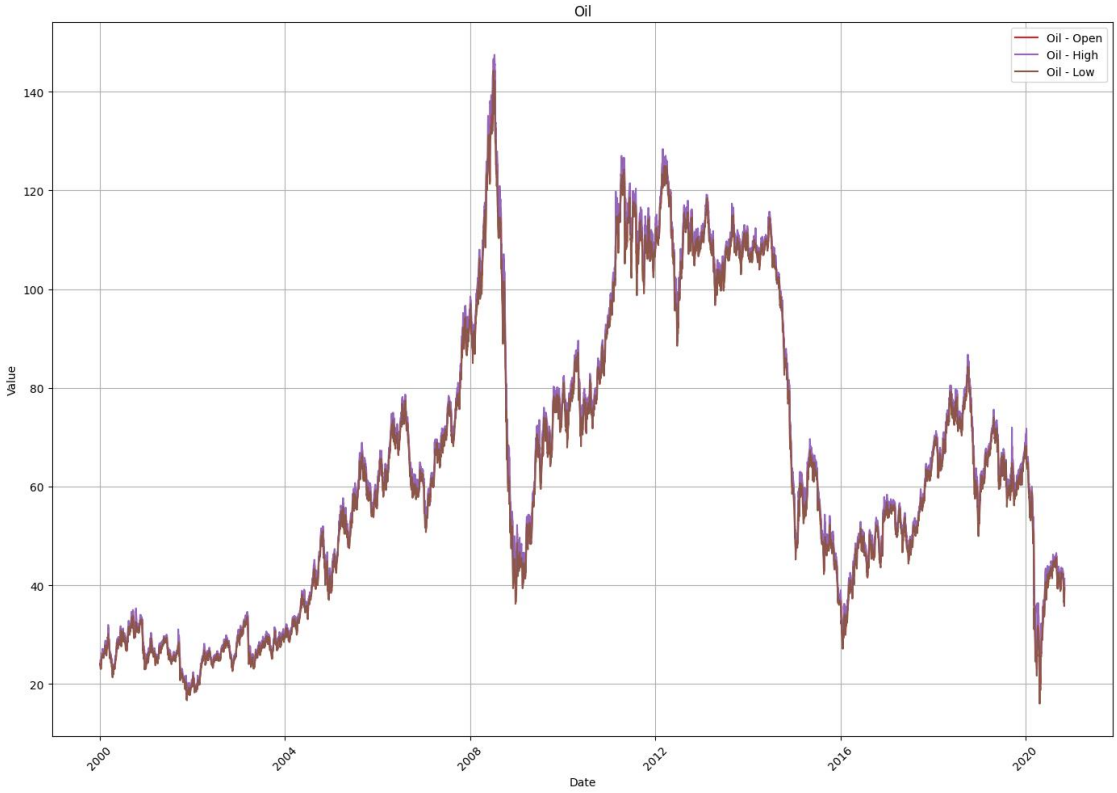

Рисунок 2.15 – Зміна показників датасету OIL(PRODUCTS)

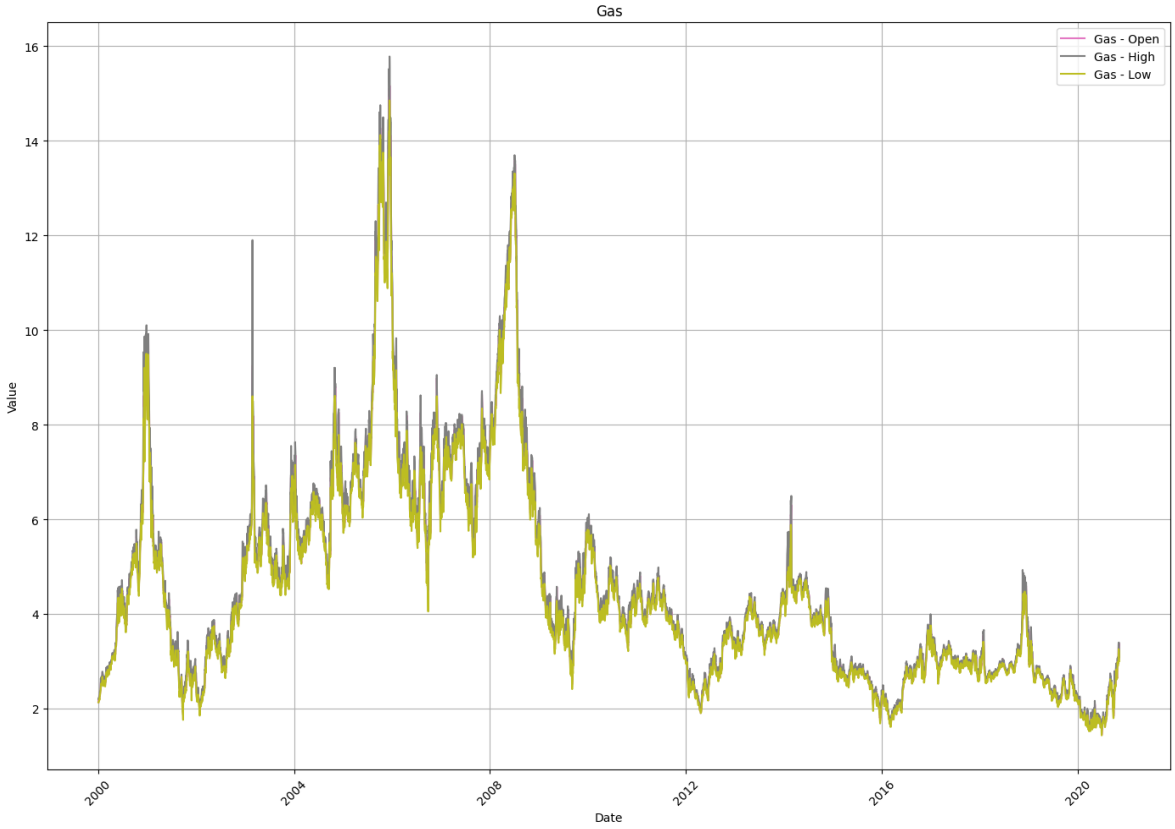

Рисунок 2.16 – Зміна показників датасету GAS(PRODUCTS)

– FED RATES – датасет який містить кредитні ставки Федерального Резерву США, це можна побачити на рисунку 2.17 на якому проілюстровано приблизні числа для ставки. Датасет має дані станом на кожний день, ілюстрацію графіку можна побачити на рисунку 2.18.

date	federal_rate
2000-01-01	3.99
2000-01-02	3.99
2000-01-03	5.43
2000-01-04	5.38
2000-01-05	5.41

Рисунок 2.17 – Приклад ілюстрації даних датасету FED RATES

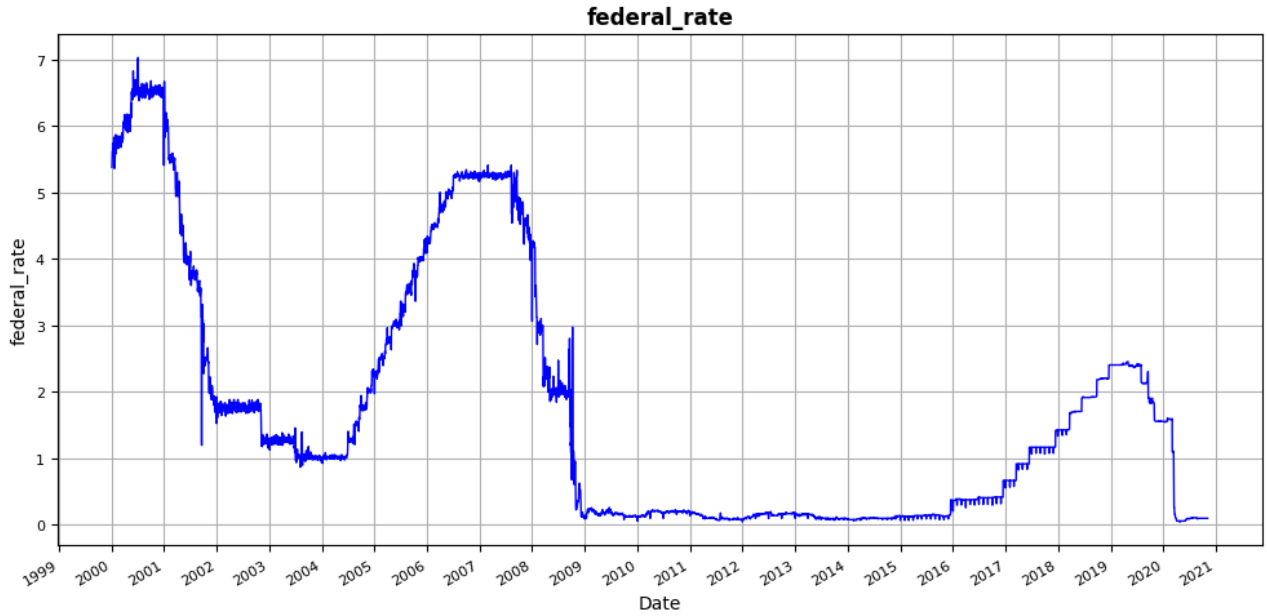

Рисунок 2.18 – Зміна кредитного рейтингу протягом часу

2.4.2 Сполучення та утворення фінального датасету

Після проведення процесу обробки та нормалізації всіх датасетів, що були зібрані у розділі "2.1.1 Збір даних", вдалося згрупувати їх згідно загальної характеристики дати. Це дозволило створити великий датасет, який використовується для тренування нейронної мережі та виявлення аномалій за допомогою алгоритму "Isolation Forest". Ілюстрацію та об'єднані параметри цього датасету можна побачити на рисунку 2.19, що наведений нижче. Зображення відображає важливу візуальну інформацію про датасет та його властивості, що допомагають зрозуміти його структуру та характеристики зібраних даних.

```
             open_sp500   high_sp500    low_sp500  close_sp500
2000-01-04  1455.219971  1455.219971  1397.430054  1399.420044  \
2000-01-05  1399.420044  1413.270020  1377.680054  1402.109985
2000-01-06  1402.109985  1411.900024  1392.099976  1403.449951
2000-01-07  1403.449951  1441.469971  1400.729980  1441.469971
2000-01-08  1416.123291  1449.099976  1414.309977  1446.846639

            adj_close_sp500  volume_sp500           GDP  inflation
2000-01-04      1399.420044  1.009000e+09  10010.273758   3.372343  \
2000-01-05      1402.109985  1.085500e+09  10012.972011   3.370839
2000-01-06      1403.449951  1.092300e+09  10015.670264   3.369334
2000-01-07      1441.469971  1.225200e+09  10018.368516   3.367830
2000-01-08      1446.846639  1.171733e+09  10021.066769   3.366325

            unemployement   open_gold  ...  high_gas   low_gas  close_gas
2000-01-04       4.009677  289.500000  ...  2.200000  2.130000   2.176000  \
2000-01-05       4.012903  283.700000  ...  2.200000  2.125000   2.168000
2000-01-06       4.016129  281.600000  ...  2.220000  2.135000   2.196000
2000-01-07       4.019355  282.500000  ...  2.230000  2.155000   2.173000
2000-01-08       4.022581  282.466667  ...  2.238333  2.158333   2.187333

              volume_gas   open_oil   high_oil    low_oil  close_oil
2000-01-04  30152.000000  23.900000  24.700000  23.890000  24.390000  \
2000-01-05  27946.000000  24.250000  24.370000  23.700000  23.730000
2000-01-06  29071.000000  23.550000  24.220000  23.350000  23.620000
2000-01-07  28455.000000  23.570000  23.980000  23.050000  23.090000
2000-01-08  28608.666667  23.393333  23.913333  23.046667  23.303333

              volume_oil  federal_rate
2000-01-04  32509.000000          5.38
2000-01-05  30310.000000          5.41
2000-01-06  44662.000000          5.54
2000-01-07  34826.000000          5.61
2000-01-08  32013.333333          5.61
```

Рисунок 2.19 – Приклад ілюстрації даних фінального датасету

2.4.2 Дослідження відношення між утвореними даними

Побудуємо матрицю відносин для того щоб показати відношення між даними, що буде свідчити, що дані які були підібрані для тренування мережі будуть мати сильну зв’язність.

