МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет інформаційних технологій
Кафедра інтелектуальних технологій

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

НА ТЕМУ

Система моніторингу фінансових метрик

Галузь знань **12 «Інформаційні технології»**

Спеціальність **122 «Комп'ютерні науки»**

Освітня програма **«Аналітика даних»**

Освітній рівень: бакалавр

Виконав: студент 4 курсу, групи АнД-41
Журавель Б.В
(прізвище та ініціали)

Керівник Самохвалов Ю. Я
(прізвище та ініціали)
д. т. н., професор
(науковий ступінь, звання)

Випускна кваліфікаційна робота бакалавра допущена до захисту
рішенням кафедри *інтелектуальних технологій*
Протокол № 11 від 06.06.2022 р.
зав. кафедри_____ доц. Іларіонов О.Є.

Київ – 2022

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра інтелектуальних технологій

Спеціальність 122 «Комп'ютерні науки»

ЗАТВЕРДЖУЮ
Завідувач кафедри
інтелектуальних технологій
Іларіонов О.Є.

_____

"___"_____2022 р.

ЗАВДАННЯ

НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Журавлю Богдану Васильовичу
(прізвище, ім'я, по батькові)

1. Тема проекту (роботи)

«Система моніторингу фінансових метрик» затверджена протоколом засідання кафедри від затверджена протоколом засідання кафедри від «23» грудня 2021 р. № 4

2. Термін здачі студентом закінченого проекту (роботи) 29 травня 2022 року

3. Вихідні дані до проекту (роботи)

Аналітика та оцінка фінансового інструмента на графічний інтерфейс користувача.

4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити)

Аналіз предметної області, аналіз проектних рішень, розробка системи.

5. Перелік презентаційного матеріалу (з точним зазначенням обов'язкових презентацій)

Мета, ціль та функції системи (1 слайд), аналіз предметної області (3 слайди), аналіз проектних рішень (2 слайди), розробка системи (4 слайди), висновки по роботі (1 слайд).

6. Консультанти з випускної кваліфікаційної роботи із зазначенням розділів випускної кваліфікаційної роботи, що їх стосуються

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 15 лютого 2022 року

Керівник _____ / Самохвалов Ю. Я. /
(підпис) (ПІБ)

Завдання прийняв до виконання _____ / Журавель Б. В. /
(підпис) (ПІБ)

КАЛЕНДАРНИЙ ПЛАН

Пор. №	Назва етапів випускної кваліфікаційної роботи	Термін виконання етапів випускної кваліфікаційної роботи	Примітка
1	Опрацювання літератури	15.02.2022 – 01.03.2022	
2	Робота над розділом 1. Аналіз предметної області, постановка задачі.	01.03.2022 – 20.03.2022	
3	Робота над розділом 2. Аналіз проектних рішень. Огляд принципів обробки фінансових даних. Аналіз існуючих рішень.	20.03.2022 – 11.04.2022	
4	Робота над розділом 3. Планування архітектури програмного забезпечення. Розробка системи.	11.04.2022 – 15.05.2022	
5	Робота над оформленням пояснювальної записки	25.05.2022	
6	Робота над презентацією	01.06.2022	

Студент-дипломник _____ / Журавель Б. В. /
(підпис) (ПІБ)

Керівник випускної кваліфікаційної роботи _____ / Самохвалов Ю. Я. /
(підпис) (ПІБ)

Анотація

Журавель Богдан Васильович виконав випускну кваліфікаційну роботу на тему «Система моніторингу фінансових метрик» за спеціальністю 122 – «Комп’ютерні науки».

У випускній кваліфікаційній роботі досліджено напрям інвестицій в Україні. Фондовий ринок обрано, як найкращий спосіб для інвестицій фізичним особам. Розроблено систему моніторингу фінансових метрик, яка у режимі реального часу обробляє фінансові дані та за допомогою сентиментального аналізу та методів глибинного навчання оцінює фінансові дані. Реалізовано чат-бот, для соціальної мережі Telegram, як найкращий графічний інтерфейс для даної системи. Розробка виконана на мові програмування Python.

Ключові слова: фінансові інвестиції, фондовий ринок, глибинне навчання, BERT, обробка даних у реальному часі, чат-бот.

