НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ
КАФЕДРА ШТУЧНОГО ІНТЕЛЕКТУ

На правах рукопису
УДК 004.852

До захисту допущено
В.о. зав. кафедри ШІ
_____О.І. Чумаченко
«___»_____2022 р.

Магістерська дисертація

на здобуття ступеня магістра

зі спеціальності 122 «Комп'ютерні науки»

на тему: «Методи прогнозування індексу акцій на основі механізмів штучного інтелекту»

Виконала:
студентка ІІ курсу, групи КІ-11мп
Міщенко Дарина Вадимівна

Керівник:
зав. кафедри СП,
д.т.н. проф. Мухін Вадим Євгенович

Рецензент:
декан ФІОТ
КПІ ім. Ігоря Сікорського, д.т.н., Корнага Я. І.

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

Студент _____

Київ
2022

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ
КАФЕДРА ШТУЧНОГО ІНТЕЛЕКТУ

Рівень вищої освіти – другий (магістерський)

Спеціальність – 122 «Комп'ютерні науки »

ЗАТВЕРДЖУЮ
В.о. зав. кафедри
_____ О.І. Чумаченко
«___» _____ 2022 р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Міщенко Дарині Вадимівні

1. Тема дисертації: «Методи прогнозування індексу акцій на основі механізмів штучного інтелекту», науковий керівник роботи Мухін Вадим Євгенович, зав. кафедри СП, д.т.н., проф. затверджено наказом по університету від «03» листопада 2022 р. № 4046-с.
2. Термін подання студентом дисертації 15.12.2022
3. Об'єкт дослідження: задача прогнозування фінансових даних.
4. Предмет дослідження: авторегресійні моделі, рекурентна нейронна мережа довгої короткострокової пам'яті та нейронна мережа, побудована на механізмі уваги.
5. Перелік завдань, які потрібно зробити:
    1) здійснити огляд технічної літератури за темою роботи;
    2) дослідити актуальність обраної теми;
    3) ознайомитись із існуючими методами та моделями прогнозування фінансових даних;

4) здійснити порівняльний аналіз наявних методів, виявити їх переваги та недоліки;

5) розробити та реалізувати систему, що використовує апарат нейрониих мереж, та вирішує задачу побудови прогнозу фінансових даних;

6) провести експеримент, що засвідчує працеспроможність запропонованої моделі, виконати аналіз результатів;

7) провести аналіз ринкових можливотсей запуску стартап проекту;

8) розробити концептуальні висновки;

9) підготувати ілюстративний матеріал;

10) оформити пояснювальну записку.

6. Перелік ілюстративного матеріалу.

7. Дата видачі завдання: 6 вересня 2022 р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів роботи	Примітка
1.	Вивчення літератури за темою роботи.	06.09.2022 – 19.09.2022	Виконано
2.	Підготовка першого розділу.	20.09.2022 – 26.09.2022	Виконано
3.	Підготовка другого розділу.	27.09.2022 – 03.10.2022	Виконано
4.	Розробка програмного продукту.	04.10.2022 – 17.10.2022	Виконано
5.	Підготовка третього розділу	18.10.2022 – 24.10.2022	Виконано
6.	Підготовка частини стартап-проєкту	25.10.2022 – 31.10.2022	Виконано
9.	Концептуальні висновки. Перспективи розвитку отриманих рішень	01.11.2022 – 14.11.2022	Виконано
10.	Оформлення пояснювальної записки	15.11.2022 – 26.11.2022	Виконано

Студент Дарина МІЩЕНКО

Керівник Вадим МУХІН

РЕФЕРАТ

Магістерська дисертація: 99 с., 26 табл., 29 рис., 25 джерел, 1 додаток.

ПРОГНОЗУВАННЯ ДАНИХ, ДОВГА КОРОТКОСТРОКОВА ПАМ’ЯТЬ, ФОНДОВІ РИНКИ, МЕТОДИ ПРОГНОЗУВАННЯ, НЕЙРОННІ МЕРЕЖІ, S&P 500, АВТОРЕГРЕСІЯ, ІНДЕКСИ АКЦІЙ, МЕХАНІЗМ УВАГИ

Об’єктом дослідження є задача прогнозування фінансових даних.

Предмет дослідження – авторегресійні моделі, рекурентна нейронна мережа довгої короткострокової пам’яті та нейронна мережа, побудована на механізмі уваги.

Мета дослідження полягає у аналізі фінансових данних, підборі моделей для прогнозування, реалізації методів прогнозування на основі механізмів штучного інтелекту та вибір найкращого методу.

Як результат дослідження було запропоновано та розроблено модель прогнозування даних, що використовує механізми штучного інтелекту, як авторегресійні моделі, нейронні мережі довгої короткострокової пам'яті та модель на основі механізму уваги.

Проведено порівняння побудованих моделей та вибрано найкращу за метриками MAPE, MAE, MSE, R^2. Результат даної роботи можна застосувати при вирішенні подібних задач короткострокового прогнозування нестаціонарних часових рядів.

ABSTRACT

Master’s thesis: 99 p., 26 tab., 29 fig., 25 references, 1 appendix.

DATA FORECASTING, LONG SHORT-TERM MEMORY, STOCK MARKETS, FORECASTING METHODS, NEURAL NETWORKS, S&P 500, AUTOREGRESSION, STOCK INDICES, ATTENTION NETWORKS

The object of research is the problem of forecasting financial data.

The subjects of current thesis are autoregressive models, recurrent neural network of long short-term memory and attention network.

The purpose of the study is to analyze financial data, select models for forecasting, implement forecasting methods based on artificial intelligence mechanisms and choose the best method.

As a result of the study, a data forecasting model was proposed and developed that uses artificial intelligence mechanisms such as autoregressive models, long short-term memory neural networks and a model based on the attention mechanism.

The comparison of the constructed models was carried out and the best one was chosen according to the metrics MAPE, MAE, MSE, R^2. The result of this work can be applied in solving similar problems of short-term forecasting of non-stationary time series.

ЗМІСТ

ВСТУП

Протягом останніх десятиліть об'єктом багатьох досліджень є прогнозування фондових ринків, яке, незважаючи на свою заплутаність, динамічність та нестабільність, є надзвичайно складною задачею. Для побудови ефективної моделі прогнозування необхідно враховувати різноманітні фактори, величезні обсяги даних та тривіальне співвідношення сигнал/шум, що значно ускладнює задачу прогнозування поведінки цін на фондовому ринку. Тим не менш, існує велика кількість різноманітних підходів, спрямованих на досягнення цієї мети.

В даній роботі пропонується розглянути задачу прогнозування фінансових даних як об’єкт дослідження з метою аналізу даних, підбору моделей для прогнозування, реалізації методів на основі механізмів штучного інтелекту та вибору найкращої.

Предметом дослідження є авторегресійні моделі, рекурентна нейронна мережа довгої короткострокової пам’яті та нейронна мережа, побудована на основі механізму уваги.

Актуальність даного дослідження полягає в тому, що прогнозування фінансових часових рядів було та завжди буде важким завданням через його чутливість до політичних, економічних і соціальних факторів, наприклад війна, чи пандемія, та навіть враховуючи ці випадки люди, які інвестують у фінансові ринки та обмін валюти, зазвичай шукають надійні моделі, які можуть гарантувати їм максимізацію свого профілю та мінімізацію втрат, наскільки це можливо в умовах постійної соціальної і економічної нестабільності.

Наукова новизна роботи полягає в розробці методу підвищення якості прогнозу індексів фондового ринку, який відрізняється процесом підготовки вхідних параметрів мережі на основі механізму уваги з застосуванням LSTM

мережі, в якому виконується аналіз оброблених даних з використанням механізму нейронних мереж, що дозволяє підвищити достовірність прогнозування індексів фондового ринку.

Рекурентні нейронні мережі (RNN) використовуються, коли модель вимагає обробки даних часових рядів або природної мови. LSTM, будучи однією з найуспішніших архітектур штучних нейронних мереж, має можливість надавати різні ваги для кожного прикладу і ненавмисно нехтувати пам'яттю, яку вона вважає неважливою для прогнозування наступного результату, щоб розрізнити поточні та попередні приклади. Даний алгоритм виявився дуже ефективним при вирішенні таких проблем. Так, на відміну від інших рекурентних нейронних мереж, він більш ефективний при обробці довгих вхідних послідовностей. Таким чином, за допомогою мережі LSTM можна досягти дуже високого рівня точності в прогнозуванні майбутніх тенденцій і оцінці цін на різні акції. Кілька інших моделей буде разглянено в роботі далі та обрано найкращу для прогнозування поведінки цін на фондовому ринку.

Мільйони людей по всьому світу щодня інвестують у фондовий ринок. Хороша модель прогнозування цін на акції допоможе інвесторам, керівництву та особам, які приймають рішення, у прийнятті правильних та ефективних рішень.

РОЗДІЛ 1 ОГЛЯД ЗАСОБІВ АНАЛІЗУ ТА ПРОГНОЗУВАННЯ ІНДЕКСУ АКЦІЙ ТА ХАРАКТЕРНИХ РІШЕНЬ

1.1 Огляд предметної області

Фондовий ринок – це сукупність бірж, через які випускаються, купуються та продаються акції публічних компаній.

Роль фондового ринку полягає в тому, щоб забезпечити компаніям спосіб залучення капіталу шляхом продажу акцій державним інвесторам. У той же час, фондовий ринок дозволяє приватним інвесторам купувати акції публічних компаній і ставати співвласниками їх бізнесу. Сукупна вартість всього фондового ринку часто відстежується та відображається за допомогою ринкових індексів, таких як промисловий індекс Доу-Джонса (Dow Jones Industrial Average) та індекс S&P 500 (S&P 500).

Коли приватна компанія бажає стати публічною, вона, як правило, здійснює первинне публічне розміщення акцій (IPO). Під час процесу IPO компанія продає акції публічним інвесторам, щоб зібрати гроші, необхідні для погашення боргу або інвестування у свій бізнес. Після завершення IPO ці акції починають торгуватися на одній або декількох фондових біржах, які складають фондовий ринок.

Інвестори купують акції в надії на те, що компанія, яка продає ці акції, з часом зросте і стане більш цінною, тим самим збільшуючи ціну кожної акції. Ціни на акції визначаються економічним законом попиту та пропозиції, і їх ціни часто коливаються щодня на основі змін у попиті інвесторів.

Перша сучасна фондова біржа була створена в Амстердамі в 1611 році, і спочатку інвестори могли купувати та продавати лише акції Голландської Ост-Індської компанії. Першою фондовою біржею США була Філадельфійська фондова біржа, яка була заснована в 1790 році. Нью-Йоркська фондова та біржова рада, попередник Нью-Йоркської фондової біржі, була офіційно

створена у 1817 році. У 1971 році була створена Національна асоціація автоматизованого котирування цінних паперів (Nasdaq). Nasdaq була новаторською, оскільки дозволила інвесторам купувати і продавати акції в цифровому вигляді в мережі комп'ютерів, а не особисто на торговому майданчику [1].

Основні терміни, що використовуються на фондовій біржі: open, close, high, low.

Почнемо з терміну open. Термін "open" фактично відноситься до певного валютного курсу. Open – це офіційний курс, за яким відкриваються торги на фондовій біржі в торговий день. Однак цей курс не буде триматися особливо довго.

Під терміном "Close" маються на увазі дві різні речі. По-перше, це офіційний обмінний курс конкретного торгового дня, тобто обмінний курс, який діє на момент закриття ринку на день. Також даний термін відноситься до точного часу дня, коли біржа фактично закривається. Це означає, що якщо транзакція встановлюється "на закриття", то час закриття – це час, коли транзакція завершується.

High – це максимальний курс обміну або для всього ринку, для певного сектора ринку, для певного індексу або для певної акції. Переконайтеся, що ви знаєте, що саме мається на увазі під максимумом. Обов’язково потрібно звертати увагу, який період часу описується: день, тиждень, місяць, рік, історія акції або індексу, що завгодно.

Мінімум - це найнижча ціна ринку, акції, сектора, індексу тощо. Він може бути на будь-якому проміжку часу [2].

1.2 Аналіз сучасних засобів прогнозування

1.2.1 Індикатори ринку цінних паперів

Серія індикаторів ринку цінних паперів (SMIS) - це ринковий індекс або середній показник, який використовує показники вибірки цінних паперів для представлення показників ринку або сегмента ринку. Найвідоміші SMIS у США включають промисловий індекс Доу-Джонса (DJIA), зведений індекс Nasdaq Composite та індекс S&P 500.

Серія індикаторів ринку цінних паперів часто використовується в бенчмаркінгу. Наприклад, аналітик може порівняти цінний папір, який широко розглядається як такий, що демонструє високі темпи зростання, з вибіркою аналогічних цінних паперів, щоб побачити, чи випереджає цей цінний папір свій ринковий сегмент, чи відстає від нього.

Аналогічно, інвестори можуть використовувати SMIS для оцінювання менеджерів з управління капіталом: професійних інвесторів, які беруть плату за розробку та реалізацію інвестиційних стратегій від імені своїх клієнтів. Щоб переконатися, що комісійні добре зароблені, клієнти можуть порівняти інвестиційні показники менеджера з порівнянними показниками SMIS. Використання ретельно підібраного SMIS може допомогти визначити, чи дійсно керуючий створює додаткову вартість відносно показників ринку в цілому. Далі розглянемо найпопулярніші індикатори ринку цінних паперів [3].

Промисловий індекс Доу-Джонса є найвідомішим фондовим індексом США. Індекс Доу-Джонса був розроблений Чарльзом Генрі Доу і спочатку містив лише 12 американських компаній. Вперше він був опублікований у травні 1896 року і відкрився на рівні 40,94 пункти. Сьогодні промисловий індекс Доу-Джонса складається з 30 найважливіших компаній-лідерів ринку на американській фондовій біржі та відображає їх зростання.

Як і Швейцарський фондовий індекс (SMI), Dow Jones є ціновим індексом. Акції, що входять до нього, зважуються за ціною; рівень індексу

являє собою середнє значення акцій, що входять до нього. Дивідендні виплати в індексі не враховуються.

S&P 500 та DJIA є двома найбільш відстежуваними фондовими індексами в США. Однак, ці два бенчмарки дуже відрізняються:

1. Промисловий індекс Dow Jones відстежує 30 акцій з великою капіталізацією, S&P 500 відстежує 500 найбільших акцій на ринку США
2. Індекс Dow Jones зважений за ціною; S&P 500 - за ринковою капіталізацією
3. Акції до індексу Dow обираються комітетом, акції до індексу S&P 500 додаються за формулою
4. Dow Jones використовує дільник; S&P 500 виражається по відношенню до базового року. [4]

Індекс S&P 500 або Standard & Poor's 500 Index - це зважений за ринковою капіталізацією індекс 500 провідних публічних компаній США.

Він не є точним переліком 500 найбільших компаній США за ринковою капіталізацією, оскільки існують й інші критерії, які включаються до індексу. Тим не менш, індекс S&P 500 вважається одним з найкращих показників діяльності провідних американських компаній, а отже, і фондового ринку в цілому.

S&P 500 використовує метод зважування за ринковою капіталізацією, надаючи більший відсоток компаніям з найбільшою ринковою капіталізацією. Алгоритм ALS вирішує наступну оптимізаційну задачу:

$$\text{Зважування компанії в } SP = \frac{\text{Ринкова капіталізація компанії}}{\text{Загальна к} - \text{ть ринкових кап.}} \qquad (1.1)$$

Визначення ваги кожної складової S&P 500 починається з підрахунку загальної ринкової капіталізації індексу шляхом додавання ринкової капіталізації кожної компанії, що входить до індексу.

Визначення ваги кожної складової S&P 500 починається з підрахунку загальної ринкової капіталізації індексу шляхом додавання ринкової капіталізації кожної компанії, що входить до індексу.

При розрахунку ринкової капіталізації S&P використовує лише акції, що перебувають у вільному обігу, тобто акції, якими може торгувати громадськість. S&P коригує ринкову капіталізацію кожної компанії, щоб компенсувати нові випуски акцій або злиття компаній. Значення індексу розраховується шляхом підсумовування скоригованої ринкової капіталізації кожної компанії та ділення результату на дільник. Дільник є комерційною таємницею S&P і не розголошується.

Проте, ми можемо розрахувати вагу компанії в індексі, яка може надати інвесторам цінну інформацію. Якщо акції зростають або падають, ми можемо зрозуміти, чи може це вплинути на загальний індекс. Наприклад, компанія з вагою 10% матиме більший вплив на значення індексу, ніж компанія з вагою 2%.

S&P 500 є одним з найбільш широко котируваних американських індексів, оскільки він представляє найбільші публічні корпорації в США. S&P 500 фокусується на секторі компаній з великою капіталізацією на американському ринку, а також є індексом, зваженим на акції в обігу (тип зважування капіталізації), що означає, що ринкова капіталізація компаній коригується на кількість акцій, доступних для публічної торгівлі.

Іншим поширеним показником фондового ринку США є промисловий індекс Доу-Джонса (DJIA). Індекс S&P 500 часто надають перевагу інституційні інвестори з огляду на його глибину та широту, в той час як DJIA історично асоціюється з важливими акціями з точки зору роздрібних інвесторів. Інституційні інвестори сприймають S&P 500 як більш репрезентативний фондовий ринок США, оскільки він включає більше акцій з усіх секторів (500 проти 30 у Dow).

Крім того, S&P 500 використовує метод зважування за ринковою капіталізацією, надаючи більший відсоток компаніям з найбільшою ринковою капіталізацією, в той час як DJIA є індексом, зваженим за ціною, що надає компаніям з вищими цінами на акції більшу вагу в індексі.

Структура, зважена на ринкову капіталізацію, як правило, є більш поширеною, ніж структура, зважена на ціну, в американських індексах.

Nasdaq - це глобальний електронний ринок для торгівлі цінними паперами. Існує декілька індексів фондового ринку, які включають акції, що торгуються на Nasdaq. Зверніть увагу, що певна акція, яка входить до індексу S&P 500, може також входити до одного або декількох різних індексів Nasdaq.

До найбільш відстежуваних фондових індексів Nasdaq відносяться: Індекс Nasdaq 100, який включає 100 найбільших, найбільш активно торгуваних звичайних акцій, що котируються на Nasdaq; Індекс Nasdaq Composite, який ЗМІ часто називають просто "Nasdaq" (і який включає понад 2 500 звичайних акцій, що торгуються на Nasdaq); Nasdaq Global Equity Index (NQGI), який включає міжнародні акції; та PHLX Semiconductor Sector Index (SOX), який є провідним барометром акцій, пов'язаних з напівпровідниковою промисловістю; OMX Stockholm 30 Index (OMXS30), який включає 30 акцій, що активно торгуються на Стокгольмській фондовій біржі. [5]

Nasdaq – це глобальний електронний ринок купівлі-продажу цінних паперів. Спочатку його назва була абревіатурою від "National Association of Securities Dealers Automated Quotations" - Nasdaq розпочала свою діяльність як дочірня компанія Національної асоціації дилерів з цінних паперів (NASD), яка зараз відома як Управління з регулювання фінансової індустрії (FINRA). Nasdaq була запущена після того, як Комісія з цінних паперів і бірж (SEC) закликала NASD автоматизувати ринок цінних паперів, що не котируються на біржі. Результатом стала перша електронна торгова система. Nasdaq відкрилася для бізнесу 8 лютого 1971 року.

Термін "Nasdaq" також використовується для позначення Nasdaq Composite, індексу з більш ніж 3700 акцій, що котируються на біржі Nasdaq, до якого входять технологічні гіганти Apple Inc., Microsoft, материнська компанія Google Alphabet, Meta Platforms Inc., Amazon.com Inc. і Tesla Inc.

Nasdaq офіційно відокремилася від NASD і почала функціонувати як національна біржа цінних паперів у 2006 році. У 2008 році вона об'єдналася зі скандинавською групою бірж OMX, щоб стати Nasdaq OMX Group.

Штаб-квартира Nasdaq знаходиться в Нью-Йорку, компанія управляє 29 ринками, що дозволяють торгувати акціями, деривативами, цінними паперами з фіксованим доходом і товарами в США, Канаді, Скандинавії та країнах Балтії. компанія також керує клірингoвою палатою і п'ятьма центральними депозитаріями цінних паперів в США і Європі.

Її торгові технології використовують 100 бірж у 50 країнах світу. Nasdaq Inc котирується на фондовій біржі Nasdaq під символом NDAQ та входить до індексу S&P 500 з 2008 року. [6]

1.2.2 Аналіз на стаціонарність

До основних видів моделей часових рядів належать моделі, що передбачають залежність одного ряду від інших рядів, моделі, що представляють залежність поточних значень ряду від його ж минулих значень, а також моделі, що є декомпозицією вихідного ряду на кілька складових, а саме тренд, періодичну і випадкову складові. В окремих випадках передбачається також наявність циклічної складової ряду. Часто модель ряду являє собою суперпозицію перелічених варіантів.

У зв'язку з цим, а також через деякі складнощі моделювання, вводиться поняття стаціонарного ряду. Для розв'язання практичних завдань зазвичай використовується поняття стаціонарності часового ряду в слабкому сенсі.

Слабо стаціонарним називається часовий ряд у разі, якщо його теоретичні математичне очікування та дисперсія не залежать від часу, а коваріація між його значеннями в моменти t та t + s залежить тільки від s, а не від часу.

Для діагностики ряду й оцінки можливого класу моделей можуть використовуватися графік ряду, корелограма, ряд тестів.

Графічно зазвичай легко помітити мінливість математичного сподівання, яка може виражатися в наявності тренда, наявної періодичності, а також у неможливості поділити ряд на кілька частин, кожна з яких матиме схожу середню.

Графічно це проілюстровано на риснках 1.1 та 1.2.

