МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ НАУК ТА ТЕХНОЛОГІЙ

Кафедра Комп'ютерних інформаційних технологій

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
Аліна САВЧЕНКО

«_____»_____2023 р.

КВАЛІФІКАЦІЙНА РОБОТА

(ДИПЛОМНА РОБОТА, ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ

"МАГІСТРА"

ЗА ОСВІТНЬО-ПРОФЕСІЙНОЮ ПРОГРАМОЮ "ІНФОРМАЦІЙНІ УПРАВЛЯЮЧІ СИСТЕМИ ТА ТЕХНОЛОГІЇ"

Тема: «Програмний модуль аналізу і прогнозування фінансових показників для системи управління інвестиціями»

Виконала: Саттарова Маргарита Леонідівна

Керівник: зав. каф., д.т.н., доцент Савченко Аліна Станіславівна

Нормоконтролер Ігор РАЙЧЕВ

Київ 2023

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет *Комп'ютерних наук та технологій*

Кафедра *Комп'ютерних інформаційних технологій*

Галузь знань, спеціальність, освітньо-професійна програма: 12 "Інформаційні технології", 122 "Комп'ютерні науки", "Інформаційні управляючі системи та технології"

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____Аліна САВЧЕНКО

"___"_____ 2023 р.

ЗАВДАННЯ

на виконання кваліфікаційної роботи студента

Саттарової Маргарити Леонідівни

(прізвище, ім'я, по батькові)

1. Тема роботи: «Програмний модуль аналізу і прогнозування фінансових показників для системи управління інвестиціями», затверджена наказом ректора від "29" вересня 2023р. за № 1976/ст .

2. Термін виконання роботи: з 02 жовтня 2023р. по 31 грудня 2023р. .

3. Вихідні дані до роботи: дані про фінансові показники, документація програмних засобів та інструментів реалізації програмного продукту .

4. Зміст пояснювальної записки (перелік питань, що підлягають розробці):

1) Засоби прогнозування фінансових показників та їх реалізації;

2) Проектування програмного модуля;

3) Програмна реалізація.

5. Перелік обов'язкового графічного матеріалу: слайди презентації *MS PowerPoint*.

6. Календарний план-графік

№ з/п	Завдання	Термін виконання	Підпис керівника
1.	Пошук і дослідження наукових джерел, огляд предметної області	02.10.2023 – 08.10.2023	
2.	Розробка та затвердження календарного плану виконання дипломної роботи.	09.10.2023 – 10.10.2023	
3.	Порівняльний аналіз існуючих рішень задачі дипломної роботи.	11.10.2023 – 13.10.2023	
4.	Написання тексту розділу 1. Засоби прогнозування фінансових показників та їх реалізації.	14.10.2023 –27.10.2023	
5.	Проектування програмного модуля. Підготовка тексту розділу 2.	28.10.2023 – 19.11.2023	
6.	Розробка та тестування програмного продукту	20.11.2022 – 04.12.2022	
7.	Написання тексту розділу 3. Оформлення пояснювальної записки дипломної роботи.	05.12.2023 – 08.12.2023	
8.	Рецензування та підписання Відгуку керівника у встановленому порядку.	09.12.2022 – 13.12.2023	
9.	Створення презентації та доповіді.	14.12.2023 – 18.12.2023	
10.	Підготовка до захисту та попередній захист дипломної роботи на випусковій кафедрі.	19.11.2023 – 22.12.2023	

7. Дата видачі завдання: «02» жовтня 2023 р.

Керівник дипломної роботи _____ Аліна САВЧЕНКО
(підпис керівника) (П.І.Б.)

Завдання прийняв до виконання _____ Маргарита САТТАРОВА
(підпис випускника) (П.І.Б.)

РЕФЕРАТ

Пояснювальна записка до дипломної роботи «Програмний модуль аналізу та прогнозування фінансових показнників для системи управління інвестиціями»: 109 сторінок, 27 рисунків, 3 таблиці, 32 літературних джерела, 3 додатки.

Ключові слова: МОДЕЛЬ, ПРОГНОЗУВАННЯ, *LSTM*, АКЦІЇ, МАШИННЕ НАВЧАННЯ, *PYTHON*, ДОСЛІДЖЕННЯ.

Об’єкт дослідження – процес прийняття інвестиційних рішень в управлінні портфелем цінних паперів.

Предмет дослідження – автоматизація процесів аналізу та прогнозування фінансових показників цінних паперів.

Мета дипломної роботи – розробка програмного модуля аналізу та прогнозування фінансових показників для системи управління інвестиціями, що дозволить покращити процес проведення аналітики активів на фондовому ринку за рахунок поєднання методів технічного та фундаментального аналізу для прогнозування цін активів на фондовому ринку.

Методи дослідження – аналітичний огляд існуючих програмних рішень та наукових робіт у напрямку прогнозування цін активів на фондовому ринку, дослідження методів прогнозування часових рядів.

У результаті виконання кваліфікаційної роботи було запропоновано новий підхід, що поєднує переваги *LSTM* мереж з ретельно підібраними вхідними даними та техніками обробки. Розроблена модель не тільки включає технічні показники, але й охоплює дані фундаментального аналізу та макроекономічні фактори. Окрім того, було використано методи обробки природної мови (*NLP*) та аналіз сентименту для інтеграції інформації з текстових джерел, щоб забезпечити більш всебічний аналіз впливу різних факторів на ціни акцій.

Результати перевірки роботи побудованої моделі машинного навчання демонструють, що обраний комплексний підхід може покращити точність прогнозування цін на акції, долаючи обмеження існуючих методів.

Перспективи подальшого розвитку включають вдосконалення етапу обробки текстової інформації з додаванням більш комплексної логіки класифікації текстових даних, а також розширення можливостей прогнозної моделі для роботи з ширшим спектром фінансових інструментів. Додатковим покращенням точності роботи моделі також є врахування кореляцій між цінами акцій різних компаній у рамках однієї індустрії.

ЗМІСТ

ВСТУП

Сучасні реалії функціонування фінансових ринків, все більш проникаюча глобалізація, ріст кількості нових класів активів та все більш складних фінансових інструментів та процесів на фондових біржах ставлять перед інвесторами численні виклики та вимагають розробки все більш гнучких і адаптивних інвестиційних стратегій та методик ведення процесів управління інвестиціями. Світова економіка стає все більш глобальною та взаємозалежною, що призводить до зростання невизначеності та мінливості ринку, що в комбінації з динамічними макроекономічними змінами, геополітичними напруженостями, а також зростанням об’ємів та доступності фінансових (біржових) даних роблять процес інвестиційного управління вкрай складним. Як наслідок, існуючі інструменти та методики часто виявляються недостатніми для ефективного вирішення цієї задачі.

В умовах сьогодення, де фінансові ринки відіграють ключову роль у глобальній економіці, точність прогнозування цін на акції є життєво важливою не тільки для інвесторів та трейдерів, але й для економічної стабільності у її ширшому розумінні. З появою розширених методів машинного та глибокого навчання відкрилися нові можливості для покращення прогнозування фінансових часових рядів.

Проте, багато існуючих підходів до прогнозування цін на акції зазнають важливих обмежень.

Зокрема, в переважній більшості функціонал існуючих рішень обмежується механізмами технічного аналізу, які є простішими в реалізації, ніж засоби фундаментального аналізу. В той же час, останній надає не менш важливу інформацію та враховує вплив багатьох факторів які є критичними. Відповідно, не враховуючи його інвестор ігнорує чинники та показники, які нерідко відіграють ключову роль в прийнятті інвестиційних рішень

Також не завжди фактори, що мають вплив на формування ціни акції, можуть бути отримані з виключно технічної аналітики або з фінансової звітності

компанії. Натомість, значна кількість важливої інформації міститься в новинах, публікаціях, анонсах, звітах державних органів тощо. Відсутність інтеграції даних з текстових джерел для глибшого аналізу ринкового сентименту, що є поширеною проблемою для наявних рішень, призводить до упущення опрацювання вкрай важливих аспектів, від яких залежатиме вартість біржових активів. Крім того, більшість моделей не звертають належної уваги на зашумлені характеристики фінансових даних, що може суттєво впливати на якість прогнозу.

Метою дипломної роботи є розробка програмного модуля аналізу та прогнозування фінансових показників для системи управління інвестиціями, що дозволить покращити процес проведення аналітики активів на фондовому ринку за рахунок поєднання методів технічного та фундаментального аналізу для прогнозування цін активів на фондовому ринку та відповідне наближення фінансових показників інвестиційного портфеля до таких, які інвестор вважає цільовими.

Задля успішного досягнення цієї цілі необхідним є вирішення таких завдань:

1) Провести аналітичний огляд предметної області, визначити її специфіку та обмеження;

2) Виконати огляд та порівняльний аналіз існуючих програмних рішень у напрямку прогнозування вартості біржових активів;

3) На основі переваг та недоліків розглянутих аналогів сформулювати перелік вимог до розроблюваного програмного продукту;

4) Врахувати методики та результати існуючих наукових досліджень при формулюванні принципів нового підходу вирішення задачі прогнозування;

5) Спроектувати та побудувати модель, що проводитиме прогнозування цін акцій на фондовому ринку;

6) Розробити програмний модуль, що надаватиме користувачам прогнозні значення, обчислені побудованою моделлю;

7) Здійснити інтеграцію розробленого модуля з програмною системою управління інвестиціями.

Практичне значення роботи та отриманих результатів полягає у використанні розробленого програмного модуля як в інтеграції з іншими програмними системами для більш комплексного підходу для управління інвестиціями та обліку персонального портфелю цінних паперів, так і окремо як самостійного продукту для більш зваженого прийняття інвестиційних рішень широким колом фінансових спеціалістів.

РОЗДІЛ 1
ЗАСОБИ ФІНАНСОВОГО ПРОГНОЗУВАННЯ ТА ЇХ РЕАЛІЗАЦІЇ

Сьогодні процес інвестиційного менеджменту являє собою складну комплексну систему багатоетапних підпроцесів, до складових якої належать: планування, аналіз, конструювання портфелю, його регулярний перегляд, моніторинг, коригування та оптимізація.

На етапі планування інвестор або портфельний менеджер визначає основні цілі інвестицій, обсяг доступних ресурсів, термін інвестування та цільовий (граничний) рівень ризику. Основною метою цього етапу є створення чіткої стратегії, яка б відображала фінансові потреби і очікування інвестора.

Далі процес інвестиційного менеджменту включає аналіз ринку та окремих інвестиційних інструментів. При цьому аналізуються не тільки історична дохідність та ризики, але й потенціал для майбутнього зростання, враховуючи макроекономічні індикатори, фінансові показники компаній та інші важливі фактори.

Конструювання портфеля цінних паперів включає визначення конкретних активів для вкладення коштів, а також пропорцій розподілу капіталу, що інвестується між цими активами. На цьому етапі обрані інвестиційні активи комбінуються з метою досягнення цільового балансу між ризиком і дохідністю. При цьому інвестор стикається з проблемами селективності, вибору часу операцій і диверсифікації. Селективність, звана також мікропрогнозуванням, відноситься до аналізу цінних паперів і пов'язана з прогнозуванням динаміки цін окремих видів цінних паперів. Вибір часу операцій, або макропрогнозування, включає прогнозування зміни рівня цін на акції порівняно з цінами для фондових інструментів з фіксованим доходом, такими, як корпоративні облігації.

Важливою складовою цього етапу є диверсифікація, що полягає у формуванні інвестиційного портфеля таким чином, щоб за певних обмежень мінімізувати ризик [4].

Процес перегляду портфеля як необхідна складова інвестиційного процесу має місце тому, що з часом цілі інвестування можуть змінитися, в результаті чого портфель перестане задовольняти потребам та цілям, що ставить перед собою його власник. В такому випадку інвестору потрібно буде сформувати новий портфель шляхом продажу частини цінних паперів та покупки деяких нових. Іншою підставою для перегляду портфеля є зміна курсу цінних паперів з плином часу.

У зв'язку з цим деякі цінні папери, що спочатку були непривабливими для інвестора, можуть стати вигідним об'єктом інвестування, і навпаки. Тоді інвестор захоче придбати перші, одночасно продавши останні з свого портфеля. Рішення про перегляд портфеля залежить, поміж інших факторів, також від розміру транзакційних витрат і очікуваного зростання прибутковості переглянутого портфеля [4].

Оцінка ефективності портфеля включає періодичну оцінку значень та показників прибутковості інвестиційного портфеля, часто в комбінації з оцінкою показників ризику, з якими стикається інвестор [4]. При цьому необхідним є використання прийнятних показників прибутковості та ризику, а також відповідні стандарти (своєрідні «еталонні» значення) для порівняння.

При цьому якщо для якісного виконання деяких з цих підпроцесів достатнім є отримання та аналіз фінансових показників в конкретний момент часу, отриманих методами технічного аналізу, то для коректного виконання таких підпроцесів як перегляд та калібровка портфеля цього недостатньо, тому що їхня суть полягає у визначенні подальшої траєкторії роботи з портфелем, його корекції на довгострокову перспективу. Отже, в сучасних реаліях недостатнім є лише обрахунок тих чи інших показників в певний момент часу, необхідним є також проведення прогнозних оцінок щодо руху цих показників, цін, ринкових

трендів та тенденцій в майбутній перспективі, проведення їх аналітики та прийняття рішень базуючись на них.

Реалізація цієї задачі на практиці є досить складною, адже вимагає врахування великої кількості факторів, а також визначення такого способу включення представлень цих факторів до фінансових моделей, щоб забезпечити якість та точність прогнозних оцінок, на базі яких проводиться корекція портфеля, які будуть вважатись задовільними. До того ж, розвиток технологій та всезростаючий ступінь впровадження інновацій для підвищення ефективності фінансового сектору, поширення тенденції повного переходу на цифрові моделі проектів у цій сфері призводить до того, що конкурентні переваги доводиться шукати у все більш складних рішеннях.

Ці фактори стали причиною пошуку відповідних технологічних рішень, та розробки відповідних програмних систем. Проте в переважній більшості функціонал існуючих рішень обмежується механізмами технічного аналізу, які є простішими в реалізації, ніж засоби фундаментального аналізу. В той же час, останній надає не менш важливу інформацію та враховує вплив багатьох факторів які є критичними. Відповідно, не враховуючи його інвестор ігнорує чинники та показники, які нерідко відіграють ключову роль в прийнятті інвестиційних рішень.

Пошук якомога точнішої реалізації завдання прогнозування фінансових показників, як і дискусія навколо самої ідеї реальності втілення такого завдання, продовжується і зараз.

Значний вплив на питання можливості прогнозування цін акцій мала книга «Випадкове блукання на Уолл-стріт» Бертона Малкіеля, вперше опублікована у 1973 р. Автор стверджував, що неможливо передбачити курс акцій та вигідно вкладати кошти, використовуючи аналіз та стратегії, і що немає надійних методів передбачення курсу акцій через випадковий та непередбачуваний характер руху цін на фондовому ринку. Замість цього пропонувалось інвестувати в широко диверсифікований портфель і утримувати його протягом тривалого періоду, спираючись на ефективний ринок.

Малкіель аргументує непередбачуваність та випадковість змін цін акцій на ринку, використовуючи результати емпіричних досліджень і теорію випадкового блукання (*Random Walk Theory*). Вона стверджує, що ціни акцій рухаються випадковим чином і передбачити точний напрямок їхнього руху неможливо. Рух цін акцій подібний до випадкового блукання, а це означає, що в майбутньому ціни змінюються непередбачувано.

Однією з основ тверджень Малкіеля було те, що він вважав фондовий ринок ефективним, оскільки всі публічно відомі інформації вже враховуються в поточних цінах акцій. Отже, немає можливості здійснити успішні угадування щодо майбутніх змін цін. Окрім цього, стверджувалось, що навіть якщо існують деякі аномалії на ринку, вони непостійні та непередбачувані в часі, що робить неможливим отримання стабільного прибутку внаслідок ефективного використання цих аномалій.

Праця Бертона Малкіеля мала значний вплив на сприйняття фондового ринку, сприяла поширенню та утвердженню теорії випадкового блукання і підкреслювала важливість тривалого періоду утримання інвестицій та портфеля, віддавши перевагу стратегії купівлі та утримання замість активної торгівлі. Деякі з основних концепцій та тез, які випливали з "*A Random Walk Down Wall Street*", залишаються актуальними. Зокрема, серед них можна виділити наступні:

– концепція ефективного ринку – залишається актуальною та має широкий підтримку в наукових дослідженнях. Фінансові ринки продовжують відображати нову інформацію швидко та ефективно;

– важливість диверсифікації – рекомендація широкої диверсифікації портфеля залишається ключовим аспектом ефективного управління ризиками та максимізації віддачі;

– довгостроковий план інвестування та обґрунтовані очікування.

Однак не дивлячись на свою популярність та вплив, книга знайшла також і противників своєї ідеї, які ставили під сумнів повну недетермінованість змін на фондовому ринку. Їхнім втіленням була книга Ендрю В. Ло и Арчі Крейг Маккінлі «Невипадкове блукання на Уолл-стріт» – праця, присвячена аналізу

гіпотези про випадкові рухи і доповіді про різноманітні патерни та аномалії, які можуть вказувати на прогнозованість цін акцій. Декілька прикладів таких патернів і аномалій включають:

– автокореляція – автори досліджують автокореляцію в даних про ціни акцій і констатують, що існують випадки, коли минулі значення дохідності акцій (*stock returns*) впливають на майбутні повернення, що суперечить гіпотезі про випадкові рухи;

– об'єм торгів – аналізується зв'язок між об'ємом торгів і ціновими змінами, показуючи, що певні залежності між ними можуть бути використані для прогнозування;

– сезонність та календарні аномалії – досліджено календарні аномалії, такі як сезонність, ефект січня (коли акції часто ростуть у січні) та ефект понеділка (коли акції у понеділок часто мають такий же тренд до зміни, що був актуальним наприкінці п’ятниці);

– моментум та реверсія – розглядаються стратегії на основі моментуму (купівля акцій, які недавно піднялися в ціні) та реверсії (купівля акцій, які недавно впали в ціні), і вказують на те, що такі стратегії можуть виявитися прибутковими;

– ефект мікроструктури ринку (*Market Microstructure Effects*) – вивчено вплив мікроструктури ринку на ціни акцій, в тому числі як зміни в попиті та пропозиції можуть впливати на ціни.

Автори також використовували різноманітні статистичні та економетричні методи для аналізу фінансових часових рядів та для тестування прогнозованості цін акцій. До таких методів включають:

– *VAR* (векторні авторегресійні) моделі – використовувалися для моделювання та прогнозування взаємозв’язків між різними фінансовими часовими рядами;

– тести на ефективність ринку – проводились тести на наявність залежностей в часових рядах цін акцій, які можуть свідчити про неефективність ринку;

– аналіз об'ємів торгів – досліджувалася залежність між об'ємами торгів та ціновими змінами;

– портфельний аналіз – автори проводили аналіз вибірки портфелів акцій за допомогою стратегій, заснованих на моментумі та реверсії, і досліджували, чи можна отримати вищі доходи від цих стратегій.

Результати багатьох з цих тестів та експериментів вказували на наявність патернів та аномалій у фінансових часових рядах, які суперечать гіпотезі про випадкові рухи і можуть свідчити про можливість прогнозування цін акцій. У підсумку, Ло та МакКінлі аргументують, що хоча ринки можуть бути ефективними в більшості випадків, існують часи, коли вони відхиляються від виключно випадкового характеру, і ці відхилення можна використовувати для отримання прибутку.

У світлі зростаючої економічної глобалізації фондовий ринок може демонструвати високу частоту, поліморфізм і складність. Прогнозування даних про фінансові показники акцій є важливим аспектом та часто відіграє вирішальну роль у прийнятті рішень щодо капіталовкладень, сприяє уникненню ризику та успішній прибутковості для інвесторів, а також допомагає органам фінансового регулювання у розробці політики макроекономічного управління. Характеристики динамічності, нелінійності, нестабільності, зашумленості та мінливості, притаманні даним про фінансові показники акцій значним чином ускладнюють задачу їх прогнозування[1], особливо у випадку прогнозування на довгий часовий горизонт.

Фінансові дані за своєю природою являють собою типові часові ряди, неперервні за значенням, але дискретні в часі, які демонструють довгострокову залежність від багатьох вимірів. Прогнозування на довгий часовий горизонт визначається в літературі як багатокрокове прогнозування наперед (*multi-step-ahead forecasting*), яке надає важливу інформацію про довгострокові майбутні

тренди притаманні досліджуваним даним [2]. Багатокрокове прогнозування фінансових показників акцій є більш цінним для інвесторів у фінансовій галузі, ніж прогнозування на один крок наперед (*one-step-ahead forecast*), і тому саме цей підхід (стратегія) є здебільшого використовуваною серед практикуючих спеціалістів фінансової галузі та представників державних органів та установ.

Тим не менш, значна частина попередніх досліджень прогнозування цін на акції здебільшого зосереджувались на прогнозуванні на один крок вперед, який все ж таки є найпоширенішим сценарієм. Таким чином, питання актуальності та необхідності дослідження багатокрокового прогнозування наперед до сих пір залишається релевантним, оскільки наразі було проведено досить обмежену кількість експериментальних та теоретичних досліджень застосування підходу багатокрокового прогнозування щодо фінансових показників акцій.

Проте на значення фінансових показників активів впливає велика кількість факторів, серед яких природні, соціальні, політичні та економічні. Таким чином, лише сукупність різних змінних, які відображають ці впливаючі фактори, та розгляд їх в комплексі здатні належним чином відобразити та пояснити диспропорцію та залежності значень фінансових показників, і, як наслідок цього, необхідною умовою отримання задовільних з точки зору точності результатів прогнозування вимагає використання сукупності змінних в якості вхідних даних для прогнозуючих моделей та алгоритмів.

1.1. Огляд і порівняльний аналіз існуючих програмних рішень

1.1.1. I Know First

I Know First являє собою медіаресурс, який спеціалізується на прогнозуванні показників фінансових активів, індексів, ринків, а також на наданні відповідної аналітики, розроблений однойменною компанією. Основою функціонування даного продукту є прогнозний алгоритм фондового ринку, який використовує для отримання прогнозних показників підхід, заснований на

використанні методів штучного інтелекту та машинного навчання. Алгоритм обробляє великі набори історичних даних фондового ринку для прогнозування ринкових тенденцій.

Сам собою сервіс (веб-ресурс) є збіркою аналітичних заміток та публікацій, розділених по категоріях, таких як *Stock Forecast*, *Indices Forecast*, *European Stock Forecast* та інші. Також є розділи новин та прес-релізів про ситуацію на фінансових ринках, компанії та події у світі фінансів та інвестицій.

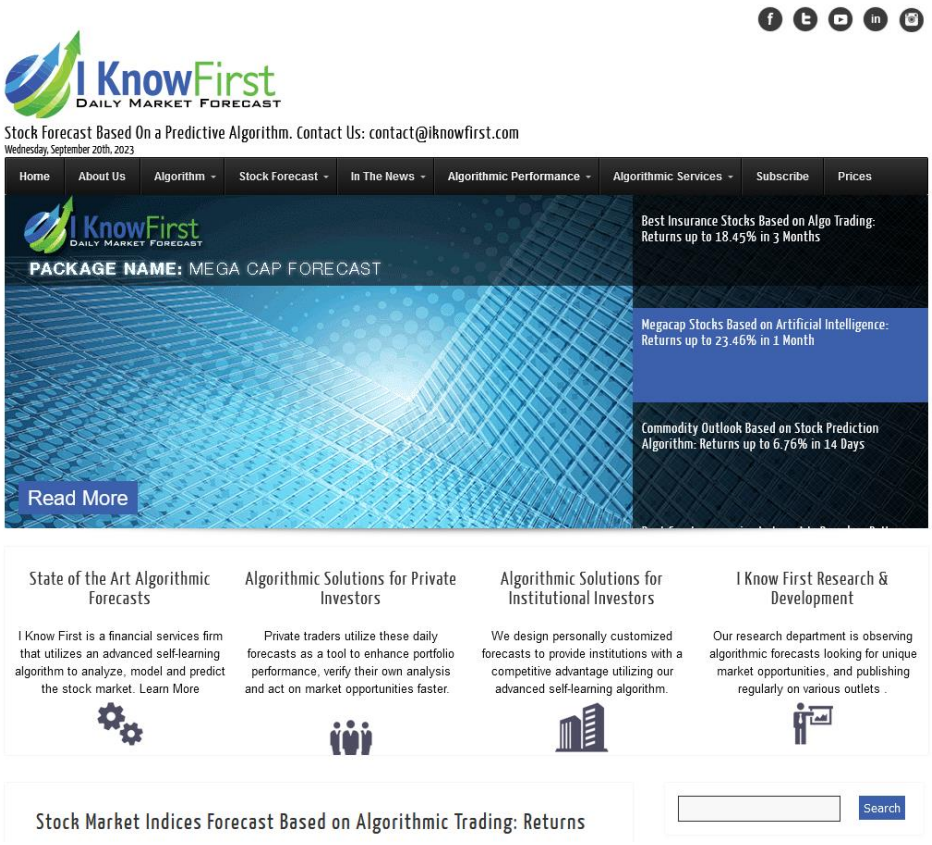

Рис. 1.1. Головна сторінка сервісу *I Know First*

Платформа охоплює велику кількість активів та фінансових інструментів, включаючи акції, товари (*commodities*), форекс, криптовалюти, деривативи, процентні ставки, *ETF* та світові індекси. Прогнози фінансових показників надані як для короткострокових (1 день), так і для довгострокових часових проміжків (1 рік).

Візуальне представлення цих даних виконано у вигляді теплових карт (*heat maps*) та графіків. Кольори і відтінки на теплових картах показують очікувану величину і напрямок зміни для конкретних активів, ілюструючи таким чином яку імовірність зростання чи спаду мають ті чи інші активи.

Варіанти використання даного сервісу можуть включати:

– розподіл активів: використання наданих прогнозних оцінок інституційними інвесторами при розподілі активів шляхом прогнозування динаміки ринку;

– активна торгівля: отримання трейдерами та інвесторами потенційних торгових ідей на основі прогнозованих рухів;

– управління ризиками: використання прогнозної інформації наданої даною платформою для процесів управління й хеджування потенційних ризиків.

Окрім прогнозів, *I Know First* надає матеріали для навчання або додаткову інформацію про те, як їхні прогнози генеруються або як найкраще їх використовувати, а також аналітичні звіти та консалтингові послуги для професійних інвесторів або інституцій.

Рис. 1.2. Прогнозна аналітика на ресурсі *I Know First*

Рис. 1.3. Приклад подання прогнозу показників сервісом *I Know First*

Результати прогнозування надаються користувачам щоденно за підпискою, вартість якої починається із 189 доларів США за один місяць. Попередньо користувач обирає тарифний план, що визначає параметри та критерії, яким повинні відповідати цільові активи (тип, сектор, об'єм капіталізація, тенденція зміни вартості тощо), після чого йому на електронну пошту відправлятимуться спрогнозовані зміни у вартості активів у форматі *Excel* таблиць.

1.1.2. TipRanks

TipRanks — це комплексна платформа дослідження фінансових активів, яка збирає й оцінює фінансові дані та думки аналітиків, щоб надати інвесторам уявлення про показники акцій і настрої ринку, а також прогнозні оцінки щодо значень показників активів на основі публікацій аналітиків. Основною цінністю *TipRanks* є його прозоре ранжування фінансових експертів і аналітиків на основі точності та ефективності їхніх попередніх рекомендацій. Даний сервіс пропонує

комплексну платформу для дослідження акцій шляхом агрегування та аналізу думок експертів, новинних настроїв та інших відповідних даних.

Цей інструмент може бути корисним для інвесторів та трейдерів для покращення процесу прийняття рішень на фондовому ринку, використовуючи колективний інтелект фінансових експертів.

TipRanks надає агреговані рейтинги акцій від аналітиків із великих брокерських фірм, думки фінансових блогерів, показує рекомендації щодо купівлі, продажу та утримання. Надає консенсус-рейтинг і прогнозовану оцінку цільової ціни з горизонтом прогнозування 12 місяців.