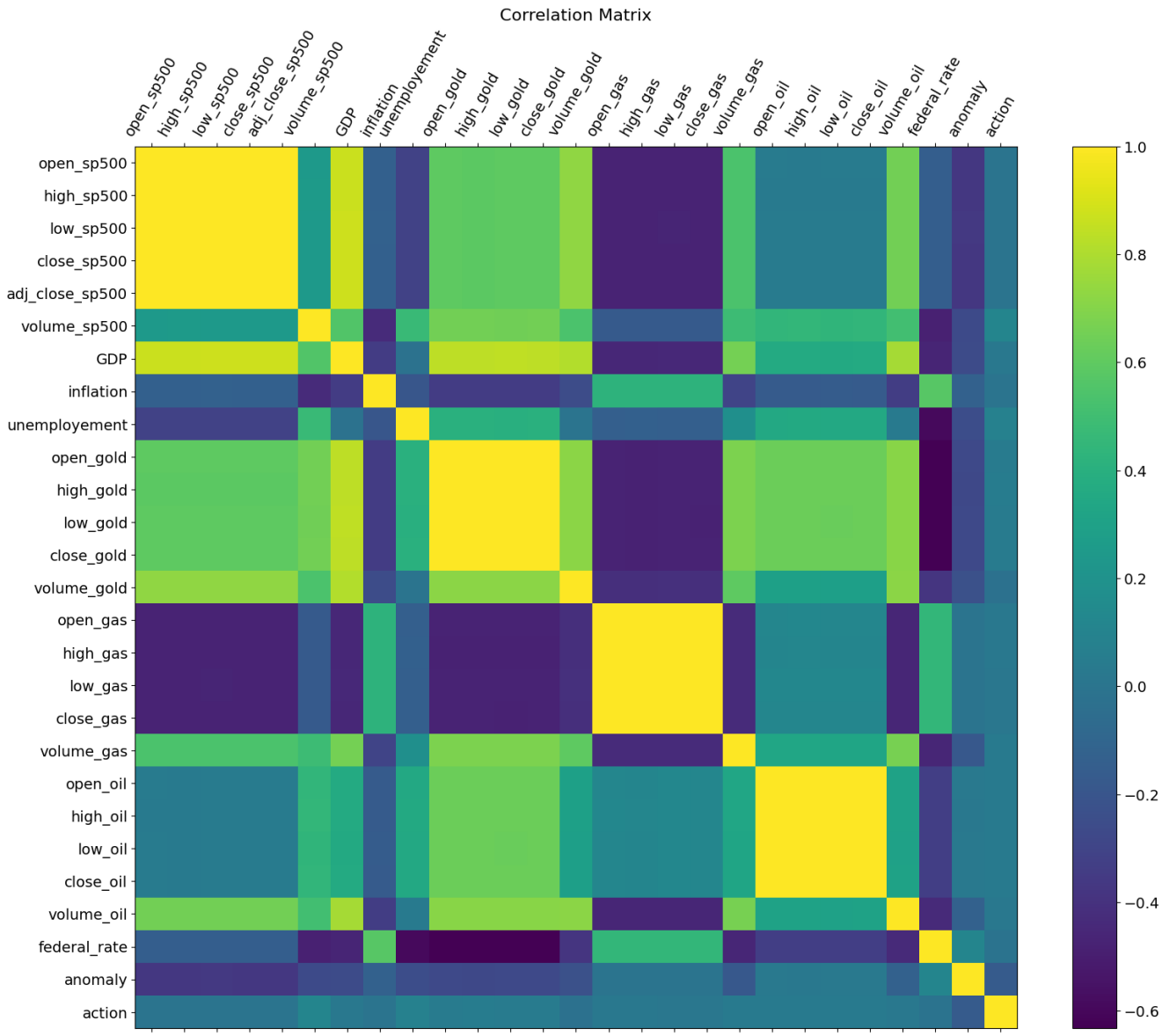

Рисунок 2.20 – Матриця відносин

Спостереження та аналіз:

– Судячи з високих кореляцій між open_sp500, high_sp500, low_sp500 та close_sp500, ці чотири показники дуже тісно пов'язані. Це логічно, оскільки вони всі представляють ціну акцій S&P 500 в різні моменти дня;

– обсяг торгів (volume_sp500) має позитивну кореляцію з цінами S&P 500, але вона не така сильна, як у випадку з іншими показниками цін;

– GDP має досить високу позитивну кореляцію з цінами на S&P 500, що свідчить про те, що з ростом економіки зростають і ціни на акції;

– Inflation та unemployment мають негативну кореляцію з цінами на акції S&P 500, що свідчить про те, що з ростом інфляції та безробіття ціни на акції, як правило, знижуються.

– Open_gold, high_gold, low_gold, close_gold мають позитивну кореляцію з цінами на S&P 500, але вона не така висока, як у випадку з GDP;

– ціни на газ (open_gas, high_gas, low_gas, close_gas) мають негативну кореляцію з цінами на S&P 500;

– ціни на нафту (open_oil, high_oil, low_oil, close_oil) мають дуже слабку позитивну кореляцію з цінами на S&P 500;

– Federal_rate має негативну кореляцію з цінами на S&P 500, що свідчить про те, що з підвищенням ставок Федеральної резервної системи ціни на акції, як правило, знижуються;

– Anomaly та action мають дуже слабку негативну кореляцію з цінами на S&P. Це логічно, оскільки anomaly та action набуті параметри під час нормалізації даних та не відносяться прямо до ціни на актив.

2.5 Вихідні дані

Видача розробленої моделі буде сигналом для торгівлі: купувати, продавати або утримуватися. Ці сигнали будуть генеруватися на основі висновків, зроблених моделлю на основі аналізу вхідних даних. Кожен сигнал буде відображати прогнозовану динаміку ринку відповідно до інтерпретації моделі виявлених аномалій та їх потенційного впливу на майбутні ціни.

Ці сигнали будуть використовуватися для визначення стратегії торгівлі. Наприклад, якщо модель прогнозує позитивну динаміку на ринку, видається сигнал "купувати", вказуючи на рекомендацію придбати активи. Навпаки, якщо прогнозується негативна динаміка, видається сигнал "продати", що вказує на потенційну вигоду від продажу або входження в короткі позиції. Якщо модель не виявляє ясних тенденцій, вона може видати сигнал "утриматися", що вказує на те, що краще утриматися від будь–яких операцій на ринку в даному моменті.

Важливо відзначити, що видача сигналів не означає автоматичного виконання торгівельних операцій. Вони служать лише рекомендаціями, які повинен розглянути фахівець з торгівлі або інвестор, оцінюючи їх в контексті загальної стратегії та інших важливих факторів, таких як управління ризиками, ліквідність і т.д.

У використанні моделі важливо зазначити, що, хоча було використовано нейронні мережі для прогнозування ринкових рухів, це не гарантує 100% точності. Модель, як і будь–який інший інструмент прогнозування, має свої обмеження та потенційні погрішності. Основними ризиками є перенавчання, коли модель стає занадто адаптованою до тренувального набору даних та не може ефективно узагальнювати на нових даних, а також шум у вхідних даних, який може викривити висновки моделі.

Тому, хоча модель може допомогти в ідентифікації потенційних торгових можливостей, важливо використовувати її в рамках збалансованої інвестиційної стратегії, яка також включає аналіз ризиків, управління капіталом та диверсифікацію портфелю. Крім того, важливо постійно оцінювати ефективність моделі, адаптуючи її та проводячи періодичне навчання на нових даних.

Ця система, що базується на штучному інтелекті, має потенціал стати важливим інструментом для інвесторів та трейдерів, що допомагає їм розуміти складність фінансових ринків та знаходити потенційно прибуткові можливості. Однак, як і в усіх інвестиційних стратегіях, вона повинна використовуватися з обережністю та в поєднанні з іншими інструментами аналізу та управління ризиками.

2.6 Структура масивів інформації

Масиви інформації в розробленій системі відіграє критичну роль в зберіганні та обробці даних. Вони дозволяють нам зберігати, обробляти та аналізувати великі обсяги фінансових даних.

Масив цін активів: Цей масив зберігає історичні дані про ціни активів. Кожен елемент масиву відповідає конкретній даті та активу, і включає значення для відкриття, закриття, максимуму, мінімуму і обсягу торгів.

Масив економічних показників: Цей масив зберігає історичні дані про ключові економічні показники, такі як ВВП, інфляція, безробіття та інше. Кожен елемент масиву відповідає конкретній даті і країні, і містить значення для кожного показника.

Масив даних про товари: Цей масив зберігає історичні дані про ціни на ключові товари, такі як золото, нафта, газ та інше. Кожен елемент масиву відповідає конкретній даті та товару, і включає ціну та обсяг торгів.

Масив облігацій та процентних ставок: Цей масив зберігає історичні дані про облігації та процентні ставки. Кожен елемент масиву відповідає конкретній даті, країні та типу облігації або ставки, і включає значення ставки.

Ці масиви з'єднані за допомогою ключів дати та активу, що дозволяє нам виконувати складні запити та аналізувати взаємозв'язок між різними видами даних.

Масиви інформації є важливою частиною структури даних розробленої системи, що дозволяє нам максимально ефективно та точно використовувати доступні нам інформаційні ресурси. У зв'язку з цим, вони були обрані як основний спосіб представлення даних у системі.

Висновок до розділу

У цьому розділі було розглянуто різні аспекти розробки системи прогнозування для торгівлі на фінансових ринках, заснованої на штучному інтелекті. Висвітлено ключові етапи, включаючи вибір даних для тренування та тестування моделі, а також важливість вибору правильного алгоритму машинного навчання.

Обговорили важливість ретельного аналізу вхідних даних, що включає історичні ціни, макроекономічні показники, дані про товари та облігації та процентні ставки. Також було розглянуто використання багатофакторної моделі для визначення аномалій в ринкових патернах та було зазначено, що важливо

розглядати механізми пошуку аномалій та моделі "LSTM" в контексті ринкових умов. Також було детально розглянули структуру даних, яка включає в себе історичні дані про ціни активів, ключові економічні показники, дані про товари та облігації та процентні ставки. Ці дані слугуватимуть вхідними даними для моделей. Також було сформовано фінальний датасет.

Загалом, пропозиція щодо використання штучного інтелекту в інвестиційному процесі показує обіцяючи перспективи. Однак, важливо розуміти, що незважаючи на велику потужність і гнучкість цього підходу, він не зменшує необхідність аналізу, осмислення та управління ризиками.

3 МЕТОДИ РОЗРОБКИ

3.1 Змістовна постановка задачі

Метою цього проекту є розробка системи, яка використовує алгоритми машинного навчання для виявлення аномалій у фінансових ринках, а також для прийняття інвестиційних рішень на основі цих аномалій.