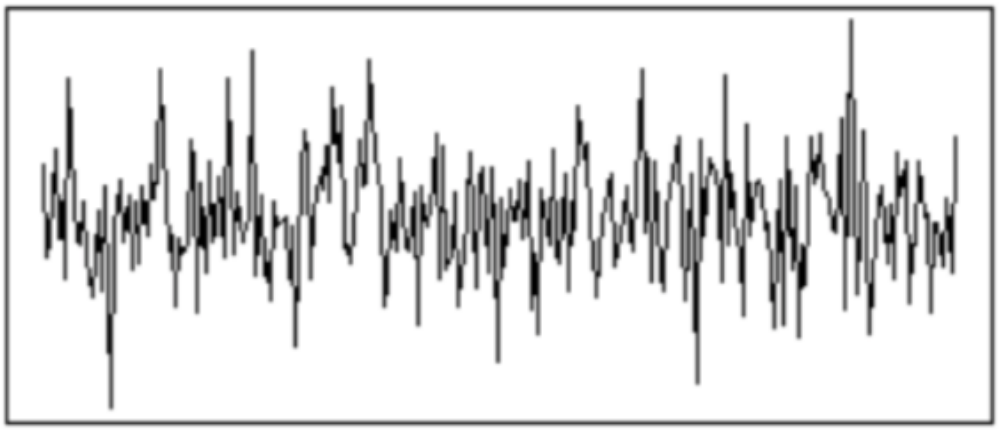

Рисунок 1.1 – Стаціонарний ряд [7]

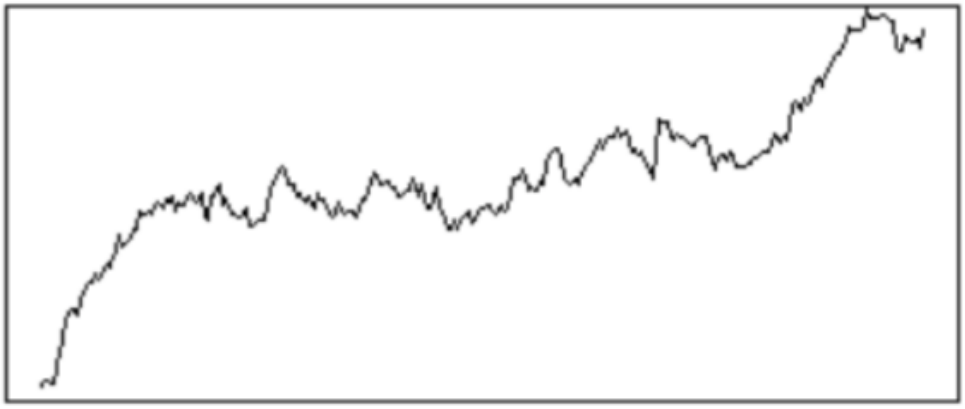

Рисунок 1.2 – Нестаціонарний ряд [7]

Нестаціонарність процесу може виявлятися по-різному, але важливою її відмінністю є наявність поодиноких коренів. Тому для визначення стаціонарності використовуються тести на перевірку нульової гіпотези щодо наявності одиничного кореня.

Стаціонарність часового ряду можна перевіряти безпосередньо, виходячи з її визначення. Для цього ряд потрібно розділити на кілька частин і перевірити гіпотези про рівність дисперсій і математичних очікувань цих частин. У разі позитивного результату перевірки відповідних гіпотез, можлива додаткова перевірка рівності АКФ (попарне порівняння коефіцієнтів кореляції кожного порядку за допомогою тесту на рівність кореляції).

Також стаціонарність часового ряду можна перевіряти за допомогою критеріїв Дікі-Фуллера. Ці критерії називають тестом на наявність одиничного кореня. Часовий ряд має одиничний корінь, або порядок інтеграції один, якщо його перші різниці утворюють стаціонарний ряд. Суть їх полягає в такому: робиться припущення про вид процесу, що породив даний часовий ряд, далі будується допоміжна модель і перевіряються гіпотези про коефіцієнти цієї моделі, після чого робиться висновок про стаціонарність/нестаціонарність вихідного ряду. [8]

Складність полягає у коректності припущення про вид допоміжної моделі та у невеликій потужності тесту для часових рядів із менш ніж 100 спостереженнями.

1.2.3 ARMA

ARMA (autoregressive moving average) – авторегресія середнього ковзного, це математична модель, що використовується для аналізу та прогнозування стаціонарних рядів у статичних даних.

У статистиці та обробці сигналів модель авторегресійного середнього ковзного, яку іноді називають моделлю Бокса-Дженкінса, застосовують для дослідження часових рядів.

Маючи часовий ряд X_t, модель авторегресії середнього ковзного дає змогу пояснити і, можливо, передбачити майбутні значення ряду. Модель складається з двох частин: авторегресійної (AR) частини та середнього ковзного (MA). Для згадки моделі зазвичай використовують позначення $ARMA(p, q)$, де p - порядок регресійної частини, а q - порядок ковзного середнього.

Поєднання $AR(p)$ використовується для позначення авторегресійної моделі порядку p. $AR(p)$ записується таким чином:

$$X_t = c + \sum_{i=1}^{p} \varphi_i X_{t-i} + \varepsilon_t$$

де $\varphi_1, \ldots, \varphi_p$ – параметри моделі, с – константа, а ε_t – білий шум. Для простоти константу часто опускають. По суті своїй авторегресійна модель є полюсним фільтром з нескінченною імпульсною характеристикою, витлумаченим у контексті аналізу часових рядів. Для того, щоб модель була стаціонарною, потрібно накласти деякі обмеження на параметри моделі. Наприклад, при $|\varphi_i| \geq 1$ модель $AR(1)$ не матиме властивості стаціонарності.

Модель середнього ковзного порядку q позначається $MA(q)$ і записується таким чином:

$$X_t = c + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} + \varepsilon_t$$

де $\theta_1, \dots, \theta_q$ – параметри моделі, а $\varepsilon_t, \dots, \varepsilon_{t-q}$ – помилки. Середню ковзну можна розглядати, як інтерпретацію фільтра з кінцевою імпульсною характеристикою.

Під позначенням $ARMA(p, q)$ розуміється модель, що містить p авторегресійних складових і q ковзних середніх. Точніше модель $ARMA(p, q)$ містить у собі моделі $AR(p)$ і $MA(q)$:

$$X_t = c + \varepsilon_t + \sum_{i=1}^{p} \theta_i \varepsilon_{t-i} + \sum_{i=1}^{p} \varphi_i X_{t-i}$$

Зазвичай значення помилки ε_t уважають незалежними та однаково розподіленими випадковими величинами, взятими з нормального розподілу з нульовим середнім: $\varepsilon_t \sim N(0, \sigma^2)$, де σ^2 – дисперсія. Припущення можна послабити, але це може призвести до зміни властивостей моделі. Наприклад, якщо не припускати незалежності та однакового розподілу помилок, поведінка моделі істотно змінюється.

Можливе й інше визначення моделі ARMA – за допомогою оператора затримки L. У цьому разі авторегресійна модель $AR(p)$ задається формулою:

$$\varepsilon_t = \left(1 - \sum_{i=1}^{p} \varphi_i L_i\right) X_t = \varphi X_t$$

де $\varphi -$ поліном

$$\varphi = 1 - \sum_{i=1}^{p} \varphi_i \, L_i$$

Модель $MA(q)$ задається таким чином:

$$X_t = \left(1 + \sum_{i=1}^{q} \theta_i \, L_i \right) \varepsilon_t = \theta \varepsilon_t$$

де $\theta -$ поліном

$$\theta = 1 - \sum_{i=1}^{p} \theta_i \, L_i$$

Нарешті, модель $ARMA(p, q)$ описується формулою

$$\left(1 - \sum_{i=1}^{p} \varphi_i \, L_i \right) X_t = \left(1 + \sum_{i=1}^{q} \theta_i \, L_i \right) \varepsilon_t$$

або коротко: $\varphi X_t = \theta \varepsilon_t$

Деякі автори, такі як, наприклад, Бокс, Дженкінс і Рейнсел (1994) обчислюють авторегресійні коефіцієнти за іншими правилами. Це дає змогу записати поліноми, що залежать від оператора затримки, у схожому вигляді. У цих позначеннях модель $ARMA(p, q)$ записується, як:

$$\left(1 - \sum_{i=1}^{p} \phi_i \, L_i \right) X_t = \left(1 + \sum_{i=1}^{q} \theta_i \, L_i \right) \varepsilon_t$$

Після вибору параметрів p і q можна за допомогою методу найменших квадратів щоб мінімізувати похибку. Зазвичай знаходять найменші p і q, за яких модель описує дані із задовільною точністю. Для налаштування "чистої" авторегресійної моделі можна використовувати систему рівнянь Юла-Волкера. [9]

1.3 Глибокі нейронні мережі в задачах прогнозування

Використання глибоких нейронних мереж для проблем регресії, у тому числі прогнозування індексу акцій, може здатися надмірним заходом (і досить часто так і є), але в деяких випадках, коли існує значна кількість багатовимірних даних, нейронні мережі можуть перевершити будь-які інші моделі ML.

1.3.1 Персептрон

Багатошаровий персептрон (MLP) — це клас feedforward штучної нейронної мережі (ANN), яка використовує backpropagation як техніку навчання. У багатошарових нейронних мережах перцептронів вихід кожного шару утворює вхід наступного шару. $P.H_1.H_2.H_3.Q$ описує архітектуру MLP із трьома прихованими рівнями, де H_1, H_2 і H_3 — приховані рівні, а P і Q — вхідний і вихідний рівні відповідно. Цю мережу можна перевести в матричну форму як функцію $f: R^D \rightarrow R^L$, де D — розмір вхідного вектора x, а L — розмір вихідного вектора f(x).

Припускаючи, що навчальний набір даних $D = \{x_n, t_n\}^N{}_{n=1}$, ми можемо сформувати матрицю даних X, що містить навчальні дані таким чином:

$$\mathrm{X}=\begin{pmatrix} x_1^{\mathrm{T}} \\ : \\ x_N^{\mathrm{T}} \end{pmatrix}$$

$$\mathrm{N\ x\ (P+1)}$$

Для того, щоб MLP міг вивчати зв’язки між змінними на основі навчальних даних, першим етапом є створення системи, що з’єднує вхідний рівень із першим прихованим шаром:

$$\begin{matrix} y_1 & \Omega_1 & x \end{matrix}$$
$$H_1 \times (P+1) = H_1 \times (P+1)(P+1) \times 1$$

Де $\Omega_1 -$ перша вагова матриця. Подібним чином перший і другий приховані шари, другий і третій приховані шари, останній прихований шар і вихідний шар з’єднані таким чином:

$$\begin{matrix} y_2 & \Omega_2 & y_1 \end{matrix}$$
$$H_2 \times (H_1+1) = H_2 \times (H_1+1) H_2 \times (P+1)$$

$$\begin{matrix} y_3 & \Omega_3 & y_2 \end{matrix}$$
$$H_3 \times (H_2+1) = H_3 \times (H_2+1) H_2 \times (H_1+1)$$

$$\begin{matrix} z & \gamma & y \end{matrix}$$
$$Q \times 1 = Q \times (H_3+1) H_3 \times (H_2+1)$$

де Ω_2, Ω_3 та γ – друга, третя та останні вагові матриці відповідно.

Розмірність кожної матриці/вектора в рівняннях змінюється залежно від кількості нейронів у вхідному, прихованому та вихідному шарах MLP. Наприклад, перше пояснює зв'язок між вхідним шаром і першим прихованим

шаром; отже, розмірність відповідної матриці/вектора є кількістю нейронів у продукті між вхідним шаром і першим прихованим шаром, $P \times H_1$. Розмірність інших матриць/векторів визначається подібним чином.

Нейрони MLP використовують нелінійні функції активації. Дві загальні функції активації MLP є сигмоїдами: функція активації Tan-Sigmoid (tansig) і функція логаритмічної активації Log-Sigmoid (logsig). Функція активації Tan-Sigmoid така:

$$tansig(n) = \frac{2}{1 + \exp(-2 * n)} - 1$$

що коливається від -1 до 1

Тоді як функція активації Log-Sigmoid виглядаэ наступним чином:

$$logsig(n) = \frac{1}{1 + \exp(-n)}$$

що коливається від 0 до 1.

MLP складається з трьох або більше рівнів, включаючи вхідний, вихідний та один або більше прихованих рівнів. Кожен нейрон одного шару приєднується з певною вагою, wij, до іншого нейрона наступного шару. Після обробки кожного фрагмента даних, тобто коли в персептроні відбувається навчання від зміни ваг з'єднання, у залежності від кількості помилок у виводі. Ця сума помилки визначається шляхом порівняння її з очікуваним результатом. Помилка у вихідному нейроні j у n-й точці даних обчислюється за допомогою

$$e_j(n) = d_j(n) - y_j(n)$$

де d — цільове значення, а y — значення, створене перцептроном. Вагові коефіцієнти коригуються відповідно до поправок, які мінімізують помилку в усьому виведенні, заданому

$$\varepsilon(n) - \frac{1}{2}\sum_j e_j^2(n)$$

Використовуючи градієнтний спуск, зміна кожної ваги обраховується як:

$$\triangle w_{ji}(n) = -\eta \frac{\partial \mathcal{E}(n)}{\partial \upsilon_j(n)} y_i(n)$$

де y_i — вихід попереднього нейрона, а η — learning rate, яка вибирається для того, щоб гарантувати швидке сходження вагових коефіцієнтів до відповіді без будь-яких коливань.

Розрахунок похідної залежить від індукованого локального поля υ_j, яке саме по собі змінюється. Можна довести, що для вихідного нейрона цю похідну можна спростити до

$$-\frac{\partial \mathcal{E}(n)}{\partial \upsilon_j(n)} = e_j(n)\phi'(\upsilon_j(n))$$

де ϕ' є похідною функції активації, як описано раніше. Аналіз є більш складним для зміни ваг щодо прихованого нейрона, але можна показати, що відповідна похідна

$$-\frac{\partial \mathcal{E}(n)}{\partial \upsilon_j(n)} = \phi'\left(\upsilon_j(n)\right)\sum_k -\frac{\partial \mathcal{E}(n)}{\partial \upsilon_j(n)} w_{kj}(n)$$

Це залежить від зміни ваг k-го нейрона, який представляє вихідний шар. Тому, щоб змінити ваги прихованого шару, ваги вихідного шару змінюються на основі похідної функції активації.

Продуктивність багатошарового персептрона можна оцінити на основі двох показників: точності та помилки узагальнення. Точність представляє здатність моделі до навчання. Помилка узагальнення показує, наскільки точно модель здатна передбачати нові дані. Помилка узагальнення є відповідним показником продуктивності нейронних мереж. Гіперпараметрична оптимізація — це процес вибору вагових коефіцієнтів з архітектури з метою покращення продуктивності алгоритмів навчання та отримання оптимальних рішень. Гіперпараметричну оптимізацію можна використовувати для досягнення найкращої архітектури мережі, що призводить до найменших помилок узагальнення.

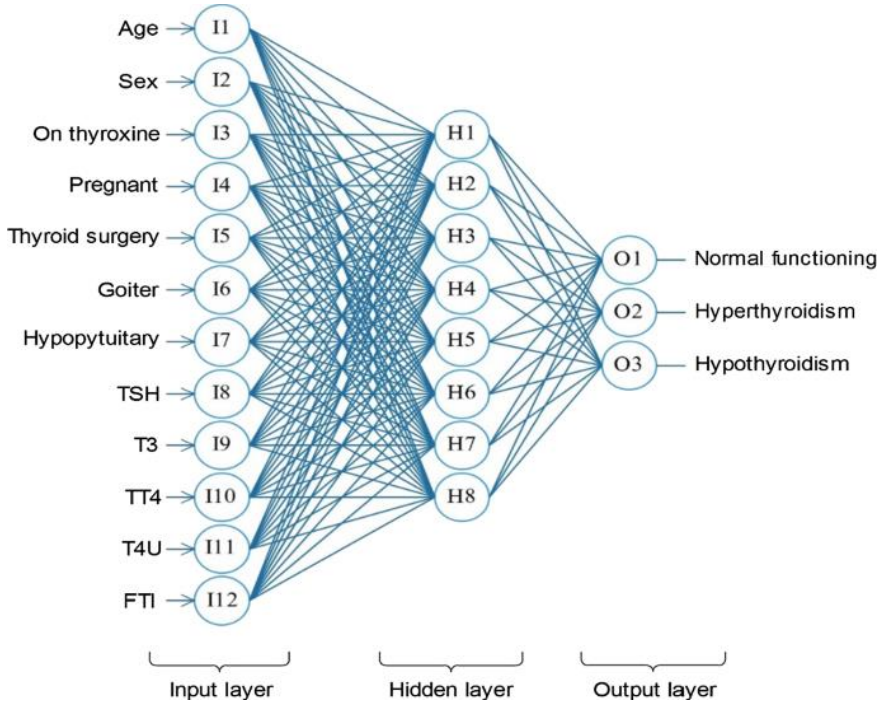

Рисунок 1.3 – Архітектура персептрону в прикладній задачі [10]

Використання нейронної мережі типу багатошаровий персептрон з адаптивним алгоритмом навчання для діагностики захворювань щитовидної залози в Інтернеті медицини зображена на рисунку 1.3. [11]

1.3.2 Згорткові нейронні мережі

На рисунку 1.4 зображена типова архітектура Convolutional Neural Network.

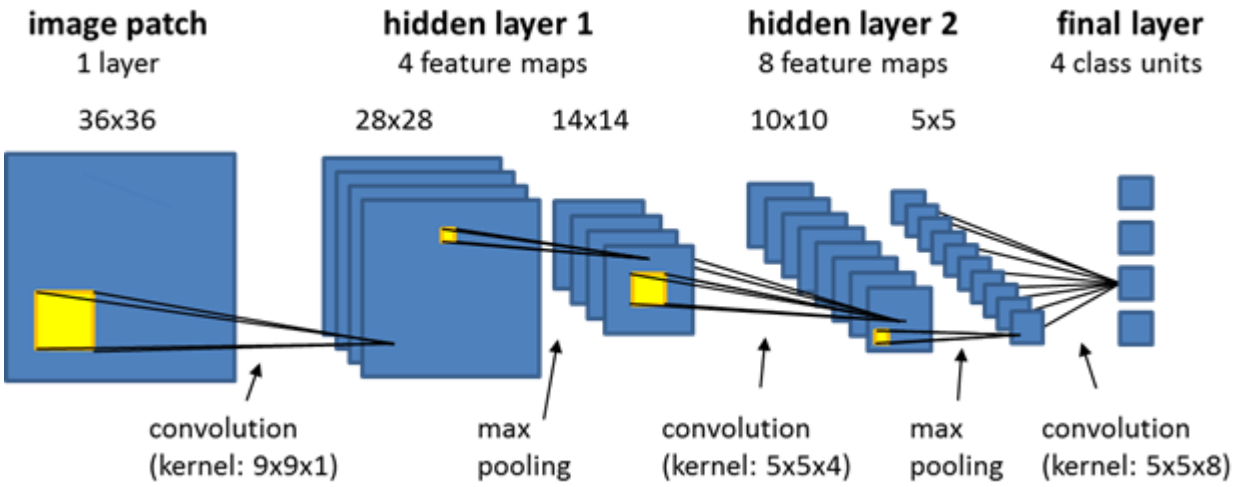

Рисунок 1.4 – Приклад типової архітектури CNN [12]

Дослідження показали, що використання CNN для класифікації часових рядів (у тому числі часових рядів що є історією зміни цій на акції та цінні папери) має кілька важливих переваг перед іншими методами. Це моделі з високою шумостійкістю, здатні витягувати дуже інформативні, глибокі риси, які не залежать від часу.

Розглянемо 1D згортку для часових рядів. Ми маємо часовий ряд довжиною n і шириною k. Довжина — це кількість часових кроків, а ширина — кількість змінних у багатовимірному часовому ряді.

Ядра згортки завжди мають таку ж ширину, як і часовий ряд, тоді як їх довжина може бути різною. Таким чином, ядро рухається в одному напрямку від початку часового ряду до його кінця, виконуючи згортку. Він не рухається ні ліворуч, ні праворуч, як це відбувається, коли метод звичайної 2-D згортки застосовується для зображень або інших двомірних наборів даних. На рисунку 1.5 зображено 1-D згортка для часових рядів.

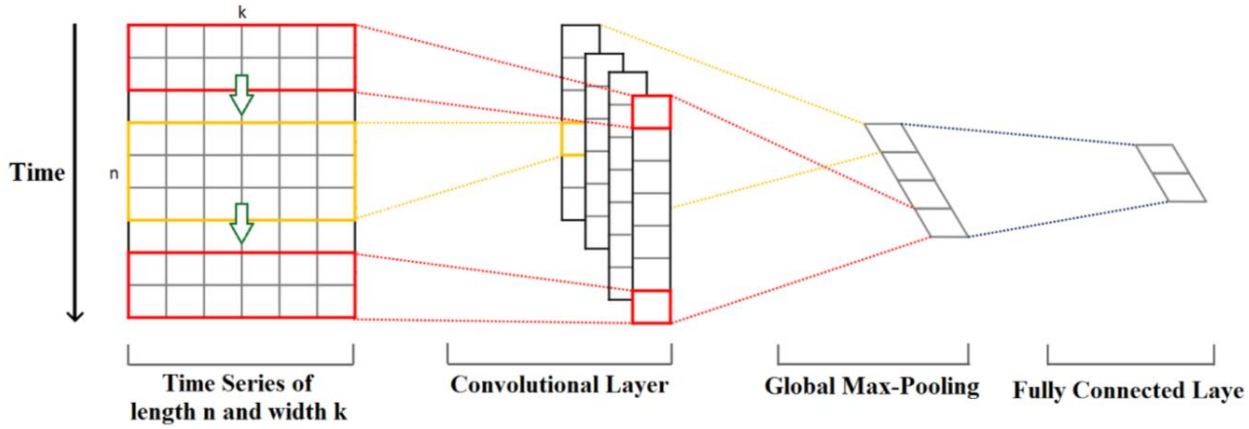

Рисунок 1.5 – 1-D згортка для часових рядів [13]

Елементи ядра множаться на відповідні елементи часового ряду, який вони охоплюють у даній точці. Потім результати множення додаються разом і до значення застосовується нелінійна функція активації. Отримане значення стає елементом нового «відфільтрованого» одновимірного часового ряду, а потім ядро рухається вперед уздовж часового ряду, щоб отримати наступне значення. Кількість нових «відфільтрованих» часових рядів дорівнює кількості ядер згортки. Залежно від довжини ядра, різні аспекти, властивості, «особливості» початкового часового ряду фіксуються в кожному з нових відфільтрованих рядів.