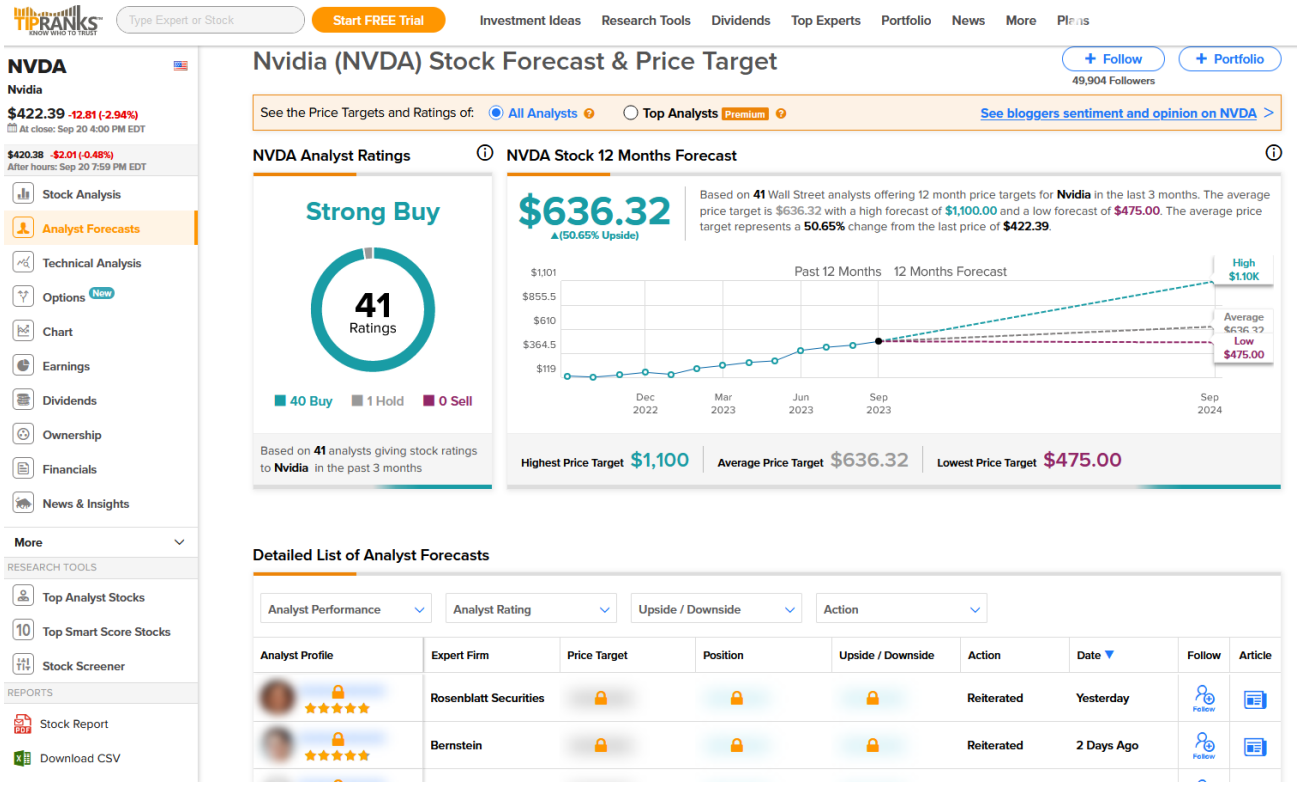

Рис. 1.4. Отримання прогнозу ціни акцій за допомогою сервісу *TipRanks* на прикладі акцій компанії *Nvidia*

Крім цього ресурс дає можливість перегляду списків, підібраних на основі зведених даних, висвітлюючи найефективніші акції, дані про діяльність хедж-фондів (надає інформацію про позиції акцій відомих хедж-фондів, вказує, чи хедж-фонди купують чи продають ті чи інші акції), дані про внутрішню торгівлю

(відстежує покупки та продажі акцій, зроблені керівниками компанії, допомагає визначити потенційні настрої інсайдерів щодо компанії), відображає показники фінансових аналітиків за попередні періоди, включно з показником успішності та середньою прибутковістю.

Функціонал *TipRanks* також включає фондовий скринінг для пошуку відповідних активів за підібраними параметрами та пошук схожих активів, можливість визначення потенційних інвестиційних можливостей на основі різних фільтрів і критеріїв, зокрема рейтингів аналітиків, сектору, ринкової капіталізації тощо. Також за допомогою інструменту *Smart Portfolio* користувач може створити власний інвестиційний портфель (або декілька портфелів) та переглядати його статистику, ключові показники, дані про загальну ефективність портфеля, інформацію про розподіл активів портфеля, розбиття за секторами, типами акцій і регіонами, показує загальну прибутковість портфеля і порівнює цей показник з попередньо обраним еталоном.

TipRanks пропонує як безкоштовне, так і платне членство. Є можливість використання обмеженої кількості функціоналу даного сервісу з будь-яким членством, проте більшість інструментів і функціоналу вимагають платного членства (підписки). Зокрема, до платних послуг відносяться практично усі результати детальної аналітики, більшість фільтрів Фондового скрінера, рейтинги фінансових експертів, інсайти, оцінки та моніторингу ризику, персоналізація інструментів для відображення даних, адаптоване до конкретної інвестиційної стратегії користувача. Мінімальна вартість підписки – $360 за рік користування.

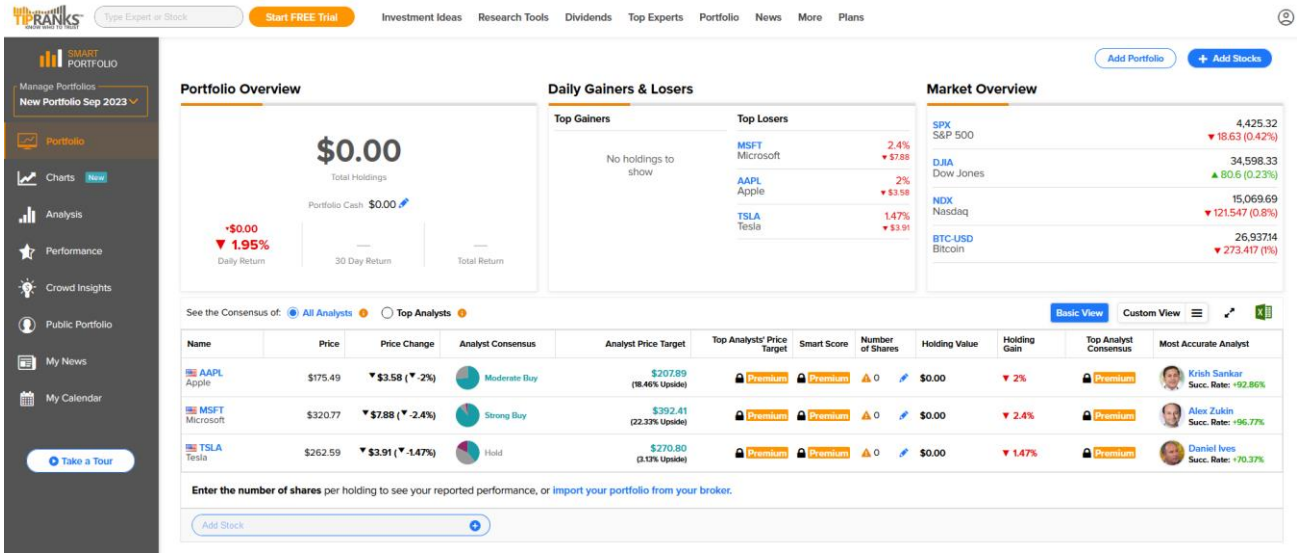

Рис. 1.5. Представлення створеного портфелю в *TipRanks*

1.1.3. FinBrain

FinBrain — це платформа фінансової аналітики, яка надає доступ до фінансової інформації значної кількості міжнародних компаній, що представлені на ряді фондових бірж. Її основна увага зосереджена на прогнозуванні майбутніх цінових коливань та надання фінансових даних різної природи, від цінових даних, технічних індикаторів та показників діяльності компаній до новин і соціальних настроїв.

Функціонально *FinBrain* надає своїм користувачам прогнозовані ціни відкриття, закриття, найвищі та найнижчі значення ціни для певного активу з горизонтами прогнозування 3 дні, 5 днів, 10 днів, та 3, 6, 12 місяців. Ці прогнози можуть супроводжуватися показником довіри або інтервалом, що дає користувачам уявлення про те, наскільки «впевненим» є алгоритм щодо своїх прогнозів.

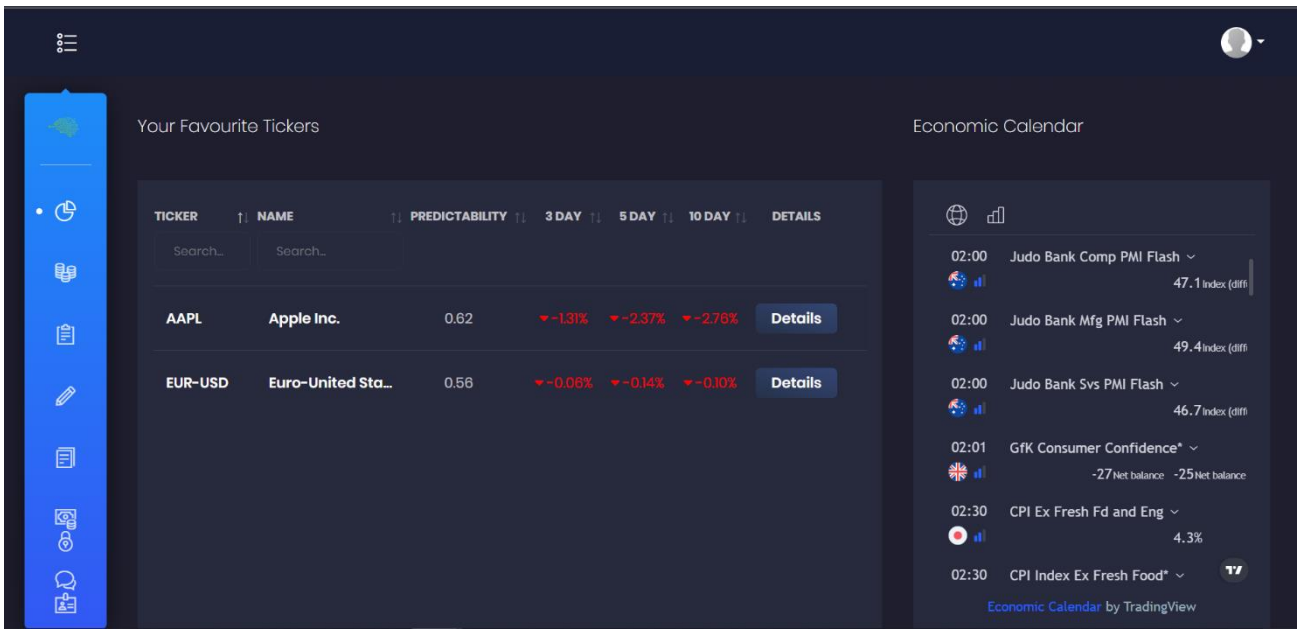

Рис. 1.6. Виведення результатів прогнозування цін активів у сервісі *FinBrain*

Крім того, сервіс надає додатковий функціонал та аналітику, зокрема економічні календарі, фінансові звіти, інсайдерські транзакції, перегляд та аналітику новинного фону та оцінку настроїв (*sentiment score*): *FinBrain* надає розрахований показник оцінки настрою для кожного активу. Ця оцінка підсумовує загальні настрої — позитивні, негативні чи нейтральні — на основі аналізу останніх новин та інших джерел даних, що стосуються цього активу, а також візуальне представлення зміни цього показнику в графічному вигляді, що відображає тенденцію зміни настрою з часом для того чи іншого активу. Платформою також надається функціонал алгоритмічної торгівлі для автоматизації торгів на основі програмно-прийнятих рішень.

Доволі широким є покриття різних класів фінансових інструментів, яке включає:

– акції: *FinBrain* пропонує прогнози для окремих акцій на основних світових фондових біржах. Це включає акції США, Європи, Азії та інших основних ринків, включаючи акції великої, середньої капіталізації та малої капіталізації;

– біржові фонди (*ETF*): платформа надає прогнози для широкого спектру *ETF*, що охоплюють різні сектори, товари та глобальні ринки;

– криптовалюти: *FinBrain* також надає прогнозні оцінки та аналітику для таких основних криптовалют, як *Bitcoin, Ethereum* та інших;

– іноземна валюта (*Forex*): платформа надає прогнози для основних валютних пар, охоплюючи такі пари валют, як долар США, євро, фунт стерлінгів та єни тощо;

– товари (*Commodities*): у сервісі *FinBrain* також доступні прогнози для основних товарів, включаючи, нафту, золото та сільськогосподарські продукти.

Також дана платформа пропонує власний *API*, що дає можливість отримання в уніфікованому форматі та використання прогнозних даних та аналітики користувачами у інших програмах, інструментах і системах та подальшої обробки цих даних програмними засобами. Це великий плюс даного сервісу, оскільки це дає гнучкість та можливість індивідульної інтеграції результатів роботи сервісу *FinBrain* користувачами у власні процеси інвестиційного менеджменту.

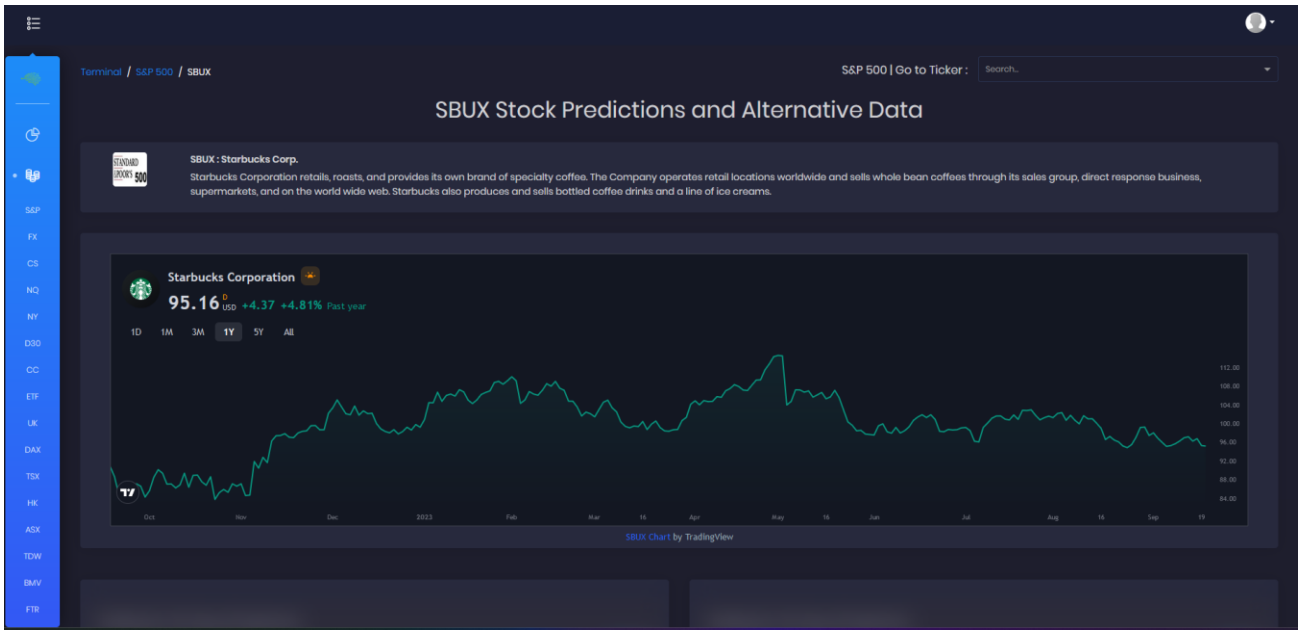

Рис. 1.7. Приклад подання аналітики платформою *FinBrain* на прикладі компанії *Starbucks*

Використання *FinBrain* можливе як на безкоштовній, так і на платній основі (за оформленням підписки). Деякий функціонал доступний та може

використовуватись без оформлення платного членства, як наприклад загальна аналітика, доступ до історичних даних, перегляд технічних індикаторів активів по компаніях, фінансової звітності та поверхневої аналітики. Тим не менш, більшість функціоналу, зокрема детальна аналітика прогнозованих даних, оцінка настроїв новинного фону по компаніям та інфографіка їх змін з часом, функціонал алгоритмічної торгівлі, а також доступ та використання *API* відносяться до платних послуг.

Мінімальна вартість підписки складає $840 за рік користування.

1.1.4. Підсумки огляду існуючих програмних рішень

Вибір існуючих рішень для прогнозування фінансових показників та інвестиційної аналітики для огляду та порівняльного аналізу був зроблений з огляду на фактори, їх використовуваності, поширеності, наданих функціональних можливостей та доступності.

Розглянуті інструменти є доволі непоганими програмними продуктами, які надають широкий функціонал, та є використовуваними великою кількістю користувачів, які ведуть різну інвестиційну діяльність. Обсяг інформації та аналітики, що пропонується користувачам цими сервісами є значним та охоплює широкий спектр ринків, бірж, фінансових інструментів, секторів та категорій. Тим не менш, кожний з цих інструментів володіє своїми сильними та слабкими сторонами та окрім запропонованого широкого спектру функціональних можливостей не позбавлений своїх недоліків.

Перш за все, варто відмітити, що спільним недоліком всіх розглянутих інструментів є обмежена гнучкість. Це простежується насамперед у неможливості гнучкого довільного вибору горизонту прогнозування (наявності прогнозів лише для заздалегідь визначених та обмежених часових інтервалів): у випадку з *I Know First* прогнози наявні лише на діапазони в 3 / 7 /14 / 30 / 90 днів та 1 рік, *TipRanks* пропонує лише один фіксований інтервал – 1 рік, а *FinBrain* діапазони 3, 5, 10 днів або ж 3, 6, 12 місяців.

Крім цього, жоден з розглянутих інструментів не надає можливості сумістити на одній координатній площині прогнозоване значення фінансового показника кількох довільних активів. Подібна функція могла б бути вкрай корисною для користувача, так як вона дає нагоду порівняти очікуваний ріст або падіння вартості кількох активів одночасно і обрати саме той, що якнайкраще відповідає інвестиційним очікуванням. До того ж, доцільним такий функціонал є при побудові та ребалансуванні портфелю, так як дозволяє вчасно помітити тренди в зміні його вартості, що можуть відрізнятись від цілей користувача.

Важливим також є той факт, що в сервісі *I Know First* можливості обрати для прогнозування вартості активу за бажанням користувача нема як такої – майбутні значення розраховуються для набору цінних паперів, і обираються для публікації лише ті, що мають найбільш сильну імовірність росту чи падіння ціни. Таким чином продукт позбавляє інвесторів в свободі вибору того, перспективи яких активів вони бажають оцінити – подається лише обмежений перелік того, що за результатами роботи алгоритму має перспективу достатньо відчутно змінитись у ціні.

Вагомим недоліком також є сума, яку повинен заплатити користувач, щоб отримати повноцінний доступ до функціоналу описаних продуктів. На відміну від *I Know First*, який надає дані виключно платним користувачам, *TipRanks* і *FinBrain* надають інформацію за моделлю *freemium* – це означає, що базовий функціонал доступний безкоштовно, однак в повній мірі сервіси розкривають свої можливості при підписці на сервіс, вартість якої вимірюється сотнями доларів. За такою завісою часто приховується саме той функціонал, який користувачу і потрібен, як-от прогноз вартості акцій по днях та місяцях від *FinBrain*, або конкретні оцінки аналітиків щодо ціни активу в майбутньому замість узагальненого прогнозу, що публікується *TipRanks*. Враховуючи це, у кінцевого користувача виникає відчутна перешкода для досягнення цілей використання сервісу загалом.

В аспекті варіативності використання важливою можна назвати можливість передачі даних з ресурсів не лише в текстовому чи візуальному

поданні, але і у форматах, що можуть бути опрацьовані програмно та/або бути експортованими для подальшого аналізу та використання. *FinBrain* надає доступ до своїх даних в тому числі через *API*, однак його використання допускається лише при купівлі платної підписки на сервіс. Разом із тим, тарифний план, що надає опцію надсилати запити до *API*, є в півтори рази дорожчим за базову версію, що надає доступ до тієї ж інформації, однак виключно через сайт.

TipRanks дозволяє експортувати лише певну частину розміщеної інформації у форматі *PDF* або *CSV* при наявності активної підписки. Це обмежує здатність користувача або клієнта опрацьовувати подібні дані автоматизовано, так як така операція ініціюється в ручному режимі, а не через програмний інтерфейс, і формат отримуваних даних потребує їх подальшої обробки перед програмним опрацюванням. Подібних труднощів зазвичай вдається уникнути за рахунок використання форматів даних, які є найбільш поширеними для обміну даними між комп'ютерами, такими як *JSON* або *XML*.

Також до недоліків *TipRanks* можна віднести сам принцип, за яким у ньому формується прогнозована вартість активів. На відміну від інших розглянутих аналогів, у ньому дані значення формуються як усереднена оцінка аналітиків, а не спеціальних алгоритмів чи штучних нейронних мереж. Як свідчить практика, людський фактор в напрямку такого роду нерідко призводить до помилкових рішень, так як людям складно зберігати об'єктивність і безпристрасність, що є ключовими поняттями при роботі з фінансовими активами. До того ж, програмні засоби володіють здатністю оперувати значно більшим об'ємом даних, мають ширший і швидший доступ до інформації, ніж це є можливим для людини.

Серйозним недоліком, який є властивим лише для одного з проаналізованих аналогів, а саме для *FinBrain*, є нестійка доступність сервісу. При відкритті сторінки будь-якого активу її завантаження або викликало помилку з'єднання, або відбувалось надто довго (до хвилини). В світі надшвидкої обробки та передачі інформації подібні затримки є неприпустимими, і в сфері фінансів особливо, так як тут лічені секунди можуть мати значення.

Тому затримка такого порядку може відштовхнути користувача від використання ресурсу, а також призвести до негативних наслідків, збитків.

Результат аналізу розглянутих існуючих систем в розрізі виявлених в них переваг та недоліків відображено в таблиці 1.1.

Таблиця 1.1

Порівняння сервісів прогнозування фінансових показників

Назва	Переваги	Недоліки
I Know First	– Широке охоплення фінансових інструментів, ринків, класів активів – Емпірично підтверджений високий рівень точності прогнозів	– Висока вартість підписки на сервіс – Відсутність свободи вибору активу для прогнозування – Застарілий графічний інтерфейс – Спосіб доставки даних користувачу унеможливлює автоматизацію взаємодії з сервісом
TipRanks	– Надання обширного переліку даних про активи – Підбір інвестиційних ідей на основі прогнозованого росту вартості активу – Можливість фільтрації та пошуку активів за бажаними для користувача показниками прогнозованої зміни вартості – Наявність прогнозу не лише на вартість активу, але й інші фінансові показники	– Обмежений функціонал безкоштовної версії, висока вартість підписки – Прогнози базуються на висновках аналітиків, а не обчислюються алгоритмічно – Вузький горизонт прогнозування – Відсутність *API* та зручного способу експорту даних – Неможливість прогнозування для кількох активів одночасно

Продовження таблиці 1.1

Назва	Переваги	Недоліки
FinBrain	– Використання нейронних мереж на основі великої кількості фінансових даних – Інтерактивність візуальних даних, інфографіка, графічний інтерфейс користувача – Наявність *API*	– Використання *API* допускається при рості вартості підписки в 1,5 рази – Нестабільна робота ресурсу – Неможливість порівняння прогнозу для кількох компаній – Фіксовані горизонти прогнозування

1.2. Постановка задачі

Зважаючи на зазначені вище результати огляду та порівняльного аналізу існуючих програмних рішень, розробка програмного модуля аналізу та прогнозування фінансових показників активів на фондовому ринку є актуальною.

Відповідно у даній роботі необхідно вирішити наступні задачі:

– дослідження властивостей, взаємозв’язків та залежностей фінансових показників та факторів, які їх формують та на них впливають;

– обгрунтований аналітичний відбір факторів, на основі яких проводитиметься прогнозування, залучення різнопланових факторів та показників для більш точного моделювання фінансових процесів;

– формування алгоритмічного підходу до прогнозування, який включатиме множину фінансових даних різного походження як базис для формування прогнозних оцінок та буде здатним відображати актуальну повістку фінансових ринків, виділяти тренди та прогнозувати динаміку цін активів.

Функціональні вимоги до розроблюваного програмного включають:

– наявність свободи вибору активів для прогнозування;

– гнучкий вибір горизонту прогнозування, можливість виконання як короткострокових, так і довгострокових прогнозів;

– можливість виконання та перегляду прогнозної аналітики в розрізі портфелю, а не лише окремих компаній, можливість перегляду значень показників сформованого користувачем інвестиційного портфелю через заданий проміжок часу в майбутньому;

– наявність механізму зручного та швидкого експорту даних (результатів роботи програмного продукту) у форматі зручному для використання цих даних в подальшій аналітиці, або для їх програмної обробки.

Основною метою розробки програмного модулю є його подальша інтеграція у інші програмні системи фінансової та інвестиційної спеціалізації, а отже важливою вимогою до нього є забезпечення уніфікованого та максимально сприятливого для цього інтерфейсу.

1.3. Висновки до розділу

В розділі було проведено огляд предметної області та визначено які проблеми та задачі є актуальними в сучасному процесі інвестиційного менеджменту, розглянуто важливість задачі прогнозування фінансових показників для якісного виконання відповідних його етапів та підпроцесів та проблематику її вирішення враховуючи природу фінансових даних, впливаючі на них фактори, їхні взаємозв’язки та залежності. Крім того було розглянуто декілька вже існуючих програмних рішень цієї задачі, проведено їх порівняння та аналіз виявлених в них переваг та недоліків.

В результаті дослідження вищезазначених питань та аспектів, а також проведеного аналізу аналогічних вже розроблених програмних рішень та їх недоліків було означено перелік задач, які залишаються актуальними для вирішення, виділено чинники та характеристики, які є важливими для розроблюваних програмних рішень цієї спеціалізації.

На основі отриманих результатів огляду предметної області та аналізу існуючих програмних рішень було проведено постановку задач для розроблюваного програмного модулю та визначено для нього функціональні та нефункціональні вимоги.

РОЗДІЛ 2
ПРОЕКТУВАННЯ ПРОГРАМНОГО МОДУЛЯ

Виконаний у розділі 1 огляд та аналіз існуючих програмних рішень дозволив виділити обмеження та недоліки інсуючих програмних продуктів для надання функціоналу прогнозування з точки зору користувача. Цей крок є необхідним для формування вимог до розроблюваного програмного продукту з перспективи потреб до функціональної його складової. Наступним важливим кроком є огляд та аналіз існуючих наукових досліджень, які присвячені вирішенню задачі прогнозування для оцінки її проблематики на більш низькому рівні – з технічної точки зору. Цей крок потрібен для поглиблення розуміння проблематики існуючих підходів та методик, і є основою для подальшого формування принципів нового підходу, окреслення необхідних для реалізації алгоритмічних аспектів та формулювання деталізованих технічних вимог та специфікації розроблюваного рішення.

2.1. Аналіз досліджень

Прогнозування цін акцій було предметом цілого ряду досліджень протягом останніх років.

Lu et al. [11] запропонували нову модель для прогнозування цін на акції за допомогою комбінації згорткової нейронної мережі (*Convolutional neural network, CNN*) та мережі з довгою короткочасною пам’яттю (*Long short-term memory, LSTM*). *CNN* використовується для ефективного вилучення ознак з даних, а *LSTM* – для прогнозування ціни на акції з отриманими ознаками даних. Модель використовує щоденні ціни на акції з 1 липня 1991 року по 31 серпня

2020 року, що охоплює 7127 торгових днів. Для цього аналізу було обрано вісім характеристик: ціна відкриття (*Open*) та закриття (*Close*), найвища (*High*) і найнижча (*Low*) ціна, обсяг (*Volume*), оборот (*Turnover*), підйоми і падіння (*Ups and Downs*) та зміни (*Change*).