Система буде використовувати набір різноманітних даних, включаючи історичні дані про ціни активів, економічні показники, дані про товари, дані про облігації та процентні ставки. Ці дані будуть використані для алгоритму для виявлення аномалій в цінах активів та навчання нейронної мережі для прийняття інвестиційних рішень на основі виявлених аномалій.

Алгоритм, який імплементує Isolation Forest, буде вивчати дані для виявлення аномалій в цінах активів. Ці аномалії можуть вказувати на незвичайні ринкові умови, які можуть впливати на ціну активу.

Нейронна мережа, яка використовує алгоритм LSTM, буде вивчати дані для прийняття інвестиційних рішень. Це може включати рішення про купівлю, продаж або втримання активу на основі виявлених аномалій та інших важливих факторів.

Ця система має потенціал збільшити ефективність та прибутковість інвестиційних стратегій, використовуючи передові технології машинного навчання для аналізу великої кількості даних та виявлення складних шаблонів та відношень, які можуть бути непомітними для людського ока.

3.2 Математична постановка задачі

Дані: Нехай D = {d1, d2, ..., dn} представляє набір даних, де di представляє i–тий запис в наборі даних. Кожен запис складається з вектору атрибутів A = {a1, a2, ..., am} та відповідного значення ціни активу.

Аномалія: Аномалія – це відхилення від норми. В контексті цього проекту аномалія відноситься до значення ціни активу, яке відхиляється від того, що очікується на основі історичних даних та інших відомих факторів.

Алгоритм виявлення аномалій: Це функція f: D –> {0,1}, яка приймає запис d з D та повертає 1, якщо d вважається аномалією, і 0 в іншому випадку.

Інвестиційне рішення: Це рішення про купівлю, продаж або втримання активу. Це може бути представлено як функція g: D –> {buy, sell, hold}.

Задача цього проекту може бути сформульована як наступна оптимізаційна проблема:

Знайти алгоритм виявлення аномалій f та алгоритм прийняття інвестиційних рішень g, які максимізують прибуток з інвестицій на основі даних D.

Це може бути вирішено за допомогою процедури навчання, яка змінює параметри алгоритмів f та g для мінімізації втрат на наборі тренувальних даних, а потім використовує ці алгоритми для прийняття інвестиційних рішень на основі нових, раніше невідомих даних.

3.3 Обґрунтування методів розв’язання

В сучасному світі є безліч методів для виявлення аномалій та прогнозування часових рядів, але не всі з них ефективні у конкретних обставинах цього проекту. Основні вимоги до методів для цього проекту включають здатність ефективно працювати з великими об'ємами даних, враховувати часові залежності, а також бути гнучкими щодо нелінійності та змін у законах розподілу.

Алгоритм Isolation Forest був обраний, тому що він відрізняється тим, що може виявляти аномалії у великих наборах даних, причому ефективно працює навіть при невеликій кількості аномальних записів.

Для Isolation Forest вектори ознак складаються з таких атрибутів:

- Ціна відкриття;
- найвища ціна;
- найнижча ціна;
- ціна закриття;
- обсяг торгів;
- економічні показники (ВВП, інфляція, безробіття і т.д.);

– ціни на ключові товари (золото, нафта, газ і т.д.);
– Облігації та процентні ставки.

Isolation Forest використовує алгоритм дерева рішень для визначення, які точки є аномальними на основі відстані від інших точок у просторі ознак.

Головний механізм цього проєкта – мережа, що використовується в цій системі, – це LSTM (Long Short–Term Memory), тип рекурентної нейронної мережі (RNN), який здатний запам'ятовувати великі послідовності даних, що робить його ідеально підходящим для аналізу часових рядів. Він підходить для задач прогнозування цін на активи, оскільки такі задачі часто включають складні часові залежності, які важко моделювати за допомогою традиційних методів. Крім того, LSTM відомі своєю гнучкістю щодо нелінійності та змін у законах розподілу.

Кожна послідовність для LSTM включає наступні атрибути:

– Ціна відкриття;
– найвища ціна;
– найнижча ціна;
– ціна закриття;
– обсяг торгів;
– економічні показники (ВВП, інфляція, безробіття і т.д.);
– ціни на ключові товари (золото, нафта, газ і т.д.);
– Облігації та процентні ставки.

Структура LSTM включає вхідний шар, декілька прихованих шарів і вихідний шар. Кожен прихований шар складається з LSTM–одиниць, які включають забувальний вентиль, вхідний вентиль та вентиль виходу.

3.4 Опис нейронної мережі

Кількість шарів в нейронній мережі і кількість нейронів в кожному шарі – це гіперпараметри, які вибираються в процесі тюнінгу моделі. На жаль, не існує конкретних правил для вибору цих гіперпараметрів, і вони можуть сильно залежати від конкретної задачі, яка була поставлена.

Було отримано 25 ознак. Зауважимо, що це не обов'язково визначає кількість шарів або нейронів в розробленій мережі. Число ознак впливає на розмір вхідного шару вашої мережі, але кількість прихованих шарів і нейронів в кожному шарі – це те, що повинно бути вирішено у процесі розробки.

Типова, можна почати з одного або двох прихованих шарів LSTM і потім експериментувати з додаванням більше. Кількість нейронів в шарах також може варіюватися. Зазвичай, більше нейронів і шарів дає моделі більше "ємності".

– Архітектура моделі включає такі рішення, як:

– Кількість шарів у вашій моделі.

– Кількість нейронів (або "одиниць") в кожному шарі.

– Типи шарів, які ви використовуєте (наприклад, LSTM, Dense тощо).

– Гіперпараметри моделі – це параметри, які встановлюються перед тренуванням моделі.

– Для LSTM мережи гіперпараметри включають такі рішення, як:

– Розмір пакету (batch size), який використовується під час тренування.

– Кількість епох (epochs), які використовуються для тренування.

– Функція втрат (loss function), яка використовується для оцінки ваг моделі.

– Оптимізатор (optimizer), який використовується для оновлення ваг моделі.

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 bidirectional_2 (Bidirectio  (None, 10, 512)          579584
 nal)

 dropout_2 (Dropout)         (None, 10, 512)           0

 bidirectional_3 (Bidirectio  (None, 256)              656384
 nal)

 dropout_3 (Dropout)         (None, 256)               0

 dense_1 (Dense)             (None, 3)                 771

=================================================================
Total params: 1,236,739
Trainable params: 1,236,739
Non-trainable params: 0
```

Рисунок 3.1 – Нейрона мережа LSTM

На рисунку 3.1 зображено структуру розробленої нейронної мережі, найцікавішим фактором тут являється двонаправлений LSTM шар, так як це не стандартний підхід розробки LSTM моделі, я наведу декілька причин щодо його застосування.

– Двонаправлені LSTM (BiLSTM) використовують інформацію з майбутніх точок даних, а також з минулих точок, що допомагає покращити точність передбачення;

– у контексті фінансових даних, де важливі зміни тенденцій, BiLSTM може виявити складні залежності, які пропускаються одно направленими LSTM;

– BiLSTM моделі можуть краще впоратися з проблемами затухання градієнта, що є типовою проблемою для LSTM;

– більш висока точність BiLSTM може бути критичною для задач інвестиційних рішень, де помилка може коштувати великої суми грошей;

– хоча BiLSTM вимагає більше обчислювальних ресурсів, ніж одно направлені LSTM, вони все ще доволі ефективні з точки зору обчислень і можуть бути навчені за розумний час, використовуючи сучасне апаратне забезпечення;

– BiLSTM здатні виявляти і використовувати довготривалі залежності в даних, які можуть бути важливими для вашої задачі;

– у випадку нерівномірно розподілених даних, BiLSTM може виявити ці складності і надати кращі результати;

– за допомогою двонаправленої структури, LSTM здатні краще моделювати сезонність і циклічність в даних, що є загальною особливістю фінансових часових рядів;

– оскільки BiLSTM використовують інформацію з майбутнього контексту, вони можуть бути особливо корисними в ситуаціях, де майбутній контекст необхідний для розуміння поточного стану, що є характерним для багатьох задач фінансового прогнозування;

– нарешті, BiLSTM може бути використаний для побудови більш складних моделей, таких як умовні випадкові поля (CRF), що можуть виявити ще більш складні залежності в даних.

Висновки до розділу

У цьому розділі було детально розглянуто постановку задачі прогнозування цін на активи та опрацювали її математичне формулювання. Було зроблено акцент на необхідності побудови відповідного набору даних для тренування та тестування моделі.

Обґрунтовано вибір методів розв'язання, зокрема, моделі LSTM, яка враховує інформацію з майбутнього контексту, тим самим покращуючи точність прогнозування. Зазначено важливість правильного налаштування гіперпараметрів цієї моделі, включаючи кількість прихованих шарів та нейронів в кожному шарі, а також вибір відповідного вихідного шару моделі.

Детально описано структуру нейронної мережі, в тому числі архітектуру моделі, методи активації та оптимізації. Було розглянуто, як вибір відповідної архітектури і методів оптимізації може вплинути на точність прогнозування моделі.

4 ДОСЛІДЖЕННЯ ТА ЕКСПЕРИМЕНТИ

4.1 Розробка нейронної мережі

У цьому розділі було розглянуто процес навчання моделі. Основні кроки включають: підготовку даних, створення моделі, її компіляцію та тренування.

Почнемо з підготовки даних. Дані перетворюються за допомогою LabelEncoder, який перетворює нечислові мітки на числові. Потім ділимо дані на тренувальний та тестовий набори. Це важливий крок, оскільки він дозволяє нам перевірити ефективність моделі на нових даних, які вона не бачила під час тренування.

Далі використовується функція з параметром time_steps для створення послідовностей з декількох послідовних записів. Після цього було застосовано one–hot encoding до міток y_train і y_test, що є необхідним для класифікаційних моделей.

Також було використано MinMaxScaler для масштабування даних в діапазоні від 0 до 1. Масштабування даних є важливим кроком перед тренуванням багатьох моделей машинного навчання, оскільки воно дозволяє моделі ефективніше навчатися.

Створюючи модель, було використано архітектуру з двонаправленими LSTM шарами. Використання двонаправленого LSTM дозволяє моделі "бачити" інформацію в обох напрямках часової шкали, що може покращити її здатність вивчити складні шаблони.

Також було додано шари Dropout для регуляризації, які допомагають зберегти модель від перенавчання.

Компіляція моделі включає визначення функції втрат та оптимізатора, які використовуються під час тренування.

Що стосується тренування, було використано метод fit моделі, передаючи йому вхідні дані та мітки, кількість епох, розмір партії, відсоток даних для валідації та вказівку, щоб дані не перемішувались.

Після тренування моделі було оцінено її на тестових даних та були зроблені прогнози. Було використано метод predict для отримання прогнозів моделі на

тестових даних, а потім конвертовано ці прогнози та дійсні мітки в масиви класів, використовуючи np.argmax.

Після тренування, як можна побачити на рисунку 4.1 отримуємо точність 62%.