Наступним кроком є застосування глобального max-pooling до кожного з відфільтрованих векторів часового ряду: з кожного вектора береться найбільше значення. З цих значень формується новий вектор, і цей вектор максимумів є остаточним вектором ознак, який можна використовувати як вхідні дані для звичайного fully connected (повнозв’язного) шару. Весь цей процес показано на малюнку вище.

Далі розглянемо багатомасштабну згорткову нейронну мережу (multi-scale convolutional neural network). Багатомасштабність цієї моделі полягає в її архітектурі: на першому згортковому шарі згортка виконується на 3

паралельних незалежних гілках. Кожна гілка витягує з даних ознаки різного характеру, діючи в різних часових і частотних масштабах.

Каркас цієї мережі складається з 3 послідовних етапів: трансформації, локальної згортки та повної згортки. Багатомасштабна архітектура згорткової нейронної мережі зображена на рисунку 1.6.

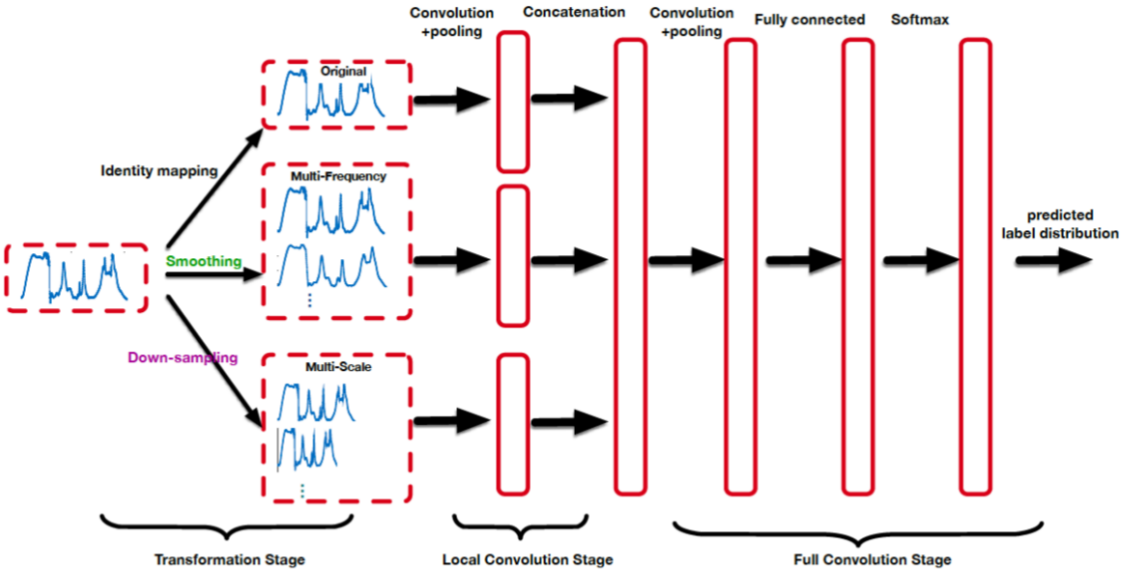

Рисунок 1.6 – Багатомасштабна архітектура згорткової нейронної мережі [14]

На етапі трансформації різні перетворення застосовуються до вихідного часового ряду на 3 окремих гілках. Перше перетворення гілки — це identity mapping, тобто вихідний часовий ряд залишається недоторканим.

Друге перетворення гілки згладжує вихідний часовий ряд за допомогою moving average з різними розмірами вікон. Таким чином створюється кілька нових часових рядів з різним ступенем гладкості. Ідея цього полягає в тому, що кожен новий часовий ряд консолідує інформацію з різних частот вихідних даних.

Нарешті, третє перетворення гілки полягає у зниженні вибірки оригінального часового ряду з різними коефіцієнтами зменшення вибірки. Чим менший коефіцієнт, тим більш детальним є новий часовий ряд, а отже, він

консолідує інформацію про особливості часового ряду в меншому часовому масштабі. Зменшення вибірки з більшими коефіцієнтами призводить до менш детальних нових часових рядів, які охоплюють і підкреслюють ті особливості вихідних даних, які проявляються у більших часових масштабах.

На етапі локальної згортки одновимірна згортка з різними розмірами фільтрів, описана вище, застосовується до часового ряду. Після кожного згорткового шару слідує max-pooling layer. У попередньому простішому прикладі використовувалося глобальне max-pooling. Тут max-pooling не є глобальним, але розмір ядра об’єднання є надзвичайно великим, набагато більшим, ніж розміри, до яких ви звикли працювати з даними зображень. Точніше, розмір ядра об’єднання визначається за формулою n/p, де n — довжина часового ряду, а p — коефіцієнт об’єднання, який зазвичай вибирається між значеннями {2, 3, 5}. Цей етап називається локальною згорткою, оскільки кожна гілка обробляється незалежно.

На етапі повної згортки всі виходи етапу локальної згортки з усіх 3 гілок об’єднані. Потім додається ще кілька шарів згортки та max-pooling. Після всіх перетворень і згорток залишається плоский вектор глибоких складних функцій, які фіксують інформацію про вихідний часовий ряд у широкому діапазоні частотних і часових масштабів. Потім цей вектор використовується як вхідні дані для повністю зв’язаних шарів із функцією Softmax на

1.3.3 Рекурентні нейронні мережі

Рекурентна нейронна мережа (RNN) - це вдосконалена форма нейронних мереж, яка має внутрішню пам'ять, що робить RNN здатною обробляти довгі послідовності. Це робить ШНМ придатною для прогнозування цін на акції, що включає довгі історичні дані.

ШНМ може забезпечити досить добрий прогноз для даних часових запасів.

Приховані стани RNN задаються рівняннями:

$$St = tanh(Wxt + USt - 1 + b)$$

$$ot = c + VSt$$

де x_t - вхідний вектор у момент часу t; b та c - значення зсуву; W, U та V - вагові матриці "вхід - прихований", "прихований - прихований" та "прихований - вихід" відповідно. При роботі з даними часових рядів (наприклад, фондового ринку) може бути використаний механізм уваги, який дозволяє розділити дані на частини, щоб декодер міг використовувати певні частини при генерації нових значень. На рисунку 1.7 показано узагальнену архітектуру ШНМ.

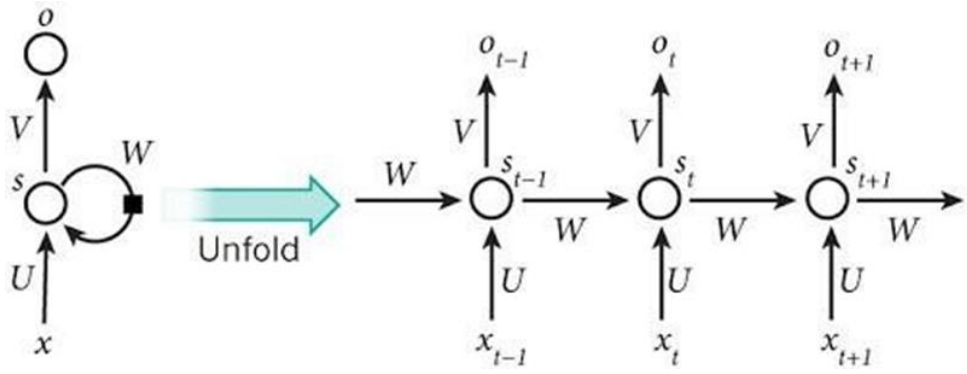

Рисунок 1.7 – Архітектура рекурентної нейронної мережі [15]

Мережа LSTM є модифікованою версією рекурентних нейронних мереж, що дозволяє легше зберігати в пам'яті минулі дані. Тут вирішується проблема зникаючого градієнта RNN. LSTM добре підходить для процесу класифікації та прогнозування часових рядів з урахуванням часових лагів невідомої тривалості. Він навчає модель за допомогою зворотного

розповсюдження. Архітектура LSTM складається з п'яти основних частин (рис. 1.8)

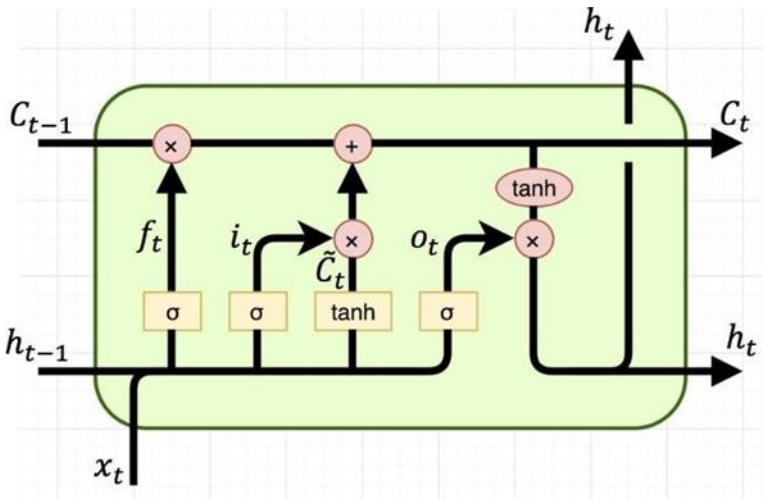

Рисунок 1.8 – Однокоміркова LSTM архітектура [15]

1. Стан комірки (c_t) - 1D вектор фіксованої форми з ініціалізацією випадкової величини. Містить інформацію, яка була присутня в пам'яті після попереднього часового кроку.
2. Forget gate (f_t) - змінює стан комірки, маючи на меті усунення неважливих значень з попередніх часових кроків. Це допомагає мережі LSTM забути неактуальну інформацію, яка не має жодного впливу на майбутній прогноз ціни.
3. Вхідні ворота (i_t) - змінює стан комірки з метою додавання нової інформації про поточний часовий крок. Він додає нову інформацію, яка може вплинути на рух ціни акцій.
4. Output gate (o_t) - вирішує, яким має бути наступний прихований стан. Новий стан комірки та нове приховане значення переноситься на наступний часовий крок. Повертає остаточну релевантну інформацію, яка буде використана для прогнозування ціни акцій.
5. Прихований стан (h_t) - обчислюється шляхом множення вектора вихідного вентилю на стан комірки вектор.

Значення цих векторів розраховуються за рівняннями:

$$it = \sigma(W^i xt + U^i ht-1 + b^i)$$

$$ft = \sigma(W^f xt + U^f ht-1 + b^f)$$

$$ot = \sigma(W^o xt + U^o ht-1 + b^o)$$

$$ct = it * ut + ft * ct-1$$

$$ht = ot * tanh(ct)$$

де *xt* – вхідний вектор, *ct*−1 – попередній стан комірки, *ht*−1 – попередній прихований стан, W та U – вагові матриці "вхід-прихований" та "прихований-прихований", σ – логістична сигмоїдальна функція, * позначає поелементне множення.

Розглянемо модель GRU, яка як і LSTM є ще однією покращеною версією стандартного ШНМ. Для вирішення проблеми зникаючого градієнта стандартного ШНМ, GRU використовує вентиль оновлення та вентиль скидання. Це два вектори, які вирішують, яку інформацію передавати на вихід. Особливістю цих векторів є те, що їх можна навчити зберігати інформацію з давніх часів, не розмиваючи її в часі, або видаляти інформацію, яка не має відношення до прогнозу. Щоб пояснити математику, що лежить в основі цього процесу, розглянемо один блок GRU, показаний на рисунку 1.9.

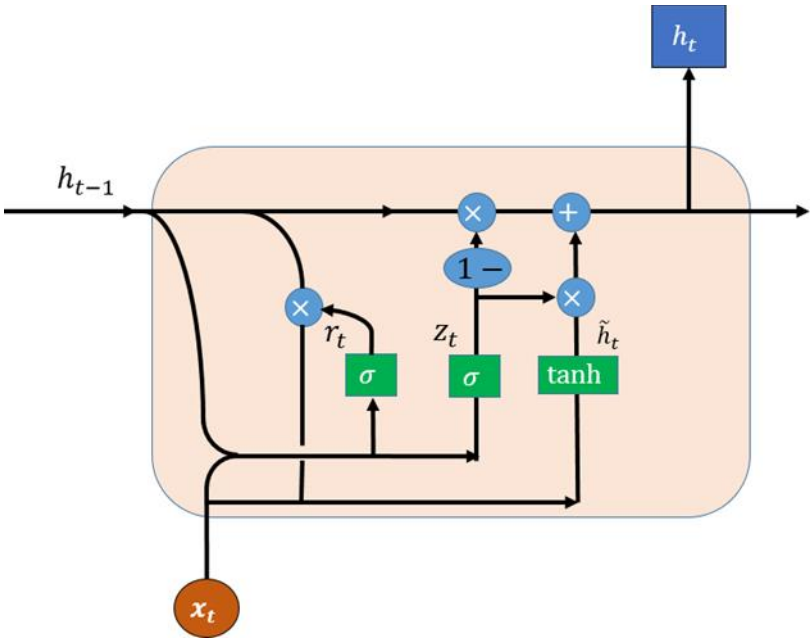

Рисунок 1.9 – Загальна архітектура окремого осередку GRU [15]

Ворота оновлення допомагають моделі визначити, скільки інформації з минулого (з попередніх часових кроків) потрібно передати в майбутнє. Це дуже потужний інструмент, оскільки модель може вирішити скопіювати всю інформацію з минулого і усунути ризик виникнення проблеми зникаючого градієнта. Він розраховується за допомогою рівняння:

$$zt = \sigma(W^z xt + U^z ht-1 + b^z)$$

Ворота скидання використовуються для прийняття рішення про те, скільки минулої інформації забути, і розраховуються за рівнянням:

$$rt = \sigma(W^r xt + U^r ht-1 + b^r)$$

Кінцевий вихід комірки розраховується за формулою:

$$rt = zt * ht-1 + (1 - zt) * h\tilde{}t)$$

де вектор активації кандидата:

$$h\tilde{}t = tanh(Wxt + rt * Uht - 1 + b(h)).$$ [15]

1.3.4 Моделі трансформери

Трансформер – це нова архітектура нейронних мереж, що спрямована перш за все на вирішення завдань пов’язаних з обробкою довгострокових залежностей.

Уперше цей підхід було запропоновано у 2017 році у роботі «Attention Is All You Need» («Увага це все, що вам потрібно») дослідниками Ашіш Васвані, Ноам Шазір, Нікі Пармар, Якоб Ушкорейт, Лліон Джонс, Ейдан Н. Гомес, Лукаш Кайзер, Ілля Полосухін на базі Корнельського Університету. За останні п’ять років технологія займає провідне місце серед сучасних технік Обробки природної мови (Natural language processing).

Обчислення вхідних і вихідних даних без використання RNN або згорток і повністю покладається на механізми самоуваги. Такий підхід у комбінації з інформативними наборами даних, тривалим тренуванням та тюнінгом мережі, дає дуже хороші результати для NLP. У тому числі, дозволяє враховувати число та стать підметів у простих та складних реченнях та застосовувати їх до присудків та інших частин речення.

Після обговорення і розуміння деяких моментів, вдосконалення моделі трансформера для цілей часових рядів (у тому числі передбачення курсу акцій) не складає багато труднощів.

На відміну від NLP, де матриці уваги повинні фокусуватись у більшості навколо сусіднього слова, тут матриці уваги тягнуться далеко в минуле, щоб врахувати такі тенденції, як сезонність (добову, тижневу, місячну чи навіть

річну). Це, звичайно, означає, що здатність обробляти довші послідовності є ключовою ціллю роботи трансформера з часовими рядами.

Через функцію активації Softmax, що широко використовується в трансформаторах для роботи з текстом, у часових рядах створює забагато ненульових ваг. Це може зменшити вагу реально релевантних даних. Саме використання функції активації Softmax стає сумнівним. Рекомендується використовувати α-entmax, простий шар, натомість. Якщо необхідно, механізм уваги може збільшити масштаб (тобто призначити 100% вагу) одному окремому токену на 500 кроків назад. Різноманітні моделі розрідженої уваги зараз є трендом у трансформаторах (як для NLP, так і для часових рядів). У 2020 році на таких моделях було випущено не менше десятка якісних і результативних моделей.

Переваги використання трасформерів для NLP (першочергового поля застосування) добре описують і позитивні наслідки перенесення цієї концепції на інші галузі дослідження.

1. Як і було згадано раніше, моделі-трансформери мають потенціал для розуміння зв’язку між послідовними елементами, навіть якщо вони знаходяться далеко один від одного.

2. Увага, що приділяється маленькому артиклю співрозмірна з будь-яким іншим елементом послідовності. Та саме актуально для дивіацій в часових рядах.

3. Ця архітектура дозволяє обробляти та тренувати значно більше даних за менший час. Навіть додаткове використання процесорів, не псує цю репутацію.

4. Універсальність у роботі з практично будь-якими видами послідовних даних, будь-то текст, зображення, звукові доріжки, часові ряди, тощо.

6. Трансформери також можуть служити для виявлення аномалій в даних, саме завдяки довгостроковій пам’яті.

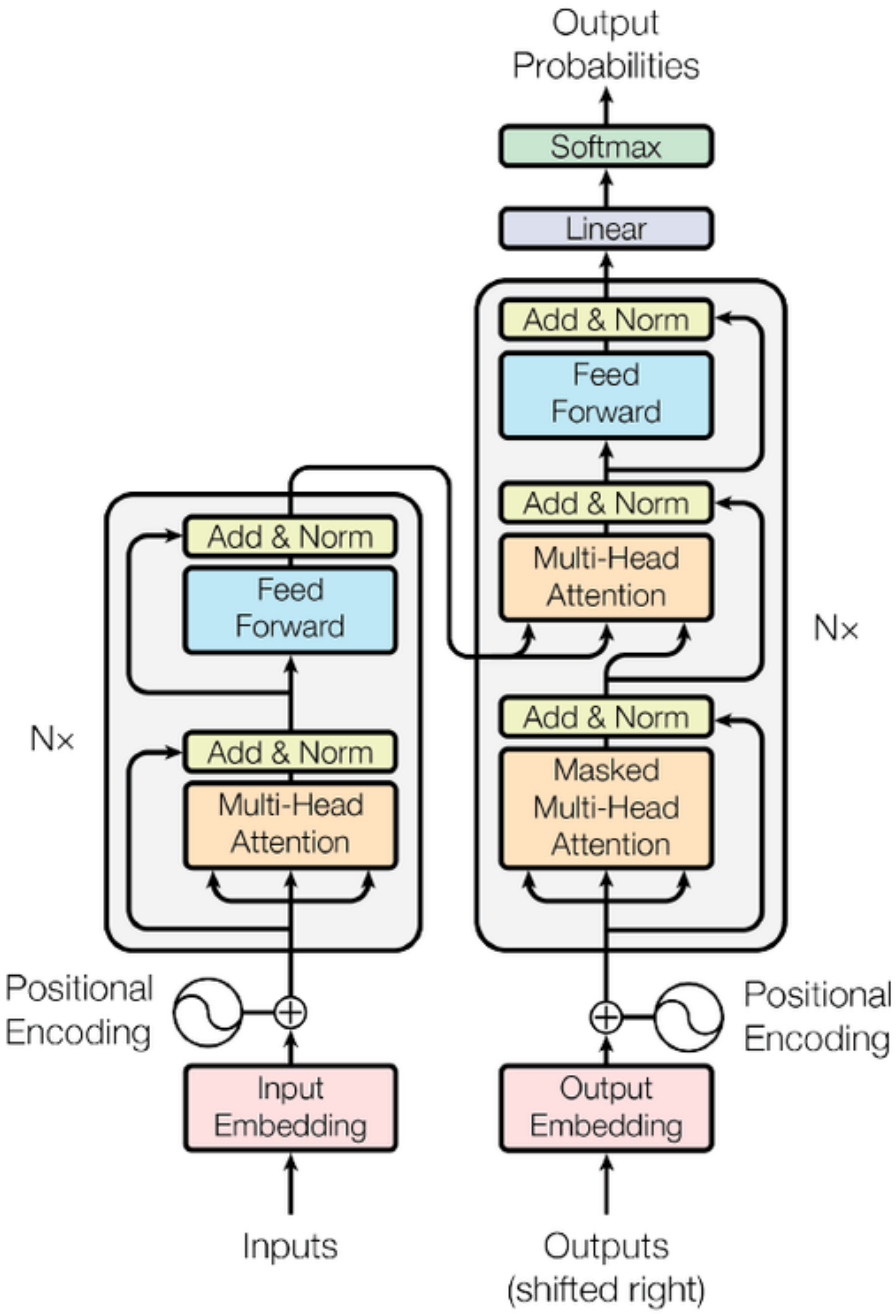

Рисунок 1.10 – Базова архітектура мережі трансформера [16]

Модель трансформер заснована на архітектурі кодер-декодер (Рис. 1.10). Ці два елементи складаються з двох і трьох підрівнів відповідно.

Енкодер (Encoder): кодер відповідає за покрокове проходження вхідних даних (часових кроків або слів) і кодування всієї послідовності у вектор фіксованої довжини, який називається контекстним вектором (context vector) (Рис. 1.11).