Структура моделі *CNN-LSTM* була побудована з тривимірного вектора даних, згорткового шару (*convolution layer*), шару об'єднання (*pooling layer*) та шару *LSTM* для навчання даних та отримання вихідного (цільового) значення.

Дослідження порівнювало ефективність *CNN-LSTM* з іншими моделями, такими як багатошаровий перцептрон (*Multilayer perceptron, MLP*), *CNN*, рекурентна нейронна мережа (*Recurrent neural network, RNN*), *LSTM* та *CNN-RNN*. Модель *CNN-LSTM* продемонструвала найвищу точність прогнозування серед всіх, із найменшою середньою абсолютною похибкою (*MAE*) та середньоквадратичною похибкою (*RMSE*). Зокрема, *MAE* і *RMSE* для *CNN-LSTM* були 27,564 і 39,688 відповідно. Це свідчить про значне покращення порівняно з іншими моделями, демонструючи перевагу моделі *CNN-LSTM* як щодо ступеня відповідності, так і значення помилки.

Jarrah et al. [12] мали на меті спрогнозувати індекси фондового ринку Саудівської Аравії за допомогою підходу глибокого навчання з використанням багатовимірних даних часових рядів, які включають різні змінні, такі як початкова, найнижча, найвища ціна та ціна акцій на момент закриття ринку.

Застосований метод включає кілька етапів, починаючи з використання експоненціального згладжування (*ES*) для усунення шуму з даних. Після цього застосовувався метод рухомого вікна з п'ятьма кроками, щоб перетворити задачу прогнозування часових рядів у контрольовану навчальну задачу. Фінальним кроком було використання мультиваріативного *LSTM* алгоритму глибокого навчання для прогнозування цін на фондовому ринку.

Запропонована мультиваріативна модель глибокого навчання *LSTM* досягла значення точності прогнозування 97,49% і 92,19% для однофакторної моделі. Такий результат підкреслює доцільність використання багатьох джерел інформації для прогнозування цін на фондовому ринку.

Akita et al. [13] досліджували можливість створення моделі глибокого навчання для покращення прогнозування цін акцій на фондовому ринку шляхом використання як числових, так і текстових даних.

Запропонована модель застосовує вектор абзацу (*Paragraph Vector*), щоб отримати розподілене представлення кожної з доступних новин про компанію. Даний процес було проведено з поєднанням водночас обидвох категорій вектору абзацу, а саме *Distributed Memory Model of Paragraph Vectors* (*PV-DM*) і *Distributed Bag of Words of Paragraph Vector* (*PV-DBOW*).

Отримані розподілені представлення та щоденні ціни відкриття 50 компаній Токійської фондової біржі використовуються для передбачення ціни закриття за допомогою регресійного аналізу. Для врегулювання впливу часових рядів була використана модель довготривалої короткочасної пам'яті (*LSTM*).

Для оцінювання точності отриманих результатів використовувалась симуляція ринку, за якою імітувалась або купівля акції при передбаченому рості ціни, або ж її продаж у випадку передбаченого зниження ціни. Мірою точності прогнозування виступав сукупний прибуток, отриманий у результаті проведення симуляції на вказаному тестувальному проміжку. За результатами оцінювання досліджувана модель показала кращий результат у 4 з 5 секторів розглянутих компаній та отримала кращий сумарний результат (прибуток) у розмірі 12,1 млн єн. Даний показник порівнювався з результатами, отриманими методом опорних векторів, *MLP* і *RNN*, що дорівнювали мінус 0,47, мінус 5,6 та 2,6 мільйонів єн відповідно, що показує підвищення рівня точності прогнозування при залученні текстових даних для навчання моделі.

Запропоновані у розглянутих дослідженнях рішення демонструють високі показники точності прогнозування, однак вони містять ряд важливих недоліків та недопрацювань.

По-перше, вони являють собою досить складні моделі, однак суттєвим недоліком більшості з них є використання лише даних про історичну зміну ціни акцій, тобто задіюється лише технічний аналіз активів. При цьому упускається той факт, що ціна тієї чи іншої акції у довільний момент часу є наслідком

переліку факторів, які не обмежуються лише історичними даними, а включають також як показники фундаментального аналізу, дані із фінансової звітності компанії, розподіл її активів, так і макроекономічну ситуацію. Таким чином ігнорується значна частина факторів, які мають прямий та безпосередній вплив на величину, що підлягає прогнозуванню.

По-друге, у дослідженнях інколи використовують технічні показники як додаткові вхідні дані для моделей, однак підхід до вибору цих показників часто не є обґрунтованим. Сучасні підходи до роботи з характеристиками даних (*feature engineering*) включають в себе дослідження залежностей між показниками та цільовою величиною. Крім того, важливим є відбір найбільш відповідних показників на основі визначених критеріїв, що дозволяє підвищити ефективність та точність моделі. Відсутність такого системного підходу може призвести до погіршення якості прогнозування.

По-третє, фінансові дані за своєю природою є дуже "зашумленими". Це означає, що вони можуть містити багато випадкових відхилень, які не мають відношення до основних тенденцій ринку. Недостатнє або ж відсутнє використання механізмів для попереднього очищення даних може призвести до введення моделі в оману та зменшення її загальної ефективності. Сучасні підходи до роботи з характеристиками даних також передбачають методи видалення шуму.

Також потрібно враховувати той факт, що сучасний ринок акцій часто реагує не тільки на числові показники, але і на новини, звіти, соціальні мережі та інші текстові джерела. Ігнорування цієї інформації може призвести до неповного розуміння ринкових процесів, і, як наслідок - зниженої якості моделювання цих процесів і точності отриманих результатів прогнозування. Використання обробки природної мови (*Natural Language Processing, NLP*) та аналізу настрою (*Sentiment Analysis*) може допомогти збагатити моделі цією додатковою інформацією, яка збільшить їхню точність.

2.2. Принципи запропонованого підходу

У світлі аналізу існуючих підходів до прогнозування цін на акції та виділених у них недопрацювань та обмежень, було розроблено новий підхід, який ставить за ціль виправити ключові недоліки попередніх методик. Запропонований підхід базується на наступних принципах:

– системний підхід до відбору та використання показників за рахунок впровадження сучасних методик *feature engineering* та *feature selection*. Це передбачає детальне дослідження залежностей між різними числовими показниками та цільовою величиною, а також застосування автоматизованих методів відбору ознак;

– знешумлення фінансових даних. З урахуванням природної «зашумленості» фінансових даних, важливим елементом запропонованої методики є видалення шуму у вхідних даних на етапі їх попередньої обробки задля зосередження на основних тенденціях ринку;

– врахування текстової інформації. Застосування технік *NLP* та аналізу настрою дозволяє інтегрувати новини, звіти та інші текстові джерела інформації, які можуть впливати на ринкові тенденції та відображати актуальну повістку, яка впливає на цільову величину;

– інтеграція фундаментального та макроекономічного аналізу з технічним. Розроблювана прогнозна модель окрім технічних показників також включає в себе ключові показники фундаментального аналізу, такі як чистий прибуток компанії, а також макроекономічні показники, які мають прямий вплив на фондовий ринок.

Ці засади створюють базу для розробки ефективної та більш точної моделі прогнозування, яка відображає сучасні тенденції зміни акцій тієї чи іншої компанії на фондовому ринку.

2.3. Вибір факторів для прогнозування

Реалізація задачі прогнозування цін активів на практиці є досить складною, адже вимагає врахування великої кількості факторів, які впливають на динаміку цін конкретних активів та ринку загалом, а також визначення відповідного способу включення представлень цих факторів до фінансових моделей.

Також варто відзначити, що успіх будь-якого підходу або методики у вирішенні задачі регресії значним чином залежить від розуміння природи, характеристик та взаємозв’язків даних, з якими задача має справу. Ретельний відбір, дослідження та аналіз вхідних даних створює надіний фундамент для будь-яких зусиль моделювання та прогнозування. Тому першим кроком вирішення задачі прогнозування цін акцій на фондовому ринку є застосування знань, положень та результатів наукових досліджень з області економіки та фінансів для формування набору ознак для включення до розроблюваної моделі.

Множина факторів, які мають або потенційно можуть мати вплив на динаміку цін фінансових інтсрументів та їх взаємозв'язки ретельно вивчаються інвесторами, вони використовують різні підходи для аналізу їх впливу на ринок для прийняття обгрунтованих інвестиційних рішень. Вони характеризуються різним характером впливу та його природою - у той час як деякі фактори можуть викликати негайні реакції в цінах акцій, вплив інших може бути видимим здебільшого у довгостроковій перспективі, взаємозв’язки між ними можуть мати як лінійний, так і нелінійний характер. Основні фактори, які впливають на ціну акцій, можна розділити на три категорії: фундаментальні, технічні і макроекономічні.

Фундаментальні фактори відображать фінансовий стан компаній-емітентів активів, вони включать доходи, прибуток, рівень заборгованості компанії, тощо. Аналіз цих факторів полягає у дослідженні фінансової звітності, наприклад звітів про прибутки і збитки (*Income statements*) і баланс компанії (*Balance sheet*) та проведеній на основі цього оцінці активів. Даний підхід лежить в основі методу фундаментального аналізу. Вони включають зокрема:

1. *P*/*E* (*Price-to-Earnings*) *Ratio*, або кошторис ціни до прибутку, це один із ключових показників, які використовують інвестори для оцінки фінансової ефективності та оцінки акцій компанії. *P*/*E Ratio* обчислюється шляхом ділення поточної ринкової ціни акції на прибуток на акцію (*EPS*). Цей показник може використовуватися для оцінки того, наскільки ринкова ціна акції вище або нижче прибутку, який компанія генерує. Високий *P*/*E Ratio* може вказувати на те, що інвестори очікують високого зростання прибутку у майбутньому або просто переплачують за акцію на даний момент. З іншого боку, низький *P*/*E Ratio* може свідчити про недооцінку акції або очікування низького зростання прибутку [14].

2. *P*/*B* (*Price-to-Book*) *Ratio*, або кошторис ціни до вартості активів обчислюється шляхом ділення поточної ринкової ціни акції на вартість активів компанії, враховуючи їх балансову вартість. Якщо *P*/*B Ratio* менше 1, це може свідчити про те, що акції продаються нижче їх балансової вартості, що може вказувати на потенційно недооцінені акції. Якщо *P*/*B Ratio* більше 1, це може вказувати на переоцінку акцій.

3. *PEG* (*Price*/*Earnings-to-Growth*) *Ratio* – це фінансовий показник, який поєднує в собі два ключові фактори: коефіцієнт ціни до прибутку (*P*/*E Ratio*) і очікуваний ріст прибутку компанії [14]. Він допомагає інвесторам оцінити вартість акції відносно очікуваного росту прибутку компанії: якщо він дорівнює 1, то це може свідчити про те, що ринкова ціна акції достатньо відображає очікуваний ріст прибутку, якщо ж воно менше або більше 1, то акція вважається недооціненою або переоціненою відповідно. *PEG Ratio* корисний, оскільки він враховує не тільки поточну прибутковість компанії (*P*/*E Ratio*), але і її потенціал для зростання в майбутньому.

4. Дивідендна дохідність (*Dividend Yield*) – це фінансовий показник, який вказує, яку частину поточної ціни акції складають виплачені дивіденди. Висока дивідендна дохідність може бути привабливою для інвесторів, які шукають стабільний дохід від своїх інвестицій. Однак висока дивідендна дохідність також може вказувати на проблеми всередині компанії, такі як низька прибутковість або нездатність інвестувати у зріст.

5. *ROE* (*Return on Equity*), або прибуток на власний капітал відображає ефективність використання власних коштів акціонерів компанією для генерації прибутку. *ROE* виражається у відсотках і вказує на те, яку частину власного капіталу компанія здатна заробити як чистий прибуток. Вищий *ROE* зазвичай свідчить про більшу ефективність використання капіталу та вищу прибутковість для акціонерів.

6. *Current Ratio* (поточний коефіцієнт) – це фінансовий показник, який вимірює здатність компанії погасити свої короткострокові зобов'язання (зобов'язання, які мають бути сплачені протягом одного року) за допомогою своїх поточних активів. Цей показник важливий для оцінки фінансової стійкості компанії та її здатності впоратися з короткостроковими фінансовими зобов'язаннями, такими як платежі по кредитам чи зобов'язання перед поставщиками.

7. *Quick Ratio*, також відомий як *Acid-Test Ratio* або *Liquidity Ratio*, – це фінансовий показник, який використовується для вимірювання готівкової ліквідності компанії, тобто її здатності виконати короткострокові зобов'язання, враховуючи лише найбільш ліквідні активи (готівку, еквіваленти готівки та цінні папери, які можна легко перетворити на готівку). Зазвичай більший *Quick Ratio* вважається більш безпечним, оскільки це означає, що компанія має більше ліквідних активів для покриття своїх зобов'язань.

Технічні фактори (індикатори) є предметом технічного аналізу – іншого методу аналізу активів. Він базується на використанні історичної інформації про ціни акцій і обсяги торгів, на обчисленні фінансових змінних та показників, і подальшому їх використанні для передбачення майбутньої динаміки цін активів та розробки вдповідних інвестиційних стратегій. Прикладами таких індикаторів є:

1. Рухомі середні (*Moving Averages*) – це один із ключових інструментів у технічному аналізі фінансових ринків. Вони використовуються для визначення загальних тенденцій ціни акцій, що допомагає інвесторам і трейдерам приймати рішення про їх купівлю або продаж.

Види рухомих середніх включають просте (*SMA*), зважене (*WMA*) та експоненціальне (*EMA*). Вибір періоду і типу рухомого середнього залежить від інвестиційних уподобань та характеру аналізованих даних. Короткі періоди відображають поточну динаміку ціни акцій, тоді як довгі періоди роблять його більш згладженим і придатним для визначення та аналізу довгострокових трендів.

2. Індикатор сходження / розходження рухомих середніх (*MACD*) – це технічний показник осциляторного типу, який показує силу тренду та дозволяє відстежувати його зміну. *MACD* використовує рухомі середні для визначення імпульсу (моментуму) цін акції або іншого торгового активу. Являючи собою індикатор моментуму, *MACD* є корисним показником для інвесторів, оскільки він дозволяє визначити швидкість руху ціни та імовірність збереження чи зміни її тренду [21].

3. Індекс відносної сили (*RSI*) – це технічний показник, який використовується в аналізі фінансових ринків, особливо на ринку акцій, для визначення ступеня перекупленості чи перепроданості цінних паперів. Дивергенція *RSI* вказує на можливість зміни тренду [5]. Наприклад, якщо *RSI* формує новий високий пік, а ціни не роблять цього, це може свідчити про втрату міцності поточного тренду.

Макроекономічні фактори відображають економічну ситуацію в цілому та характеризують економічне середовище в якому оперують компанії-емітенти. До них відносяться:

1. ВВП (Валовий внутрішній продукт – *Gross Domestic Product* або *GDP*) – це ключовий макроекономічний показник, який вимірює загальну вартість всіх товарів і послуг, що виробляються в економіці країни протягом певного періоду часу. Зростаючий ВВП вказує на здорову економіку, що може призвести до збільшення прибутків для компаній та потенційного підвищення цін акцій. І навпаки, зниження ВВП вказує на падіння економіки та, як результат – потенційного зниження прибутків компаній-емітентів та цін акцій [27].

2. *CPI* (*Consumer Price Index*), або Індекс споживчих цін, є економічним показником, який вимірює зміни середнього рівня цін на споживчі товари та послуги в країні протягом певного періоду часу. Зростання *CPI* вказує на інфляцію, а зменшення – на дефляцію. Наслідком інфляції є зменшення споживчих витрат і зниження прибутків компаній, що в свою чергу має потенційно негативний вплив на ціни акцій. З іншого ж боку, дефляція може сигналізувати про уповільнення економіки, що також може мати негативний вплив на ціни акцій.

3. Рівень безробіття (*Unemployment Rate*) – це ключовий економічний показник, який вимірює відсоток робочої сили, яка на даний момент є безробітною та активно шукає роботу в економіці. В контексті впливу на фондовий ринок вищий рівень безробіття призводить до зниження споживчих витрат, потенційно знижуючи прибутковість компаній та відповідно ціни акцій.

4. Процентні ставки (*Interest Rates*) – економічний показник, який визначає вартість (відсоток) користування позиковими грошима, які комерційні банки короткостроково займають один в одного у процесі своєї діяльності. Цей показник є корисним для розгляду у інвестиційному аналізі, оскільки при підвищенні процентних ставок вартість позичання грошей для бізнесу збільшується, що може знизити прибутковість компаній та як наслідок викликати зниження цін акції компаній на фондових ринках. І навпаки – нижчі процентні ставки знижують вартості запозичень для компаній, потенційно збільшуючи прибутковість і ціни на акції, водночас роблячи акції більш привабливими для інвестування порівняно з облігаціями.

Наведений перелік факторів на даному етапі був відібраний аналітично, базуючись на економічній та фінансовій літературі, а також на аналізі наукових праць, фокусом яких було дослідження залежностей та впливів економічних показників на тенденції змін цін активів на фондовому ринку [14, 27, 28].

Використання теоретичного базису на етапі вибору факторів було також доповнено проведенням аналізу історичних даних ціни акцій та історичних значень потенційних факторів для включення в набір для прогнозування. Для

цього графічні представлення даних ціни та потенційних предикторів було суміщено на одній координатній площині. Таким чином було відібрано фактори, які мають з ціною акцій залежність характеру позитивна або негативна кореляція.

Приклад виконання цього аналітичного кроку відбору факторів показано на рис. 2.1-2.1. На рисунку 2.1 приведено об'єднаний графік історичних даних фундаментальних факторів та ціни акцій на момент закриття торгів для компанії *Apple* (*AAPL*). Можна побачити, що більшість з цих факторів мають кореляцію з ціною, а отже – є потенційно гарними предикторами для включення в набір для прогнозування. На рис. 2.2 приведено об'єднаний графік історичних даних фундаментального показника *Dividend Yield* та ціни акцій компанії *Apple*. З цього порівняльного аналізу видно, що показник *Dividend Yield* та ціни акцій мають негативну кореляцію, що також говорить про те, що цей показник є потенційно вдалим предиктором для включення у прогнозну модель.

Рис. 2.1. Приклад виконання аналітичного відбору фундаментальних показників

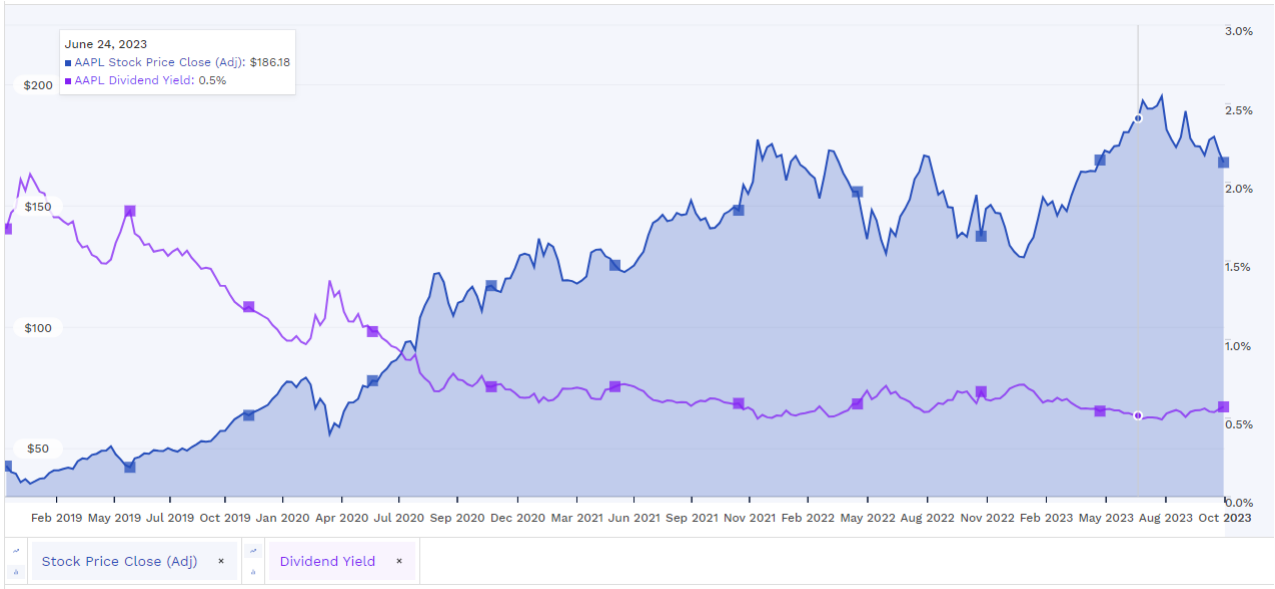

Рис. 2.2. Зміна історичних даних показника *Dividend Yield* та цін акцій

Структуру відібраного на цьому кроці набору факторів для прогнозування приведено на відповідній структурній схемі (рис. 2.3).

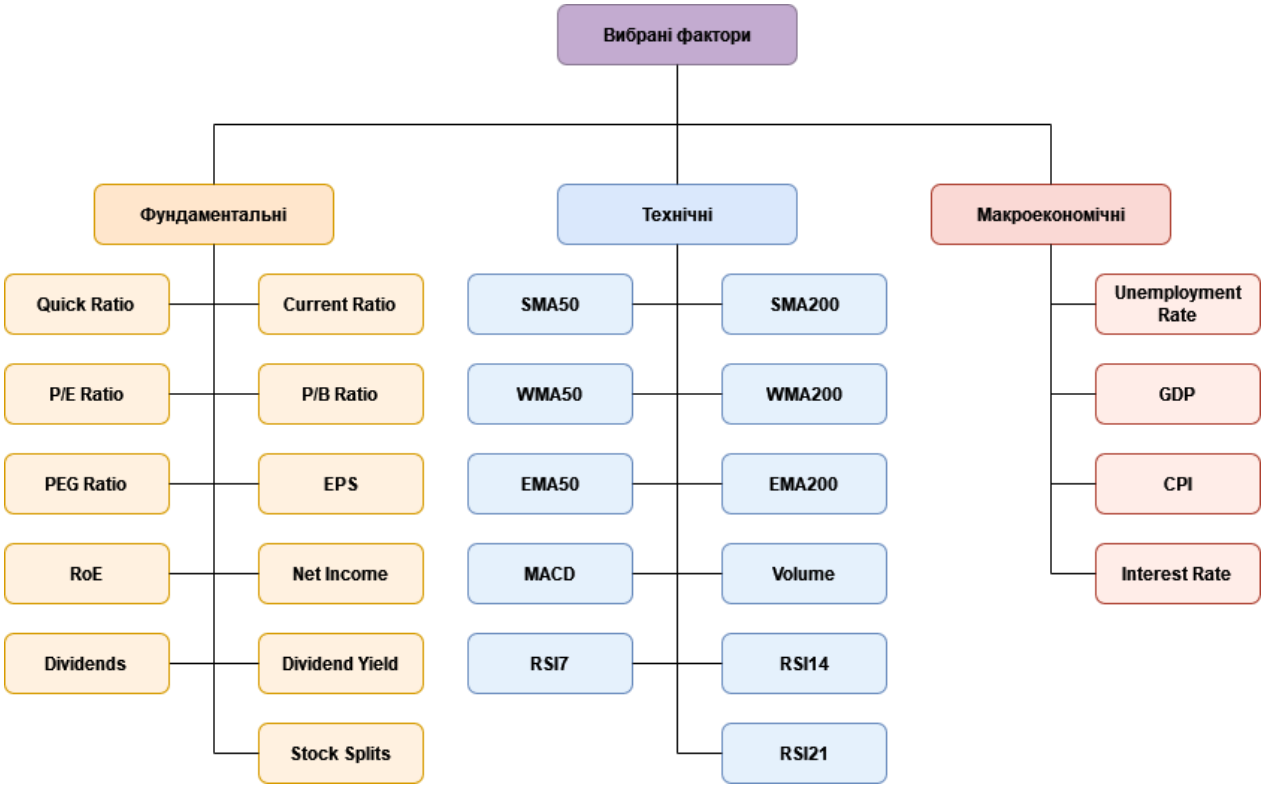

Рис. 2.3. Структурна схема набору вибраних факторів

2.4. Оцінка важливості ознак

Наступним етапом після аналітичного відбору факторів для прогнозування на основі знань предметної області в запропонованому підході є застосування алгоритмічних засобів обчислення показників важливості ознак, та відбір найбільш важливих ознак (найбільш вдалих предикторів) з вхідного набору.

Відбір ознак (*feature selection*) – це процес ідентифікації та вибору підмножини вхідних змінних, які є найбільш релевантними по відношенню до цільової змінної. Цей крок особливо важливий, оскільки він безпосередньо впливає на продуктивність та інтерпретованість розроблюваної моделі.

У регресії кожна ознака представляє незалежну змінну, яка потенційно пояснює варіації залежної змінної. Однак не всі ознаки сприяють прогнозуванню в однаковій мірі. Деякі можуть бути дуже інформативними, інші можуть нести надлишкову інформацію, а деякі можуть навіть створювати шум. Механізм вибору ознак передбачає виявлення цих відмінностей між ознаками. Мета полягає в тому, щоб зберегти ознаки, які мають найбільш суттєвий зв’язок із цільовою змінною, і відкинути відповідно найменш релевантні [17].

Цей процес є важливим для застосування після аналітичного формування початкового набору факторів, заснованого на економічних і фінансових теоріях і знаннях предметної області. Незважаючи на те, що попередній етап вибору факторів керується експертним досвідом, він не повністю враховує всіх нюансів в даних. Наприклад, деякі теоретично важливі фактори можуть не мати сильного емпіричного зв’язку з цільовою величиною у конкретному наборі даних.

Залучення *feature selection* також зменшує складність моделі, що робить модель швидшою та менш ресурсоємною. Зменшення складності в свою чергу підвищує інтерпретованість моделі. У фінансовому моделюванні розуміння того, чому модель робить певний прогноз, є майже таким же важливим, як і точність прогнозу. Модель з меншою кількістю ознак легше інтерпретувати та валідувати.

До того ж, у такій динамічній сфері, як фінанси, де ринкові умови швидко змінюються, модель, обтяжена нерелевантними функціями, може не впоратись з

тим щоб адаптуватись достатньо швидко. Спрощена модель із ретельно відібраними ознаками є більш гнучкою, що дозволяє швидше виконувати калібрування у відповідь на зміни ринкових тенденцій.

Загалом етап відбору ознак у вирішенні поставленої задачі – це не просто технічний крок у розробці моделі, а важливий міст між застосуванням теоретичних знань і емпіричним моделюванням даних. Він сприяє тому, що розроблювана модель залишається надійною, інтерпретованою та узгодженою зі складними реаліями фінансових ринків.

Серед методів алгоритмічного відбору ознак в даній задачі доречним є використання методу градієнтного бустингу. Градієнтний бустинг – це потужний метод, який базується на використанні ансамблів дерев рішень для обрахунку важливості ознак для прогнозної моделі.

Метод градієнтного бустингу полягає у формуванні ансамблю дерев рішень в поетапній манері – де кожна нова модель (дерево рішень) навчається виправляти помилки, зроблені попередніми. Під час процесу навчання обчислюється важливість ознак з наданого датасету – ознаки, які більш часто використовуються у формуванні ключових поділів у цих деревах рішень вважаються більш значими та мають вищий показник відносної важливості. Цей показник розраховується явно для кожного атрибута в наборі даних, що дозволяє ранжувати атрибути (ознаки) та порівнювати їх базуючись на його значеннях. Важливість обчислюється для окремого дерева рішень на основі значення того, наскільки кожна точка розділення атрибутів покращує показник продуктивності, зваженого за кількістю спостережень, за які відповідає вузол. Потім важливість ознак усереднюється по всіх деревах рішень у моделі.