```
Test Loss:  0.8871793150901794
Test Accuracy:  0.6245059370994568
8/8 [==============================] -
```

Рисунок 4.1 – Втрати та точність

4.2 Дослідження мережі з різними параметрами

У цьому пункті було розглянуто результати тренування нейронної мережі з використанням різних параметрів як у тренувальних даних так і в параметрах самої моделі. С початку було припущено, що треба обрати найкраще значення для параметру time_steps для того щоб з’ясувати яке значення біль підходить для LSTM моделі. Після цього було проведено експеримент с кожною групою параметрів:

– кількості шарів LSTM, кількості memory units у кожному з шарів, рівень droupout параметру

– гіперпараметри (‘епохи’, ‘learning_rate’, ‘batch_size’, ‘validation_split’)

За результатами експериментів було обрано найкращу комбінацію параметрів для максимізації результату. Ефективність зміни параметрів будемо виміряти за допомогою:

– Точності тренування: це відсоток правильних прогнозів, які модель зробила на тренувальних даних;

– точності тестування: це відсоток правильних прогнозів, які модель зробила на тестових даних;

– часу навчання: час, необхідний для навчання моделі при заданому параметрі time_steps;

– відношенню тестової точності до тренувальної;

– відношенню функції втрат на тестовому і тренувальному наборах даних.

4.2.1 Аналіз мережі зі зміною параметра time_steps

Параметр time_steps відіграє важливу роль у первинному поділі даних для їх класифікації. time_steps використовується для визначення довжини вхідної послідовності для LSTM моделі. В контексті задачі прогнозування часових рядів, time_steps представляє кількість часових одиниць (в даній задачі, днів), що беруться для прогнозування наступного значення.

Зміна time_steps може мати такий вплив на модель:

– Зменшення time_steps – якщо time_steps зменшується, модель отримує менше контексту з минулого для прогнозування майбутнього значення. Це може спростити модель, але також може призвести до погіршення якості прогнозування, якщо в минулих даних містяться важливі паттерни, які пропускаються з меншими time_steps;

– збільшення time_steps – збільшення time_steps надає моделі більше контексту з минулого для прогнозування майбутнього значення. Це може покращити якість прогнозування, особливо якщо в минулих даних містяться важливі паттерни, які з'являються протягом довгих періодів часу. Однак, збільшення time_steps також може зробити модель складнішою, що може призвести до підвищення ризику перенавчання, особливо якщо доступні дані обмежені.

Було проведено кілька експериментів для того щоб знайти бажаний варіант. Для чесності експерименту бралися історичні дані за 4 роки з 2017 по 2021, 2 шарову LSTM модель з 256 і 128 memory units для запам’ятовування та 2 DROPOUT шарами з кофіцієнтом у 0.2 також візьмемо 10 ‘епох’ за стандарт.

Припустимо, що значення time_steps дорівнює 2. Після експерименти з цим значенням було отримано такі результати: точність тренування – 50.2%, точність тестування – 31.3%, відношення функції втрат на тестовому і тренувальному наборах даних – рисунок 4.2, відношенню тестової точності до тренувальної – рисунок 4.3, часу навчання – 7.4 секунди.

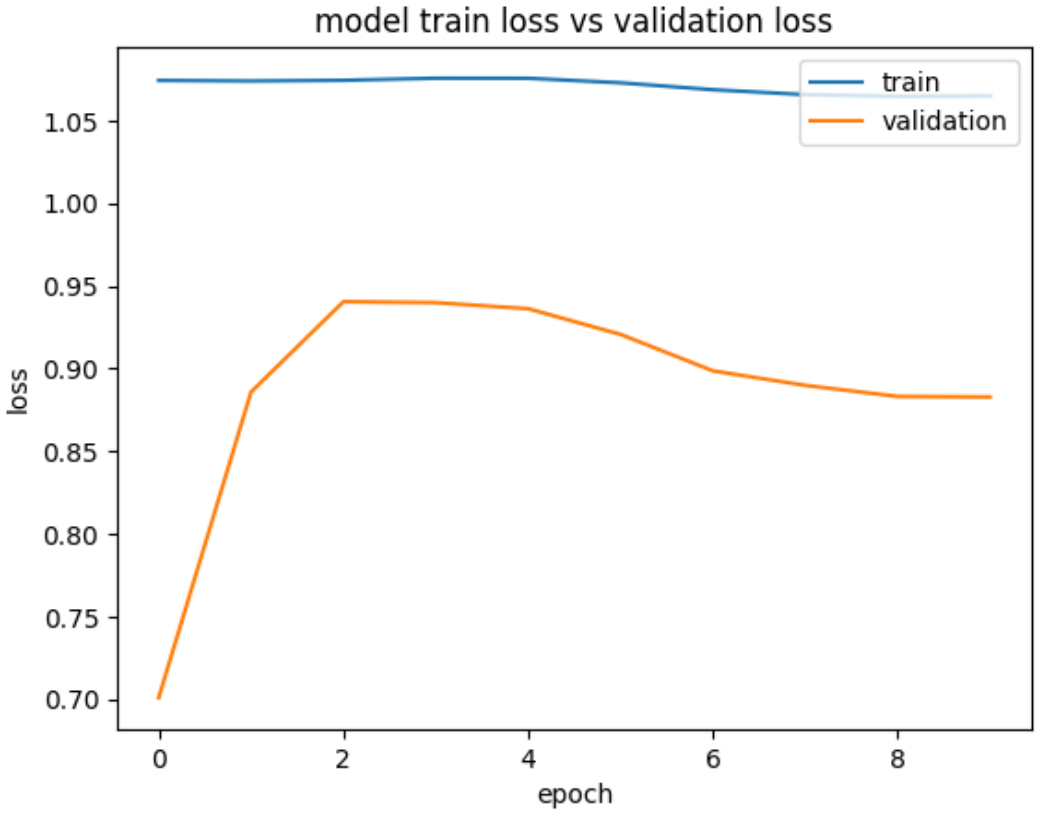

Рисунок 4.2 – Відношення функції втрат (time_steps=2)

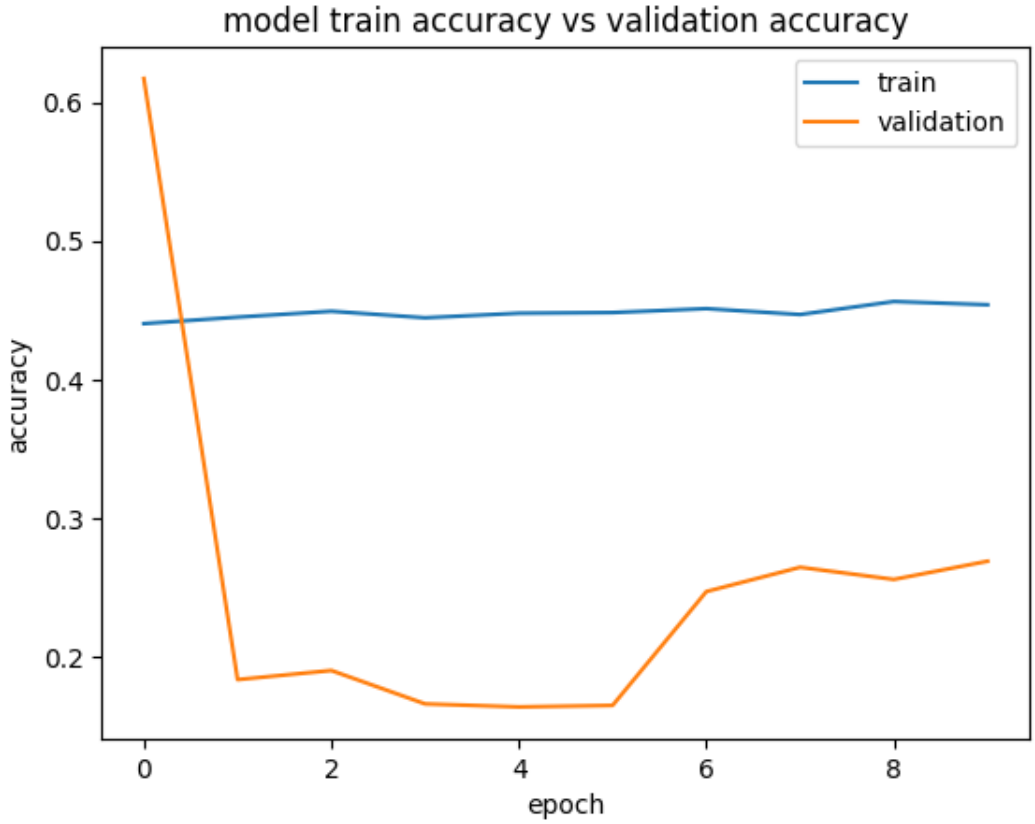

Рисунок 4.3 – Відношення точності (time_steps=2)

Якщо значення time_steps дорівнює 5. Після експерименти з цим значенням було отримано такі результати: точність тренування – 40.2%, точність тестування – 39%, відношення функції втрат на тестовому і тренувальному наборах даних – рисунок 4.4, відношенню тестової точності до тренувальної – рисунок 4.5, часу навчання – 9 секунд.

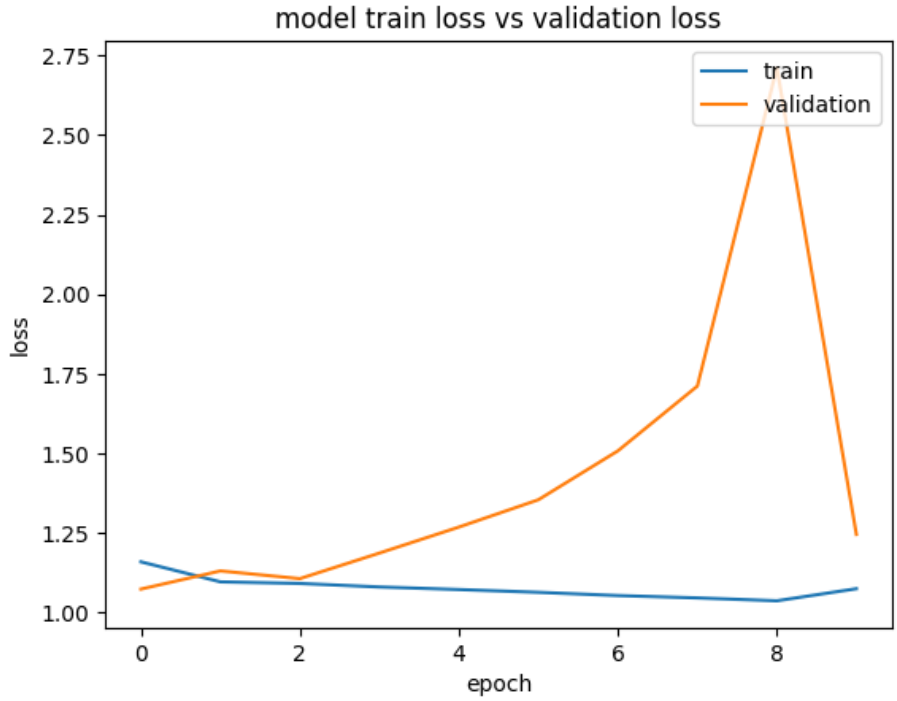

Рисунок 4.4 – Відношення функції втрат (time_steps=5)

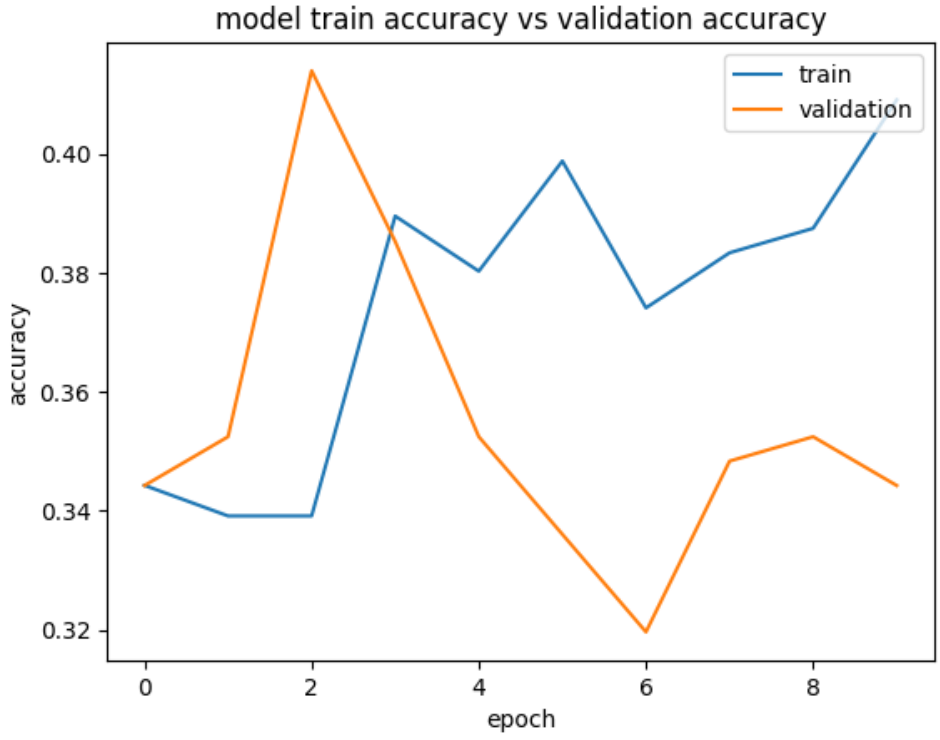