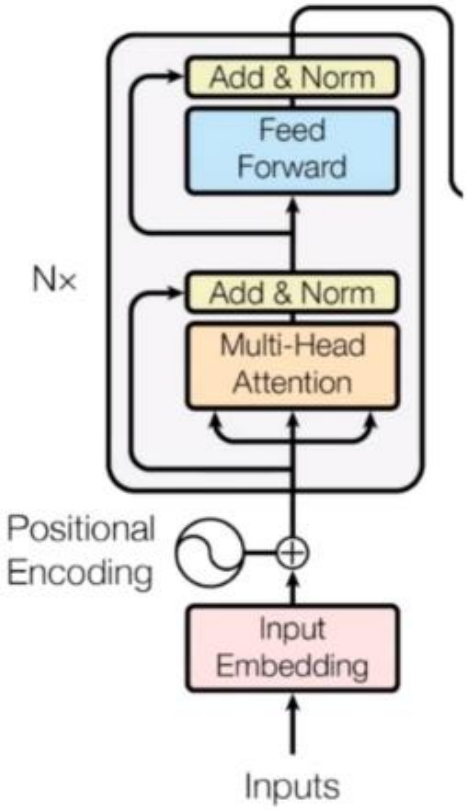

Рисунок 1.11 – Енкодер [16]

Декодер (Decoder): декодер ж у свою чергу відповідає за поступове читання з вектора контексту (по часових кроках чи словах), зображено на рисунку 1.12.

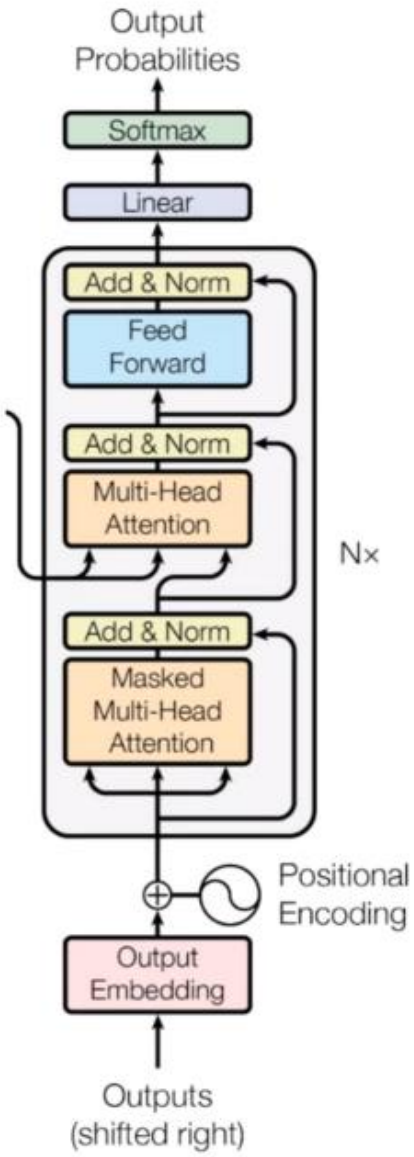

Рисунок 1.12 – Декодер [16]

Самоувага (Self-attention) – це нова техніка для технології уваги у мережі-трансформері. Замість перегляду попередніх прихованих векторів під час розгляду вбудовування слів (word embedding), самоувага є зваженою комбінацією всіх інших вбудовувань кроків (включаючи ті, які з’являються пізніше в реченні). Для прикладу слів, що послідують за обраним, або часовим періодам у майбутньому.

Самоувага виконується наступним чином:

Вбудовування слів (word embedding) перетворюється на три окремі матриці — запити, ключі та значення (queries, keys, values) — шляхом множення вбудовування слів на три матриці з вивченими вагами. Ці вектори навчаються та оновлюються в процесі навчання.

1. Оцінка за перше крок обчислюється шляхом скалярного добутку вектора запиту (q1) на ключові вектори (k1, k2, k3) усіх кроків.
2. Потім ці показники діляться на 8, що є квадратним коренем із розмірності ключового вектора (key vector).
3. Потім ці оцінки нормалізуються за допомогою функції активації softmax.
4. Потім ці нормалізовані оцінки множаться на вектори значень (v1, v2, v3) і підсумовуються результуючі вектори, щоб отримати кінцевий вектор (z1). Це результат шару самоуваги. Потім він передається в мережу прямого зв’язку (feed-forward network) як вхідний сигнал.
5. Ця процедура повторюється для усіх кроків (слів, чи часових відрізків).

Архітектура трансформера усуває аспект залежності від часу, що притаманний архітектурі RNN (і за який данну архітектуру часто критикують), обробляючи ці етапи навчання в повністю окремій архітектурі. Таким чином, трансформери мають стільки ж лінійних шарів, скільки кроків у найдовшій посліжовності, але ці шари є відносно простими та незалежними від часу, як у випадку з RNN. Це значно спрощує обчислення (звісно якщо у випадку нейронних мереж даного покоління взагалі можна говорити про простоту) і дозволяє виконувати паралельні обчислення. Останній аспект у час багатопроцесорних комп’ютерів позволяють застосовувати кілька ресурсів ЦП і введення/виведення для виконання однієї операції (у нашому випадку навчання).

Трансформери не є можна вважати цілковитим переможцем над традиційними RNN у всіх застосуваннях. Рекурентні нейронні мережі

(recurrent neural network, RNN) все ще мають перевагу у деяких контекстах. Та все ж на задачах, де перевага на стороні трансформерів або різниця у кінцевій точності є мінімальною, саме трансформери показують нижчі витрати ресурсів. Це безумовно дуже важливо у час великий даних (Big Data).

1.4 Висновки

У даному розділі було розглянено предметну область, існуючі засоби прогнозування індексу акцій, поняття стаціонарності часового ряду, аналізу на стаціонарність та тест Дікі Фуллера, також було розглянуто авторегресійну модель ARMA та виконано огляд глибоких нейронних мереж в задачах прогнозування - штучна нейронна мережа персептрон, що буде використовуватись при реалізації моделі, його архітектура в прикладній задачі. Також розглянули згорткові нейронні мережі, їх архітектуру та процес побудови. Далі ми розглянули рекурентні нейронні мережі, такі як довга короткострокова пам’ять, як кандидат на побудову та порівняння в експерименті та GRU. Застосування моделей на основі механізму уваги (моделі трасформери) також було обрано для подальшої реалізаціїї через хороші результати в прогнозі фінансових даних, в роботі розглянули більш детально як їх застосовувати.

Оскільки предметною областю є фондовий ринок, то було описано поняття фондового ринку, дані, що можуть прогнозуватись, тобто найвища ціна, найнижча та ціни відкриття та закриття ринку. В даному випадку для прогнозування було обрано дані індексу акцій S&P 500, а саме – ціна закриття. Також було розглянуто й інші індекси, такі як Dow Jones індекс та Nasdaq.

РОЗДІЛ 2 РОЗРОБКА МОДЕЛІ СИСТЕМИ ПРОГНОЗУВАННЯ ІНДЕКСУ АКЦІЙ

2.1 Формулювання задачі прогнозування фінансових даних

Прогнозування фінансових і економічних часових рядів завжди було і залишається важким завданням через його чутливість до політичних, економічних і соціальних факторів. Хорошими прикладами є пандемія у 2020 році та повномасштабне вторгнення Росії на територію України. У першому випадку, з середини лютого до середини березня 2020 ринки акцій у Європейському Союзі та Сполучених Штатах впали на 30 відсотків. Зміну очікуваного зростання дивідендів на 1 рік можна побачити на рисунку 2.1 та зміну очікуваного зростання ВВП на 1 рік на рисунку 2.2, де A. – локдаун в Ухані, B. – карантин в Італії, C – заборона США на поїздки до ЄС та D – надзвичайний стан в США.

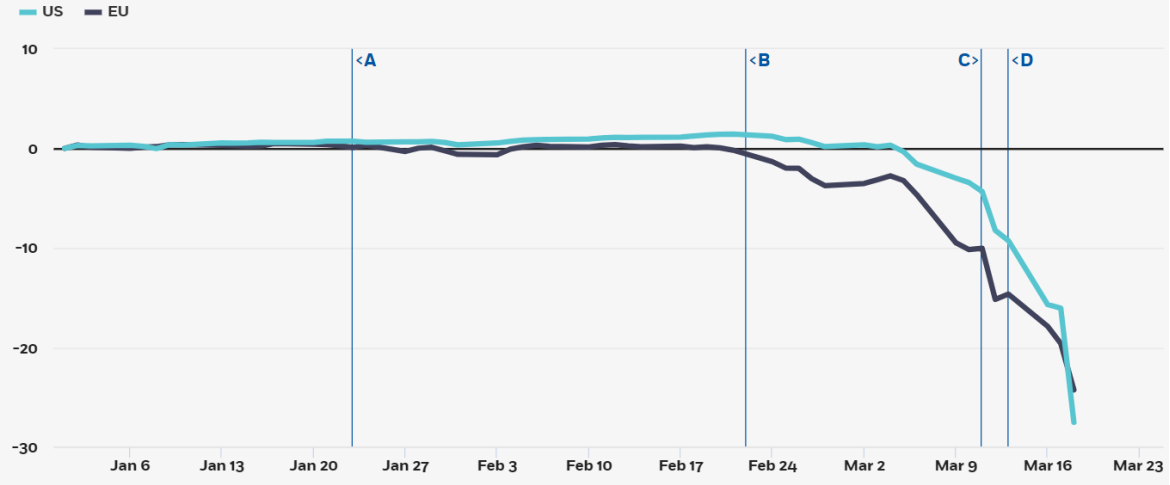

Рисунок 2.1 – Зміна очікуваного зростання дивідендів на 1 рік [17]

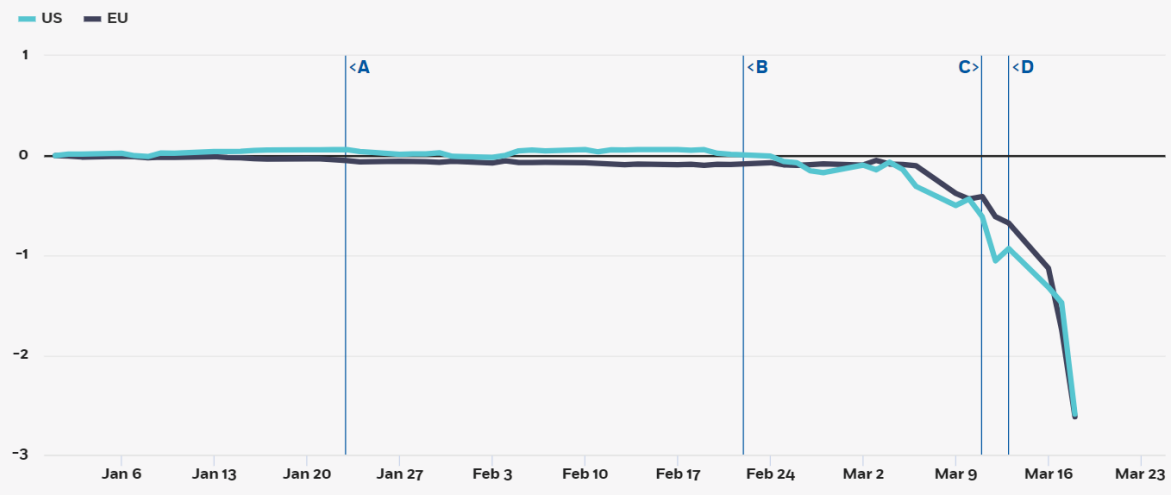

Рисунок 2.2 – Зміна очікуваного зростання ВВП на 1 рік [17]

У другому випадку війна спричинила серйозні коливання ринку і привела до певної нестабільності. Ринки здебільшого проігнорували нещодавні конфлікти, пов’язані з Близьким Сходом та Іраном. Однак війна між Росією, Україною та союзниками по НАТО може мати серйозніший вплив, особливо на ціни на нафту та інші товари. Хоча фондові ринки відновилися до рівня до вторгнення лише за кілька тижнів після вторгнення Росії в Україну, перепади спричинили серйозні фінансові втрати для багатьох інвесторів.

У США індекс S&P 500 впав більш ніж на 7% за кілька днів і тижнів відразу після вторгнення, оскільки США та інші країни посилили жорсткі економічні санкції проти Росії, а інвестори стурбовані впливом цін на сировину. Але через місяць ринки відновилися, і S&P торгувався на рівні, вищому, ніж до вторгнення, навіть якщо ціна нафти залишалася вище 100 доларів за барель. На Рис. 2.3 зображений графік стану індексу S&P 500 як почалось повномаштабне вторгнення Росії в Україну та місяць після, а на рисунку 2.4 зображена таблиця з показниками як фондові ринки відреагували на геополітичні події. На рисунку 2.5 зображений графік європейських ринків, як бачимо, вони різко впали. [20]

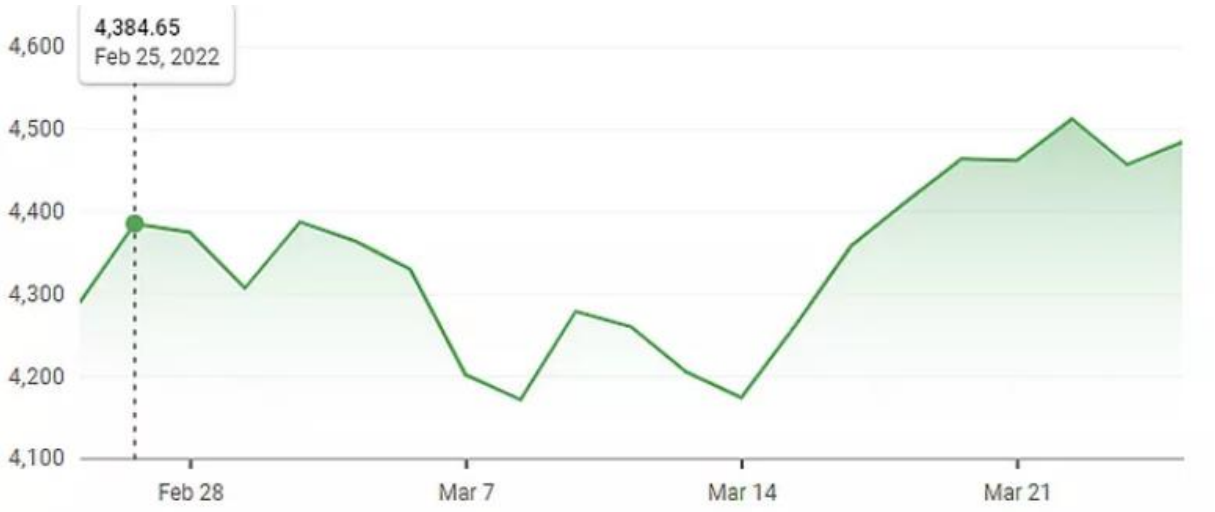

Рисунок 2.3 – Ціна індексу S&P 500 з моменту вторгнення Росії в Україну та місяць потому [18]

		S&P 500 Index		Calendar Days To	
Market Shock Events	Event Date	One-Day	Total Drawdown	Bottom	Recovery
Iranian General Killed In Airstrike	1/3/2020	-0.7%	?	?	?
Saudi Aramco Drone Strike	9/14/2019	-0.3%	-4.0%	19	41
North Korea Missile Crisis	7/28/2017	-0.1%	-1.5%	14	36
Bombing of Syria	4/7/2017	-0.1%	-1.2%	7	18
Boston Marathon Bombing	4/15/2013	-2.3%	-3.0%	4	15
London Subway Bombing	7/5/2005	0.9%	0.0%	1	4
Madrid Bombing	3/11/2004	-1.5%	-2.9%	14	20
U.S. Terrorist Attacks	9/11/2001	-4.9%	-11.6%	11	31
Iraq's Invasion of Kuwait	8/2/1990	-1.1%	-16.9%	71	189
Reagan Shooting	3/30/1981	-0.3%	-0.3%	1	2
Yom Kippur War	10/6/1973	0.3%	-0.6%	5	6
Munich Olympics	9/5/1972	-0.3%	-4.3%	42	57
Tet Offensive	1/30/1968	-0.5%	-6.0%	36	65
Six-Day War	6/5/1967	-1.5%	-1.5%	1	2
Gulf of Tonkin Incident	8/2/1964	-0.2%	-2.2%	25	41
Kennedy Assassination	11/22/1963	-2.8%	-2.8%	1	1
Cuban Missile Crisis	10/16/1962	-0.3%	-6.6%	8	18
Suez Crisis	10/29/1956	0.3%	-1.5%	3	4
Hungarian Uprising	10/23/1956	-0.2%	-0.8%	3	4
N. Korean Invades S. Korea	6/25/1950	-5.4%	-12.9%	23	82
Pearl Harbor Attack	12/7/1941	-3.8%	-19.8%	143	307
Average		**-1.2%**	**-5.0%**	**22**	**47**

Рисунок 2.4 – Реакція фондових ринків [18]

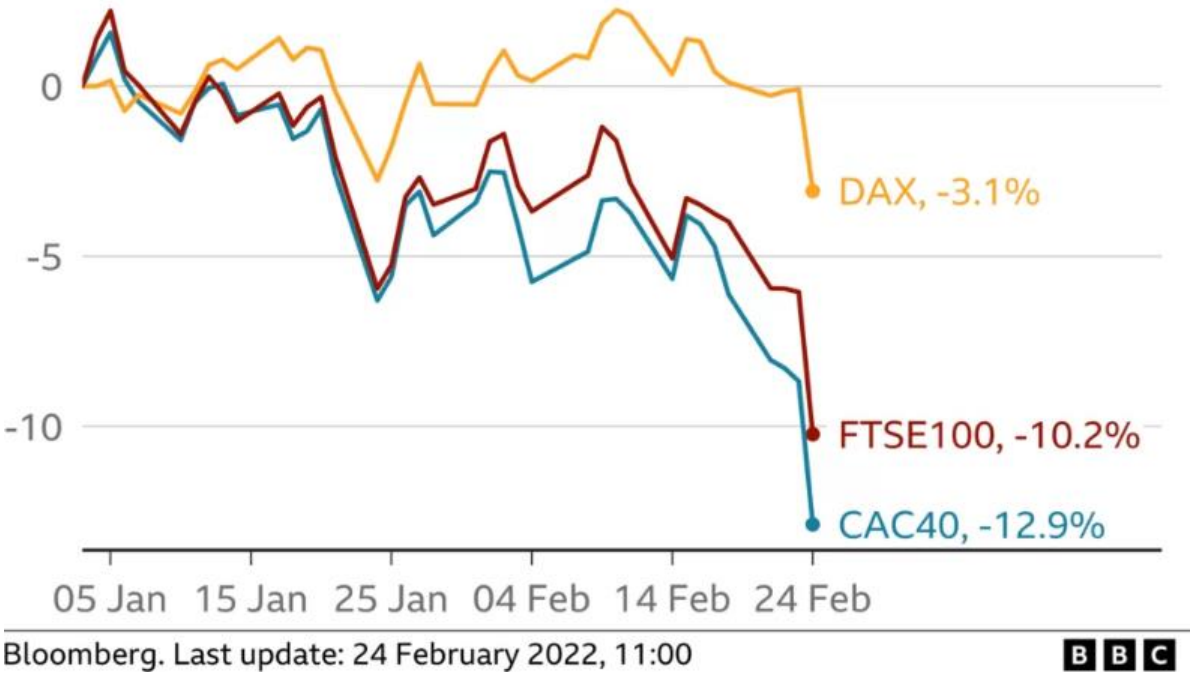

Рисунок 2.5 – Реакція фондових ринків [19]

2.2 Метрики оцінки моделі

2.2.1 Середня абсолютна похибка (MAE)

Середня абсолютна похибка – це метрика оцінки моделі, яка використовується з регресійними моделями. Середня абсолютна похибка моделі щодо тестового набору є середнім значенням абсолютних значень окремих похибок прогнозування для всіх випадків тестового набору. Кожна похибка передбачення – це різниця між справжнім значенням та передбачуваним значенням для екземпляра (формула 2.1).

$$MSE = \frac{1}{n}\sum_{i=1}^{n}|x_i - x| \tag{2.1}$$

де n – кількість похибок;

$|x_i - x|$ – абсолютна похибка. [21]

На рисунку 2.6 можна побачити графічний опис середньої абсолютної похибки, де зелена лінія – це передбачення моделі, що прогнозує, а блакитними крапками позначені наші дані. [22]

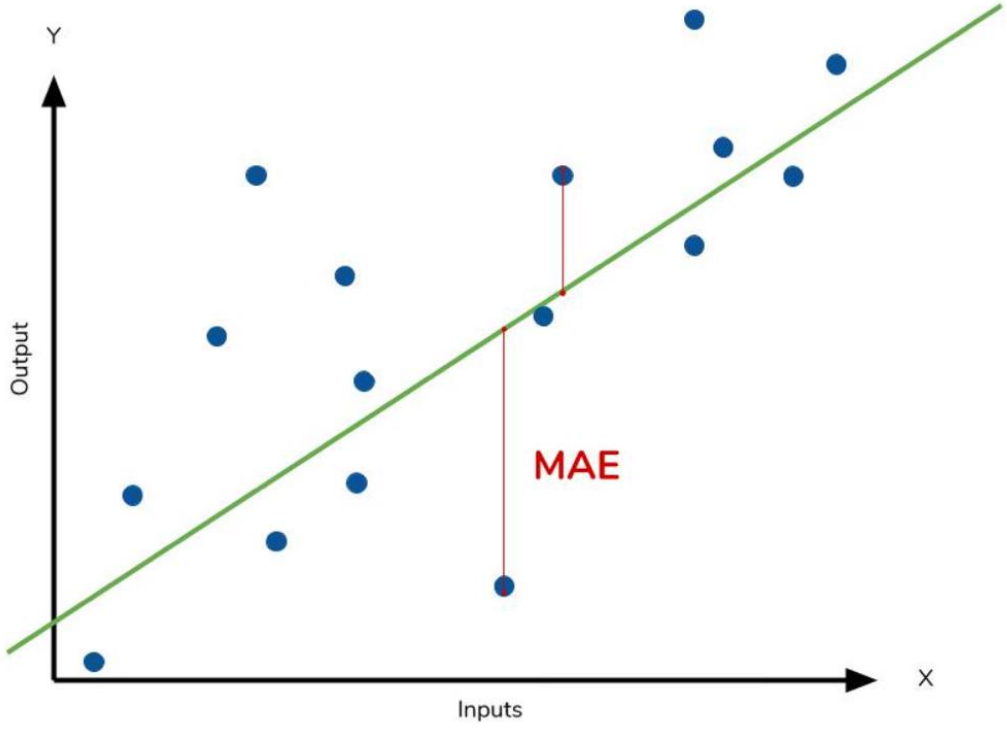

Рисунок 2.6 – Графічний опис MAE [22]

MAE також є найбільш інтуїтивно зрозумілою з метрик, оскільки просто розглядається абсолютна різниця між даними та прогнозами моделі. Оскільки ми використовуємо абсолютне значення залишку, MAE не вказує на недостатню ефективність або перевиконання моделі. Кожен залишок вносить пропорційний внесок у загальну кількість помилок, тобто більші похибки будуть лінійно сприяти загальній помилці. Невеликий MAE припускає, що модель чудово прогнозує, тоді як великий MAE припускає, що модель може мати проблеми в певних областях. MAE 0 означає, що модель є ідеальним передбачувачем результатів (але цього майже ніколи не станеться).