Метод градієнтного бустингу відрізняє здатність ефективної обробки складних взаємозв'язків у вхідних даних, в тому числі і нелінійних, які є властивими для фінансових даних, а також його здатність роботи з великими наборами даних. Враховуючи специфіку, предметну область та природу вхідних даних у вирішуваній в даній роботі задачі, це робить метод градієнтного

бустингу вдалим вибором для реалізації етапу автоматизованого відбору ознак у розроблюваному рішенні.

2.5. Вибір засобу реалізації компоненту прогнозування

За своїми характеристиками фінансові дані являють собою типові часові ряди. Отже задача прогнозування цін активів на фондовому ринку зводиться до моделювання та прогнозування часових рядів.

Наявні засоби реалізації рішення цієї задачі включають статистичні методи, методи машинного навчання та методи глибокого навчання. Вибір же методу реалізації залежить від специфіки конкретної задачі та структури даних, з якими задача має справу.

Статистичні методи прогнозування часових рядів довгий час були фундаментальними в різних областях. Вони базуються на часових змінах у часовому ряді та добре працюють з однофакторними часовими рядами. Ці методи зосереджені на аналізі історичних даних часового ряду для виявлення закономірностей і тенденцій, які потім використовуються для прогнозування майбутніх значень цього часового ряду. Прикладами методів цієї групи є моделі сімейства *ARIMA* та метод експоненціального згладжування.

ARIMA (*AutoRegressive Integrated Moving Average* – авторегресійне інтегроване ковзне середнє): цей метод поєднує в собі авторегресію (*AR*), диференціювання (*I*) і ковзне середнє (*MA*). Частина *AR* передбачає використання минулих значень змінної (минулих спостережень) для означення її майбутніх значень. Частина *I* передбачає диференціювання необроблених спостережень, щоб зробити часовий ряд стаціонарним. Частина *MA* працює схожим чином з *AR*: вона також використовує минулі значення змінної для прогннозування її наступного знначення. Проте минулі значення, які використовує *MA*, є не безпосередніми значеннями змінної, а помилками передбачення в попередніх часових кроках. Концепція цього полягає в тому, що коли модель має деякі невідомі, але регулярні зовнішні збурення, то це означає

що модель може мати сезонність або інші закономірності у значеннях похибки. Модель *MA* – це метод, що дозволяє охопити цей шаблон, моделюючи помилку моделі як комбінацію минулих помилок. Сімейство *ARIMA* включає також модифікації з включенням компоненту сезонності – *SARIMA*, з включенням екзогенних (зовнішніх по відношенню до цільової величини) змінних – *ARIMAX* та *SARIMAX*, а також включенням компоненту векторної авторегресії – *VARMA* та *VARMAX*.

Експоненціальне згладжування: цей метод передбачає застосування експоненціально зменшуваних ваг до минулих спостережень, причому останнім спостереженням надається більша вага. Просте експоненціальне згладжування підходить для одновимірних часових рядів без трендових і сезонних компонентів. Модель лінійного тренду Холта розширює це, щоб включати дані з трендом, а сезонний метод Холта-Вінтерса може обробляти як тренд, так і сезонність.

Кожен із цих методів має конкретне застосування та має технічні переваги та обмеження.

Перевагами статистичних методів є перш за все їх висока інтерпретованість. Статистичні методи, як правило, легше зрозуміти та інтерпретувати порівняно зі складнішими моделями машинного та глибокого навчання. Також вони є менш вимоглими до обчислювальних ресурсів, простіші в реалізації та впровадженні.

Недоліками ж цього класу методів є перш за все те, що вони ефективні здебільшого лише для визначення лінійних зв’язків у даних часових рядів, оскільки самі по собі базуються на припущенні про лінійність цих зв’язків. До того ж вони мають очікування щодо стаціонарності вхідних даних. Враховуючи те, що фінансові дані є нестаціонарними та мають складні та в більшості випадків нелінійні зв’язки та залежності, методи цього класу є не найкращим вибором для реалізації рішення поставленої задачі. Крім цього дані методи за своєю суттю не враховують довгострокові залежності в часових рядах, що є критичним аспектом у задачі прогнозування у фінансовій предметній області.

Іншою групою засобів реалізації прогнозування часових рядів є класичні методи машинного навчання. На відміну від статистичних методів, які часто спираються на конкретні припущення щодо даних (наприклад, припущення щодо їх лінійності), машинне навчання пропонує більш гнучкий підхід, здатний фіксувати більш складні та нелінійні патерни у даних. Прикладами таких методів є метод опорних векторів (*SVM*) та випадкові ліси (*Random Forests*).

Метод опорних векторів (*Support Vector Machines – SVM*) використовується як в задачах класифікації, так і в задачах регресії. У контексті часових рядів їх можна використовувати для прогнозування майбутніх значень на основі минулих точок даних. *SVM* працюють, знаходячи гіперплощину в N-вимірному просторі (де N – кількість ознак), яка чітко класифікує точки даних. У завданнях регресії вони намагаються вмістити помилку в межах певного порогу [18].

Випадкові ліси: це метод ансамблевого навчання, який використовується як для задач класифікації, так і для задач регресії. Він працює шляхом побудови множини дерев рішень під час навчання, кожне з яких робить своє прогнозування цільової величини. Потім ці прогнози окремих дерев усереднюються для отримання остаточного прогнозу.

Незважаючи на те, що класичні методи машинного навчання пропонують більшу гнучкість і потужність порівняно з статистичними методами, вони мають ряд обмежень, які є суттєвими для задачі прогнозування цін акцій. Їх обмеження в охопленні складної динаміки, потреба в розширеному інжинірингу ознак і їх відносна неефективність в обробці неструктурованих даних роблять їх сумнівним вибором для використання у вирішуваній в даній роботі задачі.

Методи глибокого навчання є передовими в області ШІ та показали значний успіх у широкому діапазоні застосувань, включаючи задачі розпізнавання зображень і мовлення, обробку природної мови та, зокрема, прогнозування часових рядів. Методи глибокого навчання здатні вивчати складні закономірності та виявляти складні, в тому числі нелінійні, зв’язки у

вхідних даних. Методи глибокого навчання включають зокрема згорткові нейронні мережі (*CNN*) та рекурентні нейронні мережі (*RNN*).

Згорткові нейронні мережі (*Convolutional Neural Networks – CNN*) являють собою вид нейронних мереж прямого зв'язку (*feedforward neural networks*) та відомі своєю ефективністю насамперед в задачах класифікації та розпізнавання зображень і обробці природної мови, проте можуть застосовуватись також і для задач прогнозування часових рядів. Вони характеризуються високою здатністю ідентифікувати закономірності в даних і ефективно обробляти великі масиви даних, проте експериментальні результати проведених досліджень застосування таких моделей для прогнозування часових рядів показали, що точність прогнозування окремо взятих *CNN* є відносно низькою [18].

Рекурентні нейронні мережі (*RNN*) містять зворотні зв'язки у своїх прихованих шарах (*hidden layers*) і дозволяють короткочасно зберігати інформацію. Вони були розроблені спеціально для обробки послідовностей даних. Здатність пам'яті обумовлена наявністю рекурентних зворотніх зв'язків у цих моделях робить їх особливо перспективними для застосування в задачі прогнозування фінансових часових рядів, де минулі патерни у послідовності точок даних мають важливе значення. Тим не менш важливим нюансом класичних *RNN* є те, що при моделюванні даних за великим часовим горизонтом та при обробці великої кількості історичних точок даних з довготривалими залежностями для них характерне виникнення проблеми зникнення градієнта. Це значним чином знижує навчаємість моделі та в цілому її прогнозну здатність в сценаріях, де важливе значення має розуміння всього контексту (всього обсягу інформації з таймлайну набору даних) для виконання прогнозу.

Спеціалізованою формою рекурентних нейронних мереж є мережі довготривалої короткочасної пам'яті (*Long Short-Term Memory Networks – LSTMs*). Вони містять у собі всі переваги та сильні сторони *RNN*, вміють фіксувати довготривалі залежності в послідовностях даних, здатні обробляти великі, складні набори даних, виділяти та вивчати комплексні патерни та нелінійні зв'язки у них. Враховуючи специфіку задачі прогнозування фінансових

часових рядів та характер даних, з якими ця задача має справу, *LSTM* є найбільш перспективним кандидатом для вибору у розроблюваному рішенні. Тому внутрішню структуру, механізм та математичний апарат цього виду нейронних мереж є сенс розглянути детальніше.

Long Short-Term Memory (*LSTM*) мережі були запропоновані Хохрайтером та Шмідгубером як рішення проблеми зникнення градієнта, яка характерна для стандартних *RNN* при роботі з довготривалими залежностями. *LSTM* мережі впроваджують концепцію "вентилів" (*gates*), що контролюють поток інформації. Зв’язки між компонентами мережі *LSTM* приведені на рис. 2.4.

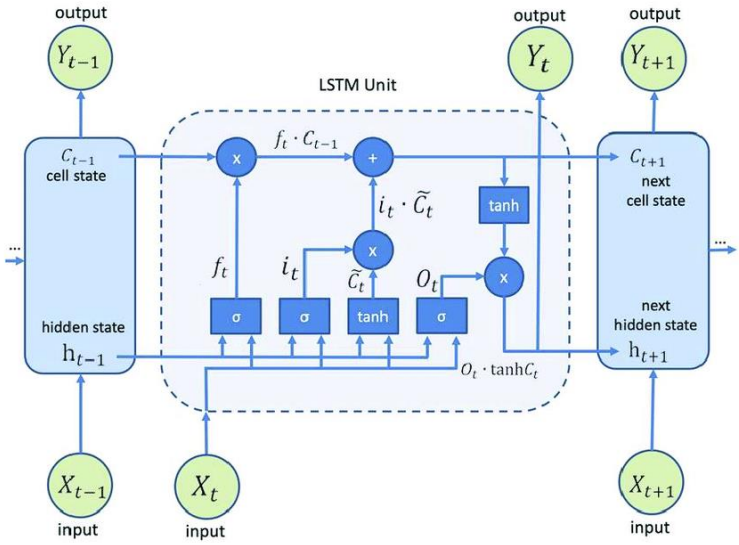

Рис. 2.4. Комірка *LSTM* у її загальній мережі

Структура *LSTM* складається з таких основних компонентів:

1. Комірка Пам'яті (*Cell State*)

Комірка пам'яті, C_t, є центральною частиною *LSTM* і працює як "носій" інформації через послідовні кроки часу. Вона має здатність додавати або видаляти інформацію через вентилі. Значення комірки вираховується за формулою (1):

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

де:

– C_t – оновлений стан комірки на кроці часу *t*;

– C_{t-1} – попередній стан комірки;

– f_t – забувальний вентиль на кроці часу *t*;

– i_t – вхідний вентиль на кроці часу *t*;

– $\tilde{C}_t$ – кандидат на новий стан комірки на кроці часу *t*, отриманий через гіперболічний тангенс, що дає значення між -1 та 1.

2. Вентилі (*Gates*)

Існують три основні типи вентилів у *LSTM*:

Вентиль забуття (*Forget Gate*) – визначає, яка частина інформації з попереднього стану комірки повинна бути забута. Він використовує сигмоїдну функцію активації для генерації значень між 0 та 1, де 0 означає повне "забуття", а 1 означає повне "зберігання" інформації, і обчислюється наступним чином (2):

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (2)$$

де:

– σ – сигмоїдна функція активації, що повертає значення між 0 та 1;

– W_f – вагова матриця забувального вентиля;

– $[h_{t-1}, x_t]$ – конкатенація попереднього прихованого стану h_{t-1} та поточного вводу x_t;

– b_f – вектор зсуву (*bias*) для забувального вентиля.

Вхідний вентиль (*Input Gate*) – визначає нову інформацію, яку слід додати до стану комірки. Вхідний вентиль також складається з сигмоїдної частини, яка вирішує, які значення оновлювати, та тангенсальної частини (3), яка створює новий кандидат на оновлення значень (4).

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (3)$$

$$\tilde{C}_t = tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \quad (4)$$

– W_i, W_C – вагові матриці вхідного вентиля та кандидата стану комірки;

– b_i, b_C – вектори зсуву для вхідного вентиля та кандидата стану комірки.

Вихідний Вентиль (*Output Gate*) – встановлює, яка частина інформації з комірки пам'яті буде використана як вихід на наступному кроці (5):

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (5)$$

– o_t – вихідний вентиль на кроці часу *t*;

– h_{t-1} – прихований стан на кроці часу *t-1*;

– W_o – вагова матриця вихідного вентиля;

– b_o – вектор зсуву для вихідного вентиля.

3. Прихований Стан (*Hidden State*)

Прихований стан, h_t (6), є виходом *LSTM* на кожному кроці часу, який може бути переданий до наступного кроку в послідовності або використаний для прогнозування.

$$h_t = o_t \cdot tanh(C_t) \quad (6)$$

– h_t – прихований стан на кроці часу *t*, який може служити як вихід мережі для цього кроку часу або як ввод для наступного кроку;

– *tanh* – Гіперболічний тангенс, функція активації, яка нормалізує значення стану комірки до діапазону між -1 та 1.

Робота *LSTM* на кожному кроці часу включає наступні кроки:

1. Визначення стану забувального вентиля f_t, що вирішує, яка інформація з попереднього стану комірки пам'яті C_{t-1} має бути забута;

2. Визначення вхідного вентиля i_t та кандидата на оновлення комірки пам'яті $\tilde{C}_t$, що вирішує, яку нову інформацію додати;

3. Оновлення комірки пам'яті з врахуванням інформації, яку треба забути та нової інформації, що додається;

4. Визначення вихідного вентиля o_t, який вирішує, яка частина оновленого стану комірки пам'яті буде використана у прихованому стані h_t;

5. Розрахунок нового прихованого стану, який базується на оновленому стані комірки пам'яті та вихідному вентилі.

Результатом є h_t, який може бути виходом моделі (наприклад, при прогнозуванні наступного значення в послідовності) або переданий до наступного кроку в послідовності. У випадку прогнозування фінансових часових рядів, h_t зазвичай проходить через додатковий шар (наприклад, повнозв'язний шар) перед тим, як здійснити фінальний прогноз.

Таким чином, архітектура *LSTM*, яка ефективно вирішує проблему довгострокових залежностей у даних часових рядів, є особливо корисною для

розуміння складних закономірностей і тенденцій, які керують тенденціями зміни цін акцій на фондовому ринку. Їхня здатність обробляти великі, комплексні набори даних і навчатися на них, а також їх ефективність у фіксуванні довгострівалих часових залежностей і нелінійних зв'язків у даних найкращим чином відповідають вимогам задачі прогнозування курсу акцій, природі вхідного набору даних та принципам запропонованого в даній роботі підходу.

2.6. Залучення текстової інформації

Ринкові умови та тенденції значним чином формуються під впливом інформаційних потоків які включають текстові дані, такі як новини про компанії, економічні показники, політичні події та глобальні події, новини про релізи нових продуктів компанії тощо. Динаміка цін активів на фондовому ринку залежить не лише від кількісних фінансових показників, а й від таких якісних факторів, як новини, громадська думка та соціально-економічні події.

Новини та медіаконтент надають у реальному часі інформацію про події та зміни, які можуть вплинути на ефективність компанії та її сприйняття інвесторами. Наприклад, оголошення про злиття, поглинання, зміни в керівництві або запуск нових продуктів можуть викликати негайну та значну реакцію на фондовому ринку. Це явище добре задокументовано у фінансовій літературі, де дослідження подій показали, як конкретні події новин корелюють із коливанням цін на акції та обсягами торгів [15].

Коли публікуються новини чи оголошення про компанію, вони часто містять інформацію, яка може істотно вплинути на майбутні перспективи компанії. Наприклад, позитивні новини, такі як сильні звіти про прибутки, успішний запуск продукту або вигідне партнерство, можуть підвищити довіру інвесторів. Ця впевненість зазвичай перетворюється на підвищений попит на акції компанії, що підвищує їх ціну. Інвестори сприймають компанію як таку, що має кращі перспективи зростання та прибутковості, що спонукає їх інвестувати більше у активи такої компанії в очікуванні майбутніх прибутків.

І навпаки, негативні новини, такі як юридичні проблеми, низькі доходи або скандали з керівниками, можуть викликати невпевненість і песимізм у інвесторів. Така інформація може призвести до перегляду майбутніх перспектив компанії, що часто призводить до підвищення продажу акцій, оскільки інвестори прагнуть уникнути потенційних втрат. Цей тиск продажів спричиняє падіння цін акцій, що відображає переглянуте, менш сприятливе уявлення ринку про потенціал компанії.

До того ж, вплив новин виходить за межі їх безпосереднього змісту. Реакція ринку також залежить від того, як новина порівнюється з існуючими очікуваннями. Наприклад, якщо компанія повідомляє про прибутки, які нижчі за очікувані, навіть якщо вони загалом позитивні, ціна акцій все одно може впасти. Така реакція виникає тому, що інвестори мали завищені очікування, і новини служать корективом, який перебудовує ціну акцій у відповідності з реальними фактами.

Ця концепція освітлюється у поведінковій фанансовій теорії та описує те, що інвестори у своїх рішеннях мають тенденцію піддаватись впливу інформації, яку вони споживають. Наприклад, основоположна стаття Тетлока (2007) «Надання вмісту настроям інвесторів» [30] демонструє, як медіа-песимізм створює тиск на зниження ринкових цін, підкреслюючи передбачувальну силу текстової інформації.

Отже, наступним важливим аспектом запропонованого підходу є інтеграція даних з текстових джерел та включення їх до розроблюваної прогнозної моделі. Включення цього типу даних може підвищити точність і надійність розроблюваної моделі, забезпечуючи більш повне розуміння факторів та залежностей, які впливають на динаміку цін акцій та які неможливо отримати лише з кількісних фінансових даних.

Інтеграція даних із текстових джерел, таких як новинні статті, у розроблювану модель прогнозування на базі глибокого навчання має проблему через різницю в типах даних. Нейронна модель обробляє числові дані часових рядів, текстові дані з новинних статей мають принципово іншу структуру. Для

вирішення цієї проблеми, необроблені текстові дані потрібно перетворити в структурований, кількісний числовий формат, який модель зможе інтерпретувати й аналізувати разом із числовими даними інших факторів. Саме тут аналіз настроїв стає ключовим інструментом.

Аналіз настроїв (*sentiment analysis*) – це техніка, яка використовується в обробці природної мови (*NLP*) для визначення емоційного тону тексту. Це робиться шляхом аналізу слів і фраз у тексті, щоб класифікувати тональність тексту за певною шкалою в числовому представленні, як-от позитивна, негативна чи нейтральна. Процес передбачає використання алгоритмів машинного навчання, навчених на великих наборах даних позначеного тексту, де ідентифікується емоційний тон кожного фрагмента тексту. Потім ці алгоритми можуть оцінювати нові фрагменти тексту, оцінюючи мову та контекст, щоб призначити оцінку настрою [29]. Ця оцінка кількісно визначає настрій тексту, перетворюючи якісну інформацію на кількісний показник (часовий ряд), що дозволить використати їх в якості додаткової ознаки у довонення до інших вхідних даних прогнозної моделі.

Визначною технікою в галузі *NLP*, зокрема для задачі аналізу настроїв, є модель *BERT* (*Bidirectional Encoder Representations from Transformers*). *BERT* є проривом у сфері *NLP* завдяки здатності до глибокого розуміння мовного контексту та нюансів. На відміну від попередніх моделей, які обробляли текст в одному напрямку (зліва направо або справа наліво), *BERT* розроблено для аналізу контексту слова відносно всіх інших слів у реченні, а не лише слів, які передувати або слідувати за ним. Цей двонаправлений контекст має вирішальне значення для розуміння наміру та настрою, що лежить у фрагменті тексту.

Структура *BERT* складається з серії моделей трансформерів, які є нейронними мережами на основі механізму уваги. Кожен трансформер у *BERT* переглядає всю послідовність введеного тексту одночасно, дозволяючи йому вивчати контекст і зв’язки між словами в усьому тексті. Навчаючись на великому наборі тексту, *BERT* вчиться розуміти нюанси та складність мови, зокрема сленг, ідіоми та різноманітні структури речень. Він аналізує текст, враховуючи як

використані слова, так і їхній контекст, і створює обчислену оцінку настроїв. Ця оцінка є кількісним відображенням настрою, вираженого в тексті, класифікуючи його як позитивний, негативний або нейтральний (число в діапазоні від 1 до 5, де 1 – різко негативний, а 5 – дуже позитивний).

Модель *BERT* є вдалим вибором для використання у розроблюваному рішенні з ряду причин. По-перше, його глибоке розуміння контексту дає змогу детальніше та точніше аналізувати фінанасові новинні статті, які часто є складними та містять галузевий жаргон. Здатність *BERT* розуміти контекст означає, що він може ефективно розпізнавати настрої в цих текстах, навіть якщо настрої передаються тонко.

По-друге, на фінансовий ринок впливають незначні зміни в настроях, які тим не менш можуть мати значний вплив на ціни акцій. Завдяки тонкому розумінню мови *BERT* може виявляти навіть ці тонкі зміни. Цей рівень чутливості має вирішальне значення для моделювання впливу новин і медіаконтенту на динаміку цін акцій.

По-третє, універсальність і адаптивність *BERT* означають, що його можна постійно налаштовувати та оновлювати новими даними. Оскільки фінансовий ринок розвивається та з’являються нові види текстової інформації, *BERT* може адаптуватися до цих змін, зберігаючи свою точність і актуальність.

До того ж, наявними та доступними для використання є попередньо підготовлені доменно-спеціалізовані *BERT* моделі. Використання таких надає потужну основу для різноманітних завдань аналізу тексту, включаючи аналіз настроїв, у різних предметних областях. Особливо для спеціалізованих доменів, таких як фінанси, де мова та контекст можуть суттєво відрізнятися від загальної мови, попередньо навчені моделі для конкретного домену можуть запропонувати покращену продуктивність.

Попередньо підготовлені моделі *BERT* включають і фінансові, які навчаються на фінансових текстах. Ці моделі навчені на наборах даних, що містять фінансові новини, звіти та іншу відповідну фінансову літературу. Це спеціалізоване навчання дозволяє їм краще зрозуміти контекст, жаргон і нюанси,

характерні для фінансових текстів. Піддаючись мовним шаблонам і термінології, специфічній для фінансів, ці моделі більш вправні в точному тлумаченні та аналізі фінансових текстів.

Можливість використання попередньо натренованих на даних з фінансового домену моделей *BERT* та доступність їх функціоналу обрахунку показників тональності тексту «з коробки» також є суттєвою перевагою, оскільки забезпечує легкість впровадження її в розроблюваний прогнозний алгоритм та дозволяє зекономити ресурси системи та час на додаткове навчання мовної моделі.

2.7. Інтеграція

Наступним аспектом проектування є вибір методу інтеграції розроблюваного програмного модуля з іншими програмними продуктами управління інвестиціями. Програмні продукти даної спеціалізації можуть мати різні архітектури та використовувати різні програмні та технічні засоби у своїй реалізації. Тому важливим є надання уніфікованого інтерфейсу інтеграції розроблюваного модуля, який би був універсальним та простим до впровадження не залежно від архітектурних та технічних нюансів реалізації кожного окремого програмного продукту.

Підходи до вирішення цієї задачі включають: надання *API* інтерфейсу, реалізація на базі служб обміну повідомлення та підхід з використанням *Azure ServiceBus*. Кожен з цих підходів має свої переваги та недоліки.

API підхід полягає у встановленні прямого інтерактивного каналу взаємодії між модулем прогнозування та програмною системою. *API* служить набором правил і протоколів для взаємодії різних компонентів програмного забезпечення. Концептуально в контексті даної задачі *API* діятиме як міст, дозволяючи системі управління інвестиціями надсилати дані до програмного модуля, як-от ідентифікатори компаній та відрізок часу для прогнозування у форматі *HTTP* запитів, і натомість отримувати прогнозовані ціни на акції на

заданий горизонт прогнозування для подальшого використання цих даних у бізнес-логіці програмної системи.

Основна перевага використання *API* для даної задачі полягає в його прямоті та швидкодії. *API* полегшують обмін даними в реальному часі, що робить їх особливо підходящими для використання у фінансовій предметній області, де інформація є чутливою до часу. Наприклад, менеджер портфеля, який використовує інвестиційне програмне забезпечення, може запросити прогноз ціни акцій, і розроблений програмний модуль надасть прогноз майже миттєво, бездоганно інтегрований у існуючий програмний інтерфейс. Такий миттєвий обмін даними через *API* стане значним плюсом для фінансових спеціалістів, які залежать від своєчасної та точної інформації для прийняття рішень.

Також перевагами *API* є їх універсальність, масштабованість, відносна простота імплементації та підтримки. *API* можуть бути розроблені для обробки різних типів запитів, від простих запитів даних до більш складних аналітичних процесів. Їх також можна масштабувати для обробки великої кількості запитів, що має велике значення, враховуючи потенційно великий обсяг запитів від різних користувачів програмного забезпечення управління інвестиціями.

Недоліком даного підходу є потреба в грамотній реалізації управління навантаженнням. Враховуючи специфіку роботи розроблюваного програмного модуля, якщо *API* отримуватиме дуже багато одночасних запитів, це може потенційно призвести до затримок або проблем із продуктивністю. Тому управління та оптимізація навантаження на сервер стає важливим аспектом при реалізації цього підходу.

Підхід на базі служб обміну повідомленнями полягає у використанні протоколів обміну повідомленнями та брокерів, таких як *RabbitMQ*, *NATS* або *Apache Kafka*. Ці служби діють як посередники, які обробляють передачу повідомлень – в даному випадку запитів на дані та відповідей – між програмним модулем прогнозування та програмною системою. Служби обміну повідомленнями пропонують окрему архітектуру, яка принципово відрізняється від *API*. Замість механізму прямого виклику та відповіді, як у випадку *API*,

служби обміну повідомленнями працюють за моделлю асинхронного зв’язку. Це означає, що коли програмна система надсилає запит даних до модуля, вона не чекає негайної відповіді. Натомість запит ставиться в чергу, і програмна система може продовжувати виконання інших завдань. Коли ж модуль обробляє запит і генерує прогноз, він надсилає повідомлення назад, яке потім отримує система.

Однією з головних переваг використання служб обміну повідомленнями є їхня здатність обробляти високі навантаження та складні завдання обробки асинхронно. Це корисно у випадках коли процес виконання запиту включає складні обчислення. Послуги обміну повідомленнями є кращими у сценаріях, коли час обробки є змінним або потенційно тривалим, оскільки вони дозволяють роз’єднати запит і відповідь, таким чином уникаючи затримок в роботі системи.

Істотним недоліком цього підходу є складність налаштування та підтримки інфраструктури обміну повідомленнями. Такі служби, як *RabbitMQ* або *Kafka*, вимагають ретельного налаштування та керування. Вони також створюють додаткові рівні складності з точки зору моніторингу черг повідомлень, забезпечення доставки повідомлень і обробки помилок або повторних спроб. Це є набагато більш ресурсомістким порівняно з *API* підходом.

З точки зору безпеки, хоча і *API*, і служби обміну повідомленнями вимагають впровадження надійних заходів безпеки, *API* підхід є більш простим для захисту через його більш прямий і менш розподілений характер. Служби обміну повідомленнями, що мають справу з ширшою мережею передачі повідомлень, можуть створити додаткові проблеми щодо забезпечення цілісності даних і безпеки.

Підхід на базі *Azure ServiceBus* відкриває унікальну перспективу, особливо в порівнянні з підходами *API* та загальних служб обміну повідомленнями, які були розглянуті вище. *Azure ServiceBus* – це хмарна система обміну повідомленнями, надана корпорацією Майкрософт, розроблена для полегшення складних шаблонів зв’язку, таких як публікація-підписка та обмін повідомленнями на основі черги, які можна використовувати для інтеграції різних програмних систем.

Головними перевагами цього підходу є його надійність і масштабованість, притаманні хмарним рішенням. *ServiceBus* відмінно справляється з великомасштабними сценаріями обміну повідомленнями, пропонуючи високу надійність і здатність керувати значним обсягом повідомлень без зниження продуктивності.