Annotation

A final Bachelor's Thesis: «Financial metrics monitoring system», 122 Computer Science specialty, was made by **Bohdan Zhuravel.**

The direction of investments in Ukraine is investigated in the final qualification work. The stock market has been chosen as the best way to invest for individuals. Financial metrics monitoring system has been developed, which processes financial data in real time and evaluates financial data with sentimental analysis and deep learning methods. Implemented a chatbot for the social network Telegram, as the best graphical interface for this system. Development is performed in the Python programming language.

Keywords: financial investments, stock market, deep learning, BERT, real-time data processing, chat-bot.

Зміст

Вступ **8**

Розділ 1. Аналіз предметної області **9**

1.1 Роль інвестиційного попиту в економічному зростанні української економіки **9**

1.2 Установи для інвестицій **9**

1.2.1 Інвестиційний банк 9

1.2.2 Інвестиційні банки України 10

1.2.3 Шляхи самостійних інвесторів 11

1.2.4. Дохідність інвестицій на фондовому ринку: альтернатива. 14

1.4 Постановка задачі **17**

1.5 Визначення профілів зацікавлених сторін **18**

1.7 Висновки до розділу **21**

Розділ 2. Аналіз проектних рішень **22**

2.1 Аналіз існуючих рішень **22**

2.2 Аналіз вхідних даних для фінансових систем **23**

2.2.1 Фундаментальний та технічний аналіз 23

2.2.3 Формування аналітичних даних 24

2.2.4 Альтернативні дані 25

2.3 Методи інтелектуального аналізу фінансових даних **25**

2.3.1 Проблема випадкового блукання 27

2.3.2 Сентиментальний аналіз фінансових даних 28

2.4 Оцінка фінансових даних на основі сентиментального аналізу **29**

2.4.1 Нейронні мережі у задачах сентиментального аналізу 29

2.4.2 Принцип донавчання нейронної мережі 32

2.5 Висновки до розділу **34**

Розділ 3. Розробка системи **35**

3.1 Python, як інструмент для реалізації системи **35**

3.2 Розробка моделі глибиного навчання **36**

3.2.1 Попередня обробка даних 36

3.2.2 Імплементація нейронної мережі 37
3.2.3 Оцінка результатів та формування аналітики 38
3.3 Архітектура програмного забезпечення 39
3.3.1 ETL модуль; збір та обробка даних 40
3.3.2 База даних; розподілений кластер 41
3.3.3 API модуль; контролер 43
3.5 Інтерфейс кінцевого користувача 45
3.6 Реалізація чат-боту 46
3.7 Висновки до розділу 48
Висновки 50
Списки використаних джерел 51
Додатки 53

Вступ

Відповідно до основних тенденцій світового економічного розвитку, в нинішніх умовах значно посилюється значення розвитку фондового ринку як одного із каталізаторів інституційних змін в функціональних, галузевих і регіональних територіальних сферах, які утворюють єдину економічну систему. Фінансовий ринок — складова фінансової системи держави. За своєю суттю це механізм перерозподілу фінансових ресурсів між окремими суб'єктами підприємницької діяльності, державою і населенням, між учасниками бюджетного процесу, деякими міжнародними фінансовими інститутами. Фінансовий ринок може успішно розвиватися і функціонувати лише в ринкових умовах. Об’єктами відносин на фінансовому ринку є грошово-кредитні ресурси, цінні папери та фінансові послуги, суб’єктами відносин — держава, підприємства різних форм власності, окремі громадяни.

Метою роботи є створення системи моніторингу фінансових ринків щодо оцінки явищ та процесів, що відбуваються у сфері фінансових ринків, у нерозривному зв’язку з їх кількісними та якісними характеристиками, та виявлення основної тенденції та закономірності їх розвитку.

Ціль роботи - спростити процес інвестиції у фінансовий ринок для користувачів системи, які не володіють базовою фінансовою грамотністю за допомогою інтелектуальної системи.

Функції. Система повинна спрощувати здатність розв’язувати складні спеціалізовані задачі та практичні проблеми в економічній сфері, які характеризуються комплексністю та невизначеністю умов, що передбачає застосування теорій та методів технологічної та економічної науки за рахунок:

- пошуку інформативних даних з багатьох популярних фінансових джерел
- швидкій обробці даних у реальному часі
- інтелектуальному аналізі даних
- навичок використання інформаційних технологій
- здатністю до пошуку, оброблення та аналізу інформації з різних джерел
- здатністю рекомендувати обґрунтовані рішення

Дослідженню питань теоритичних основ присвячені роботи багатьох зарубіжних і вітчизняних вчених економістів, математиків та докторів технічних наук.