Хоча MAE легко інтерпретується, використання абсолютного значення залишку часто не таке бажане, як квадратування цієї різниці. Залежно від того, чи є вимога, щоб модель ставилася до випадаючих або екстремальних значень у даних, можна приділити більше уваги цим викидам або применшити їх.

2.2.2 Середньоквадратична похибка (MSE)

Середньоквадратична похибка обчислюється відповідно рівнянню (формула 2.2):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(x_i - x)^2 \tag{2.2}$$

Оскільки ми обчислюємо квадрат різниці, MSE майже завжди буде більшим, ніж MAE. З цієї причини ми не можемо безпосередньо порівнювати MAE з MSE. Ми можемо порівняти показники похибок нашої моделі лише з показниками конкуруючої моделі. Ефект квадратного члена в рівнянні MSE є найбільш очевидним із наявністю відхилень у наших даних. Хоча кожен залишок у MAE вносить свій внесок пропорційно загальній похибці, в MSE помилка зростає квадратично. Це в кінцевому рахунку означає, що відхилення в наших даних сприятимуть набагато більшій загальній похибці в MSE, ніж це було б MAE. Подібним чином наша модель буде мати більші штрафи за прогнозування, що значно відрізняється від відповідного фактичного значення. Це означає, що великі відмінності між фактичними та передбачуваними штрафуються більше у MSE, ніж у MAE. Так рисунок 2.7 наочно демонструє, як може виглядати окремий залишок в MSE.

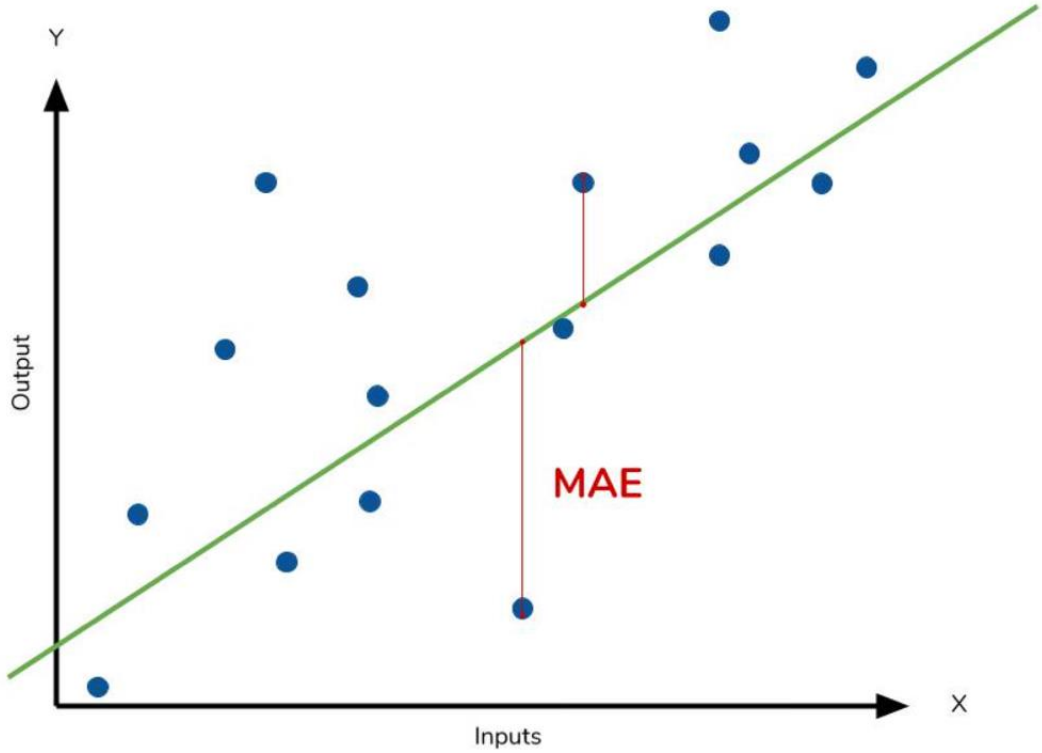

Рисунок 2.7 – Залишок в MSE [22]

як k релевантних, r_i – справжні рейтинги цих товарів, відсортовані по спаданню (ідеальне ранжування). NDCG, на відміну від MAP більше штрафує модель за рекомендації нерелевантних предметів на перших позиціях, тобто оптимізація моделі по даному критерію забезпечує видачу найрелевантніших користувачу позицій у якості перших елементів списку.

2.2.3 Середня абсолютна відсоткова похибка (MAPE)

Середня абсолютна відсоткова похибка (MAPE) - також звана середнім абсолютним відсотковим відхиленням, вимірює точність системи прогнозування. Вона вимірює цю точність у відсотках і може бути розрахована як середня абсолютна відсоткова похибка для кожного періоду часу мінус фактичні значення, поділена на фактичні значення.

Середня абсолютна похибка у відсотках (MAPE) є найбільш поширеним показником, який використовується для оцінки похибки прогнозування, ймовірно, тому, що одиниці змінної масштабуються до одиниць у відсотках, що полегшує її розуміння. Це найкраще працює, якщо в даних немає екстремальних значень (і немає нулів). Вона часто використовується як функція втрат у регресійному аналізі та оцінці моделей. [23]

Середня абсолютна відсоткова похибка (MAPE) – це відсотковий еквівалент MAE. Рівняння виглядає так само, як і MAE, але з коригуваннями для перетворення всього у відсотки (формула 2.3):

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{x_i - x}{x_i}\right| \tag{2.3}$$

Подібно до того, як MAE є середньою величиною похибки, яка оцінює модель, MAPE – це те, наскільки далеко прогнози моделі в середньому відхиляються від відповідних результатів.

Графічний опис MAPE зображено на рисунку 2.8.

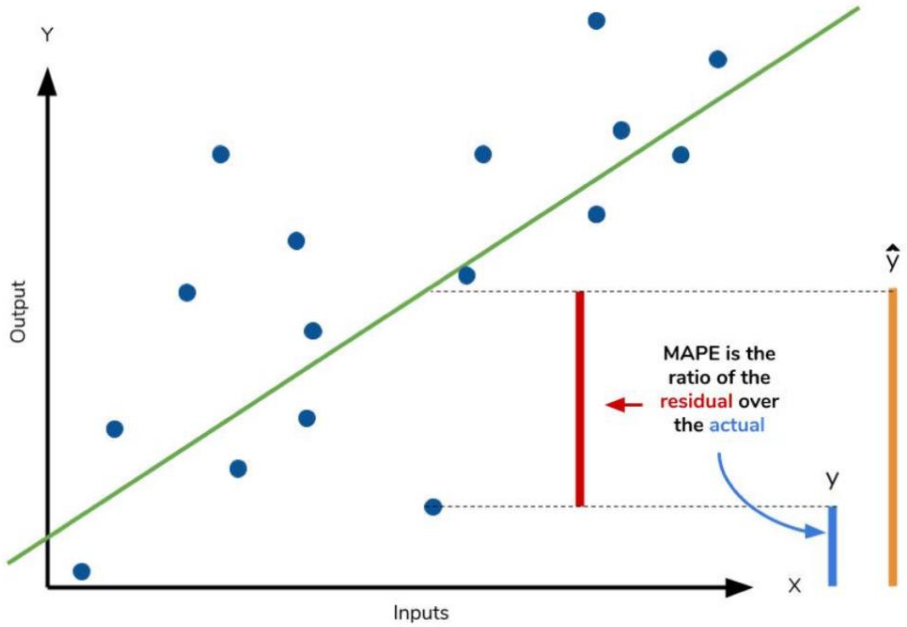

Рисунок 2.8 – Графічний опис MAPE [22]

Однак, незважаючи на всі його переваги, ми використовуємо MAPE більш обмежено, ніж MAE. Багато слабких сторін MAPE насправді пов'язані з операцією розподілу використання. Тепер, коли нам доводиться масштабувати все за фактичним значенням, MAPE не визначено для точок даних, де значення дорівнює 0. Аналогічним чином, MAPE може зростати несподівано великим, якщо фактичні значення самі по собі є виключно малими. Нарешті, MAPE схильний до прогнозів, які систематично менші за фактичні значення. Тобто, MAPE буде нижчим, коли прогноз нижчий від фактичного порівняно з прогнозом, який вищий на ту саму величину.

Відсоткові похибки розраховуються в перерахунку на абсолютні похибки, без урахування знаку. Це дозволяє уникнути проблеми взаємного погашення додатних і від'ємних похибок.

2.2.4 Коефіцієнт детермінації (R^2)

Коефіцієнт детермінації – це число від 0 до 1, яке вимірює, наскільки добре статистична модель прогнозує результат.

Інтерпретація коефіцієнта детермінації: якщо $R^2 = 0$ – це свідчить, що модель не прогнозує результат, якщо значення коливається від 0 до 1, то модель частково передбачає результат, значення, що рівне 1 показує, що модель відмінно прогнозує результат.

Для розрахунку коефіцієнта детермінації (R^2) простої лінійної регресії можна вибрати одну з двох формул (2.4 або 2.5). Перша формула є специфічною для простих лінійних регресій, а друга формула може бути використана для розрахунку R^2 багатьох типів статистичних моделей.

$$R^2 = (r)^2 \tag{2.4}$$

де r – коефіцієнт кореляції Пірсона.

$$R^2 = 1 - \frac{RSS}{TSS} \tag{2.5}$$

де RSS – сума квадратів залишків, а TSS – це загальна сума квадратів.

Коефіцієнт детермінації (R^2) можна інтерпретувати як частку дисперсії залежної змінної, яка передбачена статистичною моделлю або ж частка дисперсії, яка розподіляється між незалежною та залежною змінними. [24]

2.3 Опис запропонованого методу прогнозування

У даному розділі пропонується крок за кроком описати алгоритм роботи системи, що апроксимує наступне значення заданого часового ряду, тобто виконує прогноз на один крок.

Нехай маємо часовий ряд $Y = \{y_t \mid y_t > 0 \; \forall \; t \in \mathbb{N}\}$. Тоді алгоритм, за яким діє модель, наступний:

1. Логарифмізація ряду: $\tilde{Y} = \log Y$
2. Нормалізація ряду:
    a. Підрахунок середнього значення по вибірці: $\mu = \sum_i \frac{\tilde{y_i}}{\tilde{n}}$
    b. Підрахунок стандартної похибки по вибірці: $\sigma = \sqrt{\sum_i \left(\frac{\tilde{y_i}}{\tilde{n}} - \mu\right)^2}$
    c. Операція нормалізації: $Y_{norm} = \frac{\tilde{Y} - \mu}{\sigma}$
3. Обчислення ЧАКФ ряду Y_{norm} та виявлення порядку лагу l
4. Конструювання набору даних $X = [y_{norm_{t-1}}, \dots, y_{norm_{t-l}}]$
5. Обчислення прогнозу $\tilde{y}_{norm}$:

a. Обчислення закодованих векторів для механізму self-attention:
   i. $Q = LSTM_Q(X)$
   ii. $K = LSTM_K(X)$
   iii. $V = LSTM_V(X)$
b. Обчислення вихідного сигналу self-attention: $SA_{out} = SelfAttention(Q, K, V) = softmax(Q^T K)V$
c. Конкатенація вихідного сигналу з закодованими векторами (skip connection): $O = [Q, K, V, SA_{out}]$
d. Обчислення вектору $\tilde{y}_{norm}$ через послідовність двох повнозв'язних шарів $\tilde{y}_{norm} = W_{out}^T(W_{in}^T O)$

6. Обчислення метрик $R^2, MSE, MAE, MAPE$ відносно $\tilde{y}_{norm}$ та y_{norm}
7. Корекція прогнозу по метриці MSE методом адаптивних моментів

2.4 Компоненти системи та ресурси для їх реалізації (TensorFlow)

Глибоке навчання - це підгалузь машинного навчання, яка являє собою набір алгоритмів, натхненних структурою та функціями мозку.

TensorFlow - це другий фреймворк машинного навчання, який Google створив і використовував для проектування, побудови та навчання моделей глибокого навчання. Ви можете використовувати бібліотеку TensorFlow для чисельних обчислень, що саме по собі не здається чимось особливим, але ці обчислення виконуються за допомогою графів потоків даних. У цих графах вузли представляють математичні операції, в той час як ребра представляють дані, які зазвичай є багатовимірними масивами даних або тензорами, які передаються між цими ребрами.

Хоча на ринку існує багато фреймворків для прогнозування часових рядів, фахівці з глибоких технологій віддають перевагу TensorFlow з усіх доступних фреймворків. TensorFlow написаний на мовах Python, C++ та CUDA, які є одними з найпоширеніших мов програмування. TensorFlow - це програмна бібліотека з відкритим вихідним кодом, розроблена спеціально для машинного навчання та глибокого навчання.

TensorFlow допомагає нам у побудові ML-моделей під час аналізу часових рядів, які будуть використовуватися для прогнозування. Він пропонує користувачам будувати різні типи моделей, такі як CNN (згорткова нейронна мережа), RNN (рекурентна нейронна мережа) тощо. TensorFlow може допомогти вам спрогнозувати один часовий крок або декілька, залежно від користувача та набору даних. При прогнозуванні часових рядів за допомогою TensorFlow можна:

1. Прогнозувати одну функцію/вимір для одного часового кроку.
2. Прогнозувати всі виміри набору даних для одного часового кроку.
3. Прогнозування для декількох кроків за один раз (виконання всіх прогнозів одночасно).
4. Прогноз для декількох кроків, коли робиться один прогноз за один раз (авторегресія).
5. Підготовка даних для прогнозування часових рядів

Перед прогнозуванням часових рядів нам потрібно підготувати дані відповідним чином. Підготовка даних для прогнозування часових рядів включає в себе різні процеси, такі як вилучення даних, візуалізація, дослідження, очищення тощо. Давайте розглянемо кроки, що стоять за підготовкою даних для прогнозування часових рядів.

Основним кроком є імпорт даних або вилучення даних. Вам потрібно мати набір даних, з якого буде витягнуто багату інформацію за допомогою прогнозування часових рядів.

Наступним кроком є очищення даних, коли ви видаляєте дублікати або нерелевантні точки даних. Відсутні дані також обробляються на цьому етапі. Якщо є якісь викиди, виявлені на ранніх етапах, видаліть їх з набору даних. Структурні помилки також можуть бути виправлені під час очищення даних.

Величини в часовому ряді, які змінюються з часом, називаються сигналами. Ці сигнали також представляють різні фізичні події в часовому ряді. Наприклад, якщо є часовий ряд описів погоди в будь-якій місцевості в різні дні, то сигналами будуть фізичні події, такі як температура, кількість опадів тощо. Далі слід перетворити сигнали у форматі sin cos.

Разом з перетворенням сигналів у формат sin cos, перетворіть час і дату в секунди. Закінчивши, побудуйте графік залежності часу та функції sin cos.

Перш ніж прогнозувати та навчати наші дані, ми розділимо дані на основі часу для подальшого спрощення.

Тепер дані повинні бути нормалізовані перед навчанням. Найпростішим кроком для нормалізації даних є віднімання середнього значення набору даних з кожної точки даних, а потім ділення його на стандартне відхилення.

Після того, як ви закінчите з нормалізацією даних, ви повинні перевірити їх на зміщення. Зсув даних визначається як стан, коли деякі елементи набору даних представлені в більшій мірі порівняно з іншими елементами. Упереджені дані призводять до зниження точності, а також до викривлення аналітики. Ви можете побудувати скрипковий графік, щоб усунути будь-яке зміщення даних, наявне у вашому наборі даних.

2.5 Висновки

У даному розділі було розглянуто питання методів прогнозування фінансових та економічних рядів, наскільки зараз це є актуальним питання через вплив геополітичних подій на фондові ринки. метрики оцінки моделей,

що будуть використовуватись як критерії вибору найкращої моделі для подальшого користування. Та було описано більш детально запропонований метод прогнозування індексу акцій на основі нейронної мережі довгої короткострокової пам’яті з застосуванням нейронної мережі на основі механізму уваги. То ж для дослідженя, виходячи з аналізу існуючих методів прогнозування було обрано авторегресійну модель, мережу довгої короткострокової пам’яті та модель на основі механізму уваги для порівняння та вибору найкращої.

РОЗДІЛ 3 РЕАЛІЗАЦІЯ АЛГОРИТМІВ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

3.1 Дані для прогнозування та їх підготовка

В рамках дослідження було прийняте рішення побудувати моделі авторегресії, довгої короткострокової пам’яті та розробка методу прогнозування на основі механізму уваги (Attention network) для порівняння та вибору найкращого методу для прогнозування.

Для дослідження та побудови прогнозу було обрано дані індексу акцій S&P 500, що є зваженим за ринковою капіталізацією індексом 500 провідних публічних компаній США, а саме за період з 31.12.2019 до 31.12.2021.

Фрагмент даних наведено в таблиці 3.1.

Таблиця 3.1 – Фрагмент файлу індексу S&P 500

Date	^GSPC
2019-12-31	3230.780029
2020-01-01	NaN
2020-01-02	3257.850098
2020-01-03	3234.850098
2020-01-04	NaN

Спочатку було проведено низку маніпуляцій над вхідними даними, а саме логарифмування, визначення порядку лагу регресії для побудови вектору ознак та розділення вибірки на навчальну та тестову з подальшою нормалізацією (стандартизацією) ряду. На Рис. 3.1 зображена візуалізація логарифму показника S&P 500.

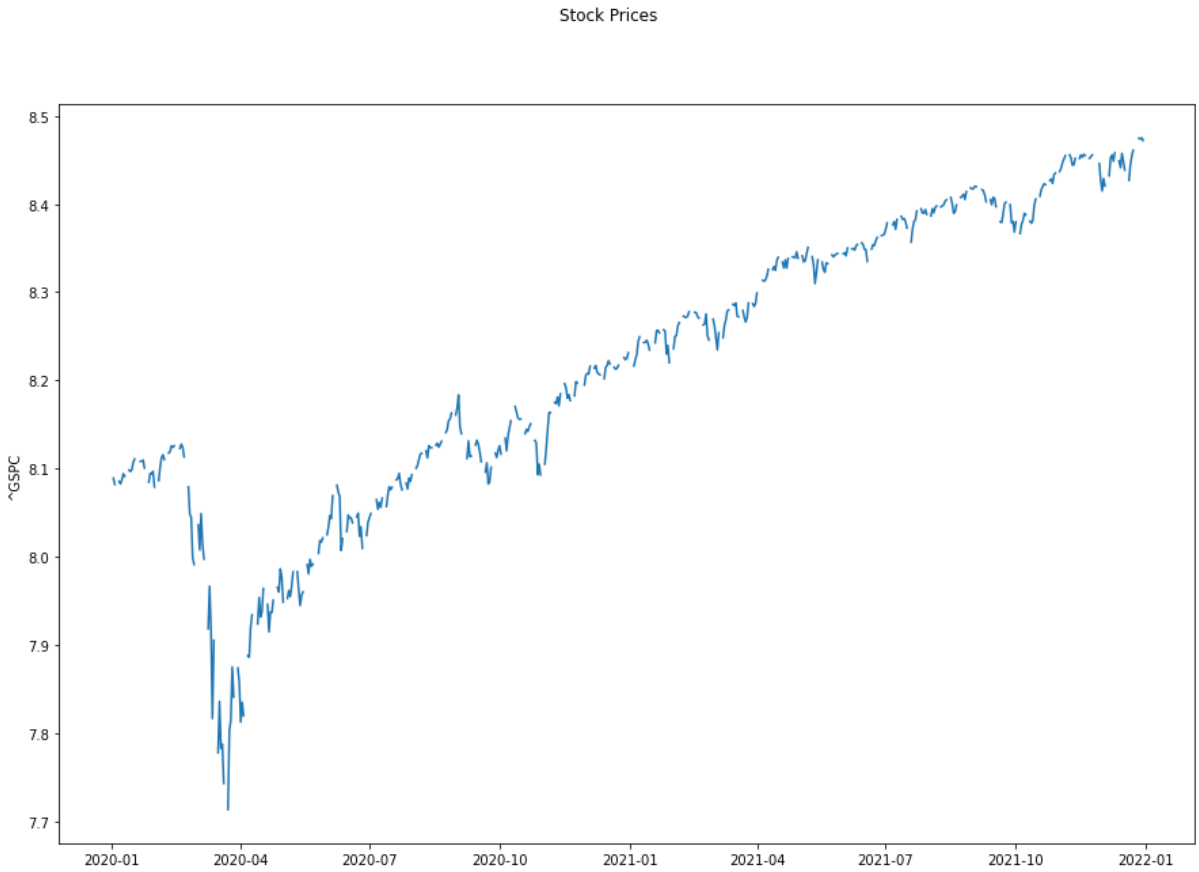

Рисунок 3.1 – Логарифм показника S&P 500

Після проведення тесту Дікі Фуллера на стаціонарність було зроблено висновок, що ряд не є стаціонарним, оскільки неможливо відхилити нульову гіпотезу про наявність одиничних коренів, про яку згадувалось в попередньому розділі. Для визначення кількості лагів, що використовуємо при прогнозуванні ряду побудуємо часткову автокореляцію, що зображена на Рис.3.2. Виходячи з візуалізації, робимо висновок, що доцільно буде використовувати три значення до прогнозованого.