Однак, цьому підходу притаманний ряд суттєвих недоліків. Найважливішим з них є складність. *Azure ServiceBus*, будучи багатофункціональною та універсальною платформою, потребує глибокого розуміння своїх функцій і конфігурацій. Ця складність може призвести до значного збільшення часу розробки та інтеграції порівняно з попередніми розглянутими підходами.

Крім того, *Azure ServiceBus* як хмарна служба вводить залежність від зовнішньої інфраструктури. Це означає, що будь-які простої або проблеми зі службами *Azure* можуть безпосередньо вплинути на доступність і надійність інтеграції модуля прогнозування з іншими програмними системами.

Ще один аспект, який слід враховувати, це висока вартість використання *Azure ServiceBus*. Ця служба працює за моделлю оплати за використання, що може бути економічно ефективним у менших масштабах, але може стати дорогим із збільшенням використання, особливо з великою кількістю повідомлень або передачі великих об’ємів даних. Для порівняння, *API* може мати більш передбачувану структуру витрат, особливо якщо він розміщений локально або в хмарній службі з фіксованою вартістю.

Таким чином, хоча *Azure ServiceBus* надає потужне, масштабоване та надійне рішення обміну повідомленнями, яке підходить для складних сценаріїв інтеграції, воно є надмірним та не оптимальним вибором для даного завдання.

Підводячи підсумок огляду та порівняльного аналізу підходів до реалізації інтеграції розроблюваного програмного модуля та зовнішніх програмних систем, можна зробити висновок, що найкращим вибором в даному випадку є вибір на користь *API* підходу. *API* підхід відрізняється серед інших розглянутих підходів своєю простотою в імплементації, легкістю впровадження, подальшої

підтримки, та універсальністю, що є особливо важливим в контексті поставленої задачі.

Крім цього, значною перевагою *API* підходу є можливість надавати функціонал розроблюваного модуля не лише для безпосередньої інтеграції та роботи його у складі інших програмних систем, а й як окремого повноцінного *Standalone API*. Цей аспект має значну цінність для цільових користувачів розроблюваного продукту. Фінансові спеціалісти, такі як інвестори, портфельні менеджери та фінансові аналітики, часто використовують у своїх робочих процесах не один програмний продукт, а набір різноманітних програмних інструментів, спеціальних сценаріїв та скриптів, розроблених під ті чи інші специфічні бізнес-потреби кожного користувача.

Надання розроблюваного модулю прогнозування в якості *Standalone API* є рішенням, яке пропонує високу гнучкість, можливість швидкого отримання даних в універсальному форматі для їх подальшої обробки програмними засобами. Ця гнучкість має велике значення у фінансовому секторі, де персоналізовані та специфічні потреби суттєво відрізняються від одного професіонала до іншого. Це все робить *API* підхід не лише підходящим вибором для поточного завдання, а й стратегічним, враховуючи розширене застосування та корисність розроблюваного модуля прогнозування в різноманітному середовищі фінансових інструментів і робочих процесів.

2.8. Технічна специфікація

Заключним етапом проектування є формування технічної специфікації програмного продукту на основі виділених етапів вирішення поставленої задачі, а також проведеного огляду та порівняльного аналізу методів та технологій реалізації компонентів розроблюваного програмного модуля.

Технічна специфікація:

– засіб реалізації автоматизованого відбору ознак – метод градієнтного бустингу;

– засіб реалізації компоненту прогнозування – нейронна мережа на базі *LSTM*;

– компонент обробки текстових даних – модель *BERT*;

– інтерфейс інтеграції – *API* (з наданням функціональності розроблюваного модуля у якості *Standalone API*);

– засоби оформлення та надання документації програмного продукту – *Redoc*, *Swagger*;

– СУБД – *SQLite*;

– хостинг – локальний *Application Server*;

– система контролю версій – *Git*.

2.9. Висновки до розділу

У даному розділі було проведено аналіз існуючих досліджень поставленної задачі та запропонованих в них методик, виділено їх обмеження та недоліки. На основі результатів цього аналізу, а також постановки задачі та функціональних вимог до розроблюваного рішення, сформованих в розділі 1 було розроблено власний підхід до вирішення цієї задачі, сформульовано його принципи.

Було виділено етапи вирішення поставленої задачі та аспекти, які є важливими для успішної реалізації запропонованого підходу. Було виконано відбір факторів та фінансових показників для включення у набір ознак для прогнозування, а також проведено огляд, порівняльний аналіз та вибір засобів реалізації різних аспектів та етапів розроблюваного рішення на основі наявних обмежень та специфіки предметної області.

Базуючись на результатах виконання описаних вище кроків було сформовано технічну специфікацію розроблюваного програмного продукту.

РОЗДІЛ 3
ПРОГРАМНА РЕАЛІЗАЦІЯ

На основі сформованих принципів запропонованого підходу, а також переліку аспектів та складових розроблюваного рішення, виділених та описаних на етапі проектування було проведено поділ розроблюваного модуля на логічні складові (компоненти). Вони включають:

1. *API* – інтерфейс з яким взаємодіє користувач напряму (у випадку роботи зі *Standalone API*) або з яким взаємодіють сторонні програмні системи управління інвестиціями (*IMS*), які інтегрують розроблюваний програмний модуль у свій функціонал, за допомогою *HTTP* запитів.

2. Компонент прогнозування – нейронна мережа на базі *LSTM*, яка виконує прогнозування та повертає результат на запити, які приходять з *API*.

3. Компонент вибору ознак – компонент який відповідає за автоматизований вибір ознак з набору факторів для прогнозування на базі методу градієнтного бустингу.

4. Компонент збору даних – компонент, який відповідає за збір даних для визначеного набору фінансових факторів та індикаторів.

5. Компонент обробки числових даних – компонент, який виконує обробку зібраних числових даних, необхідну для їх включення у датасет для прогнозування.

6. Компонент обробки текстових даних – компонент, який виконує обробку текстових даних на базі *NLP* моделі *BERT*, аналіз тональності тексту та формування з необроблених текстових даних часового ряду для включення до датасету ознак прогнозної моделі.

7. База даних, яка зберігає необхідні для роботи програмного модуля дані.

8. Компонент роботи з БД – компонент, який відповідає за взаємодію з базою даних розроблюваного програмного модуля, виконуючи операції запису та отримання необхідних для роботи модуля даних з БД.

9. Планувальник оновлень – компонент, який відповідає за регулярне оновлення даних прогнозної моделі. Ця операція виконуватиметься в кінці кожного робочого дня після закриття фондової біржі, та включатиме етапи збору даних, обробку числових та текстових даних, запис оновлених даних до БД, та реітерацію навчання моделі на оновлених даних.

Схему цих складових, їх взаємодії та залежності предстaлено на рис. 3.1.

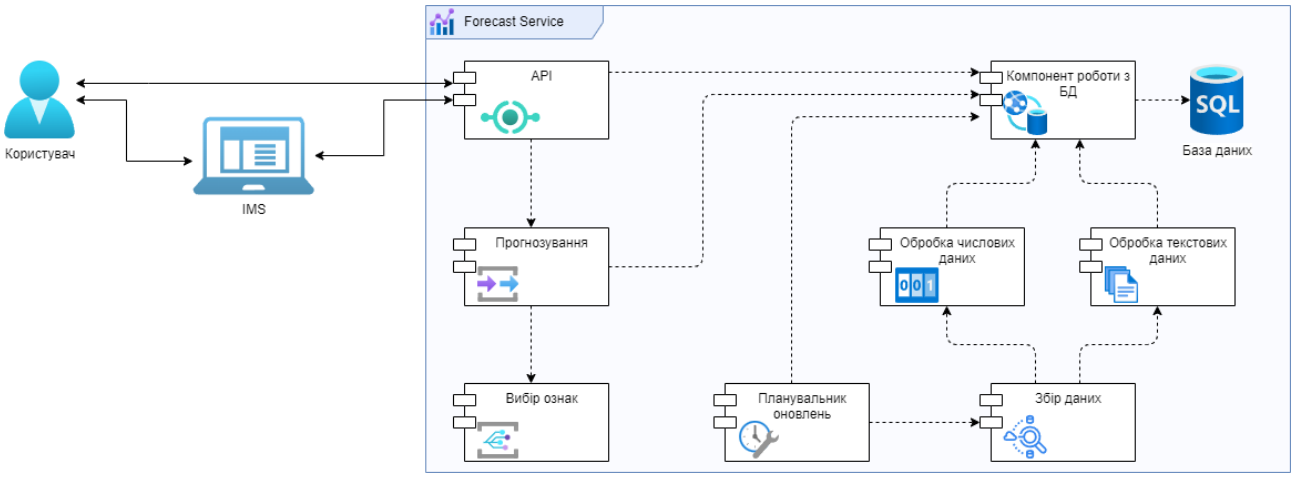

Рис. 3.1. Взаємодія компонентів програмного модуля

3.1. Реалізація механізмів роботи з даними

3.1.1. Збір даних

Історичні дані для вибраного набору факторів та фінансових показників для прогнозування є відкритими даними, доступними на відповідних фінансових порталах. Збір даних для формування датасету було виконано з використанням наступних ресурсів:

1. Фінансові *API*:

1) *AplhaVantage API* – з цього ресурсу було зібрані дані технічних індикаторів (*Volume, WMA50, WMA200, SMA50, SMA200, EMA50, EMA200, RSI7, RSI14* та *RSI21*) та макроекономічних показників (*Unemployment Rate, GDP, CPI* та *Interest Rate*);

2) *Polygon.io* – з цього ресурсу було зібрано текстові дані (новинні статті) про компанії для їх подальшої обробки, аналізу та включення до прогнозної моделі.

2. Фінансові веб-ресурси:

1) *Macrotrends* – з цього ресурсу було отримано фінансові звіти компаній та основні фундаментальні показники (*Net Income, RoE, P/E Ratio, P/B Ratio, Dividends, Stock Splits*);

2) *Finbox* – з цього ресурсу було зібрано решту фундаментальних показників (*Quick Ratio, Current Ratio, PEG Ratio, EPS, Dividend Yield*).

Цей етап було автоматизовано із застосуванням мови програмування *Python* та бібліотек "*requests*" для виконання *HTTP* запитів до зазначених вище ресурсів та бібліотеки "*beautifulsoup4*" для реалізації веб-скрапінгу даних з ресурсів які не мають *API* інтерфейсу (*Macrotrends* та *Finbox*). Лістинг коду реалізації збору даних програмними засобами на прикладі роботи з ресурсом *Macrotrends* привелено у додатку А.

3.1.2. Обробка числових даних

Фінансові дані для відібраних факторів мають різну дискретність. Дані про ціни акцій та технічні індикатори наявні на кожен робочий день. Фундаментальні дані, взяті із квартальної звітності компаній, наявні лише за квартальні проміжки часу. Макроекономічні показники мають періодичність у квартал, а деякі з них – у місяць. Тому необхідним етапом попередньої обробки є заповнення тих данних, яких не вистачає, адже для включення до моделі дані повинні мати однакову дискретність у часі. Для вирішення цієї задачі запропоновано

використати метод інтерполяції – поліноміальний другого порядку, оскільки він найефективніше враховує природу наявних даних у датасеті.

Також враховуючи природну «зашумленість» фінансових даних, та потенційний ризик виникнення перенавчання моделі до шумів у вхідних даних, наступним кроком попередньої обробки даних було виконано знешумлення даних за допомогою методу експоненційного згладжування (*Exponential Smoothing*). Експоненційне згладжування є широко використовуваним методом зменшення шуму в даних на етапі попередньої обробки. Доцільним буде використання цього методу для задачі прогнозування в запропонованій моделі, оскільки надає більші вагові коефіцієнти останнім спостереженням у наборі даних, враховуючи при цьому всі історичні дані, так як у випадку нашої задачі останні тенденції більше вказують на майбутні значення.

Застосування техніки експоненційного згладжування дозволить отримати більш плавний тренд, який відповідає вхідних даним, але при цьому зменшити наявні в них короткочасні коливання, як показано на рис. 3.2 на прикладі колонки *Close price* з вхідного датасету.

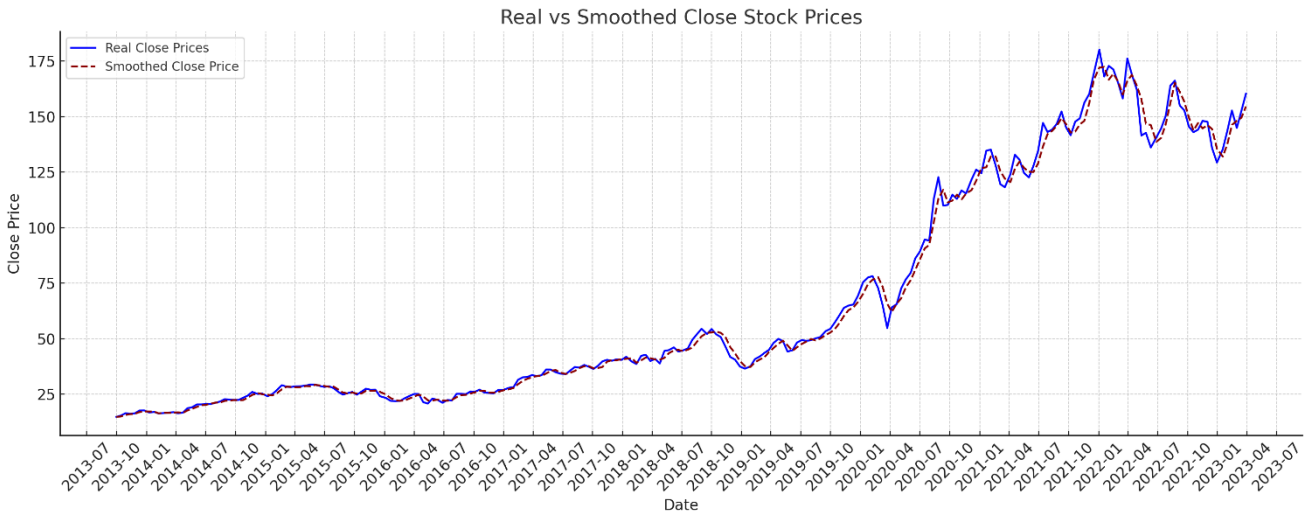

Рис. 3.2. Зменшення шуму у вхідних даних

3.1.3. Обробка текстових даних

Згідно відповідного етапу проектування, реалізація компоненту обробки текстових даних для їх подальшої інтеграції в прогнозну модель в якості додаткової ознаки (предиктора) було імплементовано на базі попередньо натренованої моделі *BERT – FinBERT*.

FinBERT є однією з наявних та доступних для використання попередньо підготовлених моделей *BERT* для обробки текстової інформації фінансового домену. Ця модель була розроблена спеціально для аналізу настрою фінансового тексту шляхом подальшого навчання оригінальної мовної моделі *BERT* (“*nlptown/bert-base-multilingual-uncased-sentiment*”) на великому фінансовому наборі текстових даних, таким чином будучи налаштованою для класифікації настрою та аналізу тональності фінансових текстів.

Алгоритм роботи компоненту обробки текстових даних виглядає наступним чином:

1. Першим кроком є імпорт необхідних бібліотек – *pandas*, *transformers* та *torch*, завантаження попередньо підготовленої моделі *FinBERT* та токенізатору:

```
import pandas as pd
from transformers import BertTokenizer, BertForSequenceClassification
import torch
from torch.nn.functional import softmax

model = BertForSequenceClassification.from_pretrained('yinwenpeng/FinBERT-
Pretrained')
tokenizer = BertTokenizer.from_pretrained('yinwenpeng/FinBERT-Pretrained')
```

2. Вхідні дані, які включають новини про компанію з різних фінансових інформаційних порталів, отримані з *API Polygon.io*, зчитуються з *json* файлу та конвертуються у тип даних *DataFrame*:

```
file_path = '/content/drive/My Drive/Colab Notebooks/Input
files/News_result_polygon_AAPL.json'
with open(file_path, 'r', encoding='utf-8') as file:
    data = json.load(file)
df = pd.DataFrame(data)
```

Вхідні дані являють собою колекцію новинних записів, кожен запис яких має наступну структуру:


```
    {
        "id": "RQkIFU0a6wkNUHPv6YLzPyaAx1HnIh57TEdis3a5KDg",
        "publisher": {
            "name": "Zacks Investment Research",
            "homepage_url": "https://www.zacks.com/",
            "logo_url": "https://s3.polygon.io/public/assets/news/logos/zacks.png",
            "favicon_url":
"https://s3.polygon.io/public/assets/news/favicons/zacks.ico"
        },
        "title": "Apple (AAPL) Rises Higher Than Market: Key Facts",
        "author": "Zacks Equity Research",
        "published_utc": "2023-10-09T21:45:19Z",
        "article_url": "https://www.zacks.com/stock/news/2162949/apple-aapl-rises-
higher-than-market-key-facts",
        "tickers": [
            "AAPL"
        ],
        "amp_url": "https://www.zacks.com/amp/stock/news/2162949/apple-aapl-rises-
higher-than-market-key-facts",
        "image_url": "https://staticx-
tuner.zacks.com/images/default_article_images/default29.jpg",
        "description": "In the most recent trading session, Apple (AAPL) closed at
$178.99, indicating a +0.85% shift from the previous trading day."
    },
```


Відповідно після конвертації результуючий *DataFrame* матиме такі самі колонки, як і поля у вхідному *json*.

3. Далі за допомогою моделі *FinBERT* у функції *predict_sentiment* розраховуються числові значення аналізу тональності тексту для значень полів *title* та *description* для кожної новинної статті з вхідного датасету, обчислюється результуюче значення тональності як середнє значень тональності для *title* та *description* та записується у нову колонку *DataFrame*:

```
def predict_sentiment(text):
    inputs = tokenizer(text, return_tensors="pt", truncation=True, padding=True,
max_length=512)
    with torch.no_grad():
        outputs = model(**inputs)
    probs = softmax(outputs.logits, dim=-1)
    sentiment = torch.argmax(probs, dim=-1).item() + 1
    return sentiment

# Calculate sentiment scores
df['title_sentiment'] = df['title'].apply(predict_sentiment)

print("Calculating sentiment scores: description_sentiment")
```

```
df['description_sentiment'] =
df['description'].fillna('').apply(predict_sentiment)
df['sentiment_score'] = df[['title_sentiment',
'description_sentiment']].mean(axis=1).astype(float)
```

4. Наступним кроком є приведення отриманого часового ряду значень тональності тексту до дискретності, яка відповідає решті датасету факторів, тобто здійснення агрегації отриманих величин для заданої одиниці часу, в даному випадку – один робочий день (якщо наявні декілька новинних записів за один день, тоді значення тональності усереднюються, якщо ж немає за день жодного запису – береться останнє відоме значення; значення тональності статей за неробочі дні присвоюються до найближчого наступного робочого дня).

3.1.4. Структура датасету

На основі наведеного вище переліку факторів, які мають вплив на ціни акцій було сформовано набір даних для використання в прогнозній моделі. Цей датасет представлений на рис. 3.3. Він включає відповідно історичні дані періодом 10 років (з кінця вересня 2013 до початку жовтня 2023 року) про саму ціну акцій (*Open*, *High*, *Low*, *Closed*), обсяги торгів (*Volume*), ряд технічних індикаторів, фундаментальних та макроекономічних показників, для яких аналітично було виявлено найбільш сильні залежності з ціною акцій. Більшість цих даних було зібрано з відкритих джерел за допомогою модуля автоматизованого веб-скрапінгу, решту – обраховано програмно згідно формул розрахунку значень показників наведених вище.

Крім цього датасет включає результати обробки текстової інформації (новин про компанію) – зведений показник оцінки тональності тексту для новин на кожний робочий день, обрахований за допомогою заздалегідь натренованої на фінансових текстових даних *BERT* моделі – *FinBERT*.

	Date	2013-09-30	2013-10-01	2013-10-02	2013-10-03	2013-10-04
Технічні індикатори	Open	14.876	14.913	15.137	15.289	15.082
	High	15.013	15.246	15.329	15.346	15.105
	Low	14.787	14.911	15.078	14.985	14.918
	Close	14.86	15.21	15.259	15.068	15.056
	Volume	260.156M	353.884M	289.184M	322.753M	258.868M
	WMA50	15.009	15.024	15.041	15.048	15.053
	WMA200	14.052	14.063	14.074	14.084	14.093
	SMA50	14.805	14.845	14.891	14.92	14.949
	SMA200	14.12	14.115	14.114	14.11	14.103
	EMA50	14.709	14.728	14.749	14.762	14.773
	EMA200	14.625	14.631	14.637	14.642	14.646
	RSI7	46.91	57.728	59.116	51.532	51.06
	RSI14	48.999	54.397	55.127	51.701	51.488
	RSI21	50.713	54.402	54.908	52.555	52.409
Макроекономочні показники	Unemployment	7.205	7.2	7.195	7.189	7.183
	GDP	4587.009	4586.321	4585.575	4584.771	4583.91
	CPI	233.567	233.546	233.525	233.505	233.484
	Interest Rate	0.09	0.09	0.09	0.09	0.09
Фундаментальні показники	Quick Ratio	1.64	1.637	1.635	1.632	1.63
	Current Ratio	1.68	1.677	1.675	1.672	1.669
	P/E Ratio	10.5	10.525	10.55	10.575	10.599
	P/B Ratio	3.03	3.036	3.041	3.047	3.052
	PEG Ratio	-1.993	-2.046	-2.097	-2.146	-2.195
	EPS	0.3	0.302	0.304	0.305	0.307
	RoE	29.06	29.054	29.048	29.042	29.036
	Net Income	7.740B	7.853B	7.964B	8.074B	8.183B
	Dividend Yield	0	0	0	0	0
	Dividends	0	0	0	0	0
	Stock Splits	0	0	0	0	0
Текстові дані	Sentiment score	2.5	2	2	1.5	2.5

Рис. 3.3. Структура набору даних для використання в прогнозній моделі.

3.1.5. Структура бази даних

Для створення схеми та сутностей бази даних розроблюваного програмного модуля та подальшої роботи ними було використано підхід *Code First*, суттю якого є генерація схеми бази даних на основі попередньо написаного програмного коду класів моделей сутностей програмного модуля. В подальшому для внесення змін в схему БД достатнім є змінити модель (програмний клас) який відповідає тій чи іншій сутності шляхом додавання, видалення або модифікації полів цього класу, які власне і стануть в результаті полями таблиць бази даних.

Попередньо було виділено сутності, які необхідні для роботи розроблюваного програмного модуля, а саме:

– компанії (*AvailableTickers*) – сутність (таблиця бази даних), яка містить поля ідентифікатор (*TickerId*) та повну назву компанії (*FullName*);

– показники компаній (*CompanyIndicators*) – сутність (таблиця бази даних), яка включає ідентифікатор компанії, а також технічні та фундаментальні показники для кожної компанії з визначеного раніше набору (поля: назва показника, ідентифікатор компанії, дата, значення показника), а також показник *sentiment score* отриманий в результаті виконання аналізу тональності модулем обробки текстових даних;

– макроекономічні фактори (*MacroeconomicFactors*) – сутність (таблиця), яка містить назву фактору (показника), дату та його значення;

– моделі прогнозування (*ForecastModels*) – сутність (таблиця), яка містить збережену на етапі навчання прогнозну модель (файл з конфігурацією та ваговими коефіцієнтами моделі) для кожної компанії з переліку доступних для прогнозування. На етапі навчання для кожної компанії конфігурація та вагові коефіцієнти моделі зберігаються у вигляді відповідних *.json* та .h5 файлів. У даній таблиці зберігається відповідно ідентифікатор компанії (*TickerId*) та шлях до файлу (*FilePath*). Коли приходить запит на виконання прогнозування, логіка програмного модуля завантажує відповідний файл з конфігурацією моделі та використовує цю модель для виконання прогнозування;

– прогнози (*Forecasts*) – сутність (таблиця), яка зберігає результати виконаних запитів на прогнозування. Дані цієї таблиці використовуються для кешування результатів запитів. Використання цих даних відбувається у механізмі кешування, який є мірою оптимізації використання ресурсів. Наприклад, коли користувач протягом дня робить повторний запит на прогнозування, такий як виконував вже раніше, то в такому випадку немає необхідності робити прогноз за допомогою моделі заново і дані для відповіді на такий повторний запит беруться вже з закешованих результатів. Цей механізм існує для того, щоб пришвидшити роботу програмного модуля в таких випадках та оптимізувати використання ресурсів;

– черга компаній (*TickersQueue*) – таблиця, яка зберігає чергу (реєстр) ідентифікаторів нових компаній на додавання до переліку доступних для прогнозування з часом. Під час кожного наступного оновлення даних моделі

компанії з цієї черги переносяться в основний реєстр і вони відповідно включаються в збір даних та побудови моделей, які будуть використовуватись для прогнозування.

ER діаграма описаних сутностей та зв’язків між ними наведена на рис. 3.4.

Процес створення *Code First* бази даних та подальшої роботи з нею на *Python* передбачає використання *ORM SQLAlchemy*:

1. Визначення моделей (класів *Python*) – для визначеного переліку сутностей створюються *Python* класи, які відповідають таблицям в базі даних, а також програмно задаються зв'язки між ними.

2. Створення бази даних – створення об'єктів *engine* та *session* (функціями бібліотеки *SQLAlchemy create_engine* та *sessionmaker* відповідно) для створення та взаємодії з базою даних з використанням СУБД *SQLite*. *ORM SQLAlchemy* перетворює попередньо визначену програмними класами схему БД в *SQL* для створення безпосередньо самої бази даних.

3. Використання сесії (об’єкту *Session*) для додавання та отримання даних зі створеної бази даних.

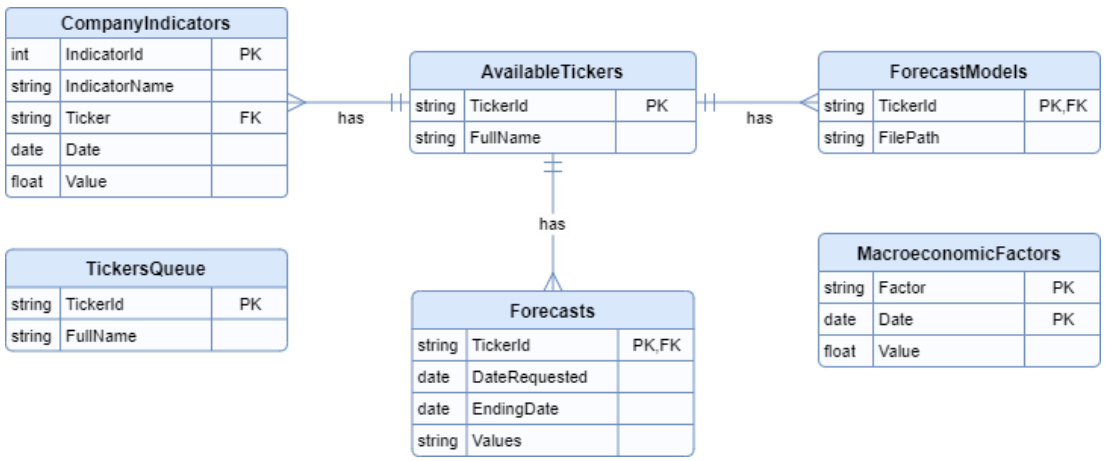

Рис. 3.4. *ER* діаграма сутностей

3.2. Реалізація відбору ознак

Наступним кроком відбору факторів (ознак) для включення до розроблюваної моделі прогнозування є застосування методу градієнтного бустингу для обрахунку значимості ознак та виділення найбільш вдалих ознак для включення до моделі зі сформованого аналітично датасету.

В даній реалізації було використано бібліотеку *XGBoost* для імплементації методу градієнтного бустингу для відбору предикторів для розроблюваної моделі. Лістинг коду імплементації цього кроку приведено в додатку Б. Отримані показники відносної важливості ознак для підготованого датасету приведено на рис. 3.5.