Розділ 1. Аналіз предметної області

1.1 Роль інвестиційного попиту в економічному зростанні української економіки

З II кварталу 2016 року інвестиційний попит закріпив за собою роль локомотива відновлення та росту української економіки (внесок валового нагромадження основного капіталу (ВНОК) у зростання реального ВВП зріс до 2.3 в.п. із 0.2 в.п. у IV кварталі 2015 року та 0.7 в.п. у I кварталі 2016 року). Значні темпи зростання ВНОК протягом 2016-2021 років підтримувалися зокрема поліпшенням ділових очікувань підприємств на тлі достатньо сприятливої зовнішньо цінової кон'юнктури та поступового покращення фінансових результатів підприємств.

1.2 Установи для інвестицій

1.2.1 Інвестиційний банк

Інвестиційний банк – це спеціалізована кредитна установа, яка залучає довгостроковий капітал і надає його у розпорядження шляхом випуску облігацій та інших видів боргових зобов'язань. Основні функції інвестиційного банку полягають у визначенні характеру та обсягу фінансових споживчих позицій, з'ясуванні вартості фінансвого інструмента, визначення термінів їх емісії та розміщення серед інвесторів. Інвестиційний банк – це не просто середина між інвестором та оцінювачем, а й інвестиційний гарант та організатор ринку, який може займати велику частину об'єму на фінансовому ринку.

Основні функції і-банків**:**

* Інвестиційний банкінг — Інвестиційний консультант для будь якої форми державної влади, корпорації чи бізнесу. Кредитор. Керуючий активами

* Брокерські послуги — Ринок акцій, ринок цінних паперів з фіксованим відсотком, іпотечний ринок і ринок нерухомості, товарний ринок

* Послуги інвестиційного менеджменту — Створення під-установ, для залучення капіталу під довірче управління. Створення фондів різних типів. Передання капіталу під професійне управління.

* Сервісні послуги — Кредитна підтримка інвесторів, кредитна підтримка емітентів, проведення валютно-обмінних операцій, розрахунково-клірингове обслуговування, страхування, аналітична підтримка

* Дослідження фінансових ринків. Аналітика фінансового інструмента, моделювання поведінки ринку, симуляції, розгляд усіх можливих сценаріїв, та продаж даного аналізу.

Топ 10 і-банків світу

Десять найбільших глобальних інвестиційних банків світу, станом на 31 грудня 2010 року, є наступним (за загальними гонорарами):

Таблиця 1 10 найбільших інвестиційних компаній

Номер	Компанія	Гонорари ($млн)
1.	J.P. Morgan	$5,533.85
2.	Bank of America Merrill Lynch	$4,581.59
3.	Goldman Sachs	$4,386.52
4.	Morgan Stanley	$4,055.48
5.	Credit Suisse	$3,379.12
6.	Deutsche Bank	$3,286.80
7.	Citi	$3,238.67
8.	Barclays	$2,864.44
9.	UBS	$2,614.44
10.	BNP Paribas	$1,433.89

Найбільші світові банки отримують рейтингові бали залежно від їх M&A порад, синдикованих позичок, еквіті ринків капіталу та боргових ринків капіталу.

1.2.2 Інвестиційні банки України

В Україні, як правило, інвестиційні банки не називають інвестиційними банками (ІБ), а натомість використовують назву інвестиційні компанії. Щодо українського законодавства, то воно також змушує підприємства використовувати назву інвестиційна компанія, а не інвестиційний банк, у зв'язку з тим, що слово «банк» у назві дозволяється лише тим юридичним особам, які зареєстровані Національним банком України як банк та мають банківську ліцензію (не всі інвестиційні банки України мають банківську ліцензію, оскільки часто не надають послуги типові для комерційних банків).

Банк - слово "банк" та похідні від нього дозволяється використовувати у назві лише юридичним особам, які зареєстровані Національним банком України як

банк та мають банківську ліцензію. Винятком є міжнародні організації, що діють на території України відповідно до міжнародних договорів, згода на обов'язковість яких надана Верховною Радою України та законодавства України. Інвестиційна компанія - це торговець цінними паперами, який, крім здійснення інших видів діяльності, може залучати кошти для здійснення спільного інвестування шляхом емісії цінних паперів та їх розміщення.