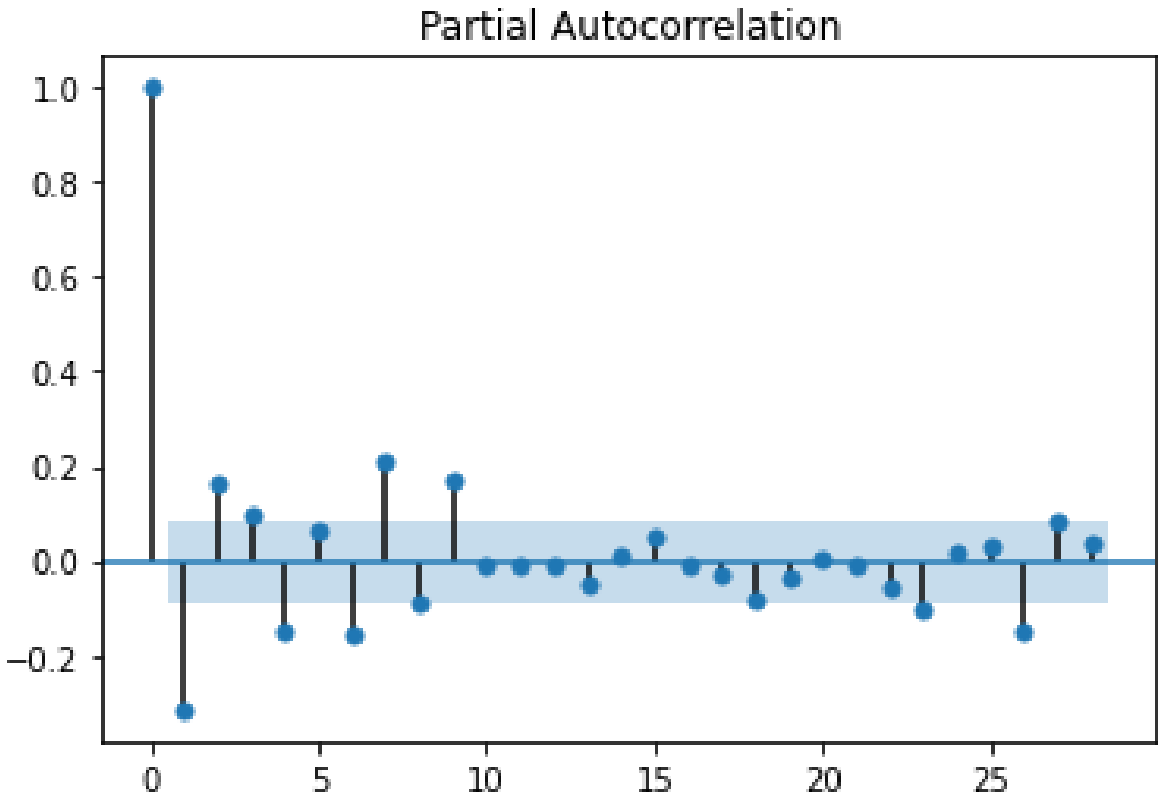

Рисунок 3.2 – Часткова автокореляція ряду

3.2 Реалізація моделей та аналіз отриманих результатів

Далі було реалізовано модель авторегресії порядку 3, у таблиці 3.2 представлено значення похибок прогнозу даної моделі. Для оцінки якості прогнозу було обрано метрики R^2, MSE, MAE та MAPE.

Таблиця 3.2 – Значення похибок прогнозу моделі лінійної регресії ARMA

	R^2	MSE	MAE	MAPE
Тренувальна	0.979615	0.000330	0.011232	0.001396
Тестова	0.957944	0.000067	0.006250	0.000745

Порівняємо прогноз моделі та справджні дані на рисунку 3.3. На верхньому графіку відображена візуалізація справжніх даних, а на нижньому отриманий прогноз за допомогою моделі лінійної регресії.

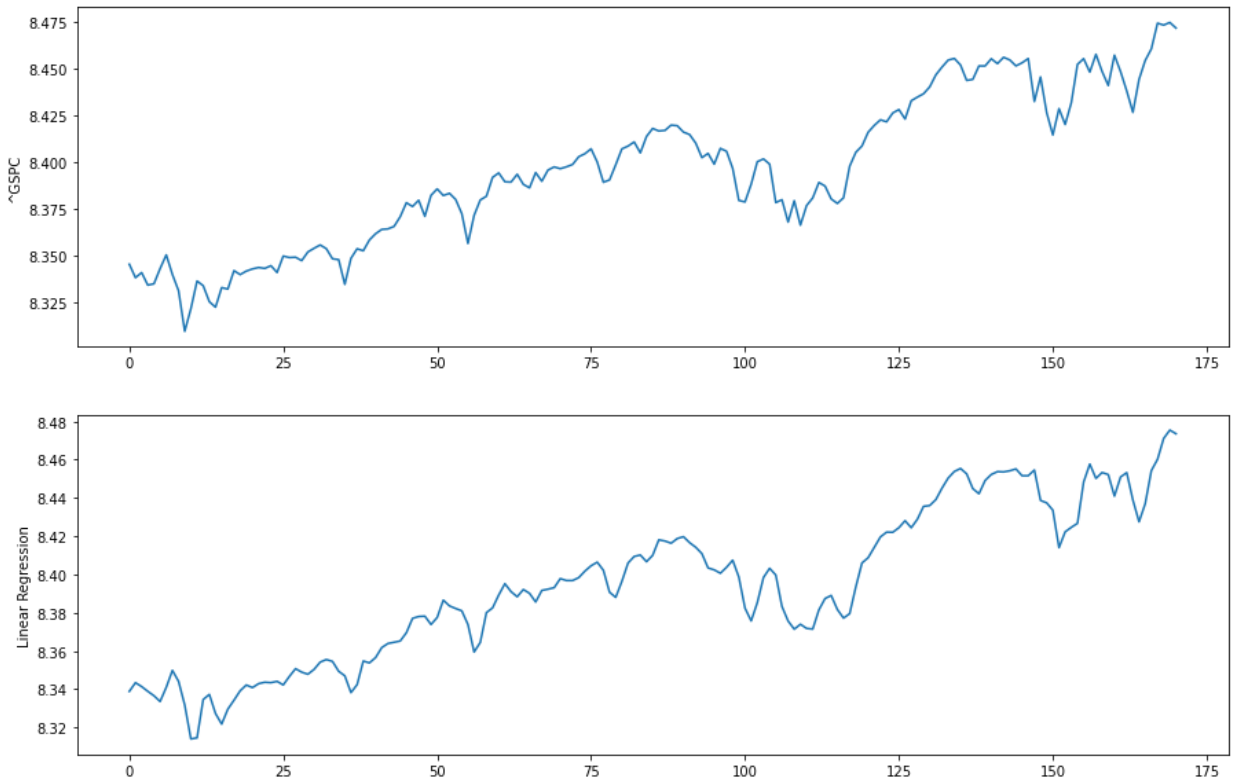

Рисунок 3.3 – Прогноз лінійної авторегресійної моделі в порівнянні зі справжніми даними

Далі було реалізовано модель довгої короткострокової пам’яті в різних конфігураціях для знаходження найкращої моделі прогнозування ряду. У таблиці 3.3 показані значення похибок прогнозу LSTM моделі для різних конфігурацій.

Таблиця 3.3 – Значення похибок прогнозу моделі LSTM

Конфігурації/ похибка	Датасет	R^2	MSE	MAE	MAPE
lstm_128_lstm_layers_128_dense_layers	Train	0.938707	0.000992	0.020548	0.25536
	Test	0.523463	0.000759	0.024207	0.288116
lstm_128_lstm_layers_64_dense_layers	Train	0.944707	0.000895	0.019438	0.241648
	Test	0.818033	0.00029	0.013642	0.162508
lstm_128_lstm_layers_32_dense_layers	Train	0.976457	0.000381	0.012501	0.155453
	Test	0.952235	7.61E-05	0.006499	0.077416
lstm_64_lstm_layers_128_dense_layers	Train	0.975333	0.000399	0.012299	0.153083
	Test	0.446925	0.000881	0.023112	0.274365
lstm_64_lstm_layers_64_dense_layers	Train	0.976643	0.000378	0.012285	0.152821
	Test	0.940457	9.49E-05	0.007643	0.090915
lstm_64_lstm_layers_32_dense_layers	Train	0.975706	0.000393	0.012973	0.161163
	Test	-0.05207	0.001676	0.035015	0.415929
lstm_32_lstm_layers_128_dense_layers	Train	0.974312	0.000416	0.012858	0.159895
	Test	-42.2321	0.068883	0.171424	2.031888
lstm_32_lstm_layers_64_dense_layers	Train	0.031065	0.015677	0.122378	1.510335
	Test	-218.689	0.350039	0.513514	6.100405
lstm_32_lstm_layers_32_dense_layers	Train	0.972265	0.000449	0.01442	0.179075
	Test	-0.10021	0.001753	0.034899	0.414404

З отриманих результатів робимо висновок, що найкраще себе показала модель з однією LSTM коміркою розміром 128 та одним прихованим повнозвязним шаром розміром 32. З візуалізацій на рисунку 3.4 видно, що модель згладжує ряд та показує непогані результати.

Порівняємо прогноз реалізованих моделей та справджні дані на рисунку 3.4. На верхньому графіку відображена візуалізація справжніх даних, далі – лінійна регресія та на нижньому LSTM.

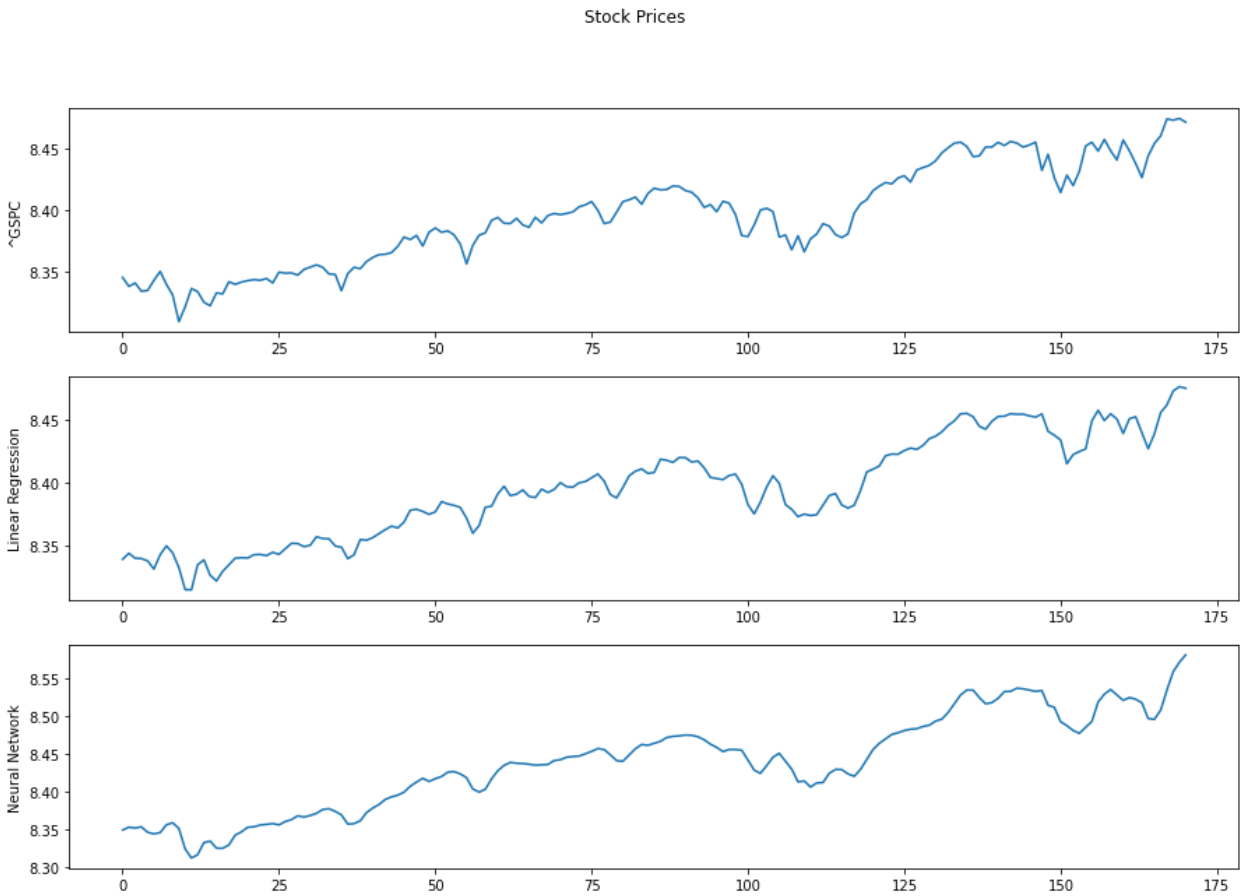

Рисунок 3.4 – Прогноз реалізованих моделей в порівнянні зі справжніми даними

Далі було реалізовано модель LSTM, що повертає послідовності, які інтерпертуються як вхідні вектори для механізму уваги, механізм уваги повертає вектор, який об’єднується з вхідною послідовністю, передається в персептрон та отримкємо нове значення. В таблиці 3.4 показані значення похибок прогнозу моделі на основі механізму уваги. На рисунку 3.5 зображені графіки реальних даних та побудованих моделей. Як бачимо, наша модель, що була побудована на основі механізму уваги, в даному випадку модель згладжує шуми та є більш точною за інші з гарною тенденцією.

Таблиця 3.4 – Значення похибок прогнозу моделі на основі механізму уваги

	R^2	MSE	MAE	MAPE
Тренувальна	0.9723	0.00023	0.0134	0.1669
Тестова	0.8824	0.00013	0.0108	0.1280

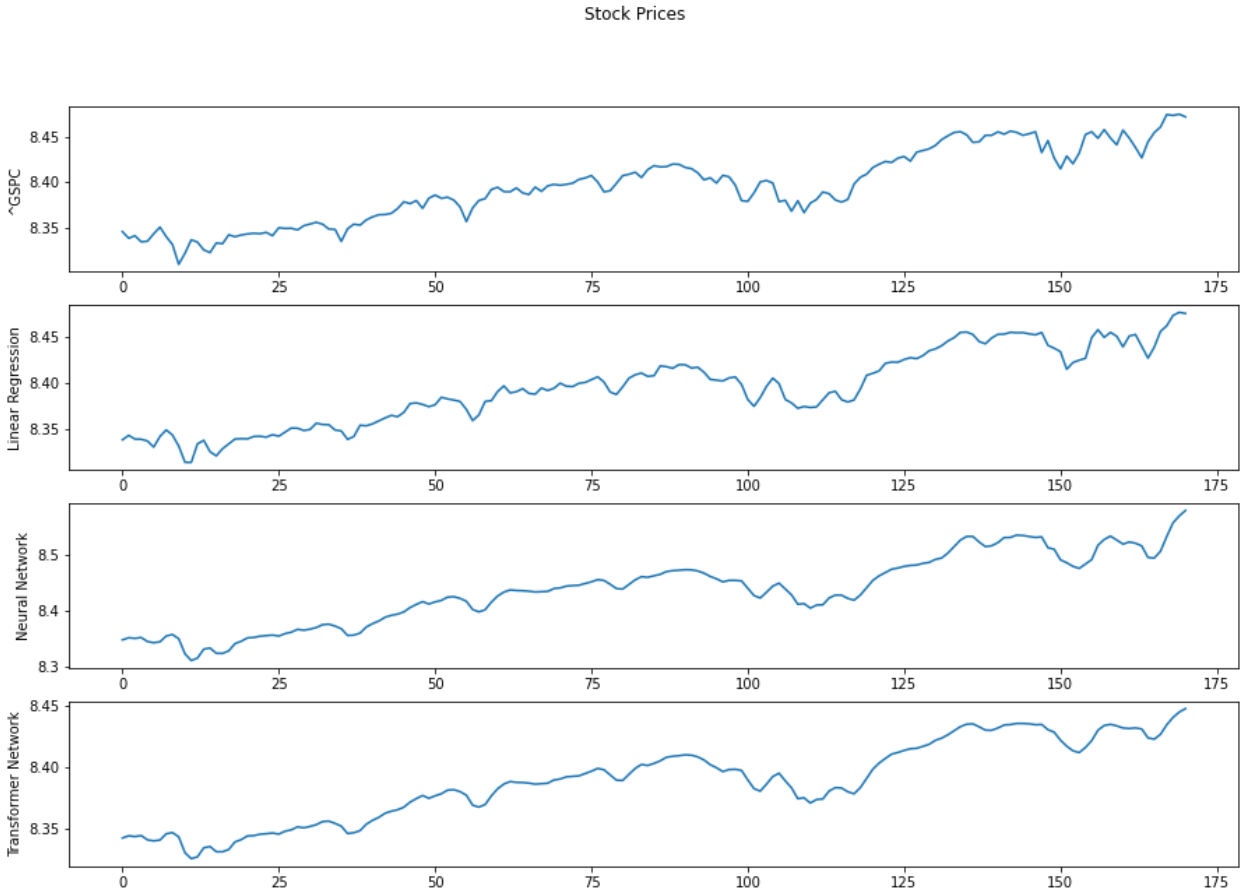

Рисунок 3.5 – Прогноз всіх реалізованих моделей в порівнянні зі справжніми даними

На рисунках 3.6 – 3.8 зображено як покращувались значення похибок з епохами найкращої моделі.

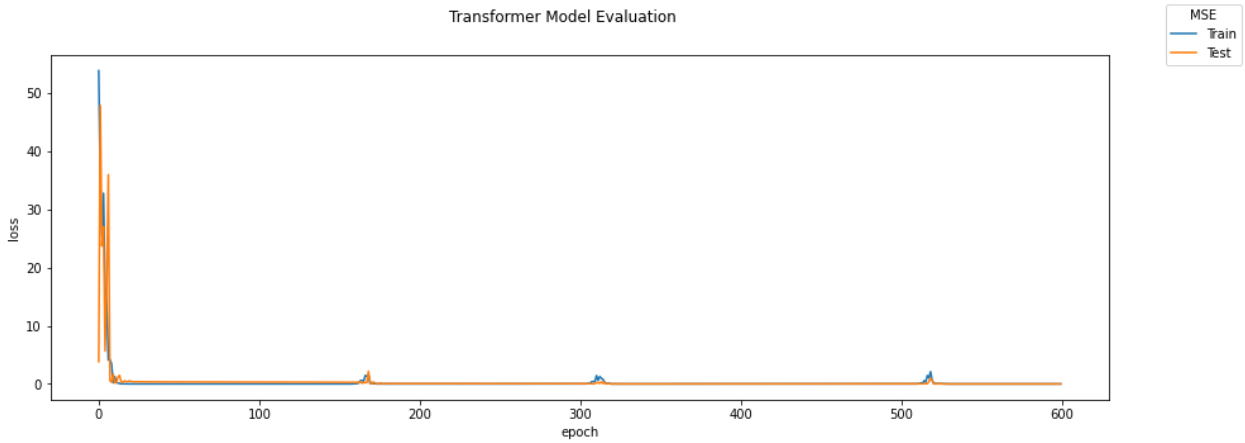

Рисунок 3.6 – Зміна MSE з плином часу

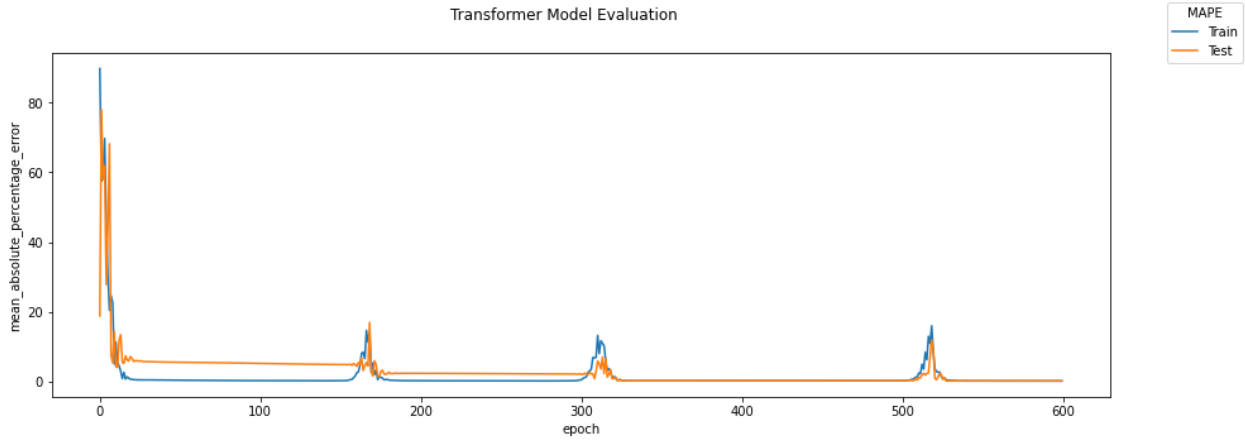

Рисунок 3.7 – Зміна MAPE з плином часу

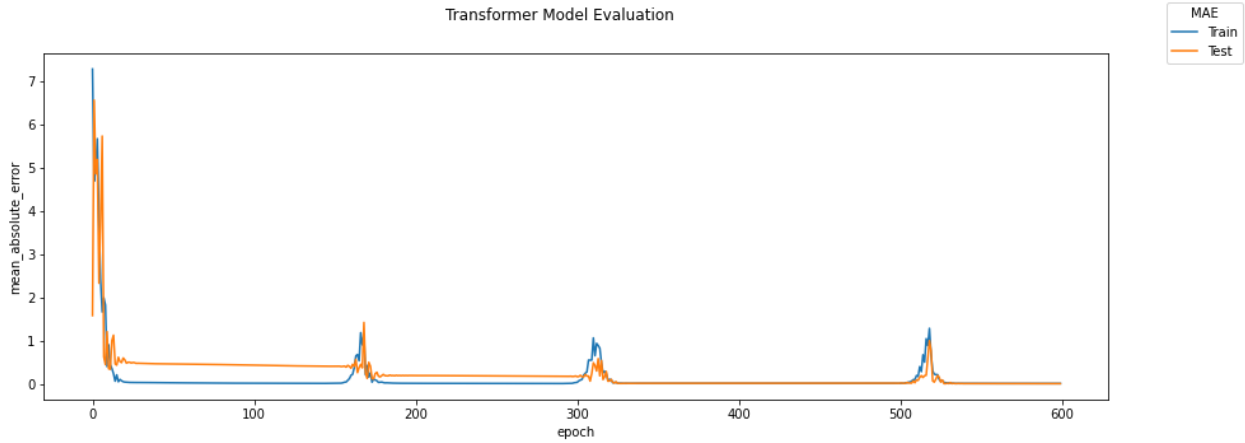

Рисунок 3.8 – Зміна MAE з плином часу

На рисунку 3.9 зображено розподіл похибок на реалізованих моделях, з чого робимо висновок, що моделі добре навчані, оскільки похибки

відцентровані навколо 0 (окрім моделі довгої короткострокової пам’яті, що показала себе найгірше).