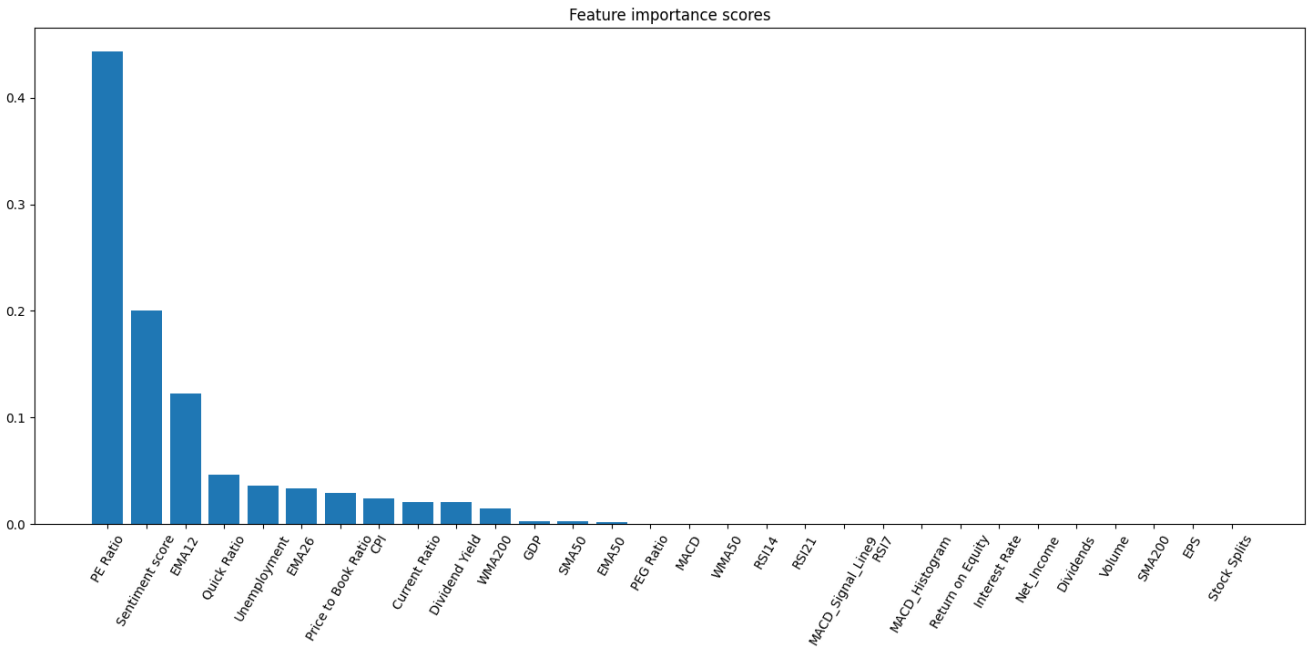

Рис. 3.5. Показники важливості ознак

На основі отриманих значень було відібрано 10 ознак з найбільшими значеннями відносного показника важливості для включення до прогнозної моделі.

3.3. Реалізація компоненту прогнозування

Після етапу автоматизованого відбору ознак на основі обчислених для них покаників важливості, відібраний набір з 10 ознак з найбільшими значеннями показників важливості передаються на вхід компоненту прогнозування. Компонент прогнозування включає в себе безпосередньо прогнозну модель на базі *LSTM* та «обгортку» над нею, яка реалізує логіку формування та перетворень структур даних перед передачею їх на вхід моделі, а також збереження та передачу отриманих з моделі результатів до інших компонентів програмного модуля.

Алгоритм роботи цього компоненту полягає у виконанні ітеративного прогнозування ціни акцій для заданої в запиті компанії на заданий горизонт прогнозування (відрізок часу), та включає наступні кроки:

1. Отримання вхідних даних – даних набору відібраних на попередньому етапі ознак (предикторів) до останнього відомого моменту часу (t).

2. Прогнозування значення ціни на наступний часовий крок (t+1) на базі набору значень предикторів. Його довжина – це значення відповідного параметру моделі, яке задає скільки останніх точок даних використовуватимуться для прогнозування наступного значення цільової величини. Після здійснення прогнозування відбувається накопичення результатів шляхом додавання результату у буфер.

3. Для всіх вхідних факторів з датасету, які не включають значення ціни у свою формулу розрахунку, виконується прогнозування значення на наступний часовий крок (t+1). В свою чергу для факторів, формула яких включає ціну, значення для часового кроку t+1 обчислюється за відповідною формулою з використанням спрогнозовованого на попередньому кроці значення ціни.

4. Отримані значення ціни та факторів для кроку часу t+1 додаються в кінець датасету, а часове вікно зсувається на один крок часу вперед, таким чином виконуючи зсув датасету.

5. Оновлений датасет подається на вхід моделі для прогнозування значення вже для наступного часового кроку (t+2).

6. Описані вище кроки повторюються ітеративно до тих пір, поки не досягнуто кінця заданого горизонту прогнозування.

Схема описаного вище алгоритму наведена на рис. 3.6.

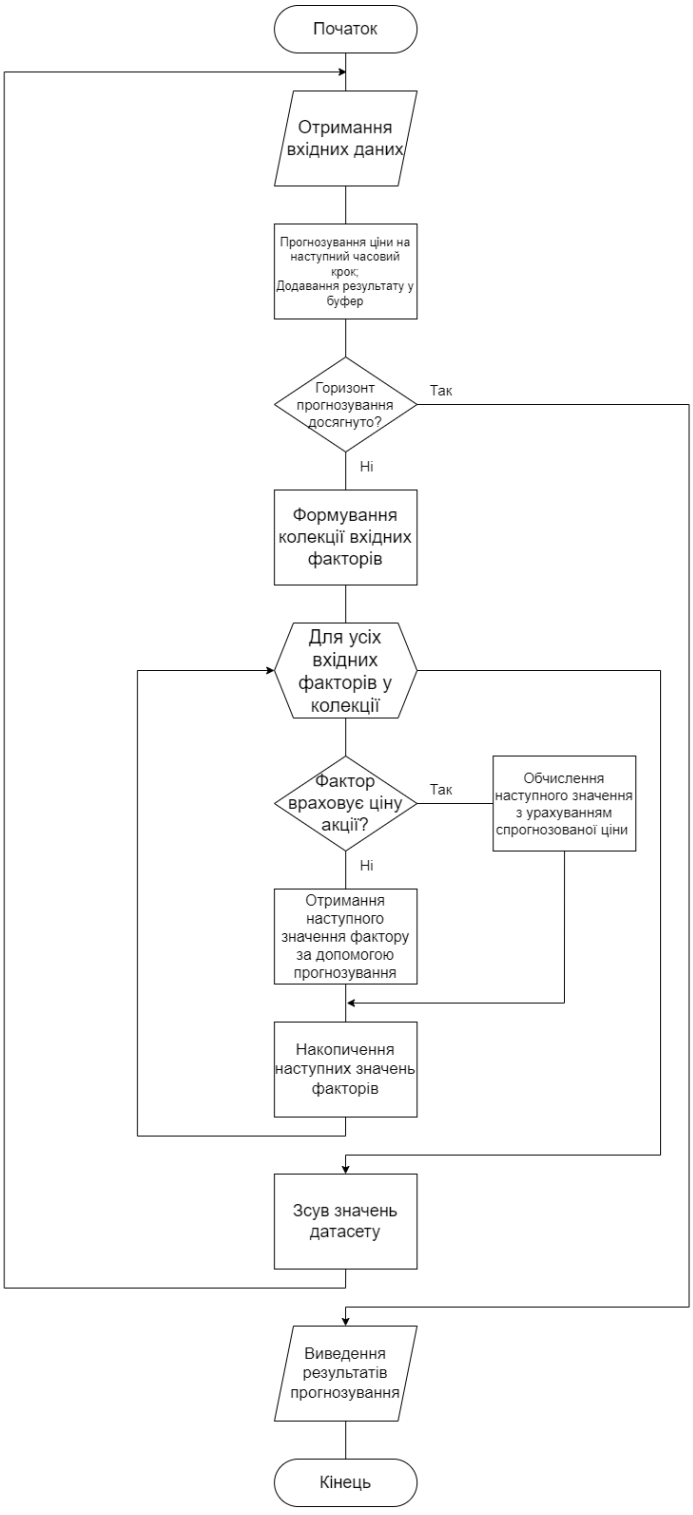

Рис. 3.6. Схема алгоритму компоненту прогнозування

3.3.1. Навчання моделі

Один з ключових аспектів, що впливає на точність результатів прогнозування, є правильний підбір функції активації та втрат, а також оптимізатора, так як саме ці параметри безпосередньо впливають на формування результуючого значення та швидкість навчання моделі.

Функція активації є математичним перетворенням між входом, що подається на нейрон, та його виходом, що йде на наступний шар. Ця функція визначає, наскільки сильно активується нейрон, відповідно до вхідного сигналу [25]. Вона додає нелінійності до моделі, що є необхідним для навчання складних шаблонів.

Для даної предметної області доречним вибором функції активації є гіперболічний тангенс (*tanh*), що виводить значення в діапазоні від -1 до 1. Він центрований в нулі, що робить навчання ефективнішим, оскільки середнє значення виходів шарів наближається до нуля, сприяючи стабільнішому градієнтному спуску. Перевагою гіперболічного тангенсу є його ефективність у мережах, де потрібно, щоб вихідні значення могли мати знак, що є особливо корисним у рекурентних нейронних мережах, які працюють із послідовностями даних.

Функція втрат визначає, наскільки точно модель працює під час навчання. Для задачі прогнозування ціни акції найкращим чином проявила себе середньоквадратична помилка (*Mean Squared Error, MSE*). *MSE* є мірою середнього квадрату відхилень прогнозованих значень від реальних. Оскільки *MSE* підносить помилки до квадрату, великі помилки «штрафуються» надмірно, що спонукає модель точніше прогнозувати значення.

При навчанні модель калібрує вагові коефіцієнти, маючи на меті зниження значення функції втрат. Спосіб безпосередньо калібрування визначається оптимізатором – алгоритмом, що використовується для зміни вагових коефіцієнтів та підвищення швидкості навчання. Для поставленої задачі було обрано оптимізатор *Adam* (*Adaptive Moment Estimation*), який поєднує ідеї двох

інших популярних методів оптимізації: *RMSprop* і *Stochastic Gradient Descent with Momentum*. *Adam* адаптує швидкість навчання для кожного параметра моделі індивідуально, використовуючи оцінки перших та других моментів градієнтів. За рахунок своїх переваг, а саме адаптивності, ефективності на різноманітних даних, а також швидкості збіжності, у контексті прогнозування цін акцій, де модель може зіткнутися з даними, що сильно коливаються, *Adam* може допомогти моделі швидше адаптуватися та зменшити помилки прогнозування.

Важливим явищем, що може значно повпливати на точність та гнучкість роботи моделі в негативну сторону, є перенавчання (*overfitting*), що наступає, коли модель надто точно відтворює навчальний набір даних, втрачаючи при цьому здатність до узагальнення на нових даних. Це означає, що модель "запам'ятовує" навчальні дані, включаючи шум та випадковості, замість того, щоб вивчити загальні закономірності. Результатом перенавчання є висока точність на навчальному наборі даних, але погана продуктивність на тестовому наборі або на нових даних, що не брали участь у тренуванні.

Одним із способів недопущення перенавчання, задіяних у моделі, стало додавання шарів відкидання (*dropout*)**,** що є реалізацією техніки регуляризації. У кожній ітерації навчання, визначений відсоток нейронів ігнорується, що запобігає їхній активації та впливу на вихід моделі. Така техніка ефективно симулює тренування великої кількості мереж та їхнє усереднення, що покращує загальну стійкість моделі до перенавчання. Таким чином, відкидання допомагає моделі підвищити рівень робастності, змушуючи її не покладатися на будь-який конкретний набір нейронів, а навчатися більш рівномірно по всій мережі.

Результати навчання моделі з обраними параметрами представлені на рис. 3.7 в розрізі зміни функції втрат протягом етапу навчання з залученням окремого тестувального набору даних.

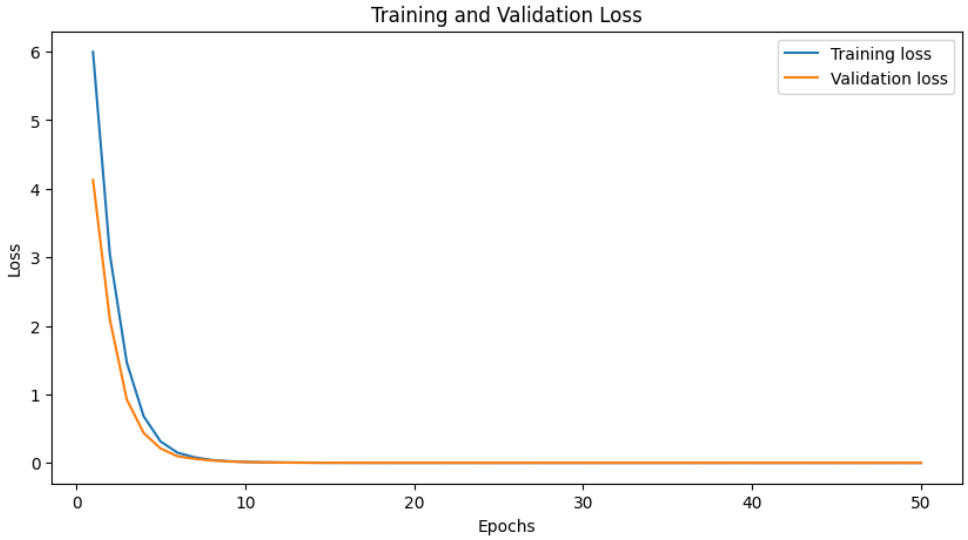

Рис. 3.7. Зміна значення функції втрат протягом навчання моделі

3.3.2. Архітектура та параметри розробленої моделі

Побудована модель визначається та характеризується параметрами, приведеними в табл. 3.1.

Таблиця 3.1

Параметри розробленої моделі

Параметри	Значення
Кількість входів	10
Кількість *LSTM* шарів	5
Кількість комірок *LSTM* шару	120
Функція активації *LSTM* шару	*tanh*
Крок часу (часове вікно)	120
Розмір пакета	64
Оптимізатор	*Adam*
Функція втрат	Середня квадратична помилка
Рівень відкидання	0.3
Епохи	50

1. Кількість входів (*Number of inputs*) – це кількість вхідних змінних, які модель використовує для прогнозування. Наприклад, це можуть бути різні технічні показники на фондовому ринку.

2. Кількість комірок *LSTM* шару (*Number of units in LSTM layer*) – кількість нейронів в кожному з *LSTM* шарів. Більше нейронів може забезпечити більшу здатність моделі до навчання, але також збільшує ризик перенавчання.

3. Функція активації *LSTM* шару (*LSTM layer activation function*) – функція, яка застосовується до обчислення виходів нейронів *LSTM* шару;

4. Крок часу (*Time step*) – кількість часових точок, які модель використовує для прогнозування наступного значення.

5. Розмір пакета (*Batch size*) – кількість зразків даних, які обробляються за один крок навчання.

6. Оптимізатор (*Optimizer*) – алгоритм, який використовується для оновлення ваг моделі під час навчання.

7. Функція втрат (*Loss function*) – критерій, за яким модель оцінює помилки прогнозування та прагне мінімізувати.

8. Рівень відкидання (*Dropout rate*) – відсоток нейронів, які випадково ігноруються під час навчання, щоб запобігти перенавчанню.

9. Епохи (*Epochs*) – кількість повних проходів навчального набору даних, які виконуються під час навчання моделі.

Відповідно до приведених параметрів та архітектури моделі, її робота відбувається наступним чином. Вхідний набір навчальних даних є тривимірним вектором розміром (*None x* 120 *x* 10), де 120 є підібраним кроком часу, а 10 представляє кількість вхідних параметрів (ознак). Спершу ці дані поступають на вхідний шар, з якого вони поступово передаються на задану кількість шарів *LSTM*, за кожним з яких слідує шар відкидання (*Dropout layer*). Обидва види шарів формують вихідні значення у вигляді вектору розміром (*None x* 120 *x* 120), і результуючий набір даних потрапляє на повнозв’язний шар (*Dense layer*), який і формує скалярне значення, що є результатом роботи моделі.

Лістинг програмного коду імплементації прогнозної моделі наведено в додатку В.

3.4. Аналіз отриманих результатів

Для проведення оцінки ефективності роботи розробленої моделі необхідним етапом є вибір базової моделі для порівняння результатів. Ціни акцій по своїй природі близько слідують гіпотезі про «випадкове блукання» (*Random walk hypothesis*), що саме по собі означає що зміни цієї величини є близькими за своїм характером до випадкових. Як відомо, для таких величин найкращим прогнозуванням є так зване «наївне прогнозування» (*Naive forecast*), суттю якого є копіювання останнього відомого значення прогнозованої величини на весь часовий проміжок прогнозування. Тому оцінку результатів роботи розробленої моделі було вирішено проводити у порівнянні з моделлю наївного прогнозування.

Для оцінки якості роботи розробленої моделі та базової (моделі наївного прогнозування) було вибрано наступні метрики: *MSE, RMSE, MAPE* та R^2. *MSE* було вирішено включити до набору обрахованих метрик для аналізу оскільки ця метрика є тою, що мінімізуєтьсян а етапі навчання нашої моделі. Вибір *RMSE* для включення до метрик обумовлений тим, що метрики *MSE* та *RMSE* відображають якість роботи моделі різним способом: *MSE* дає приблизне уявлення про величину помилки, а *RMSE* є квадратним коренем із *MSE*. Цей квадратний корінь є значущим, оскільки він означає, що *RMSE* має той самий масштаб, що й вихідні дані, що робить його більш зручним для інтерпретації, так як відображає помилку в тих самих одиницях, що й вихідна змінна. Крім цього, застосування у оцінці якості моделі комбінації метрик *MSE* і *RMSE* обумовлене тим, що оскільки помилки зводяться в квадрат перед усередненням, *RMSE* надає відносно високу вагу великим помилкам, що означає, що *RMSE* є чутливим до викидів.

Однак, оскільки помилки зводяться в квадрат перед усередненням, як *MSE*, так і *RMSE* в цілому більш чутливі до викидів, ніж інші показники, такі як середня абсолютна похибка. Тому для оцінки якості розробленої моделі було вирішено також включити метрики *MAPE* та R^2.

Після проведення навчання моделі, їй на вхід були передані тестувального набору, що включає дані за 30 робочих днів, і отримано прогнозоване значення ціни закриття для кожної з поданих дат. Результати прогнозування разом з базовим варіаном для порівняння (наївним прогнозом) на прикладі компанії *Apple* (*AAPL*) приведені на рис. 3.8.

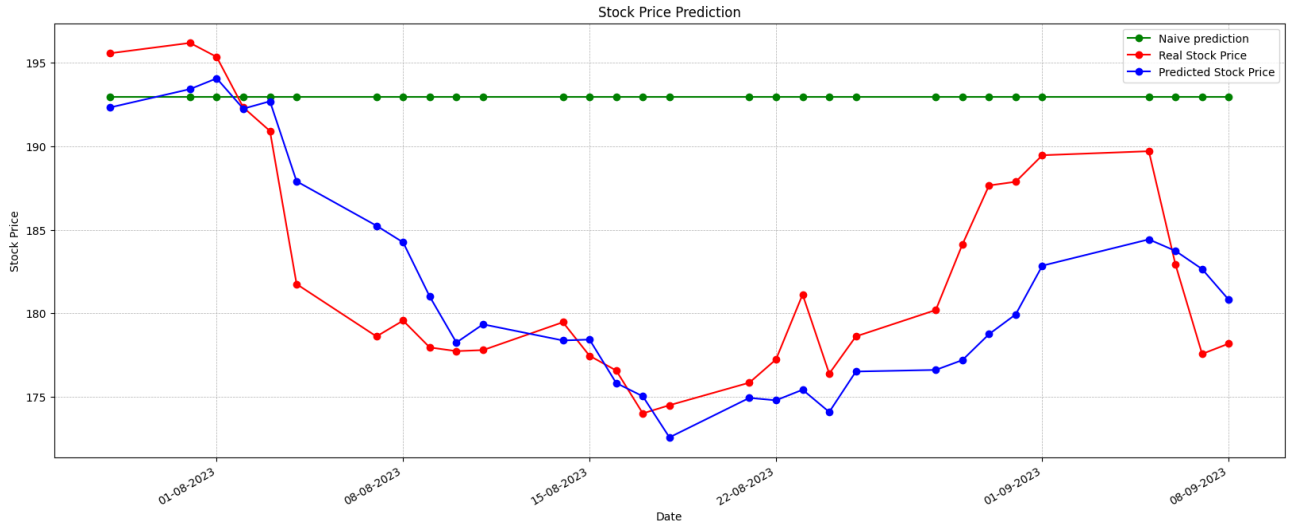

Рис. 3.8. Результати прогнозування моделі для компанії *Apple* (*AAPL*)

На основі отриманих значень, були вирахувані попередньо обрані метрики для прогнозу моделі та базового сценарію (табл. 3.2).

Таблиця 3.2

Обчилені метрики прогнозування

Метрика	Розроблена модель	Наївний прогноз
R^2	0,714	-2,527
MAPE	0,015	0,062
MSE	12,584	155,143

Продовження таблиці 3.2

Метрика	Розроблена модель	Наївний прогноз
RMSE	3,547	12,456

Як видно з приведених вище результатів, розроблена модель з використанням *LSTM* змогла перевершити базову модель прогнозування за усіма обраними метриками, довівши свою високу ефективність у порівнянні з наївним прогнозом. Розробленій моделі вдалось досягти близькості із дійсними даними, а також виділяти тренди і передбачати напрямок зміни ціни.

3.5. Реалізація інтеграції

Робочий процес розробленого програмного модулю характеризується набором станів, представлених на відповідній діаграмі станів (рис. 3.9).

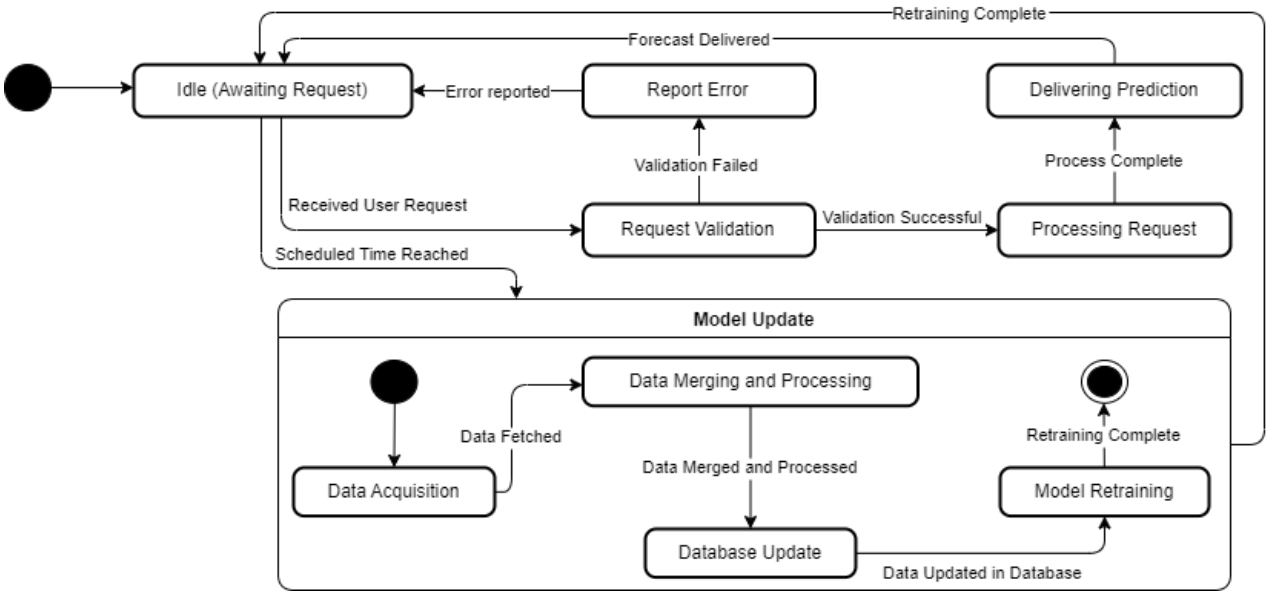

Рис. 3.9. Діаграма станів розробленого модулю

Відповідно, за замовчуванням програмний модуль перебуває у стані простою (*Idle*) очікуючи надходження запитів на прогнозування. Запити на прогнозуування можуть прийти від користувача напряму (за допомогою будь-

якого інструменту, який робить *HTTP* запити), або від будь-якої програмної системи управління інвестиціями, в яку програмний модуль інтегровано.

При отриманні такого запиту спочатку відбувається валідація запиту, яка включає перевірку авторизації та перевірку коректності значень заданих параметрів запиту. Якщо валідація не пройшла (запит або передані параметри запиту мають некоректні значення), програмним модулем повертається відповідь з помилкою. Якщо валідація запиту виконана успішно, модуль переходить у стан обробки запиту. На цьому етапі параметри запиту передаються вже до компоненту прогнозування та оброблюються його логікою. Після завершення обробки запиту компонент прогнозування повертає результат роботи прогнозної моделі, тобто програмний модуль переходить у стан відправки відповіді на запит з результатом прогнозування у відповідності з параметрами, які були передані у запиті.

Крім цього наявний механізм регулярного оновлення даних, який реалізує компонент «Планувальник оновлень». Планувальник оновлень має таймер з заданим значення часу для запуску процесу оновлення даних моделі. Коли цей таймер спрацьовує модуль переходить у стан оновлення даних моделі, цей процес відображений на діаграмі композитним станом «Оновлення моделі» (*Model Update*). Цей процес включає стани збору даних (*Data Aquisition*), на якому відпрацьовує компонент збору даних, який збирає з фінансових порталів дані для набору використовуваних факторів, які були оновлені (додані) за день. Далі виконується обробка зібраних даних та формування з них об’єднаного датасету (стан “*Data Merging and Processing*”). Наступним кроком після виконання обробки та об’єднання даних є оновлення бази даних програмного модуля, модуль переходить у стан «Оновлення бази даних» (“*Database Update*”). Заключним етапом цього композитного стану є виконання реітерації навчання прогнозної моделі на вже оновлених даних.

3.5.1. API

Згідно з обраним на етапі проектування методом реалізації інтерфейсу для взаємодії з програмним модулем (*API* інтерфейс для інтеграції зі сторонніми програмними системами та окремий *Standalone API*) було створено компонент, який реалізує ініціалізацію точок входу *API* (*API endpoints*), їх документацію з описом параметрів та моделей за допомогою *Swagger* та інкапсулює у логіку точок входу *API* виклик функцій прогнозування, реалізованих у компоненті прогнозування.