Рисунок 4.5 – Відношення точності (time_steps=5)

Якщо значення time_steps дорівнює 15. Після експерименту з цим значенням було отримано такі результати: точність тренування – 81%, точність тестування – 38%, відношення функції втрат на тестовому і тренувальному наборах даних – рисунок 4.6, відношенню тестової точності до тренувальної – рисунок 4.7, часу навчання – 7.9 секунд.

Рисунок 4.6 – Відношення функції втрат (time_steps=15)

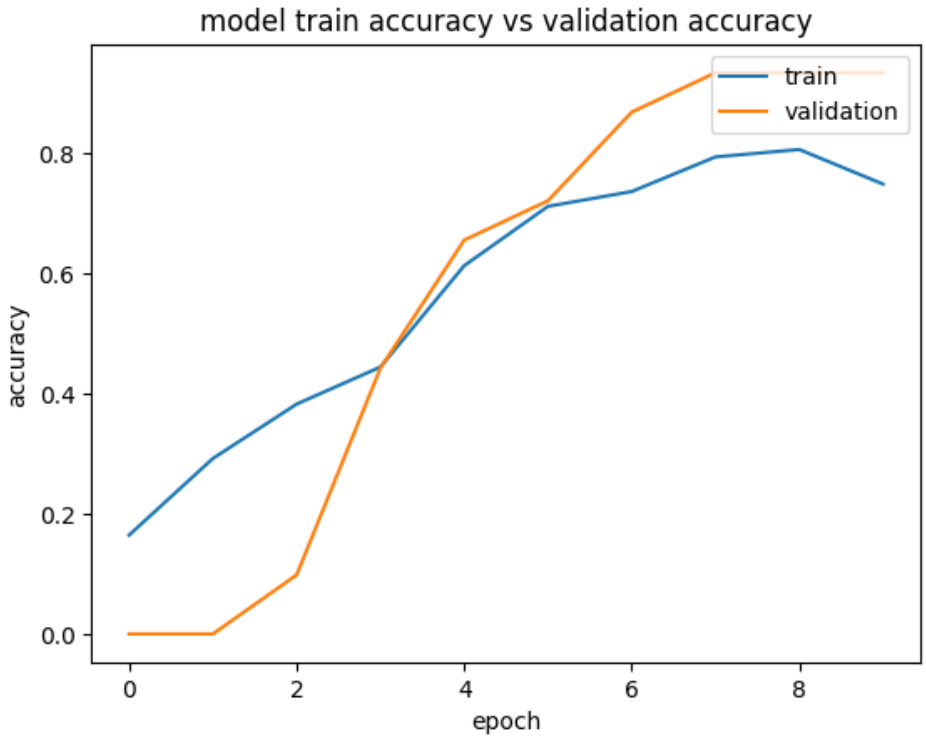

Рисунок 4.7 – Відношення точності (time_steps=15)

Якщо значення time_steps дорівнює 30. Після експерименту з цим значенням було отримано такі результати: точність тренування – 89%, точність тестування – 52%, відношення функції втрат на тестовому і тренувальному наборах даних – рисунок 4.8, відношенню тестової точності до тренувальної – рисунок 4.9, часу навчання – 7.9 секунд.

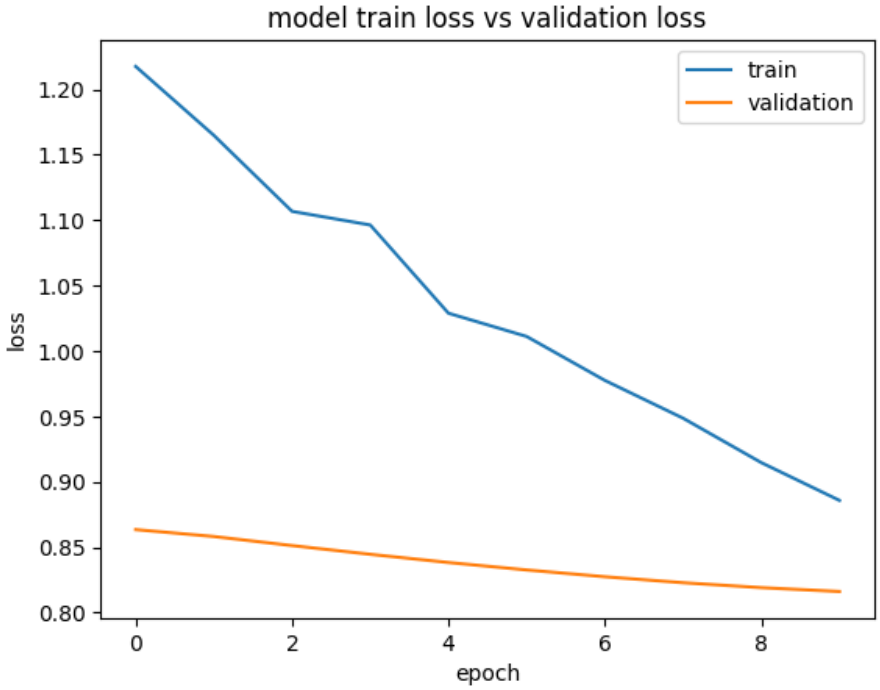

Рисунок 4.8 – Відношення функції втрат (time_steps=30)

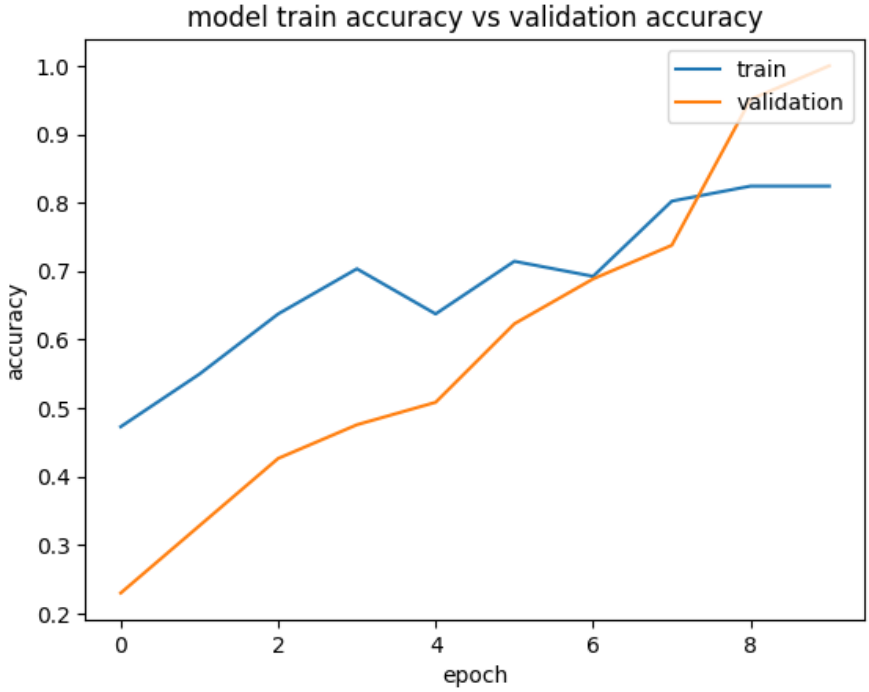

Рисунок 4.9 – Відношення точності (time_steps=30)

Експерименти показали, що модель працює ефективніше на коли кількість елементів в послідовності знаходиться в діапазоні з 15 до 30 днів. Якщо буде проведено експеримент зі значенням time_steps більше ніж 30, то в наявності не буде достатньо даних для тренування моделі, що буде негативно позначиться на результатах під час тренування та тестування.

Отже, було вирішено, брати значення кількості елементів у послідовності в проміжку від 15 до 30, як за еталоне значення.

4.2.1 Аналіз мережі зі зміною кількості шарів LSTM, кількості memory units у кожному з шарів, рівень droupout параметру

Так як коли значення параметру time_steps знаходиться між 15 та 30, то кількість елементів у датасеті зменшується, тому важливо правильно підібрати параметри для кількості шарів, кількості запам’ятовуючих комірок та рівень фільтрації. Важливо зазначити, що droupout шар буде кожного разу йти за LSTM шаром для фільтрації результатів.

Було проведено експерименти з одним та двома послідовними комбінацією шарів LSTM та droupout. Кожний експеримент був проведений зі зміною кількості memory units, а також рівнем droupout для знаходження еталонного значення.

Припустимо, що модель має тільки один droupout шар з рівнем фільтрації 0.2 та LSTM шар з мінімально допустимою кількістю memory units – 32, так як одна послідовність має від 15 до 30 елементів, мати менше просто призведе до недотренування. Після експерименти з такою конфігурацією було отримано такі результати: точність тренування – 67%, точність тестування – 62%, відношення функції втрат на тестовому і тренувальному наборах даних – рисунок 4.10, відношенню тестової точності до тренувальної – рисунок 4.11, часу навчання – 1.75 секунд.

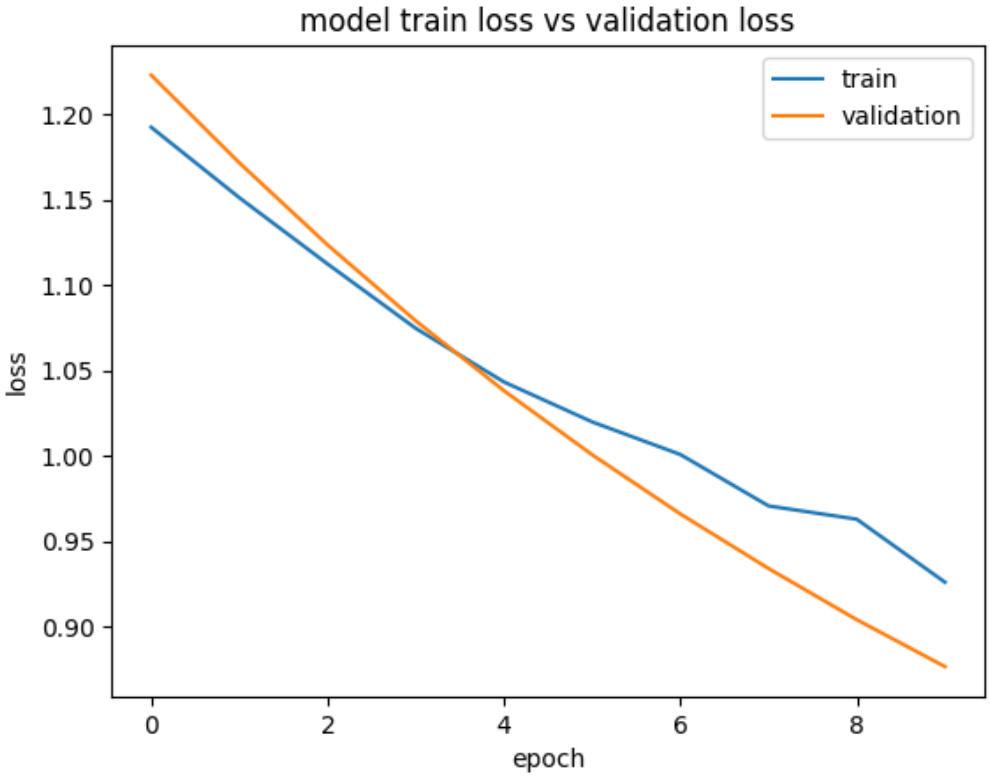

Рисунок 4.10 – Відношення функції втрат (LSTM – 1 | 32)

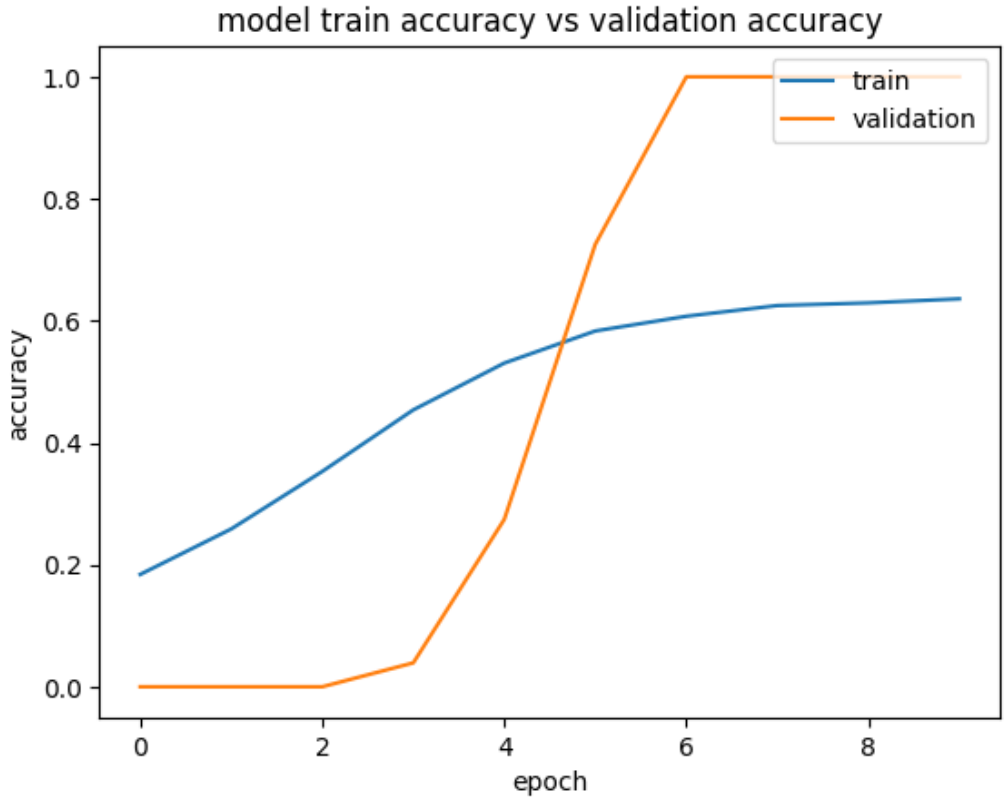

Рисунок 4.11 – Відношення точності (LSTM – 1 | 32)

Змінимо кількість memory units на 64 для підвищення ефективності. Після експерименту з цим значенням було отримано такі результати: точність тренування – 67%, точність тестування – 62%, відношення функції втрат на тестовому і тренувальному наборах даних – рисунок 4.12, відношенню тестової точності до тренувальної – рисунок 4.13, часу навчання – 1.75 секунд.

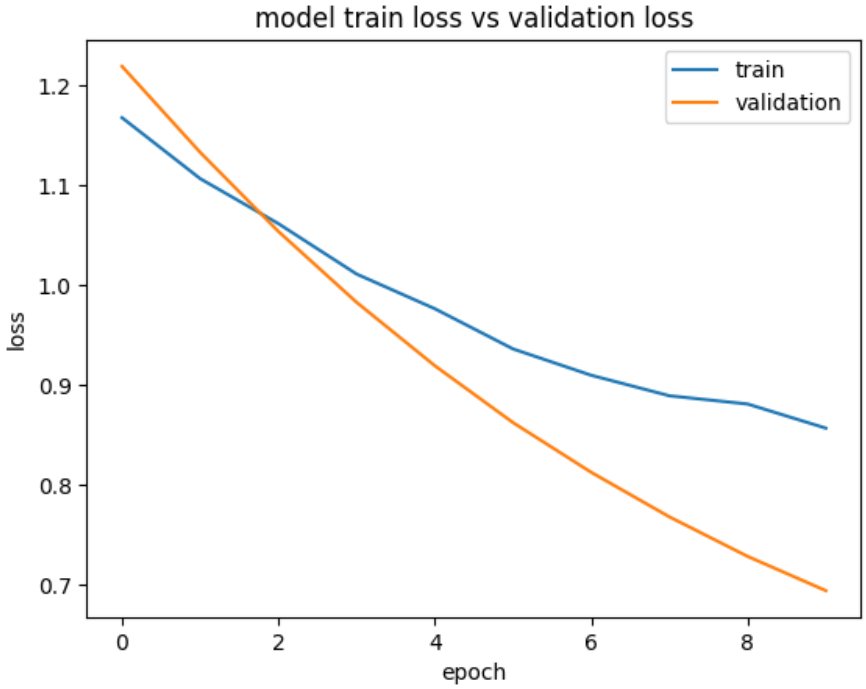

Рисунок 4.12 – Відношення функції втрат (LSTM – 1 | 32)

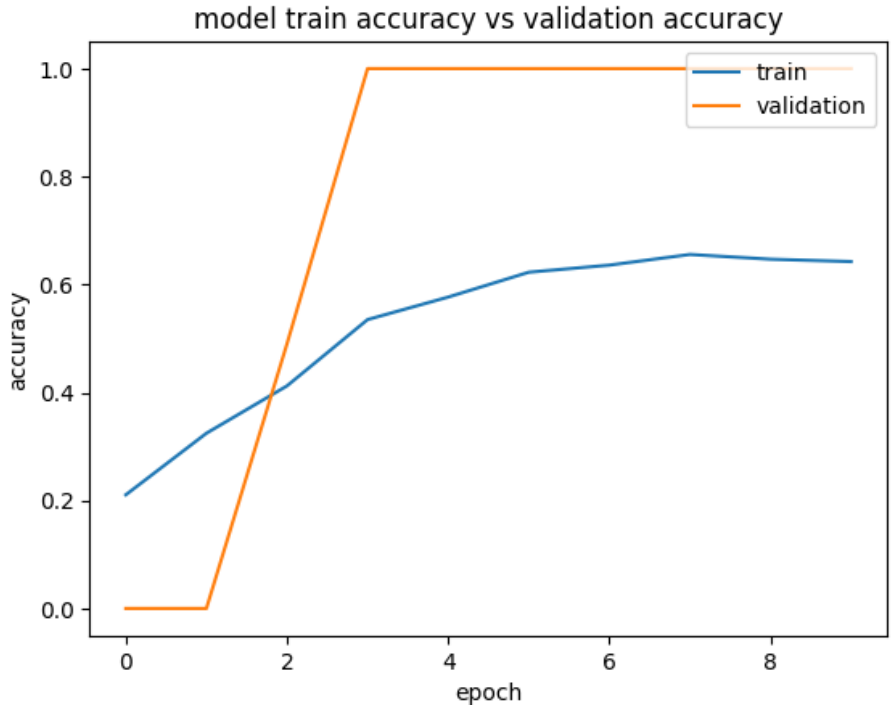

Рисунок 4.13 – Відношення точності (LSTM – 1 | 32)

Припустимо, що модель має два droupout шар з рівнем фільтрації 0.2 та LSTM шар з мінімально допустимою кількістю memory units – 64 та 32. Після експерименту з такою конфігурацією було отримано такі результати: точність тренування – 78%, точність тестування – 74%, відношення функції втрат на тестовому і тренувальному наборах даних – рисунок 4.14, відношенню тестової точності до тренувальної – рисунок 4.15, часу навчання – 4.75 секунд.