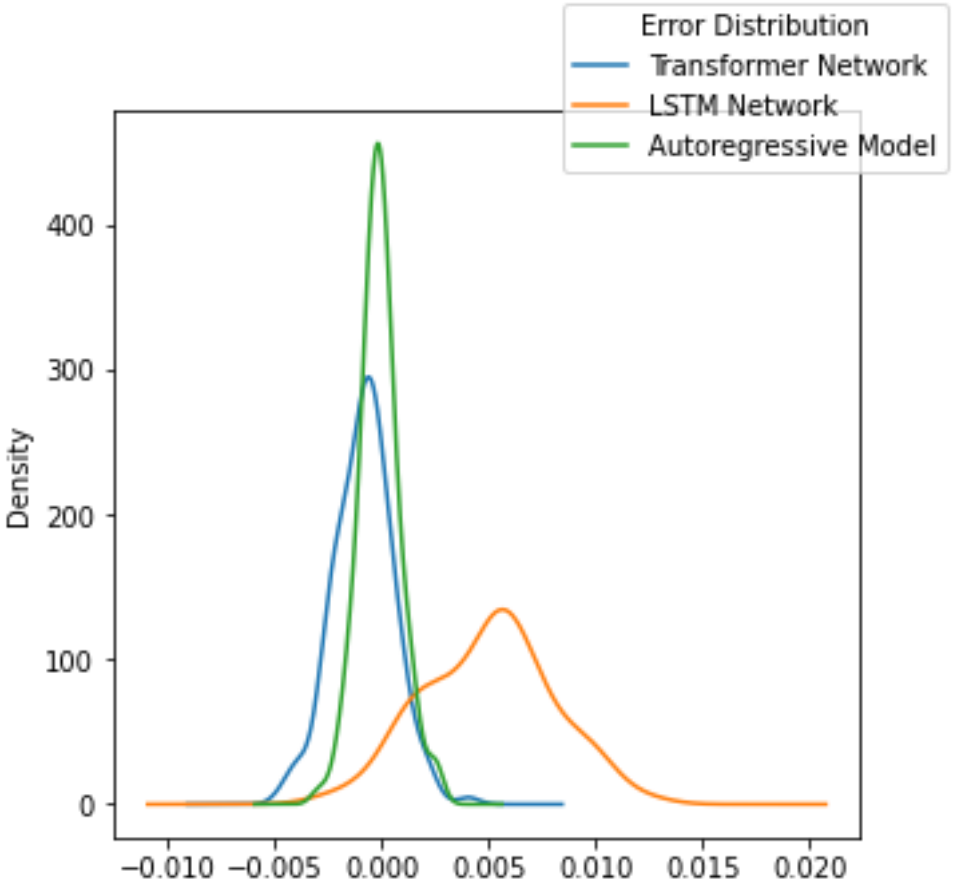

Рисунок 3.9 – Розподіл похибок

3.3 Висновки

У даному розділі було виконано аналіз вхідних даних, що було обрано для прогнозування, реалізовано моделі та проведено експерименти, результатом яких є побудовані прогнози та оцінка їх метриками.

У якості моделей для реалізації експерименту було обрано три кандидати – авторегресійна модель, модель нейронної мережі довгої короткострокової пам’яті та модель на основі механізму уваги. Також вищезгадані нейронні мережі було побудовано з різними конфігураціями

кількостей шарів та розмірністю. Дивлячись на результати можна зробить висновки, що реалізовані моделі пройшли навчання успішно та є придатними для побудови короткострокових прогнозів фінансового ряду. Також з мінімальної різниця деяких метрик на тестовій та навчальній вибірці робимо висновок, що моделі мають гарні узагальнюючі властивості.

Як результат проведення експериментів було встановлено, що найкращою виявилася модель нейронної мережі, побудована на основі механізму уваги, що було підтверджено обчисленням та порівнянням метриками оцінки прогнозуючих моделей.

РОЗДІЛ 4 РОЗРОБКА СТАРТАП-ПРОЕКТУ

Стартап як форма малого ризикового підприємництва впродовж останнього десятиліття набула широкого розповсюдження у світі через зниження бар'єрів входу в ринок, і вважається однією із наріжних складових інноваційної економіки, оскільки за рахунок мобільності, гнучкості та великої кількості стартап-проектів загальна маса інноваційних ідей зростає.

Проте створення та ринкове впровадження стартап-проектів відзначається підвищеною мірою ризику, ринково успішними стає лише невелика частка, що за різними оцінками складає від 10% до 20%. Ідея стартаппроекту, взята окремо, не вартує майже нічого. Головним завданням керівника проекту на початковому етапі його існування є перетворення ідеї проекту у працюючу бізнес-модель, що починається із формування концепції товару (послуги) для визначеної клієнтської групи за наявних ринкових умов.

Розроблення та виведення стартап-проекту на ринок передбачає здійснення низки кроків, в межах яких визначають ринкові перспективи проекту, графік та принципи організації виробництва, фінансовий аналіз та аналіз ризиків і заходи з просування пропозиції для інвесторів. Далі наведено маркетинговий аналіз стартап проекту. [25]

4.1 Опис ідеї проекту

В межах підпункту було проаналізовано і подано у вигляді таблиць:

- зміст ідеї;
- можливі напрямки застосування;
- можливі напрямки застосування;
- основні вигоди, що може отримати користувач товару;

– чим відрізняється від існуючих аналогів та замінників.

Перші три пункти подані у вигляді таблиці (таблиця 4.1) і дають цілісне уявлення про зміст ідеї та можливі базові потенційні ринки, в межах яких потрібно шукати групи потенційних клієнтів.

Таблиця 4.1 — Опис ідеї стартап-проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Дана комплексна система дозволяє розвязати проблему прогнозування фінансових даних	Аналіз даних	Прорахування ризиків при торгівлі акціями
	Прогнозування індексу акцій	Імовірний прибуток з отриманих результатів при торговлі

Аналіз потенційних техніко-економічних переваг ідеї (чим відрізняється від існуючих аналогів та замінників) порівняно із пропозиціями конкурентів передбачає:

– визначення переліку техніко-економічних властивостей та характеристик ідеї;
– визначення попереднього кола конкурентів (проектів-конкурентів) або товарів-замінників чи товарів-аналогів, що вже існують на ринку, та проводиться збір інформації щодо значень техніко-економічних показників для ідеї власного проекту та проектів-конкурентів відповідно до визначеного вище переліку;
– проводиться порівняльний аналіз показників: для власної ідеї визначаються показники, що мають:
  – гірші значення (W, слабкі);
  – аналогічні (N, нейтральні) значення;
  – кращі значення (S, сильні) (табл. 4.2).

Таблиця 4.2 – Визначення сильних, слабких та нейтральних характеристик ідеї проекту

№ п/п	Технікоекономічні характеристики ідеї	(потенційні) товари/концепції конкурентів			W (слабка сторона)	N (нейтральна сторона)	S (сильна сторона)
		Мій проект	Screener Plus	Thinkorswim			
1.	Форма виконання	Надавання послуг	Надавання послуг	Надавання послуг		+	
2.	Собівартість	Низька	Висока	Висока			+
3.	Функціонал	Широкий	Широкий	Широкий	+		

Визначений перелік слабких, сильних та нейтральних характеристик та властивостей ідеї потенційного товару є підґрунтям для формування його конкурентоспроможності.

4.2 Технологічний аудит ідеї проекту

В межах даного підрозділу було проведено аудит технології, за допомогою якої можна реалізувати ідею проекту (технології створення товару). Визначення технологічної здійсненності ідеї проекту передбачає аналіз таких складових (таблиця 4.3):

1. За якою технологією буде виготовлено товар згідно ідеї проекту?

2. Чи існують такі технології, чи їх потрібно розробити/доробити?
3. Чи доступні такі технології авторам проекту?

Таблиця 4.3 – Технологічна здійсненність ідеї проекту

№ п/п	Ідея проекту	Технології реалізації	Наявність технологій	Доступність технологій
1	Створення системи прогнозування індексу акцій	Використання мови програмування Python	Наявна	Доступна
		Використання мови програмування C#	Відсутні	Недоступні
		Використання мови C++	Відсутні	Недоступні
Обрана технологія реалізації ідеї проекту: мова програмування Python.				

За результатами аналізу таблиці зроблено висновок щодо можливості технологічної реалізації проекту. Технологічним шляхом реалізації проекту було обрано такі технології, як Python через доступність та безкоштовність.

Визначення ринкових можливостей, які можна використати під час ринкового впровадження проекту, та ринкових загроз, які можуть перешкодити реалізації проекту, дозволяє спланувати напрями розвитку проекту із урахуванням стану ринкового середовища, потреб потенційних клієнтів та пропозицій проектів-конкурентів.

Спочатку було проведено аналіз попиту: наявність попиту, обсяг, динаміка розвитку ринку (таблиця 4.4).

Таблиця 4.4 – Попередня характеристика потенційного ринку стартап-проекту

№ п/п	Показники стану ринку (найменування)	Характеристика
1	Кількість головних гравців, од	50
2	Загальний обсяг продаж, грн/ум.од	10000
3	Динаміка ринку	Зростає
4	Наявність обмежень для входу	Немає
5	Специфічні вимоги до стандартизації та сертифікації	Немає
6	Середня норма рентабельності в галузі, %	30%

За результатами аналізу таблиці 4.4 було зроблено висновок, що ринок є привабливим для входження.

Надалі були визначені потенційні групи клієнтів, їх характеристики та сформовано орієнтовний перелік вимог до товару для кожної групи (табл. 4.5).

Таблиця 4.5 – Характеристика потенційних клієнтів стартап-проекту

Потреба, що формує ринок	Цільова аудиторія	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару

Точні та швидкі прогнози	Власник бізнесу	Велика кількість даних	Простота використання, висока точність
Аналіз даних	Кінцевий користувач	Цікавить простота у використанні, низька ціна підтримки системи	Швидкість створення, низька ціна

Після визначення потенційних груп клієнтів було проведено аналіз ринкового середовища: складено таблиці факторів, що сприяють ринковому впровадженню проекту, та факторів, що йому перешкоджають (табл. 4.6, 4.7).

Таблиця 4.6 – Фактори загроз

№ п/ п	Фактор	Зміст загрози	Можлива реакція компанії
1	Конкуренція	Вихід на ринок продуктів з кращими характеристиками	Передбачити додаткові переваги власного програмного продукту (ПП) для того, щоб повідомити про них саме після виходу на ринок конкурентів. Вдосконалення технічних моментів власного продукту. Обрати нову цільову аудиторію і зосередитися на ній: зниження цін.

2	Зміна потреб користувачів	Користувачам необхідний сервіс з більшим/новим функціоналом.	Розроблення гнучкої архітектури програмного забезпечення для легшого впровадження нового функціоналу

Таблиця 4.7 – Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1	Гнучкі ціни	Зменшення ціни товару задля збільшення попиту	Введення власних гнучких цін
2	Поява нових методів квантування зображення	З’являться нові методи, що будуть швидше, та більш точно квантувати зображення	Покращити ПП додаванням нового функціоналу, розширення можливостей

Надалі було проведено аналіз пропозиції: визначили загальні риси конкуренції на ринку (таблиця 4.8).

Таблиця 4.8 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства
1. Тип конкуренції-чиста	Існує величезна кількість конкурентів на ринку.	Введення тестового періоду користування, щоб запевнитись в користі.

2. За рівнем конкурентної боротьби - міжнародний	Компанії-конкуренти з інших країн	Створити основу ПП таким чином, щоб можна було легко переробити даний ПП для використання у галузях інших країн.
3. Конкуренція за видами товарів: - товарно-видова	Конкуренція між видами ПП, їх особливостями.	Створити ПП, враховуючи недозображення конкурентів
4. За характером конкурентних переваг - неціновa	Вдосконалення технології створення ПП, щоб собівартість була нижчою	Удосконалення моделі. Використання більш дешевих технологій для розробки, ніж використовують конкуренти, але тільки якщо ці технології відповідають необхідним вимогам якості.
5. За інтенсивністю - не марочна	Бренд присутній, але його роль незначна	Реклама, участь у конференціях, семінарах.

Було проведено аналіз конкуренції у галузі за моделлю М. Портера (табл. 4.9).

Таблиця 4.9 – Аналіз конкуренції в галузі за М. Портером

Склад ові	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товаризамінники

ана лізу	Навести перелік прямих конкурентів	Визначити бар'єри входження в ринок	Визначити фактори сили постахальників	Визначити фактори сили споживачів	Фактори загроз з боку замінників
	Thinkorswim	Наявність вже існуючих рішень	-	Контроль якості продукту	Наявність більш широкого функціоналу, зручнішого інтерфейсу та авторитет
Висновки:	Досить інтенсивна конкурентна боротьба з іншими гравцями	Є можливості і виходу на ринок, але є і конкуренти.	-	Клієнти диктують умови роботи на ринку: зручний інтерфейс	Необхідно випускати ПЗ не гірше, ніж у конкурентів та розширяти функціонал.

За результатами аналізу було зроблено висновок про можливість роботи на ринку з огляду на конкурентну ситуацію.

Цей висновок був врахований при формулюванні переліку факторів конкурентоспроможності у наступному пункті. На основі аналізу конкуренції, проведеного в таблиці, а також із урахуванням характеристик ідеї проекту (табл. 4.2), вимог споживачів до товару (табл. 4.5) та факторів маркетингового середовища (табл. 4.6, 4.7) визначається та обґрунтовується перелік факторів конкурентоспроможності.

Аналіз оформлено у (табл. 4.10).

Таблиця 4.10 – Обґрунтування факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування
1	Ціна	Один із факторів для вибору продукту клієнтом.
2	Якість	Один із факторів для вибору продукту клієнтом.
3	Зручність роботи з програмою	Дозволяє користувачу легко працювати з програмою

За визначеними факторами конкурентоспроможності (табл. 4.10) проведено аналіз сильних та слабких сторін стартап-проекту (табл. 4.11).

Таблиця 4.11 – Порівняльний аналіз сильних та слабких сторін проекту

№ п/ п	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні						
			–3	–2	–1	0	+1	+2	+3
1	Ціна	15					*		
2	Якість	10			*				
3	Зручність роботи з програмою	15					*		

Фінальним етапом ринкового аналізу можливостей впровадження проекту є складання SWOT-аналізу (табл. 4.12) на основі виділених ринкових загроз та можливостей, та сильних і слабких сторін (таблиця 4.11).

Перелік ринкових загроз та ринкових можливостей було складено на основі аналізу факторів загроз та факторів можливостей маркетингового середовища. Ринкові загрози та ринкові можливості є наслідками впливу факторів, і, на відміну від них, ще не є реалізованими на ринку та мають певну ймовірність здійснення.

Таблиця 4.12 – SWOT-аналіз стартап-проекту

Сильні сторони: Якість Простота використання Висока швидкодія	Слабкі сторони: Дуже насичений ринок, мала кількість функціоналу, відсутня кросплатформеність.
Можливості: насичення ринку новим підходом до прогнозування; різноманітна клієнтура, вдосконалення системи	Загрози: Конкуренція

На основі SWOT-аналізу було розроблено альтернативи ринкової поведінки (перелік заходів) для виведення стартап-проекту на ринок та орієнтовний оптимальний час їх ринкової реалізації з огляду на потенційні проекти конкурентів, що можуть бути виведені на ринок.

Визначені альтернативи були проаналізовані з точки зору строків та ймовірності отримання ресурсів (таблиця 4.13).

Таблиця 4.13 – Альтернативи ринкового впровадження стартап-проекту

№ п/п	Альтернатива ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	PR, просування бренду	50%	6 місяців

2	Перехід на безкоштовне розповсюдження	75%	3 місяців
3	Партнерство для об’єднання продукції	65%	2 місяці

Після аналізу було обрано альтернативу №2.

4.3 Аналіз ринкової стратегії проекту

Розроблення ринкової стратегії першим кроком передбачає визначення стратегії охоплення ринку: було проведено опис цільових груп потенційних споживачів (таблиця 4.14).

Таблиця 4.14 – Вибір цільових груп потенційних споживачів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Великі компанії	Середня: велика конкуренція і можливість власних веб-відділів	Середній	Сильна	Просто

2	Маленькі компанії.	Висока	Високий	Слабка	Середня
Які цільові групи обрано:1,2					

За результатами аналізу потенційних груп споживачів було обрано цільові групу, для яких буде запропоновано даний товар, та визначено стратегію охоплення ринку - стратегію диференційованого маркетингу (компанія працює з декількома сегментами).

Для роботи в обраних сегментах ринку сформовано базову стратегію розвитку (таблиця 4.15).

Таблиця 4.15 – Визначення базової стратегії розвитку

№ п/п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
1	Постійне оновлення і покращення продукту	Ринкове позиціонування на індивіуальних користувачів	Швидкодія, якість продукту	Концентрований маркетинг

Наступним кроком обрано стратегію конкурентної поведінки (таблиця 4.16).

Таблиця 4.16 – Визначення базової стратегії конкурентної поведінки

№ п/п	Чи є проект «першопрохідцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки
1	Ні.	Компанія буде шукати нових споживачів та забирати існуючих у конкурентів	Буде копіювати, удосконалювати та створювати свої унікальні пропозиції	Зайняття конкурентної ніші

На основі вимог споживачів з обраних сегментів до постачальника (стартап-компанії) та до продукту (табл. 4.5), а також в залежності від обраної базової стратегії розвитку (табл. 4.15) та стратегії конкурентної поведінки (таблиця 4.16) розроблено стратегію позиціонування (таблиця 4.17), що полягає у формуванні ринкової позиції (комплексу асоціацій), за яким споживачі мають ідентифікувати торгівельну марку/проект.

Таблиця 4.17 – Визначення стратегії позиціонування

№ п/п	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартаппроекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту

1	Легкість розуміння, зручний інтерфейс, надійний, швидкий, точний та достовірни ПП для генерації рекомендацій.	Стратегія дифере нціації	Позиція на основі порівняння фірми з товарами конкурентів; Відмінні особливості споживача	Економія часу; Зручність застосування; Практичність та точність результату

Результатом виконання підрозділу стала узгоджена система рішень щодо ринкової поведінки стартап-компанії, яка визначає напрями роботи стартап- компанії на ринку.

4.4 Розроблення маркетингової програми стартап-проекту

Сформовано маркетингову концепцію товару, який отримає споживач. Для цього підсумовано результати попереднього аналізу конкурентоспроможності товару (таблиця 4.18). Концепція товару – письмовий опис фізичних та інших характеристик товару, які сприймаються споживачем, і набору вигод, які він обіцяє певній групі споживачів.

Таблиця 4.18 – Визначення ключових переваг концепції потенційного товару

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити)

1	Швидкість отримання результату	Швидка побудова прогнозу ціни	Необхідно покращити швидкість навчання моделі для більш чіткого прогнозування
2	Зручність застосування	Нативна підтримка мови програмування Python	Розробка зручного прикладного програмного інтерфейсу
3	Практичність та точність результату	Користувач отримує точні (з малою похибкою розбіжності) результати прогнозу.	Користувач на виході роботи ПП отримує модель та прогноз, котрі відповідають необхідним показникам варіативності та точності.

Розроблено трирівневу маркетингову модель товару: уточнюється ідея продукту та/або послуги, його фізичні складові, особливості процесу його надання (таблиця 4.19).

1-й рівень. При формуванні задуму товару вирішується питання щодо того, засобом вирішення якої потреби і / або проблеми буде даний товар, яка його основна вигода. Дане питання безпосередньо пов’язаний з формуванням технічного завдання в процесі розробки конструкторської документації на виріб.

2-й рівень. Цей рівень являє рішення того, як буде реалізований товар в реальному/ включає в себе якість, властивості, дизайн, упаковку, ціну.

3-й рівень Товар з підкріпленням (супроводом) – додаткові послуги та переваги для споживача, що створюються на основі товару за задумом і товару в реальному виконанні (гарантії якості , доставка, умови оплати та ін).