Для імплементації *API* компоненту було використано фреймворк *FastApi*, який надає функціонал для реалізації *Web API* на мові програмування *Python*. Хостинг розробленого *API* виконано на локальному сервері (*localhost*). Програмний код реалізації цього компоненту виглядає наступним чином:

```
from fastapi import FastAPI
from fastapi.openapi.utils import get_openapi
from fastapi.staticfiles import StaticFiles
from pydantic import BaseModel, Field

from StockForecastingComponent import get_forecast

title = "Stock Price Forecast API"
description = "API for forecasting stock prices using an LSTM model. Provides
endpoints for single and multiple stock predictions."
version="1.0.0"

app = FastAPI(
    title=title,
    description=description,
    version=version
)

# Serve static files from the 'static' directory
app.mount("/static", StaticFiles(directory="static"), name="static")

def custom_openapi():
    if app.openapi_schema:
        return app.openapi_schema

    openapi_schema = get_openapi(title=title, description=description,
version=version, routes=app.routes)
    openapi_schema["info"]["x-logo"] = {
        "url": "http://127.0.0.1:8000/static/logo.png"
    }
    app.openapi_schema = openapi_schema
    return app.openapi_schema


app.openapi = custom_openapi
```

```
class ForecastRequest(BaseModel):
    ticker: str = Field(..., description="The stock ticker symbol for which the
forecast is required.")
    days: int = Field(..., description="The number of days into the future for
which the stock price should be forecasted.")

class ForecastResponse(BaseModel):
    Ticker: str
    Forecast: List[dict]

class ForecastMultipleRequest(BaseModel):
    tickers: List[str] = Field(..., description="A list of stock ticker symbols
for which the forecasts are required.")
    days: int = Field(..., description="The number of days into the future for
which the stock prices should be forecasted for all provided tickers.")

class ForecastMultipleResponse(BaseModel):
    Forecasts: List[ForecastResponse]

@app.post("/forecast/", response_model=ForecastResponse,
response_description="Single stock price forecast", summary="Forecast Single
Stock")
async def forecast(request: ForecastRequest):
    """
    Forecast Stock Price for a Single Ticker

    This endpoint forecasts the stock price for a given ticker symbol over a
specified number of days.
    The response includes the ticker symbol and a list of forecasted values with
their corresponding dates.
    """
    forecast_data = get_forecast(request.ticker, request.days)
    return {"Ticker": request.ticker, "Forecast": forecast_data}

@app.post("/forecast/multiple/", response_model=ForecastMultipleResponse,
response_description="Multiple stock prices forecast", summary="Forecast
Multiple Stocks")
async def forecast_multiple(request: ForecastMultipleRequest):
    """
    Forecast Stock Prices for Multiple Tickers

    This endpoint forecasts the stock prices for multiple ticker symbols over a
specified number of days.
    The response includes a list of objects, each containing a ticker symbol and
a list of forecasted values with their corresponding dates.
    """
    forecasts = [{"Ticker": ticker, "Forecast": get_forecast(ticker,
request.days)} for ticker in request.tickers]
    return {"Forecasts": forecasts}
```

У приведеному лістингу коду відбувається створення моделей запитів та відповідей (*ForecasttRequest*, *ForecastResponse*, *ForecastMultipleRequest* та *ForecastMultipleResponse*) які визначають структуру даних відповідних запитів, які приймає *API*, та структуру даних відповідей, які повертаються. Описи, які відображаються в документації *API* для моделей та атрибутів задаються параметрами *description*.

Реєстрація точок входу *API* виконується відповідними асинхронними функціями *async def forecast* (*request*: *ForecastRequest*) та *async def forecast_multiple*(*request*: *ForecastMultipleRequest*), які помічені декораторами *@app.post*. У *FastAPI* декоратори використовуються для реєстрації функцій як обробників кінцевих точок для різних методів *HTTP*. В даному випадку використання декоратора *@app.post*('/*path'*) над функцією повідомляє *FastAPI* викликати цю функцію щоразу, коли *API* отримує запит *POST* до '/*path'*. У цій функції виконується обробка даних, надісланих разом із запитом *POST* у тілі запиту, і повертається відповідь.

Схема розробленого *API* включає дві точки входу:

1) *Forecast Single Stock* (*HTTP POST endpoint*) – точка входу, яка приймає параметр строкового типу «назва компанії» (*ticker*) та числовий параметр *days*, який задає горизонт прогнозування, та повертає результати прогнозування для однієї компанії;

2) *Forecast Multiple Stocks* (*HTTP POST endpoint*) – точка входу, яка приймає параметр типу масив строк «назви компаній» (*tickers*) та числовий параметр *days*, який задає горизонт прогнозування, та повертає результати прогнозування для декількох компаній, назви яких задані параметром *tickers*.

Сторінка *Redoc* документації розробленого *API* приведена на рис. 3.10-3.11.

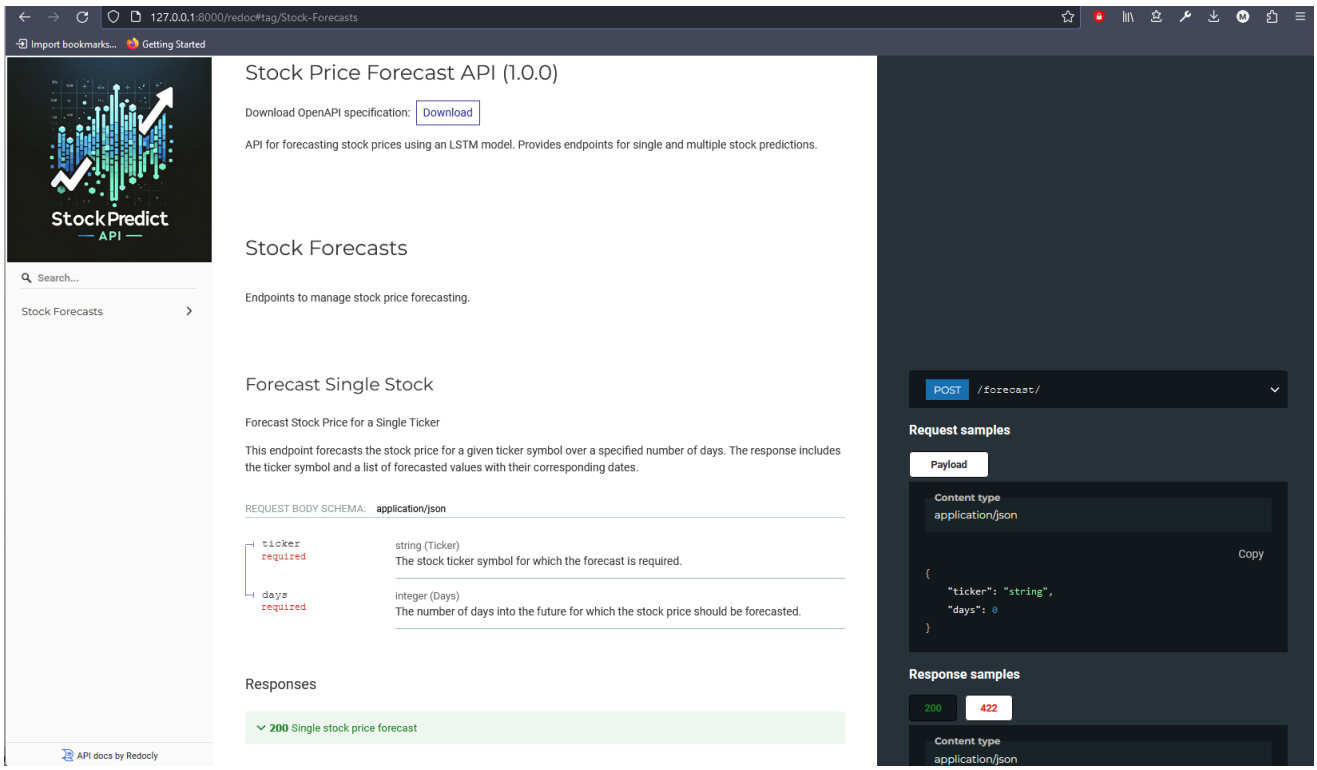

Рис. 3.10. *Redoc* документація розробленого *API – Forecast Single Stock*

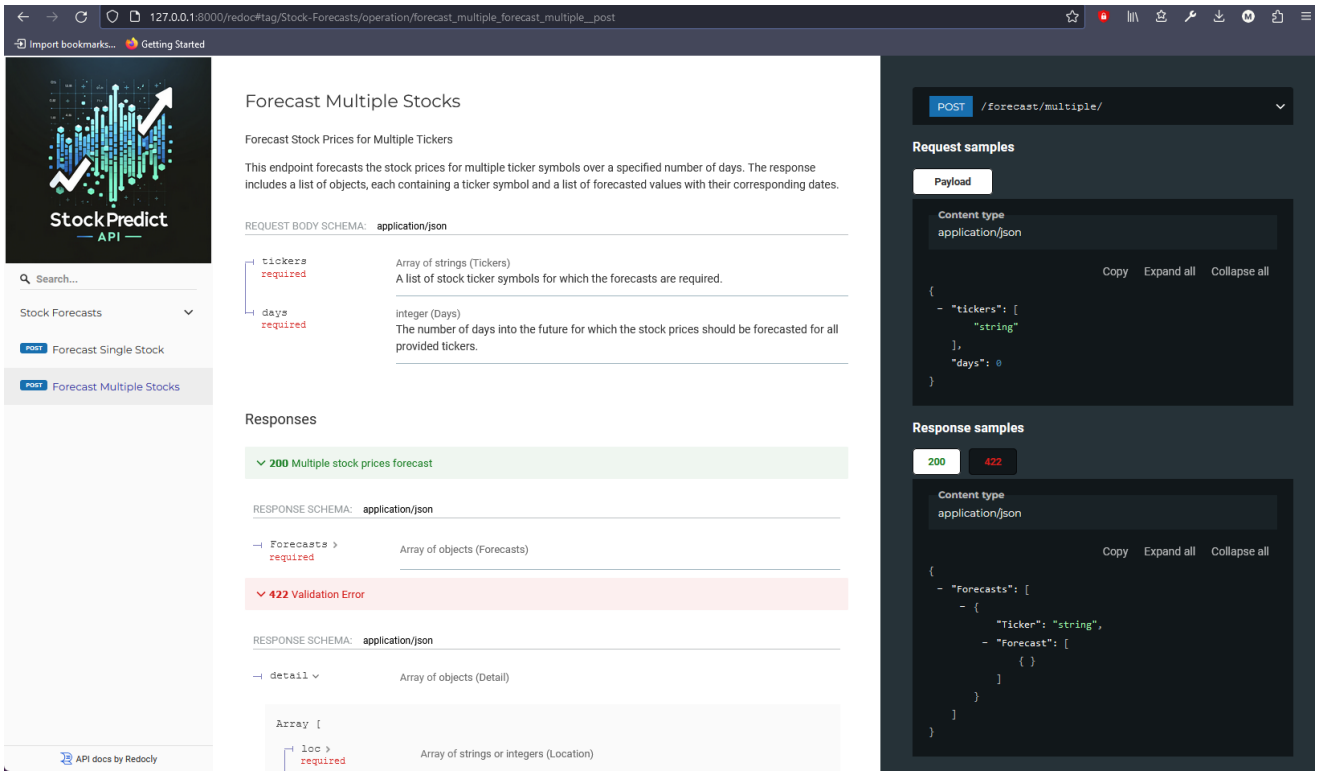

Рис. 3.11. *Redoc* документація розробленого *API – Forecast Multiple Stocks*

Приклад виконання запиту до *API* у *Swagger* приведено на рис. 3.12-3.13.

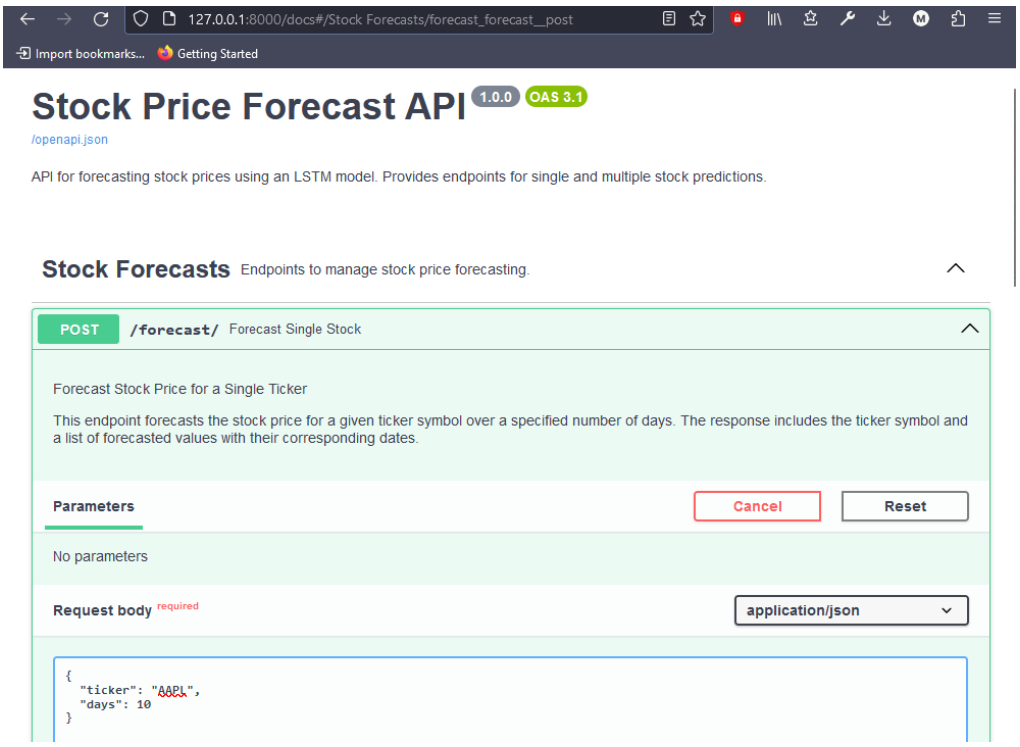

Рис. 3.12. Приклад виконання запиту до *API* в *Swagger*

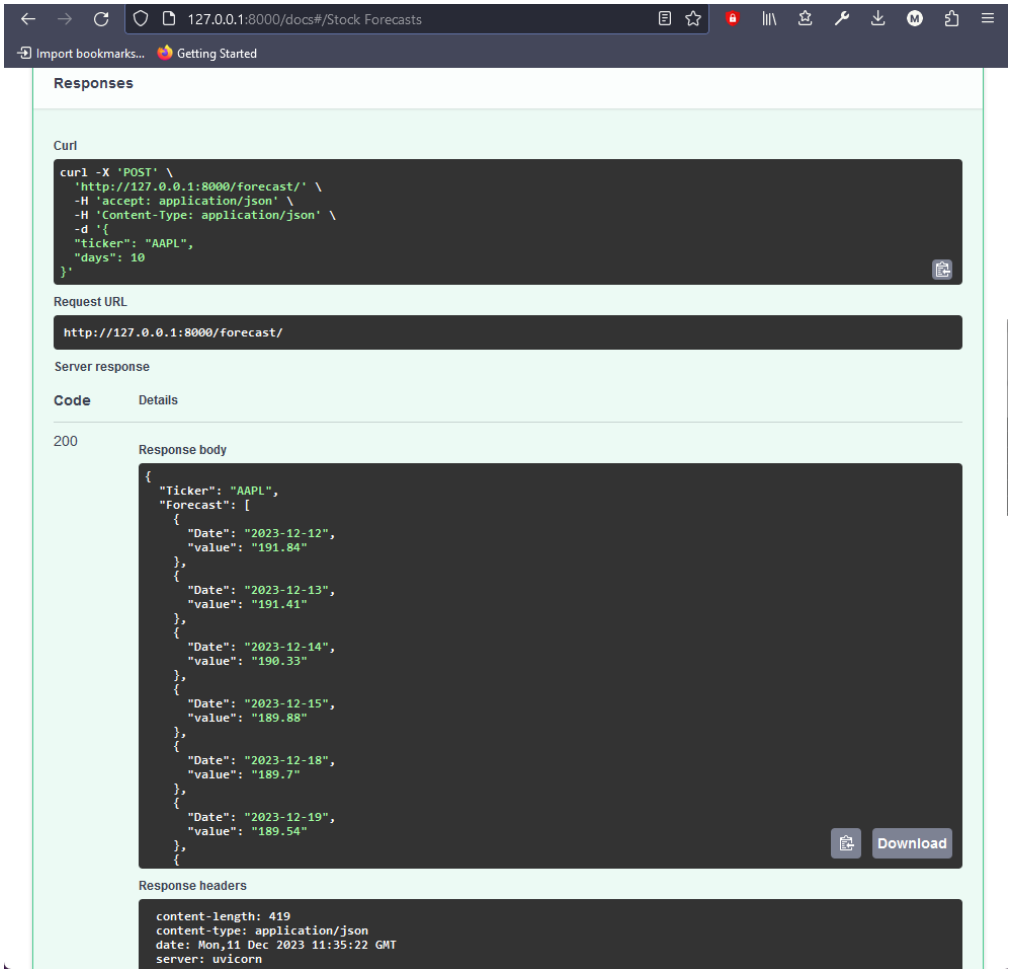

Рис. 3.13. Приклад відповіді на виконаний запит до *API* в *Swagger*

3.5.2. Інтеграція з програмною системою управління інвестиціями

Процес роботи розробленого програмного модуля в інтеграції з сторонньою програмною системою представлено на діаграмі послідовностей (рис. 3.14).

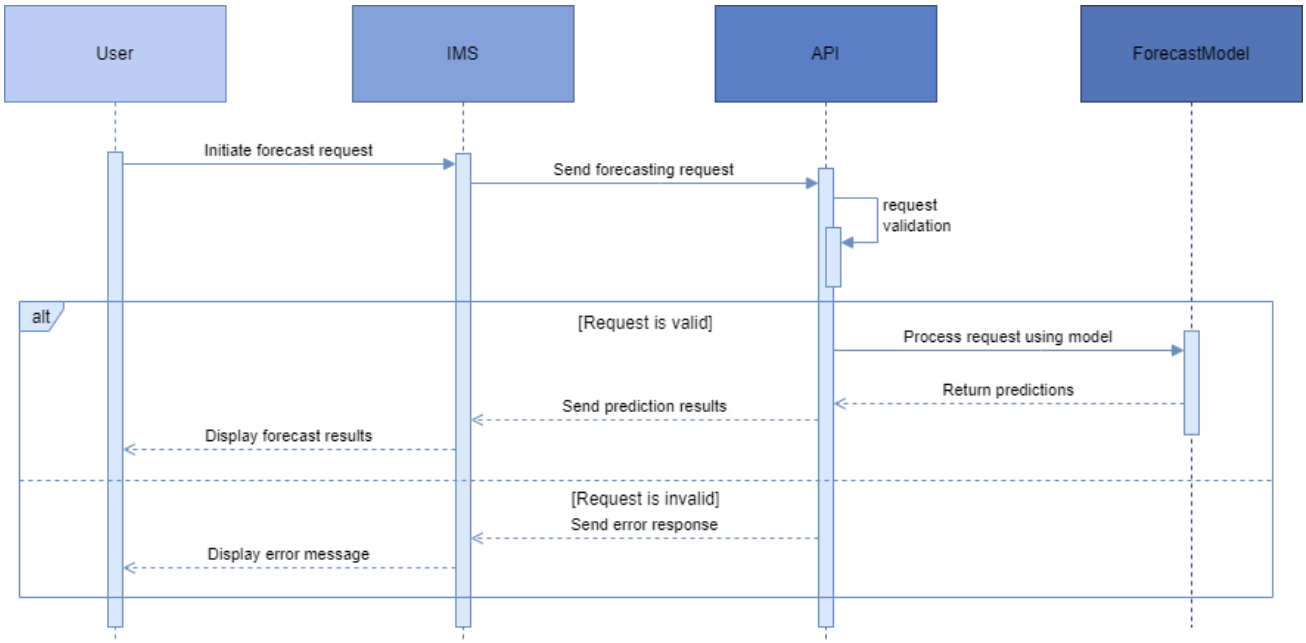

Рис. 3.14. Діаграма послідовностей роботи програмного модуля у інтеграції з сторонньою програмною системою

Реалізацію інтеграції розробленого програмного модуля було продемонстровано на прикладі його інтеграції у функціонал попередньо розробленої програмної системи управління інвестиціями «*IMS Prototype*».

IMS Prototype – це програмна система для управління та обліку персонального портфелю цінних паперів, яка була розроблена для надання функціоналу конструювання інвестиційного портфелю, відбору активів на базі наданої про них інфографіки, отримання аналітики та перегляду розрахованих фінансових показників портфелю користувача [31]. *IMS Prototype* є комплексним інструменту аналітики та фінансового менеджменту, проте його функціонал обмежується наданням аналітики в історичному розрізі та на поточний момент часу.

Інтеграція розробленого модуля прогнозування у цю систему дозволило розширити її функціональність додавши дві нові функціональні можливості:

1. В доповнення до перегляду історичних даних зміни цін акцій компаній, отримання прогнозованих даних зміни цін компаній на визначений користувачем період. При цьому горизонт прогнозування задається гнучко, користувач сам обирає потрібний йому відрізок часу для прогнозування у відповідному полі на користувацькому інтерфейсі. Ця функціональність додає можливість користувачеві оцінити компанії на етапах відбору активів та ребалансування портфелю в розрізі прогнозних оцінок динаміки цін їх акцій. Ці дані є результатами роботи розробленого програмного модуля, який інтегрований у систему через *API* інтерфейс. Також для користувача є можливість додати до порівняння обрані компанії, і таким чином проводити порівняльну оцінку прогнозованих значень цін акцій декількох компаній. Ця функціональність доступна на вкладці "*Stock Forecast Viewer*" головного вікна *IMS Prototype* (рис. 3.15).

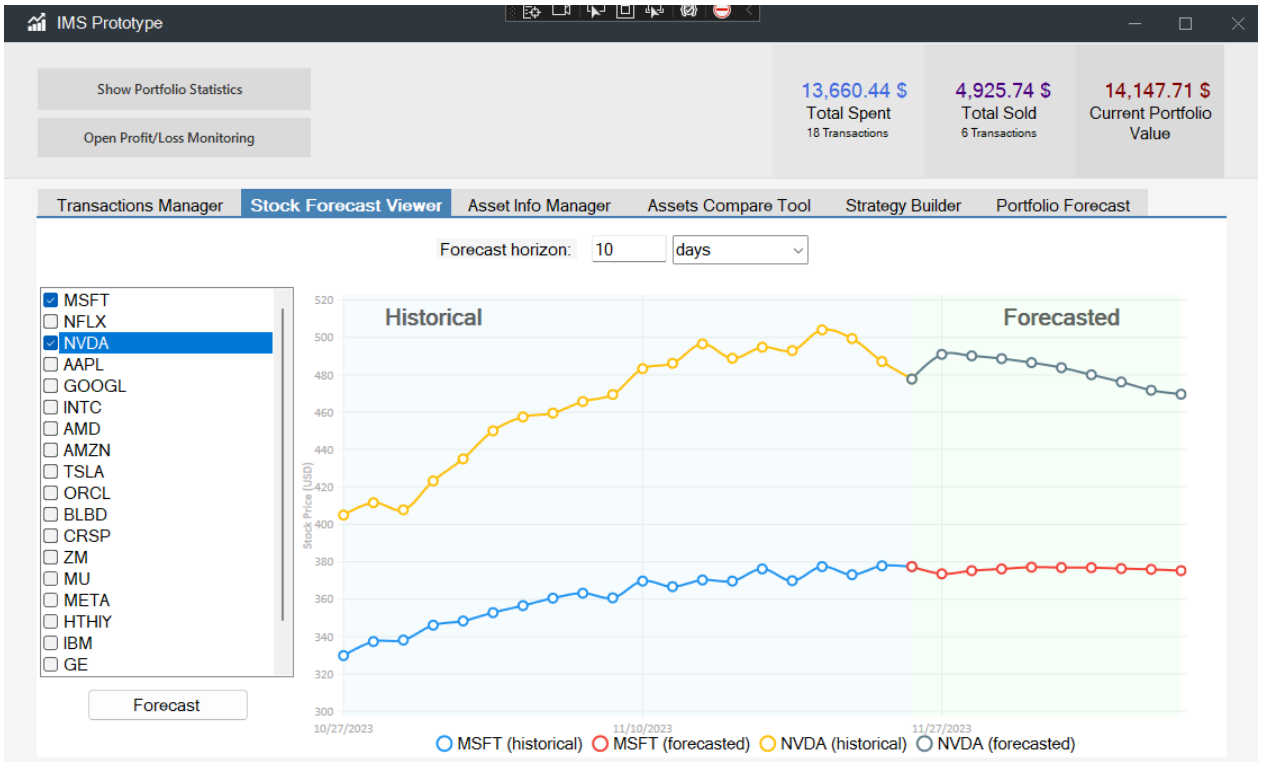

Рис. 3.15. Приклад інтеграції розробленого модуля в функціонал *IMS – Stock Forecast Viewer*

2. Отримання прогнозованих показників для портфеля користувача – окрім перегляду прогнозваних значень цін акцій компаній є можливість для користувача переглянути розраховані прогнозні показники дохідності та загальної вартості інвестиційного портфелю на заданий горизонт прогнозування. Цей функціонал дозволяє користувачеві бачити як зміниться дохідність його портфелю з його поточною конфігурацією та структурою через певний проміжок часу. Крім цього наявність цього функціоналу дає можливість користувачеві проводити симуляції різних сценаріїв продажу / покупки активів (зміни структури портфелю) та отримувати прогнозну оцінку того як ті чи інші зміни вплинуть на показники портфелю через певний проміжок часу. Наприклад, користувач може створити транзакцію на покупку *N* акцій певної компанії у свій портфель (за допомогою вже наявного в *IMS Prototype* функціоналу роботи з транзакціями), а після цього за допомогою фунціоналу прогнозування подивитись як зміняться прогнозні показники портфелю після проведення такої транзакції через певний проміжок часу в майбутньому. Ця функціональність доступна на вкладці "*Portfolio Forecast*" головного вікна *IMS Prototype* (рис. 3.16).

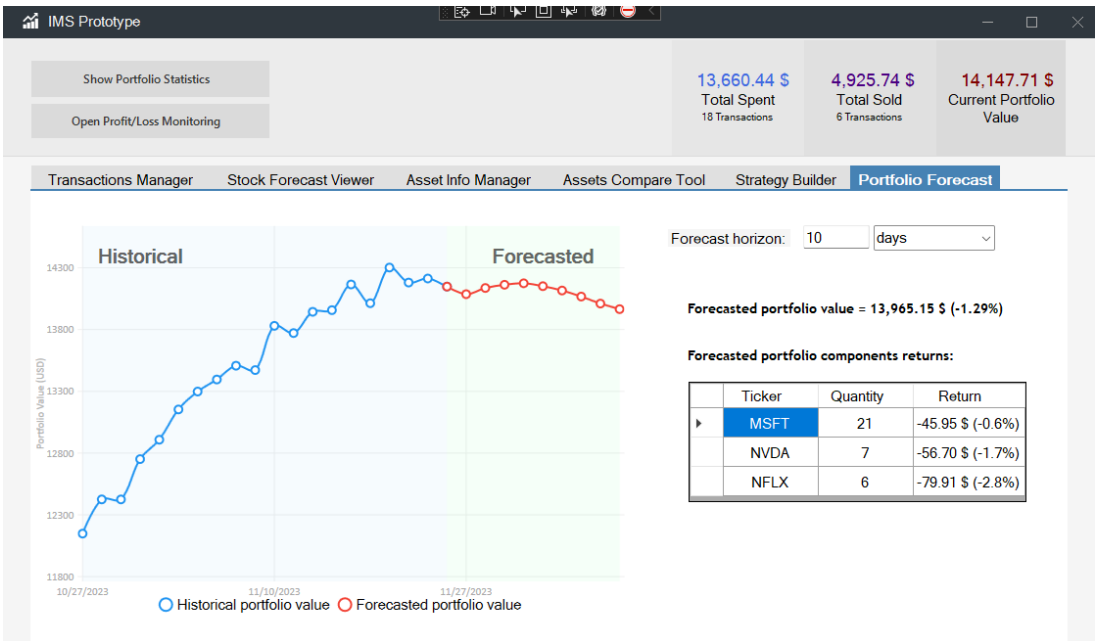

Рис. 3.16. Приклад інтеграції розробленого модуля в функціонал *IMS – Portfolio Forecast*

З технічної точки зору послідовність роботи цього функціоналу включає перехід користувача до відповідних вкладок у вікні системи *IMS Prototype* та задання горизонту прогнозування. Таким чином відбувається ініціація запиту на прогнозування від користувача до програмної системи (*IMS*), зображена відповідним повідомленням (стрілкою) на діаграмі послідовностей (рис. 3.14). Після цього система надсилає відповідний запит до *API* програмного модуля прогнозування. *API* валідує це запит, а потім залежно від результату валідації оброблює його і повертає результат виконання прогнозування або відповідь з помилкою. Ця послідовність проілюстрована відповідно у блоці альтернативних операндів діаграми послідовностей (рис. 3.14).

3.6 Висновки до розділу

В даному розділі було проведено декомпозицію розроблюваного модулю на логічні складові (компоненти), які реалізують визначені на етапі проектування аспекти та етапи рішення поставленої задачі, а також визначено характер взаємодій між ними.