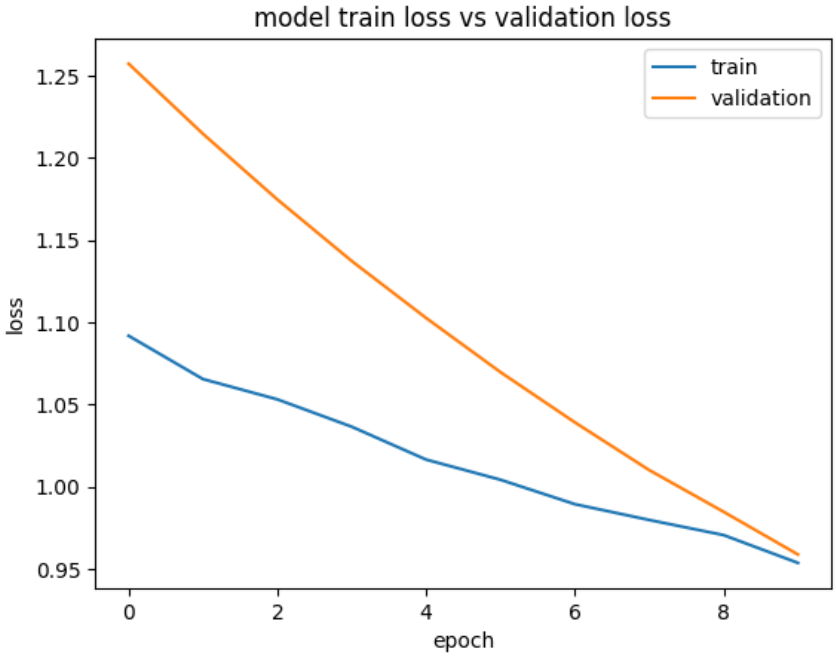

Рисунок 4.14 – Відношення функції втрат (LSTM – 2 | 64 | 32)

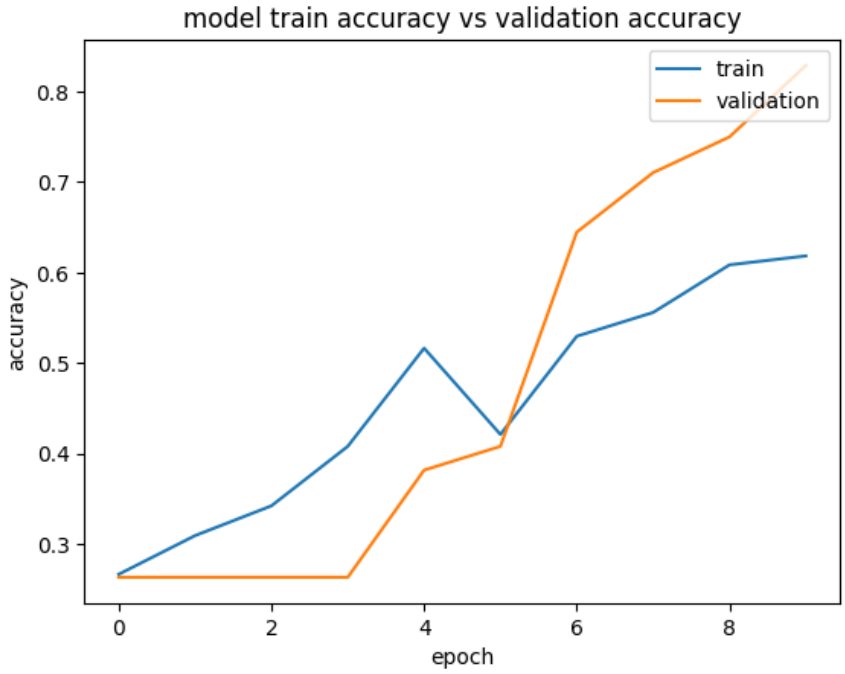

Рисунок 4.15 – Відношення точності (LSTM – 1 | 64 | 32)

Отже, експерименти показали, що найвигідніша комбінація це використання 2 шарів LSTM з 64 та 32 нейрони відповідно.

4.2.2 Гіперпараметри

Після аналізу у пунктах 4.2.1 та 4.2.2 було показано, що найефективніше використовувати модель з 2 шарами LSTM з послідовностями довжиною з 15 до 30 елементів, тому підібрати гіперпараметри буде не складно.

Важно зазначити що:

– Епохи (Epochs) – це кількість проходів через весь набір даних під час тренування моделі. Більша кількість епох може допомогти моделі краще навчитися, але також може призвести до перенавчання, якщо модель починає просто запам'ятовувати дані замість вивчення загальних шаблонів;

– швидкість навчання (Learning rate) – це параметр, який контролює, наскільки швидко модель оновлює ваги в процесі навчання. Занадто велика швидкість навчання може призвести до нестабільності і поганих результатів, тоді як занадто низька швидкість навчання може змусити процес навчання йти надто повільно;

– розмір пакета (Batch size) – це кількість прикладів, які обробляються одночасно в процесі тренування. Занадто великий розмір пакета може призвести до недостатньої обучаємості моделі, тоді як занадто малий може зробити процес навчання повільним і нестабільним.

– Validation split – це відсоток даних, які видаляються з набору тренувальних даних і використовуються для перевірки ефективності моделі під час тренування. Це допомагає контролювати перенавчання, дозволяючи перевіряти результати моделі на невиданих даних під час навчання;

Кожен з цих гіперпараметрів може значно вплинути на ефективність LSTM моделі, а їх оптимальні значення можуть залежати від конкретного набору даних і задачі.

Було проведено декілька експериментів, найцікавіші було задекларовано. Першим експериментом були проставлені ось такі параметри: Epochs – 20, Learning rate – 0.001, Batch size – 32, Validation split – 0.1. Після експерименту з такою конфігурацією було отримано такі результати: точність тренування – 98%, точність тестування – 60%, відношення функції втрат на тестовому і тренувальному наборах даних – рисунок 4.16, відношенню тестової точності до тренувальної – рисунок 4.17, часу навчання – 4 секунд.

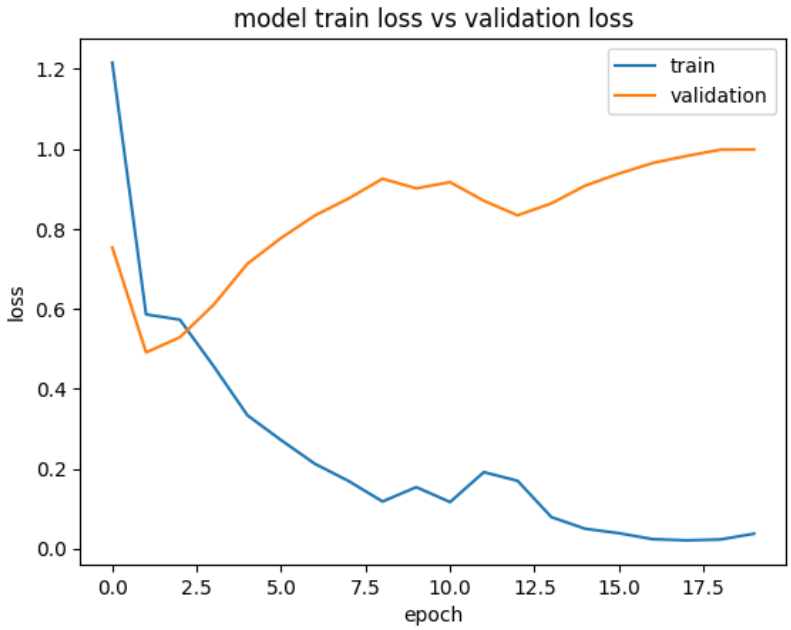

Рисунок 4.16 – Відношення функції втрат (Експеримент 1)

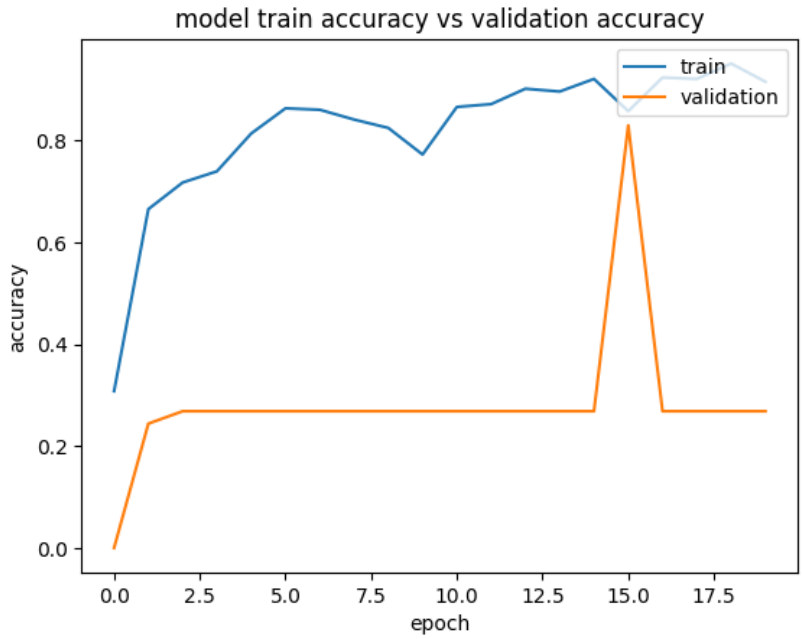

Рисунок 4.17 – Відношення точності (Експеримент 1)

Наступний та фінальний експериментом буде проставлені ось такі параметри: Epochs – 20, Learning rate – 0.00005, Batch size – 16, Validation split – 0.2. Після експерименту з такою конфігурацією було отримано такі результати: точність тренування – 79%, точність тестування – 66%, відношення функції втрат на тестовому і тренувальному наборах даних – рисунок 4.18, відношенню тестової точності до тренувальної – рисунок 4.19, часу навчання – 5.20 секунд.

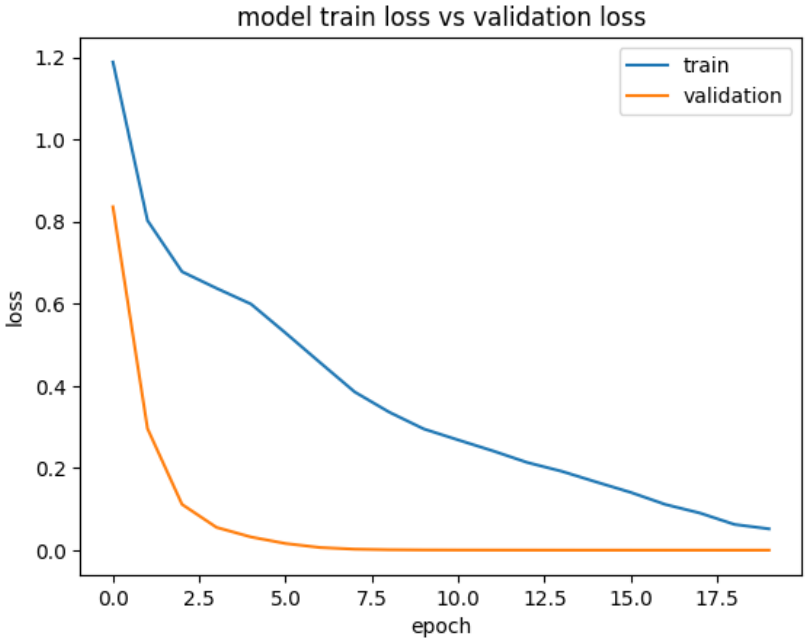

Рисунок 4.18 – Відношення функції втрат (Експеримент 2)

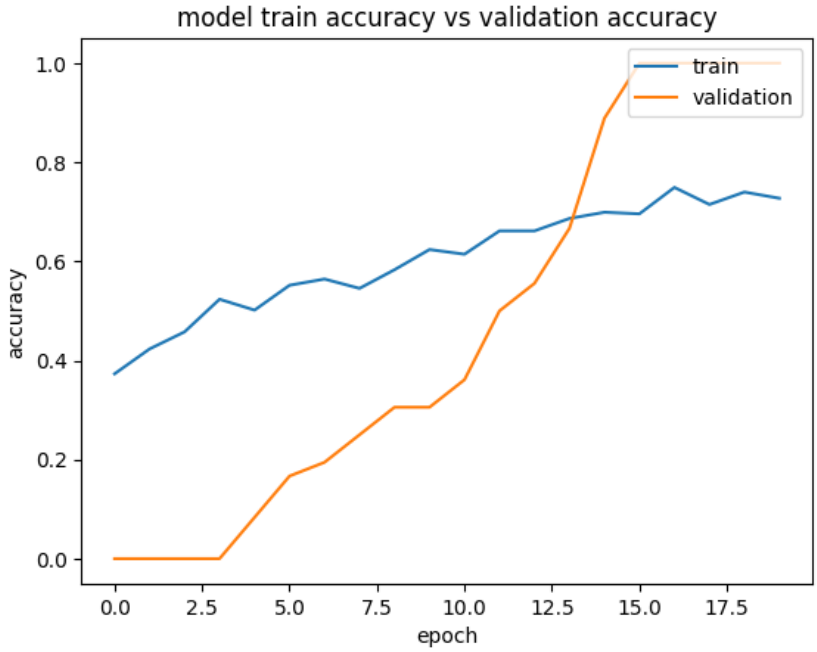

Рисунок 4.19 – Відношення точності (Експеримент 2)

Після проведених експериментів, було зроблено висновок щодо використання параметрів. Еталонними значеннями параметрів стали time_steps, які знаходиться між 15 та 30, 2 шари з 64, 32 комірками пам’яті та гіперпараметри у 20 ‘епох’, 0.00005 – ступінь вивчення, розмір одного батчу 16 разом з поділом валідаційних даних у 0.2.

Саме такі параметри з невеличким відклоном будуть використовуватися користувачем.

4.3 Використання системи для аналізу