Відповідно, найчастіше українські інвестиційні банки називають свої підприємства інвестиційними компаніями. Цікавою особливістю вітчизняних інвестиційних компаній є те, що вони, як і західні інвестиційні банки є структурами, в яких діють основні інвестиційні підрозділи: інвестиційний банкінг, управління активами, брокерські послуги, дослідження фін. ринків тощо.

Інвестиційні компанії мають право відмовитися від деяких інвестиційних підрозділів замінивши, як правило консалтинговими послуги інших компаній. Зазвичай, інвестиційні компанії звертаються до дослідницьких компаній, які в свою чергу розробляють системи для аналізу фондових ринків.

Станом на 2022 рік, в Україні немає інвестиційної компанії, яка займається внутрішнім дослідженням фінансів та надає відкрито усі послуги інвест банкінгу. Отже шлях для інвестицій обирає самостійний інвестор.

1.2.3 Шляхи самостійних інвесторів

Кожна людина, що заробила і акумулювала певний обсяг заощаджень, рано чи пізно замислюється як його зберегти і примножити. Сьогодні потенційним і діючим інвесторам доступна безліч інформації, в якій важко орієнтуватись. Доволі широкий спектр інструментів для інвестування важко оцінити людині без інвестиційного досвіду.

Класичні інструменти для самостійного інвестора, це нерухомість, цінні папери, криптовалюти, дорогоцінні метали.

Нерухомість**.** Не зважаючи на кризи останніх років, ускладнення оподаткування, скандальні історії із забудовниками, величезна кількість українців купують нерухомість на початку будівництва і продають її ближче до здачі. В деяких випадках залишають нерухомість у власності і здають її в оренду (“орендна” модель). Інвестор може придбати житлову або комерційну нерухомість. У проєкти є сенс входити при 15-30% готовності об'єкта. Продавати квартири найкраще при 90% готовності, коли проводяться фасадні, оздоблювальні та внутрішні інженерні роботи в будинку.

Після отримання документів на право власності при подальшому продажі нерухомості, інвестор буде додатково платити податки: 5% податку на доходи,

1,5% військового збору та 1% державного мита. Щоб уникнути цього податкового навантаження варто продавати недобудовані квадратні метри. Під час продажу нерухомості квартиру через договір відступлення, права вимоги переходять до нового інвестора. Лише в 2020 році в Україні уклали біля 261,7 тис. таких договорів.

Рентабельність "орендної" моделі в операціях з нерухомістю має чисту доходність на рівні 4-6% річних у валюті. Отже "окупність" нерухомості буде довгою і потребуватиме постійних додаткових вкладень грошей і часу на поточні ремонти, пошук орендарів і т.ін. Крім того, в даному виді інвестицій не варто забувати про моральне застарівання самих споруд і будинків, що безпосередньо впливає на остаточну вартість нерухомості. Враховуючи, що банківські депозити в Україні у валюті представлені на рівні 0,3-1% річних, цей варіант виглядає непогано.

Вихід з такої інвестиції в гроші може бути ускладненим або розтягнутим в часі. Тому що попит на нерухомість коливається і нерухомість може подешевшати з часом.

Цінні папери**.** Міжнародний фондовий ринок і акції закордонних компаній також є актуальним інвестиційним інструментом, який користується попитом в українських інвесторів. Частина інвесторів обирає тактику спекулятивних операцій на коливаннях окремих волатильних інструментів, частина має довгострокову стратегію і вкладається в індекси. Хтось віддає перевагу дивідендним акціям, хтось планує заробити на здорожчанні самих акцій. Безліч стратегій, кожна з яких має бути ретельно продумана. Але передбачити все неможливо. Фактична дохідність такого інструмента як цінні папери може коливатись в залежності від багатьох чинників.

На фондовий ринок напряму впливають багато факторів: ситуація в самій компанії-емітенті, випуск нових продуктів, економічний клімат в країні компанії-емітента, злиття, поглинання, дії конкурентів та ще безліч мікро та макро чинників. Чим краще ситуація, тим дорожче коштує частка компанії. Спекулятивні угоди, вимагають специфічних знань і величезного запасу нервових клітин в моменти корекції фондового ринку.

Вихід з такої інвестиції в гроші реалізований в безготівковій формі, через продаж цінних паперів і зарахування грошей на банківський рахунок інвестора.