Таблиця 4.19 – Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові		
I. Товар за задумом	Генерація палітри кольорів зображення за допомогою інтелектуальних систем		
II. Товар реальному виконанні	Властивості/характеристики	М/Нм	Вр/Тх/Тл/Е/Ор
	1. Індивідуальний підхід. 2. Низька ціна. 3. Простота у використанні.	1.Нм 2.Нм 3.Нм	1.Технологічна 2.Економічна 3.Технологічна
	Якість: тестування фірмами аудиторами		
	Пакування: відсутнє		
	Марка: MissPredict		

Наступним кроком є визначення цінових меж, якими необхідно керуватись при встановленні ціни на потенційний товар (остаточне визначення ціни відбувається під час фінансово-економічного аналізу проекту), яке передбачає аналіз ціни на товари-аналоги або товари субститути, а також аналіз рівня доходів цільової групи споживачів (таблиця 4.20). Аналіз проведено експертним методом.

Таблиця 4.20 – Визначення меж встановлення ціни

№ п/п	Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
1	1800$	2500$	У всіх трьох груп високий рівень доходів	500$--

Наступним кроком є визначення оптимальної системи збуту, в межах якого було прийняте рішення (таблиця 4.21).

Таблиця 4.21 – Формування системи збуту

№ п/п	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
1	Канал нульового рівня	Продаж	0(напряму)	Власна

Останньою складової маркетингової програми є розроблення концепції маркетингових комунікацій, що спирається на попередньо обрану основу для позиціонування, визначену специфіку поведінки клієнтів (таблиця 4.22).

Таблиця 4.22 – Концепція маркетингових комунікацій

№ п/ п	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення

	Встановлення програми у персональний комп’ютер і його використання	Інтернет	Низька ціна, простота використання, універсальність	Показати переваги рішення над конкурентами, виділити ключові особливості	Створення сайту продукту, розповсюдження інформації про продукт на спеціалізованих ресурсах.

Було визначено, що придбання продукту буде проводитись через мережу Інтернет або при безпосередньому спілкуванні із представниками компанії. Розповсюдження інформації про продукт буде проводитись виключно через Інтернет, адже аудиторія даного продукту активно користується всесвітньою мережею.

Результатом підрозділу стала ринкова (маркетингова) програма, що включає в себе концепції товару, збуту, просування та попередній аналіз можливостей ціноутворення, спирається на цінності та потреби потенційних клієнтів, конкурентні переваги ідеї, стан та динаміку ринкового середовища, в межах якого впроваджено проект, та відповідну обрану альтернативу ринкової поведінки.

4.5 Висновки

В даному розділі проведено підготовчий аналіз для впровадження розробленої системи в якості стартап проекту. Досліджено аналогічні конкурентні системи, встановлено сильні та слабкі сторони системи в порівнянні з ними. Також було досліджено можливі шляхи розповсюдження

продукту та його ймовірну аудиторію, рівень доходів та ймовірну ціну продукту, що розробляється.

Було проведено аналіз потенційних ризиків і можливостей, а також розраховані основні фінансово-економічні показники проекту. Отримані результати кажуть про те, що реалізація проекту є доцільною. Було визначено сильні сторони проекту: зручність у використанні, ціна, якість та широкий функціонал. Серед слабких варто виділити повільне навчання. Варто відмітити можливість реклами продукту на спеціалізованих ресурсах із зазначенням сильних сторін проекту.

ВИСНОВКИ

Метою дослідження було проаналізувати дані та реалізувати кілька методів прогнозування на основі механізмів штучного інтелекту та обрати найкращий з них для подальшого використання при вирішенні задачі прогнозування фінансових показників у короткостроковій перспективі.

У ході проведення дослідження були вивчені властивості досліджуваного часового ряду, а саме – показника індексу S&P 500 на період з 31.12.2019 до 31.12.2021, після чого було встановлено нестаціонарність часового ряду та зроблено висновок щодо автокорельованості ряду та доцільності застосування авторегресійних моделей, в якості референсних моделей.

Після виконання низки маніпуляцій над вхідними даними були проведени експерименти з побудовою та тестування моделей глибинного навчання з метою визначення найкращої моделі за показниками R^2, MSE, MAE та MAPE.

Однією з головних задач, що було поставлено – була побудова моделей прогнозування на основі методів штучного інтелекту, а саме – авторегресійна модель (ARMA), модель довгої короткострокової пам’яті та модель на основі механізму уваги. В якості експериментів було виконане тренування нейронних мереж з різними конфігураціями для вибору найкращої.

Як результат проведення експериментів було встановлено, що найкращою виявилася модель нейронної мережі, побудована на основі механізму уваги, як серед досліджуваних моделей глибинного навчання, так і в порівнянні з авторегресійними лінійними моделями.

Результат даної роботи можна застосувати при вирішенні подібних задач короткострокового прогнозування нестаціонарних часових рядів.

ПЕРЕЛІК ПОСИЛАНЬ

1. What Is the Stock Market and How Does It Work? URL: https://learn.financestrategists.com/finance-terms/stock-market/ (Дата звернення 10.10.2022).

2. What is Open High Low Close in Stocks? URL: https://analyzingalpha.com/open-high-low-close-stocks (Дата звернення 10.10.2022).

3. Security Market Indicator Series (SMIS). URL: https://www.investopedia.com/terms/s/security-market-indicator-series-smis.asp (Дата звернення 21.10.2022).

4. Dow Jones Industrial Average: What It Is? URL: https://www.investopedia.com/terms/d/djia.asp (Дата звернення 21.10.2022).

5. S&P 500 Index: What It's for and Why It's Importing in Investing? URL: https://www.investopedia.com/terms/s/sp500.asp (Дата звернення 21.10.2022).

6. Nasdaq. URL: https://www.investopedia.com/terms/n/nasdaq.asp (Дата звернення 21.10.2022).

7. Stationary in time series analysis. URL: https://towardsdatascience.com/stationarity-in-time-series-analysis-90c94f27322 (Дата звернення 21.10.2022).

8. Interpreting Results of Dicky Fuller Test for Time Series Analysis. URL: https://medium.datadriveninvestor.com/interpreting-results-of-dicky-fuller-test-for-time-series-analysis-4bb1e98f242b (Дата звернення 21.10.2022).

9. Авторегресія з середнім ковзним URL: http://www.machinelearning.ru/wiki/index.php?title=ARMA (Дата звернення 21.10.2022).

10. A multiple multilayer perceptron neural network with an adaptive learning algorithm. URL: https://link.springer.com/article/10.1007/s11227-020-03404-w (Дата звернення 26.11.2022).

11. A Multilayer Feedforward Perceptron Model in Neural Networks for Predicting Stock Market Short-term Trends. URL: https://pure.strath.ac.uk/ws/portalfiles/portal/124295806/Namdari_Durrani_ORF_2021_A_multilayer_feedforward_perceptron_model_in_neural_networks_for_predicting_stock_market.pdf (Дата звернення 26.11.2022).

12. Using Deep Learning Models / Convolutional Neural Networks. URL: https://docs.ecognition.com/eCognition_documentation/User%20Guide%20Developer/8%20Classification%20-%20Deep%20Learning.htm (Дата звернення 26.11.2022).

13. How to Use Convolutional Neural Networks for Time Series Classification. URL: https://towardsdatascience.com/how-to-use-convolutional-neural-networks-for-time-series-classification-56b1b0a07a57 (Дата звернення 30.11.2022).

14. How to Develop Convolutional Neural Network Models for Time Series Forecasting. URL: https://towardsdatascience.com/how-to-use-convolutional-neural-networks-for-time-series-classification-56b1b0a07a57 (Дата звернення 30.11.2022).

15. Comparative Analysis of Recurrent Neural Networks in Stock Price Prediction for Different Frequency Domains. URL: https://www.researchgate.net/publication/354061072_Comparative_Analysis_of_Recurrent_Neural_Networks_in_Stock_Price_Prediction_for_Different_Frequency_Domains (Дата звернення 30.11.2022).

16. Attention is All You Need. URL: https://arxiv.org/abs/1706.03762 (Дата звернення 02.12.2022).

17. How the Coronavirus Affects Stock Prices and Growth Expectations. URL: https://www.chicagobooth.edu/review/how-coronavirus-affects-stock-prices-and-growth-expectations (Дата звернення 21.10.2022).

18. How War Affects the Modern Stock Market. URL: https://www.investopedia.com/solving-the-war-puzzle-4780889 (Дата звернення 22.10.2022).

19. Oil hits seven-year high but shares rebound on Russian war. URL: https://www.bbc.com/news/business-60502451 (Дата звернення 21.10.2022).

20. How Do Conflicts and War Affect Stocks? URL: https://money.usnews.com/investing/articles/how-do-conflicts-and-war-affect-stocks (Дата звернення 21.10.2022).

21. Mean Absolute Error. URL: https://www.statisticshowto.com/absolute-error/ (Дата звернення: 26.11.2022).

22. Tutorial: Understanding Regression Error Metrics in Python. URL: https://www.dataquest.io/blog/understanding-regression-error-metrics/ (Дата звернення: 26.11.2022).

23. Mean Absolute Percentage Error URL: https://www.statisticshowto.com/mean-absolute-percentage-error-mape/ (Дата звернення: 26.11.2022).

24. Coefficient of Determination URL: https://www.scribbr.com/statistics/coefficient-of-determination/ (Дата звернення: 26.11.2022).

25. What is A Startup? URL: https://www.feedough.com/what-is-startup/ (Дата звернення 03.12.2022).

ДОДАТОК А. ЛІСТИНГ ПРОГРАМНОГО КОДУ

1. Файл StockPriceDiploma.ipynb

```
import pandas as pd
import numpy as np
from datetime import datetime
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression,
ElasticNetCV
from sklearn.tree import DecisionTreeRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import (mean_absolute_error,
                             mean_squared_error,

mean_absolute_percentage_error)
import statsmodels.api as sm
import yfinance
import seaborn as sns
from matplotlib import pyplot as plt


sp_500_ticker = ['^GSPC']
end_date = '2021-12-31'
start_date = '2019-12-31'

def read_tickers(tickers):
  """
  Read and process stock prices fro given list of tickers
  """
  df = pd.DataFrame(columns=['Date'])
  df['Date'] = pd.date_range(start_date, end_date,
freq='D')
  df.set_index('Date', inplace=True)
  for ticker in tickers:
    ticker = yfinance.Ticker(ticker)
    history_prices = (ticker
```

```
                .history('1d', start=start_date,
end=end_date)
                [['Close']]
                )
    history_prices.index =
history_prices.index.tz_localize(None)

    df[ticker.ticker] = history_prices
  return df

def plot_tickers(tickers_dataset):
  """
  Plot tickers from tickers dataset
  """
  number_of_plots = len(tickers_dataset.columns)
  fig, ax = plt.subplots(number_of_plots, figsize=(15, 10))
  fig.suptitle('Stock Prices')
  for idx, ticker in enumerate(tickers_dataset.columns):
    if number_of_plots > 1:
      axis = ax[idx]
    else:
      axis = ax
    axis.plot(tickers_dataset.index.values.ravel(),
tickers_dataset[ticker].values.ravel())
    axis.set(ylabel=ticker)
  plt.show()

def plot_correlation(tickers_dataset):
  """
  Plot Correlation
  """
  fig = plt.figure(figsize=(10,10))
  sns.heatmap(df.corr(), cmap="YlGnBu", annot=True)
  fig.show()

def shift_data(tickers_dataset, y_col, days_shift,
include_lag=None):
  """
  Add to data some lags for given column
  """
  new_dataset = tickers_dataset.copy()
  for lag in range(1, days_shift + 1):
```

```
    new_dataset[f'{y_col}_-{lag}'] =
new_dataset[y_col].shift(lag)

  return new_dataset

def dickey_fuller_pval(tickers_dataset, column, trend):
  """
  Calculate p-value for Augmented Dickey-Fuller Test
  """
  return sm.tsa.stattools.adfuller(tickers_dataset[column],
regression=trend)[1]

def get_scores_for_model(ml_model, X_train, y_train,
X_test, y_test):
  """
  Score model (R^2, MAE, MSE, RMSE)
  """

  report_df = {
      "Sample Type":[],
      "Coef. of determination":[],
      "Mean Squared Error": [],
      "Mean Absolute Error": [],
      "Mean Absolute Percentage Error": []}

  train_r2 = ml_model.score(X_train, y_train)
  test_r2 = ml_model.score(X_test, y_test)
  y_train_pred = ml_model.predict(X_train)
  y_test_pred = ml_model.predict(X_test)

  train_mae = mean_absolute_error(y_train_pred, y_train)
  test_mae = mean_absolute_error(y_test_pred, y_test)

  train_mse = mean_squared_error(y_train_pred, y_train)
  test_mse = mean_squared_error(y_test_pred, y_test)

  train_mape = mean_absolute_percentage_error(y_train_pred,
y_train)
  test_mape = mean_absolute_percentage_error(y_test_pred,
y_test)
  report_df["Sample Type"].extend(["Train", "Test"])
```

```
  report_df["Coef. of determination"].extend([train_r2,
test_r2])
  report_df["Mean Squared Error"].extend([train_mse,
test_mse])
  report_df["Mean Absolute Error"].extend([train_mae,
test_mae])
  report_df["Mean Absolute Percentage
Error"].extend([train_mape, test_mape])

  return pd.DataFrame(report_df).set_index("Sample Type")


df = read_tickers(sp_500_ticker)
df.head(5)


plot_tickers(np.log(df))


df.dropna().diff().dropna()


print(dickey_fuller_pval(np.log(df.dropna()),
sp_500_ticker, 'c'))
print(dickey_fuller_pval(np.log(df.dropna()).diff().dropna(
), sp_500_ticker, 'c'))


sm.graphics.tsa.plot_pacf(np.log(df.dropna()).diff().dropna
()[sp_500_ticker])


sm.graphics.tsa.plot_acf(np.log(df.dropna()).diff().dropna(
)[sp_500_ticker])

df = np.log(df.dropna())
df_ma = pd.DataFrame({'MA':
np.random.default_rng().standard_normal(len(df))}).set_inde
x(df.index)

df = shift_data(df, '^GSPC', 3).dropna()
df_ma = shift_data(df_ma, 'MA', 3).dropna()
```

```
df = df.join(df_ma)

X_cols = ['^GSPC_-1', '^GSPC_-2', '^GSPC_-3', 'MA', 'MA_-
1', 'MA_-2', 'MA_-3']


X_train, X_test, y_train, y_test =
train_test_split(df[X_cols], df[sp_500_ticker],
train_size=0.66, shuffle=False)
predictions_df = y_test.copy().reset_index(drop=True)
y_train = y_train.values.ravel()
y_test = y_test.values.ravel()


sc = StandardScaler()
X_train_sc = sc.fit_transform(X_train)
X_test_sc = sc.transform(X_test)


lr = LinearRegression()
lr.fit(X_train_sc, y_train)
predictions_df["Linear Regression"] = lr.predict(X_test_sc)
get_scores_for_model(lr, X_train_sc, y_train, X_test_sc,
y_test)


plot_tickers(predictions_df)


from itertools import product
from collections import defaultdict

import tensorflow as tf
import tensorflow_addons as tfa
from tensorflow import keras
from tensorflow.keras import layers


X_train = X_train_sc[:, :3]
X_test = X_test_sc[:, :3]
```

```
def construct_lstm_model(lstm_layer_sizes: int,
dense_layer_sizes: int):
  model_layers = [layers.Input(shape=(3, 1))]
  model_layers.extend([layers.LSTM(layer_size,
return_sequences=True) for layer_size in
lstm_layer_sizes[:-1]])
  model_layers.extend([layers.LSTM(lstm_layer_sizes[-1])])
  model_layers.extend([layers.Dense(layer_size,
activation='relu') for layer_size in dense_layer_sizes])
  model_layers.extend([layers.Dense(1)])
  lstm_model = keras.models.Sequential(model_layers)
  return lstm_model

def benchmark_lstm_model(lstm_layer_sizes_list,
dense_layer_sizes_list):
  results_df = defaultdict(list)
  for lstm_layers, dense_layers in
product(lstm_layer_sizes_list, dense_layer_sizes_list):
    lstm_model = construct_lstm_model(lstm_layers,
dense_layers)
    lstm_model.compile(

optimizer=tf.keras.optimizers.Adam(learning_rate=0.005),
      loss=tf.keras.losses.MSE,
      metrics=[tf.keras.metrics.MAPE,
               tfa.metrics.RSquare(),
               tf.keras.metrics.MAE]
      )
    print(lstm_model.summary())
    print('.'*10, end='')
    model_name = f"lstm_{'-'.join(map(str,
lstm_layers))}_lstm_layers_{'-'.join(map(str,
dense_layers))}_dense_layers"
    history = lstm_model.fit(train_dataset.batch(64),
validation_data=test_dataset.batch(64), epochs=35,
verbose=False)
    results_df["Model Name"].extend([model_name]*2)
    results_df["Sample Type"].extend(["Train", "Test"])
    results_df["Coef. of
determination"].extend([history.history["r_square"][-1],
history.history["val_r_square"][-1]])
```

```
    results_df["Mean Squared
Error"].extend([history.history["loss"][-1],
history.history["val_loss"][-1]])
    results_df["Mean Absolute
Error"].extend([history.history["mean_absolute_error"][-1],
history.history["val_mean_absolute_error"][-1]])
    results_df["Mean Absolute Percentage
Error"].extend([history.history["mean_absolute_percentage_e
rror"][-1],
history.history["val_mean_absolute_percentage_error"][-1]])
    print('Done')

  return pd.DataFrame(results_df)


df = benchmark_lstm_model(
    [
        [64, 64],
        [32, 32],
        [128],
        [64],
        [32]
    ],
    [
        [128, 128],
        [64, 64],
        [32, 32],
        [128],
        [64],
        [32]
    ]
     )

df = df.set_index(["Model Name", "Sample Type"])

df[df.index.isin(['Test'], level=1)].sort_values(by='Mean
Absolute Percentage Error')

best_model = construct_lstm_model([64], [32])
best_model.compile(

optimizer=tf.keras.optimizers.Adam(learning_rate=0.01),
```

```
      loss=tf.keras.losses.MSE,
      metrics=[tf.keras.metrics.MAPE,
               tfa.metrics.RSquare(),
               tf.keras.metrics.MAE]
      )
best_model.fit(train_dataset.batch(64),
validation_data=test_dataset.batch(64), epochs=85)


predictions_df['Neural Network'] =
best_model(np.reshape(X_test_sc[:, :3], (*X_test_sc[:,
:3].shape, 1))).numpy().ravel()

plot_tickers(predictions_df)


(predictions_df['Neural Network'] / predictions_df['^GSPC']
- 1).plot.density()

(predictions_df['Linear Regression'] /
predictions_df['^GSPC'] - 1).plot.density()


class TransformerTimeSeries(tf.keras.Model):
  def __init__(self, input_dim, q_k_dims,
dense_layer_sizes_list):
    super().__init__()
    self.q_k = tf.keras.layers.LSTM(q_k_dims,
return_sequences=True, activation='relu')
    self.attn = tf.keras.layers.Attention()
    self.pooling = tf.keras.layers.GlobalAveragePooling1D()
    self.concat = tf.keras.layers.Concatenate()
    self.dense_layers = [
        tf.keras.layers.Dense(layer_size,
activation='relu')
        for layer_size in dense_layer_sizes_list
    ]
    self.out = tf.keras.layers.Dense(1)

  def call(self, input_tensor):
    q_out = self.q_k(input_tensor)
    k_out = self.q_k(input_tensor)
```

```
    attn_out = self.attn([q_out, k_out])
    q_pooling_out = self.pooling(q_out)
    k_pooling_out = self.pooling(k_out)
    pooling_out = self.pooling(attn_out)
    out = self.concat([q_pooling_out, k_pooling_out,
pooling_out])
    for dense_layer in self.dense_layers:
      out = dense_layer(out)
    out = self.out(out)
    return out


transformer_network = TransformerTimeSeries(3, 128, [128])


transformer_network.compile(

optimizer=tf.keras.optimizers.Adam(learning_rate=0.01),
      loss=tf.keras.losses.MSE,
      metrics=[tf.keras.metrics.MAPE,
               tfa.metrics.RSquare(),
               tf.keras.metrics.MAE]
      )
history = transformer_network.fit(train_dataset.batch(64),
validation_data=test_dataset.batch(64), epochs=600,
verbose=True)

predictions_df['Transformer Network'] =
transformer_network(np.reshape(X_test_sc[:, :3],
(*X_test_sc[:, :3].shape, 1))).numpy().ravel()


plot_tickers(predictions_df)


history_df = pd.DataFrame(history.history)


fig = plt.figure(figsize=(15, 5))
fig.suptitle('Transformer Model Evaluation')
sns.lineplot(data=history_df, x=history_df.index,
legend=False, y='loss')
```

```
sns.lineplot(data=history_df, x=history_df.index,
legend=False, y='val_loss')
fig.legend(title='MSE', loc='upper right', labels=['Train',
'Test'])
plt.xlabel('epoch')
fig.show()

fig = plt.figure(figsize=(15, 5))
fig.suptitle('Transformer Model Evaluation')
sns.lineplot(data=history_df, x=history_df.index,
legend=False, y='mean_absolute_percentage_error')
sns.lineplot(data=history_df, x=history_df.index,
legend=False, y='val_mean_absolute_percentage_error')
fig.legend(title='MAPE', loc='upper right',
labels=['Train', 'Test'])
plt.xlabel('epoch')
fig.show()

fig = plt.figure(figsize=(15, 5))
fig.suptitle('Transformer Model Evaluation')
sns.lineplot(data=history_df, x=history_df.index,
legend=False, y='mean_absolute_error')
sns.lineplot(data=history_df, x=history_df.index,
legend=False, y='val_mean_absolute_error')
fig.legend(title='MAE', loc='upper right', labels=['Train',
'Test'])
plt.xlabel('epoch')
fig.show()


fig = plt.figure(figsize=(5, 5))
(predictions_df['Transformer Network'] /
predictions_df['^GSPC'] - 1).plot.density()
(predictions_df['Neural Network'] / predictions_df['^GSPC']
- 1).plot.density()
(predictions_df['Linear Regression'] /
predictions_df['^GSPC'] - 1).plot.density()
fig.legend(title='Error Distribution', loc='upper right',
labels=['Transformer Network', 'LSTM Network',
'Autoregressive Model'])

fig.show()
```