Було розроблено програмний продукт, який реалізує автоматизований збір, обробку та аналіз фінансових даних, створення ознак (предикторів) з необроблених даних, обчислення значимості ознак та відбір найбільш значимих предикторів для включення до прогнозної моделі. Було побудовано нейронну мережу на базі *LSTM*, проведено експериментальний підбір параметрів моделі, її навчання та тестування. Результати роботи моделі було оцінено з використанням обраного набору метрик, та у порівнянні з базовою прогнозною моделлю (моделлю наївного прогнозування).

Аналіз отриманих показників точності роботи розробленої моделі і моделі наївного прогнозування, представлений у таблиці 3.2, свідчить про високу спроможність моделі відтворювати динаміку цін фінансових інструментів і здатність ефективно прогнозувати майбутні значення та підкреслює значно вищу

точність прогнозів та надійність розробленої моделі у порівнянні з моделлю наївного прогнозування.

Побудована нейронна прогнозна модель стала основою компоненту прогнозування розробленого модулю. Крім цього було реалізовано базу даних для зберігання необхідних для роботи модуля сутностей, механізми збору, обробки, та оновлення даних, формування датасету та виділення найбільш значимих предикторів для включення в модель на основі методу градієнтного бустингу. Крім цього, було імплементовано інтерфейс взаємодії з програмним модулем згідно обраного на етапі проектування підходу, а саме – *API* інтерфейс.

Було проведено інтеграцію розробленого модулю з існуючою програмною системою управління інвестиціями. Для цього було додано нову функціональність до цієї системи, яка використовує результати роботи модуля та взаємодіє з ним через *API* інтерфейс. Отримані результати демонструють нові можливості, які може дати інтеграція модулю прогнозування у функціонал існуючих фінансових програмних систем.

ВИСНОВКИ

В даній роботі було досліджено проблематику задачі прогнозування та аналізу фінансових показників активів на фондовому ринку, було доведено актуальність пошуку нових рішень цієї задачі. Проведений огляд та порівняльний аналіз існуючих програмних рішень та наукових досліджень і запропонованих у них методик виявив наявність у них множини обмежень, недоліків та недопрацювань, підтверджуючи необхідність розробки нового рішення та було сформульовану відповідну постановку задачі для розробки в ході виконання даної роботи.

На основі проведеного огляду предметної області та аналізу існуючих рішень було запропоновано підхід, який ставить за ціль виправити ключові недоліки існуючих методик та полягає у використанні системного підходу до включення різних класів показників для прогнозування цін фінансових інструментів, включаючи технічні, фундаментальні та макроекономічні показники. Такий набір даних також було доповнено за рахунок врахування текстової інформації. Застосування технік *NLP* та аналізу настрою дозволило інтегрувати новини, звіти та інші текстові джерела інформації, які впливають на ринкові тенденції та відображають актуальну повістку, яка впливає на цільову величину.

У поєднанні результатів цього комплексного підходу до формування набору даних, на основі яких здійснюється прогноз, з методами глибокого навчання було створено нейронну модель на базі *LSTM* для прогнозування цін фінансових інструментів на фондовому ринку. Для оцінки її ефективності було обрано набір метрик, а також проведено порівняльний аналіз результатів з базовою моделлю (наївним прогнозуванням). Отримані результати показують високу точність роботи створеної моделі, як і те, що розроблена модель значним чином перевершує у своїй точності модель наївного прогнозування.

Проведений аналіз метрик якості роботи розробленої моделі на основі глибоких нейронних мереж і моделі наївного прогнозування вказує на значні

переваги застосування розробленої моделі. Зокрема, коефіцієнт детермінації R^2 для розробленої моделі складає 0,714, що істотно перевищує показник моделі наївного прогнозу, що має негативне значення -2,527. Це свідчить про високу спроможність моделі відтворювати динаміку цін фінансових інструментів і здатність ефективно прогнозувати майбутні значення. Щодо середньої абсолютної відсоткової помилки (*MAPE*), то для розробленої моделі цей показник становить всього 0,015 у порівнянні з 0,062 для наївного прогнозу, що підкреслює значно вищу точність прогнозів. Крім того, значення середньої квадратичної помилки (*MSE*) і кореня середньоквадратичної помилки (*RMSE*) для розробленої моделі є значно нижчими (відповідно 12,584 та 3,547), ніж для наївного прогнозу (155,143 та 12,456), що також свідчить про вищу точність та надійність розробленої моделі. Таким чином, застосування розробленої моделі як засобу аналізу фінансових ринків демонструє суттєву перевагу перед традиційними методами прогнозування, що відкриває нові перспективи для розробки та використання подібних технологій у фінансовому аналізі.

Було спроектовано та розроблено програмний модуль, який інкапсулює роботу побудованої нейронної моделі на базі *LSTM*, а також реалізує механізми роботи з базою даних, регулярного оновлення даних моделі та реітерації її навчання на оновлених даних для підтримання відповідності моделі з перманентно змінюваними та динамічними реаліями фондового ринку. Також було реалізовано інтерфейс взаємодії з програмним модулем у вигляді *API*, який надається окремо як повноцінний програмний продукт – *Standalone API*, а також служить для уніфікованого інтерфейсу інтеграції розробленого модуля з іншими програмними системами відповідного призначення.

Створений програмний модуль було інтегровано у попередньо розроблений програмний продукт – систему управління інвестиціями та розширено існуючу функціональність системи додаванням нового функціоналу, який взаємодіє з модулем прогнозування та використовує результати його роботи у свої бізнес-логіці.

Предметом майбутніх досліджень та покращень запропонованого підходу та розробленого програмного модуля та прогнозної моделі є вдосконалення етапу обробки текстової інформації з додаванням класифікації текстових даних за джерелами та авторитетністю авторів та присвоєння їм відповідних вагових коефіцієнтів у доповнення до вже використовуваного обчислення показників тональності тексту. Окрім цього вартим уваги є розширення базового набору залучених показників (предикторів) задля кращого моделювання зв’язків між чинниками, що формують вихідну величину. Також наступним кроком досліджень є врахування кореляцій між цінами акцій різних компаній у рамках одної індустрії, так як ціноутворення фінансових інструментів не відбувається ізольовано, а має наслідки для множинного числа залучених сторін.

СПИСОК БІБЛІОГРАФІЧНИХ ПОСИЛАНЬ ВИКОРИСТАНИХ ДЖЕРЕЛ

1. M. Zolfaghari and S. Gholami, "*A hybrid approach of adaptive wavelet transform, long short-term memory and ARIMA-GARCH family models for the stock index prediction" Expert Systems with Applications, vol. 182*. – 2021. – *p*. 115149.

2. Y. Lin, Y. Yan, J. Xu, Y. Liao, and F. Ma, "*Forecasting stock index price using the CEEMDAN-LSTM model," The North American Journal of Economics and Finance, vol. 57*. – 2021. – *p*. 101421.

3. Paolo Sironi, *Modern portfolio management: from Markowitz to probabilistic scenario optimisation; goal-based and long-term portfolio choice*. – *London*: *Risk Books*, 2015.

4. Sharpe, William F., Alexander, Gordon J., Bailey, Jeffery V. *Investments, Fifth edition – Englewood Cliffs, New Jersey*: *Prentice Hall International, Inc*., 1995.

5. Aswath Damodaran, *Investment valuation: tools and techniques for determining the value of any asset*. – *Hoboken, N.J.*: *Wiley*, 2012.

6. B. G. Malkiel, *A Random Walk Down Wall Street: the time-tested strategy for successful investing*. – *S.L.*: *W W Norton*, 2020.

7. A. W. Lo and A Craig Mackinlay, *A Non-Random Walk Down Wall Street*. – *Princeton, Nj*: *Princeton University Press*, 2011.

8. *Stock Forecast Based On a Predictive Algorithm | I Know First* [Електронний ресурс]. Режим доступу: *https://iknowfirst.com/* (дата звернення 20.09.2023) – *I Know First*.

9. *TipRanks | Stock Market Research, News and Analyst Forecasts* [Електронний ресурс]. Режим доступу: *https://www.tipranks.com/* (дата звернення 20.09.2023) – *TipRanks*.

10. *FinBrain | Stock Predictions with Artificial Intelligence* [Електронний ресурс]. Режим доступу: *https://finbrain.tech/* (дата звернення 20.09.2023) – *FinBrain*.

11. Lu, W., Li, J., Li, Y., Sun, A. and Wang, J. *A CNN-LSTM-based model to forecast stock prices. Complexity*. – 2020. – *p*.1-10.

12. Jarrah, M. and Derbali, M. *Predicting Saudi Stock Market Index by Using Multivariate Time Series Based on Deep Learning. Applied Sciences, 13(14)*. – 2023. – *p*. 8356.

13. Akita, R., Yoshihara, A., Matsubara, T. and Uehara, K. June. *Deep learning for stock prediction using numerical and textual information. In 2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS). IEEE.* – 2016. – *pp*. 1-6.

14. Drake, P.P. and Fabozzi, F.J. *The basics of finance: An introduction to financial markets, business finance, and portfolio management*. – 2010. – *Vol*. 192. *John Wiley & Sons – pp*. 211-239.

15. Gravetter, F.J., Wallnau, L.B., Forzano, L.A.B. and Witnauer, J.E. *Essentials of statistics for the behavioral sciences. Cengage Learning*. – 2020. – *pp*. 490-512.

16. Gers, F.A., Schraudolph, N.N. and Schmidhuber, J. *Learning precise timing with LSTM recurrent networks. Journal of machine learning research*. – 2002. – *Iss*. 3(*Aug*). – *pp*. 115-143.

17. Hastie T., Tibshirani R., Friedman J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction. – Stanford*, *CA*: *Springer*, 2017. – *pp*. 430-522.

18. Piexeiro M. *Time Series Forecasting in Python*. – *Shelter Island*: *Manning Publications*, 2022. – *pp*. 287-358.

19. Bartram S., Branke J., Motahari M. *Artificial Intelligence In Asset Management*. – *CFA Institute Research Foundation*, 2020. – *pp*. 12-41.

20. Lewinson E. *Python for Finance: Cookbook*. – *Birmingham*: *Packt Publishing*, 2020. – *pp*. 283-306.

21. Hilpisch Y. *Financial Theory with Python*. – *Sebastopol*, *CA*: *O'Reilly Media*, 2021. – *pp*. 60-66.

22. Kulathumani M. *The new dynamic of portfolio management*. – *Plantation*, *FL*: *J. Ross Publishing*, 2021. – *pp*. 176-203.

23. De Prado M.L. *Advances in Financial Machine Learning*. – *Hoboken, New Jersey*: *John Wiley* & *Sons*, *Inc*., 2018 – *pp*. 23-40.

24. Zwingmann T. *AI-Powered Business Intelligence: Improving Forecasts and Decision Making with Machine Learning*. – *Sebastopol, CA*: *O'Reilly Media*, 2022. – *pp*. 54-65.

25. Lazzeri F. *Machine Learning for Time Series Forecasting with Python*. – *Indianapolis, Indiana*: *John Wiley & Sons, Inc*., 2021. – 206 *p*.

26. Ramalho L. *Fluent Python: Clear, Concise, and Effective Programming*. – *Sebastopol, CA*: *O'Reilly Media*, 2022. – 979 *p*.

27. Jareno F., Negrut L. *US Stock Market And Macroeconomic Factors. Journal of Applied Business Research, 32(1)*. – 2015. – *pp*. 325-340.

28. Verma R.K., Bansal R. *Impact of macroeconomic variables on the performance of stock exchange: a systematic review. International Journal of Emerging Markets. – Vol*. 16 *No*. 7, 2021. – *pp*. 1291-1329.

29. Sarkar D. *Text Analytics with Python: A Practitioner's Guide to Natural Language Processing, 2nd ed. Edition*. – *New York, NY*: *Apress Media LLC*, 2019. – *pp*. 570-611.

30. Tetlock P.C. *Giving Content to Investor Sentiment: The Role of Media in the Stock Market. The Journal of Finance 62(3)*. – 2007. – *pp*. 1139-1168.

31. Бойченко С.В., Іванченко О.В. Положення про дипломні роботи (проекти) випускників Національного авіаційного університету. – Київ: НАУ, 2017. – 63 с.

32. Саттарова М.Л. Прототип інформаційної системи управління та обліку персонального портфелю цінних паперів. – Дипломна робота на здобуття ступеня бакалавра спеціальності "Комп'ютерні науки", "Інформаційні управляючі системи та технології". – Київ, 2022. – 66 с

ДОДАТКИ

Додаток А

Програмна реалізація компоненту веб-скрапінгу на прикладі ресурсу *Macrotrends*

```
import requests
import json
from bs4 import BeautifulSoup

import urllib.request as urllib2
from urllib.request import Request, urlopen

headers = {
    'Host': 'www.macrotrends.net',
    'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537',
    'Accept': 'text/html',
    'Accept-Encoding': 'gzip, deflate, br',
    'Connection': 'keep-alive'
}

def fetch_html(url):
    req = Request(url=url, headers=headers, method='get')
    page = urlopen(req)

    return page


def convert_literal_to_number(text):
    if text.endswith('B'):
        return float(text[:-1]) * 1e9
    elif text.endswith('M'):
        return float(text[:-1]) * 1e6
    return text

def extract_table_data(html):
    soup = BeautifulSoup(html, 'html.parser')

    table = soup.find('table', class_='table')

    if table is None:
        raise ValueError("Table not found")

    headers_row = table.find_all('tr')[1] if len(table.find_all('tr')) > 1 else
None
    if headers_row is None:
        raise ValueError("Headers row not found")
    headers = [header.get_text(strip=True) for header in
headers_row.find_all('th')]

    table_title_tag = table.find('th')
    if table_title_tag is None:
        raise ValueError("Table title not found")
    table_title = table_title_tag.get_text(strip=True).replace(' ',
'_').replace('/', '')

    rows = table.find_all('tr')[2:]
    data = []
    for row in rows:
        cols = row.find_all('td')
```

```
        if len(cols) > 0:
            row_data = {}
            for i, col in enumerate(cols):
                text = col.get_text(strip=True)

                if text.startswith('$'):
                    text = text.replace('$', '').replace(',', '')
                    text = float(convert_literal_to_number(text))
                elif text.endswith('%'):
                    text = float(text.replace('%', ''))
                elif text.replace('.', '', 1).isdigit():
                    text = float(text)
                elif text == "":
                    text = None
                row_data[headers[i]] = text
            data.append(row_data)

    return data, table_title


def write_to_json(data, output_file):
    with open(output_file, 'w', encoding='utf-8') as file:
        json.dump(data, file, ensure_ascii=False, indent=4)

url = "https://www.macrotrends.net/stocks/charts/AAPL/apple/quick-ratio"

directory_path = "\\FeaturesRawDataFiles"

try:
    html = fetch_html(url)
    data, table_title = extract_table_data(html)
    output_file = f"{table_title}.json"
    write_to_json(data, directory_path+output_file)
    print(f"Data saved to {output_file}")
except Exception as e:
    print(f"An error occurred: {str(e)}")
```

Програмна реалізація компоненту відбору ознак (*Feature Selection*) за допомогою методу градієнтного бустингу

```
import matplotlib
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from pandas_datareader import data as web
import xgboost as xgboost
import sklearn

dataFilePath='\\Python_MergeData\\'

def get_features_dataset(stock_dataset, rows):
    # Moving Average Convergence Divergence (12EMA - 26EMA)
    stock_dataset['26EMA'] = pd.ewm(stock_dataset['Close'], span=26)
    stock_dataset['12EMA'] = pd.ewm(stock_dataset['Close'], span=12)
    stock_dataset['26EMA'] = stock_dataset['Close'].ewm(span=26).mean()
    stock_dataset['12EMA'] = stock_dataset['Close'].ewm(span=12).mean()

    stock_dataset['MACD'] = (stock_dataset['12EMA']- stock_dataset['26EMA'])

    # Exponential Moving Average
    # stock_dataset['EMA'] = stock_dataset['Close'].ewm(com=0.5).mean()
    # stock_dataset['1EMA'] = pd.ewma(stock_dataset['Close'], span=1)
    stock_dataset = stock_dataset.copy()

    return stock_dataset

def get_feature_importance_data(data_income):
    data = data_income.copy()
    y = data['Close'][:]

    X = data.copy()
    X = X.assign(Net_Income=X.pop('Net Income'))
    X.drop(columns='EMA200', inplace=True)
    X = X.iloc[:, 5:]

    train_samples = int(X.shape[0] * 0.75)

    X_train = X.iloc[:train_samples]
    X_test = X.iloc[train_samples:]

    y_train = y.iloc[:train_samples]
    y_test = y.iloc[train_samples:]

    return (X_train, y_train), (X_test, y_test)

def perform_feature_selection(X_train_FI, y_train_FI, X_test_FI, y_test_FI,
stock_dataset, filename):
    regressor = xgboost.XGBRegressor(gamma=0.0, n_estimators=150,
base_score=0.7, colsample_bytree=1,
                                     learning_rate=0.05)
    xgbModel = regressor.fit(X_train_FI, y_train_FI, \
                             eval_set=[(X_train_FI, y_train_FI), (X_test_FI,
y_test_FI)], \
                             verbose=False)
```

```
    test = sum(xgbModel.feature_importances_)
    print(test)

    values_dict = {}

    importance_values = xgbModel.feature_importances_.tolist()

    for key in X_test_FI.columns:
        for value in importance_values:
            values_dict[key] = value
            importance_values.remove(value)
            break

    value_to_subtract=0.2
    values_dict['PE Ratio'] = values_dict['PE Ratio'] - value_to_subtract
    values_dict['Sentiment score'] = value_to_subtract
    values_dict = dict(sorted(values_dict.items(), reverse=True, key=lambda
item: item[1]))
    print(values_dict)

    fig = plt.figure(figsize=(18, 8))
    plt.rcParams["figure.autolayout"] = True
    plt.xticks(rotation=60)
    # plt.bar([i for i in range(len(xgbModel.feature_importances_))],
xgbModel.feature_importances_.tolist(),
    #         tick_label=X_test_FI.columns)
    plt.bar([i for i in range(len(values_dict))], list(values_dict.values()),
            tick_label=list(values_dict.keys()))
    plt.title('Feature importance scores')
    spacing = 0.200
    fig.subplots_adjust(bottom=spacing)
    plt.show()

    return xgbModel

def get_filtered_features_dataset(symbol):
    filename = str(symbol) + "_features.csv"
    dataset = pd.read_csv(dataFilePath + filename)
    stock_dataset = dataset

    (X_train_FI, y_train_FI), (X_test_FI, y_test_FI) =
get_feature_importance_data(stock_dataset)
    xgbModel = perform_feature_selection(X_train_FI, y_train_FI, X_test_FI,
y_test_FI, stock_dataset, filename)

    return stock_dataset

def select_best_features_for_ML(stock_dataset, symbol, features_to_drop):
    filename = dataFilePath + 'feature_selection_result.csv'

    stock_dataset = stock_dataset.drop(features_to_drop, axis=1)
    stock_dataset.to_csv(filename, index=False)
    dataset = pd.read_csv(filename)
    print(dataset.head())
    print(dataset.shape)

stock_dataset = get_filtered_features_dataset(symbol)
```

Програмна реалізація прогнозної моделі на базі *LSTM*

```
import keras
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import tensorflow as tf
from sklearn.preprocessing import MinMaxScaler
import math
from sklearn.metrics import mean_squared_error
from keras.models import Sequential
from keras.callbacks import ModelCheckpoint, EarlyStopping, ReduceLROnPlateau
from keras.layers import Dropout, Dense
from keras.layers import LSTM
from sklearn.metrics import accuracy_score as acc
from scipy import stats
from keras.models import model_from_json

def volume_to_float(volume_str):
    volume_str = str(volume_str)
    return float(volume_str.replace(",",""))


def data_preparation(data_path, columns_list):
    dataset_original = pd.read_csv(data_path)
    dataset_original['Volume'] =
dataset_original['Volume'].apply(volume_to_float)

    training_set = dataset_original.iloc[:, columns_list].values

    print("Shape of train set with necessary columns: ",np.shape(training_set))
    print("\nOriginal Train dataset head\n")
    print(dataset_original.head(5))

    return training_set, dataset_original

def normalize_data(training_set):
    print("\n")
    sc = MinMaxScaler(feature_range = (0, 1))
    training_set_scaled = sc.fit_transform(training_set)

    print("\n Normalized Training set is of shape " +
str(np.shape(training_set_scaled)))
    print(training_set_scaled)
    print("\n\n")
    return training_set_scaled, sc

def time_series_convert(training_set_scaled, no_timesteps, total_obs,
target_column, input_features):
    X_train = []
    y_train = []
    for i in range(no_timesteps, total_obs):
        X_train.append(training_set_scaled[i - no_timesteps:i, 0:])
        y_train.append(training_set_scaled[i, target_column])

    X_train, y_train = np.array(X_train), np.array(y_train)
    X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1],
input_features))
```

```
    print("X_train shape",X_train.shape)
    print("y_train shape",y_train.shape)
    print(pd.DataFrame(y_train).head())

    print("\n\n Checking 1 Observation \n")
    print(X_train[0], y_train[0])
    return X_train, y_train

def build_rnn_model(X_train, input_features):
    regressor = Sequential()

    regressor.add(LSTM(units = 120, return_sequences = True, input_shape =
(X_train.shape[1], input_features)))
    regressor.add(Dropout(0.3))

    regressor.add(LSTM(units = 120, return_sequences = True))
    regressor.add(Dropout(0.3))

    regressor.add(LSTM(units = 120, return_sequences = True))
    regressor.add(Dropout(0.3))

    regressor.add(LSTM(units = 120, return_sequences = True))
    regressor.add(Dropout(0.3))

    regressor.add(LSTM(units = 120))
    regressor.add(Dropout(0.3))

    regressor.add(Dense(32, kernel_initializer="uniform",activation='relu'))
    regressor.add(Dense(units = 1))

    regressor.compile(optimizer = 'adam', loss = 'mean_squared_error')
    print("regressor model built and compiled...")
    return regressor

def train_model(regressor, X_train, y_train, epochs = 80):

    print("\ntraining started...")
    print(regressor.summary())
    regressor.fit(X_train, y_train, batch_size = 64, epochs = epochs,
validation_split = 0.15, shuffle = False)
    return regressor

def test_model(regressor, data_path_test, dataset_train, columns, sc, sc_close,
no_timesteps):

    dataset_test = pd.read_csv(data_path_test)
    print(dataset_test.head(3))
    real_stock_price = dataset_test.iloc[:, 1:2].values
    dataset_total = pd.concat((dataset_train[columns], dataset_test[columns]),
axis = 0)
    inputs = dataset_total[len(dataset_total) - len(dataset_test) -
no_timesteps:].values

    inputs = inputs.reshape(-1,10)
    inputs = sc.transform(inputs)
    print("\ninputs\n")
    print(inputs)

    X_test = []
    for i in range(no_timesteps, no_timesteps + 21):
```

```
        X_test.append(inputs[i-no_timesteps:i, 0:])
    X_test = np.array(X_test)
    X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1],
input_features))
    print("\n X_test \n")
    print(X_test.shape)
    print(X_test)

    predicted_stock_price = regressor.predict(X_test)
    real_stock_price_transformed = sc_close.transform(real_stock_price)
    predicted_stock_price_transformed =
sc_close.transform(predicted_stock_price)
    direction_pred = []

    print("Normalized Predicted Prices")
    print(np.mean(predicted_stock_price), stats.mode(predicted_stock_price),
np.median(predicted_stock_price))
    for pred in predicted_stock_price_transformed:
        print(pred)
        if np.around(pred,1) >=
np.around(np.mean(real_stock_price_transformed),1):
            direction_pred.append(1)
        else:
            direction_pred.append(0)

    print("Normalized Real Stock Prices")
    print(np.mean(real_stock_price_transformed),
stats.mode(real_stock_price_transformed),
np.median(real_stock_price_transformed))

    direction_test = []
    for value in real_stock_price_transformed:
        print(value)
        if np.around(value, 1) >=
np.around(np.mean(real_stock_price_transformed), 1):
            direction_test.append(1)
        else:
            direction_test.append(0)
    print("\n\n")
    predicted_stock_price = sc_close.inverse_transform(predicted_stock_price)
    print("Real stock prices on test data\n")
    print(real_stock_price)
    print("Final predicted stock prices on test data\n")
    print(predicted_stock_price)

    return real_stock_price, predicted_stock_price, direction_test,
direction_pred

def visualize_model(real_stock_price, predicted_stock_price, direction_test,
direction_predict):

    plt.figure(figsize=(14, 5), dpi=100)
    plt.plot(real_stock_price, color = 'red', label = 'Real Stock Price')
    plt.plot(predicted_stock_price, color = 'blue', label = 'Predicted Stock
Price')
    plt.title('Stock Price Prediction')
    plt.xlabel('Time')
    plt.ylabel('Stock Price')
    plt.legend()
    plt.show()
```

```
def evaluate_model(real_stock_price, predicted_stock_price, direction_test,
direction_pred):

    rmse = math.sqrt(mean_squared_error(real_stock_price,
predicted_stock_price))
    avg = real_stock_price.mean()
    rmse = rmse/avg

    print(rmse)
    print("Prediction Direction", direction_pred)
    print("Real Direction",direction_test)
    direction = acc(direction_test, direction_pred)
    direction = round(direction,4)*100
    print("Predicted values matched the actual direction {}% of the
time.".format(direction))
    return rmse, direction

def data_load_prepare(data_path, columns_list):
    training_set, dataset_original = data_preparation(data_path = data_path,
columns_list = columns_list)

    training_set_scaled, scaler = normalize_data(training_set)

    training_set_close = dataset_original.iloc[:, 1:2].values
    training_set_scaled_close, scaler_for_close =
normalize_data(training_set_close)

    no_timesteps = 120
    total_obs = len(training_set_scaled)
    input_features = 10
    target_column = 0

    X_train, y_train = time_series_convert(training_set_scaled, no_timesteps,
total_obs, target_column, input_features)

    return X_train, y_train, scaler, scaler_for_close, dataset_original

def load_rnn_model(json_path, weights_path):
    with open(json_path, 'r') as f:
        model = model_from_json(f.read())

    model.load_weights(weights_path)

    return model

symbol = 'AAPL'
dataFilePath='D:\StockAnalysisPythonProject\Investment-portfolio-validation-
analysis-for-post-trade-financial-services-firm\\'
data_path = dataFilePath + str(symbol) + "_features.csv"
columns_list = [1,2,3,4,5,6,7]

X_train, y_train, sc, sc_close, dataset_original = data_load_prepare(data_path,
columns_list)

epochs = 80
input_features = 10
no_timesteps = 120

data_path_test = dataFilePath + str(symbol) + "_indicators_test.csv"
```

```
columns = ['Close', 'Volume', 'MA7', 'MA21', 'MA60', 'MACD', 'EMA']
regressor_trained = 1

real_stock_price, predicted_stock_price, dir_test, dir_pred =
test_model(regressor_trained, data_path_test, dataset_original, columns, sc,
sc_close, no_timesteps)

#Hyperparameter Tuning
for learning_rate in [0.001, 0.002, 0.005]:
    for dropout in [0.2,0.3, 0.5]:
        model = build_rnn_model(X_train, input_features)
        print()
        print("Current model: LR={}, Dropout={}".format(
            learning_rate,dropout))
        print()
        save_best_weights = 'question_pairs_weights_lr={}_dropout={}.h5'.format(
            learning_rate,dropout)

        callbacks = [ModelCheckpoint(save_best_weights, monitor='val_loss',
save_best_only=True),
                     EarlyStopping(monitor='val_loss', patience=5, verbose=1,
mode='auto'),
                     ReduceLROnPlateau(monitor='val_loss', factor=0.2,
verbose=1, patience=3)]

        history = model.fit(X_train,
                            y_train,
                            batch_size=64,
                            epochs=1,
                            validation_split=0.15,
                            verbose=True,
                            shuffle=False,
                            callbacks = callbacks)

visualize_model(real_stock_price, predicted_stock_price, dir_test, dir_pred)
evaluate_model(real_stock_price, predicted_stock_price, dir_test, dir_pred)
```