Криптовалюти**.** Щоденний обіг криптовалюти в Україні складає біля 1 млрд грн. На крипторинку постійно виникають десятки тисяч нових проектів та стартапів. Для залучення грошей на свої проекти вони використовують первинну пропозицію монет (ICO) або розміщення монет через криптобіржу (IEO). Так з 2016 року було проведено більше ніж 5600 ICO, через які залучили біля 64,5 млрд. дол, як зазначає аналітик The Block Ларі Чермак. При цьому середня

рентабельність проектів склала біля 87% і величезна кількість з них вже закрились та мають неліквідні токени.

Криптовалюта є високоризиковим інвестиційним інструментом. Світова практика формування інвестиційного портфелю віддає таким інструментам 2-5% портфелю. Українські інвестори, менш обережні і віддають криптовалютам до 15%.

Як приймаються рішення купити ті або інші монети? Традиційно на початку інвестування проводять класичний для крипторинку аналіз 4T:

1. Theme (концепція);

2. Tech (технологія, код);

3. Team (команда);

4. Token (криптомонета).

Кожен з етапів оцінки токену вимагає від інвестора чудово розбиратися в профільній сфері та бути постійно у стані недовіри та напруження. Навіть якщо дорожня карта проекту ідеальна на перший погляд , все одно немає гарантії, що проект не скам (“мильна булька”). Як приклад - проект PlexCoin зібрав 15 млн. дол. та мав чудову дорожню карту, проте був закритий Комісією з цінних паперів та бірж США (SEC). Аналіз монети - ювелірна і комплексна робота. Наприклад, низька капіталізація може свідчити про можливість маніпуляції нею, обсяг торгів може бути штучним. Навіть якщо монету включила до свого переліку велика торгова площадка (Binance, Coinbase, Huobi Global та ін.), це не убезпечує інвестора від ризиків шахрайства.

Законодавче поле в питаннях криптовалют в Україні невизначене. В 2014 році НБУ взагалі визначав криптовалюту як “грошовий сурогат”.

В 2017 році вийшла спільна заява регулятора, де було зазначено “складна правова природа криптовалют не дозволяє визнати їх грошовими коштами, валютою і платіжним засобом іншої країни, валютною цінністю, електронними грошима, цінними паперами, грошовим сурогатом».

Ще один крихітний крок було здійснено в 2021 році, коли український регулятор ухвалив “Основні засади грошово-кредитної політики на 2022 рік та середньострокову перспективу”, в яких продовжує вбачати ймовірну нестабільність гривні, неконтрольованість у проведенні фінмоніторингу в сфері обігу криптовалюти.

Проте українці мають вільний доступ до криптобірж, найбільші з яких Binance, Kuna Exchange, EXMO та BTC Trade UA і активно торгують Bitcoin, Ethereum та іншими альткоінами. За умов існуючого конфлікту інтересів українців і регулятора, Україна перебуває у списку країн з найприбутковішим ICO.

Як висновок - інвестиції у крипторинок є високоризиковими. Корекції в межах 20% в день звичне явище. Законодавче поле для цього інструменту в Україні відсутнє, неможливо спрогнозувати наскільки легким буде вихід з інвестиції в гроші за умови пильної уваги контролюючих інституцій.

Дорогоцінні метали**.** Золото або інший аналогічний актив – можна придбати як фізично так і через фінансові інструменти. Золото – ліквідний актив. Його можна швидко перевести в грошові кошти. Для банку, який продає і купує золоті злитки, це маржинальний дохід на рівні 5 - 25%. Тобто для інвестора це “гра у довгу”, а не на місяць чи рік. Що далі термін продажу, тим вищий прибуток. При цьому якщо інвестор може зменшити супутні витрати купівлі/ продажу, якщо купуватиме дорогоцінний метал не у фізичному вигляді, а, наприклад через ETF (Exchange Traded Fund). Безумовно, дорогоцінні метали є надійними, але реальний інвестиційний ефект дуже пролонгований.

Доволі широкий спектр інструментів для інвестування важко оцінити людині без інвестиційного досвіду та фінансової грамоти. Сервіс, який надає рекомендації щодо інвестицій значно спрощує процес прийняття рішень для самостійного інвестора.

1.2.4. Дохідність інвестицій на фондовому ринку: альтернатива.

Цінні папери. 2022 рік почався на хвилі позитиву: американські фондові індекси та ринки були налаштовані на активне зростання по всьому світу. Однак усі позитивні прогнози та очікування перекреслило згортатання стимуляції економіки та війна в Україні.

Люди стали набагато обережнішими в інвестуванні, а аналітики – у прогнозах.