Як було описано у розділі ‘1.2 опис функціональної моделі’ користувач має можливість подивитися результат аналізу аномалій на даних які він зазначив, а також подивитися результат аналізу.

Після того як юзер обрав який актив він буде аналізувати, а також визначив для себе якими проміжками часу користувач опирує – система нормалізує дані та починає аналіз.

Далі система знаходить аномалії та запитує користувача за яким параметром вона може вивести результат, припускається що користувач вибирає параметр який найбільше підходить, а саме ціну. Результатом аналізу на аномалії користувач отримує графік, його приклад можно подивитися на рисунку 4.20, який він може використати для того щоб зрозуміти загальну ситуацію на ринку. Червоним помічені проміжки де алгоритм пошуку знайшов аномалію.

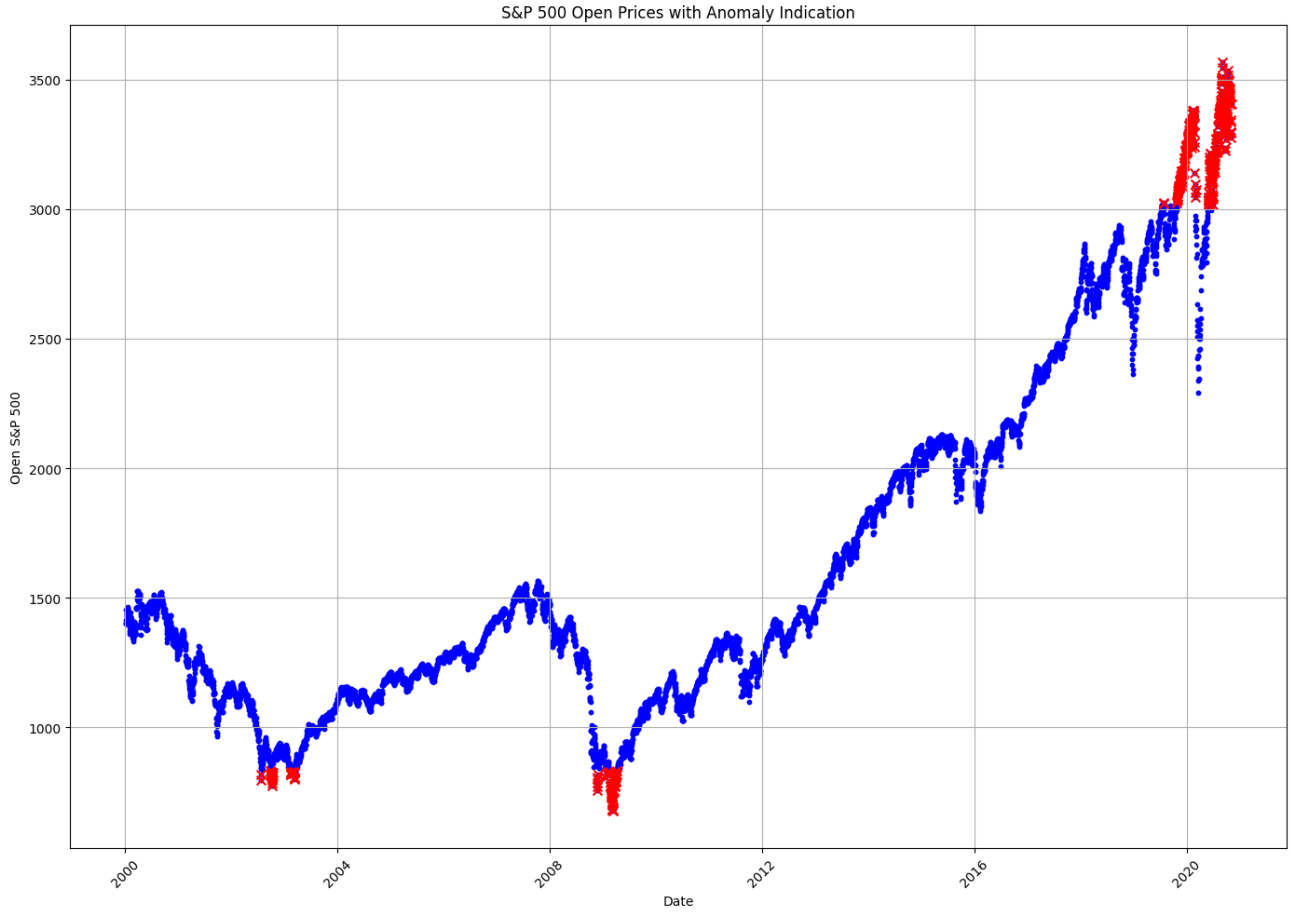

Рисунок 4.20 – Дослідження аномалій користувачем

Після знаходження та аналізу аномалій користувач має можливість продовжити аналіз. Після тесту отримуємо результат, що треба йти у закупівлю. Було зроблено передбачення маючи послідовність даних для 15 послідовних днів та отримали сигнал sell. Це рішення може бути далі прийняте до розглядання. Також треба брати до уваги, що кількість елементів у послідовності може мінятися.

Висновки до розділу

У розділі 4 було розглянуто критичні аспекти розробки нейронної мережі. Було обґрунтовано важливість масштабування даних до оптимального діапазону та використання кодування для подальшої обробки міток.

Архітектура нейронної мережі була створена з використанням двонаправлених LSTM шарів для кращого вивчення складних шаблонів в даних.

Використання шарів Dropout було обґрунтовано як ефективний метод регуляризації для запобігання перенавчання.

Тренування моделі включало визначення оптимальних гіперпараметрів, таких як кількість епох, розмір пакета і розмір відкладеної вибірки для валідації. Здатність моделі передбачувати була протестована, використовуючи метод predict, що дозволило отримати прогнози для тестових даних і перетворити їх у масиви класів.

Загалом, розроблена модель продемонструвала позитивні результати, виказавши точність 74%. Цей результат показує потенціал моделі, хоча подальша оптимізація і налаштування можуть покращити цей показник.

Також було розглянуто як користувач використовує додаток у 2 етапи: було показано як знаходяться аномалії, а також отримання сигналу щодо майбутньої інвестиції.

ВИСНОВКИ

Метою цієї роботи було підвищення ефективності аналізу фінансових ринків за допомогою розробки програмного забезпечення на базі нейронних мереж.

Було проаналізовано та описано предметне середовище, також розібрано наявні аналоги вже існуючих рішень. Далі було розписано потрібність такої системи. Також була описана функціональна модель, яка послугувала планом при створенні додатка, були створені вимоги до нього, а також можливості програми.

У першому розділі було проведено детальний аналіз існуючих рішень та стратегій щодо аналізу фінансових ринків. Було зазначено, які рішення можуть бути успішними для впровадження. Було розглянуто наявні методи машинного навчання та зроблено висновки щодо їх використання у роботі. Також було поставлено цілі на розробку і зроблені вимоги до розроблення додатку.

У другому розділі було проведено детальний аналіз предметного середовища. Було розглянуто різні мови програмування, а також обрано яка більше всього підходить для вирішення поставленої задачі. Також було проаналізовано наявні бібліотеки для спрощення вирішення наявної задачі. Було розглянуто структуру різних видів вхідних даних, які мають в собі якості часових рядів, а також способи їх нормалізації. Було знайдено і сформовано датасет а також проаналізовано відношення вхідних даних між собою. Також було виконано детальний опис вигляду вихідних даних.