Проте, гроші повинні завжди залишатися в обороті та в Україні стає відкритим питанням куди ж краще інвестувати, типова та надійна нерухомість чи незрозумілий для багатьох фондовий ринок.

Починаючи з часів пандемії COVID-19 у першу чергу, зросли акції компаній hitech-індустрії. В умовах пандемії коронавірусу, коли більшість людей перебували на самоізоляції та працювали віддалено, послуги цих компаній виявилися найбільш затребуваними.

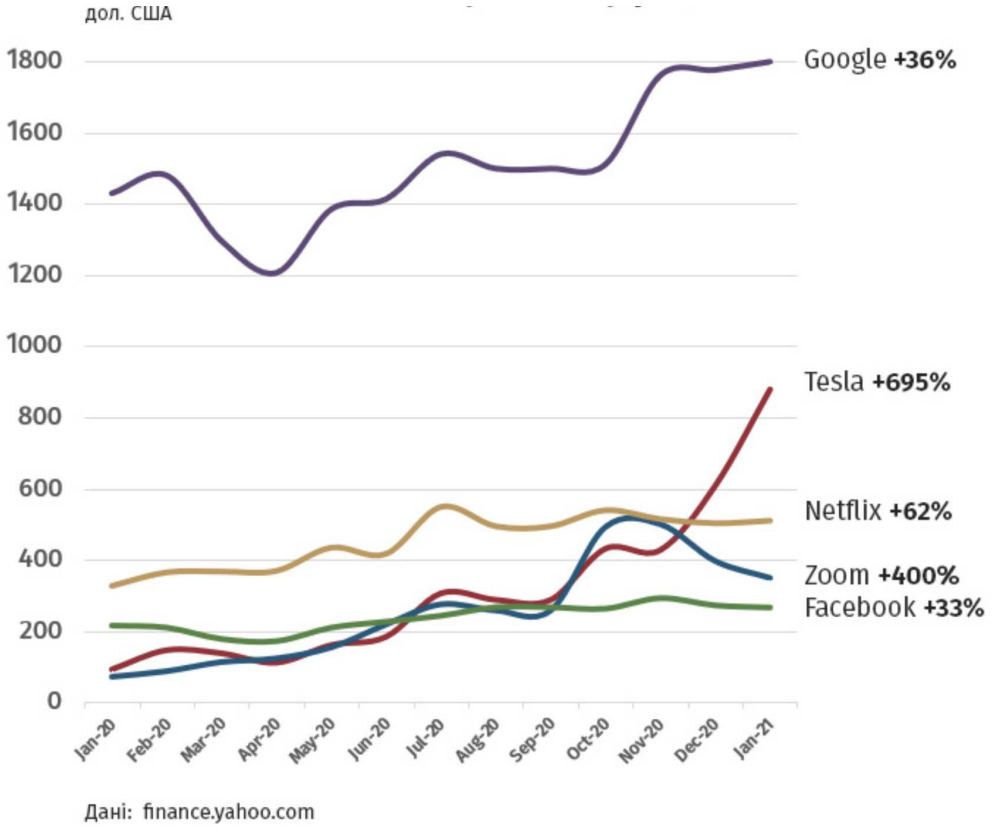

Рис. 1 Графік зміни ціни цінних паперів на фондовому ринку

На рисунку 1 лідером стала компанія “Zoom”. Два роки тому вартість акцій стартапу з організації відеоконференцій становила 62-67 дол. На піку їх ціна сягала 550 дол. Той, хто придбав акції в січні 2020 року і продав їх на максимумах, збільшив свої статки усемеро.

Навіть з урахуванням падіння капіталізації Zoom в кінці року після виходу новин про успішне тестування вакцини проти COVID-19 зростання за рік становило 400%.
Збільшення виручки технологічних гігантів штовхало вартість акцій угору. Звісно, не такими темпами, як Zoom, проте прибутковість інвестицій була значною. Наприклад, папери Facebook за 2020 рік подорожчали на 33%, інтернет-корпорації Alphabet (Google) – на 36%, відеосервісу Netflix – на 62%.

Нерухомість. Інвестиції в нерухомість в Україні більш традиційні, ніж купівля цінних паперів чи золота. Особливо популярні вони у Києві та на його околицях, а також в інших великих містах.
Криза 2020 року не сильно вдарила по ринку житлової нерухомості. За розрахунками City Development Solutions, обсяги продажів квартир у столиці зросли на 30%, а ціни на нерухомість відновилися до рівня 2013-2014 років.