У третьому розділі було описано методи розробки до системи кількісного аналізу фінансових ринків. Було детально описано роботу системи, включаючи опис алгоритму пошуку аномалій і вид нейронної мережі. Було математично описано задачу та обґрунтовано спосіб вирішення за допомогою системи, що розробляється. Також було описано структуру та тип нейронної мережі для аналізу часових рядів.

У четвертому розділі було проведено дослідження та експерименти з розробленою системою. Було продемонстровано процес розробки нейронної мережі та процес її вдосконалення методом впровадження великої кількості експериментів із долученням різних видів параметрів. Також було показано як система працює для вирішення поставленої задачі, системою було побудовано

графік відображення знайдених аномалій на часовій послідовності, а також нейронною мережею було видано сигнал щодо послідуючих рекомендованих дій.

Розроблена система у цьому дипломному проекті може подальше бути використана як допоміжний інструмент в аналізі фінансових ринків. Система має середню точність, що пояснюється волатильністю фінансових ринків а також складністю самої моделі.

До подальших покращень системи можна віднести впровадження цієї системи на більш складних даних для фокусування системи у роботі з алгоритмічною торгівлею. Це не тільки зможе покращити її точність, а й зробити її основним інструментом впровадження фінансових рішень.

Розробивши систему для кількісного аналізу фінансових ринків за допомогою нейронних мереж та оцінивши її на отриманих результатах, можна сказати, що ціль та мета цієї дипломної роботи була виконана.

Список використаних джерел

1. Tatsat, H., Puri, S., Lookabaugh, B. (2020). Machine Learning and Data Science Blueprints for Finance. United States: O'Reilly Media. 49–55 p.
2. International Monetary Fund [Електронний ресурс] – Режим доступу до ресурсу: https://www.imf.org/external/datamapper/NGDP_RPCH@WEO/OEMDC/ADVEC/WEOWORLD
3. Analyzing Alphpa [Електронний ресурс] – Режим доступу до ресурсу: https://analyzingalpha.com/algorithmic–trading–statistics
4. Facebook [Електронний ресурс] – Режим доступу до ресурсу: https://facebook.github.io/prophet/docs
5. Bayes, T. 1764. "An Essay Toward Solving a Problem in the Doctrine of Chances". 370–418 p.
6. Dennis M. Dimiduk, Elizabeth A. Holm, Stephen R. Niezgoda. (2018). "Perspectives on the Impact of Machine Learning, Deep Learning, and Artificial Intelligence on Materials, Processes, and Structures Engineering". 103–169p.
7. Xing Lin, Yair Rivenson, Nezih T. Yardimci, Muhammed Veli, Yi Luo, Mona Jarrahi, Aydogan Ozcan. (2018). "All–optical machine learning using diffractive deep neural networks". 1–50p.
8. Christian Janiesch, Patrick Zschech, Kai Heinrich. (2021). "Machine learning and deep learning".
9. Sahand Hariri, M. Carrasco Kind, Robert J. Brunner. (2021). "Extended Isolation Forest".
10. Kishwar Sadaf, Jabeen Sultana. (2020). "Intrusion Detection Based on Autoencoder and Isolation Forest in Fog Computing".
11. Cheng–Few Lee, Alice C Lee, John D. Lee. (2010). "Handbook of Quantitative Finance and Risk Management".
12. Francesco Rundo, Francesca Trenta, Agatino Luigi di Stallo, Sebastiano Battiato. (2019). "Machine Learning for Quantitative Finance Applications: A Survey".

13. Jan De Spiegeleer, Dilip B. Madan, Sofie Reyners, Wim Schoutens. (2018). "Machine learning for quantitative finance: fast derivative pricing, hedging and fitting".

14. Python [Електроний ресурс] – Режим доступу до ресурсу: https://www.python.org/

15. NumPy [Електроний ресурс] – Режим доступу до ресурсу: https://numpy.org/

16. TensorFlow [Електроний ресурс] – Режим доступу до ресурсу: https://www.tensorflow.org/

17. Keras [Електроний ресурс] – Режим доступу до ресурсу: https://keras.io/

18. Scikit–image [Електроний ресурс] – Режим доступу до ресурсу: https://scikit–image.org