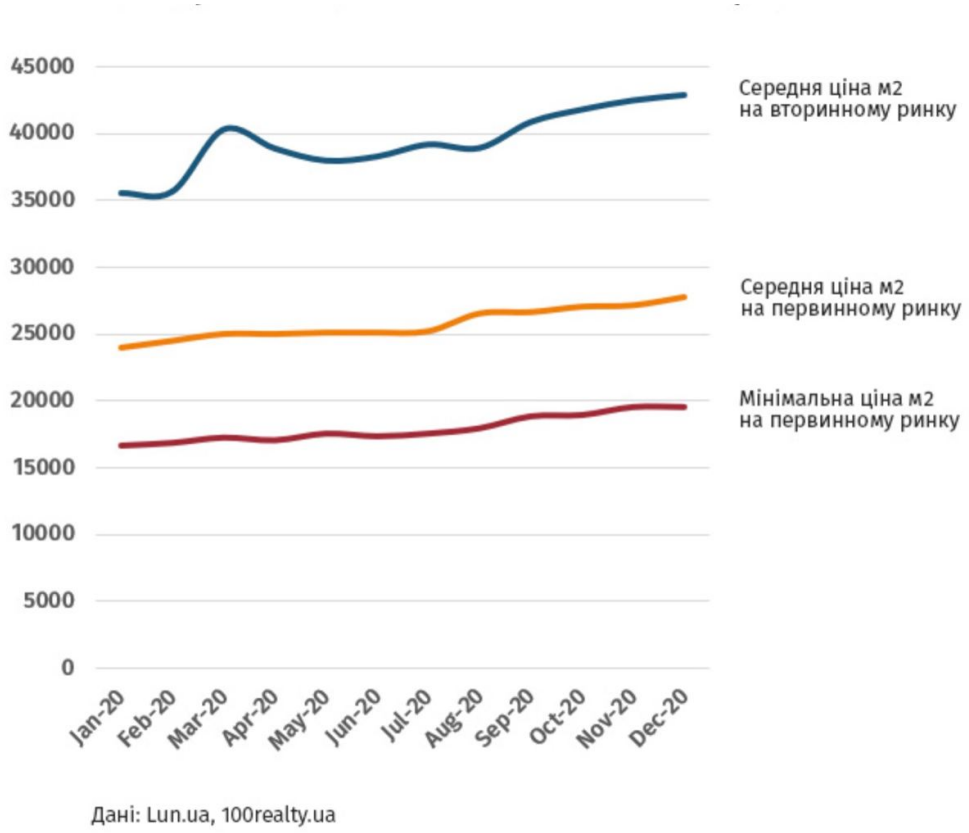

Рис. 2 Зміни цін на нерухомість у Києві

Згідно з рисунку 2, найбільший попит у 2021 році мали квартири бізнес-класу та сегменту "комфорт". Також експерти звертали увагу на житлові комплекси, збудовані за концепцією "місто в місті" або "0 км", тобто ті, які мають розвинену інфраструктуру.

"Потенціал для зростання цін у 2022 році є. За рік ціни можуть зрости на 7-10% при збереженні темпів, які були наприкінці 2021 року. Водночас у популярних форматах "місто в місті", багатофункціональних та сервісних комплексах, еко-проєктах ціни росли набагато швидше, враховуючи високий рівень платоспроможного попиту та стабільно високі темпи будівництва", – повідомляла ЕП експерт на ринку нерухомості Вікторія Берещак.

Загалом інвестиції в нерухомість підходять не всім. Для них характерні високий поріг входу і тривалий період окупності. Якщо ж грошей для інвестицій небагато, то вкладати їх усі в купівлю нерухомості – не найкраща ідея.

"Якщо є 50-100 тисяч доларів, то це про диверсифікацію. Квартира – це один об'єкт. З її купівлею можна і вгадати, і програти. Не треба забувати і про великі транзакційні витрати на оформлення житла. Краще такі кошти інвестувати у кілька інструментів", – вважає “Forbes-Україна”. За консенсус-прогнозом “Forbes-Україна”**,** інвестиції в житлову нерухомість у Києві та передмісті можуть дати дохід 10-15%.

Згідно оцінених плюсів та мінусів інвестування значно переважають інвестиції і фондовий ринок. Станом на 2022 Україна не володіє власним ліквідним та досить популярним фондовим ринком, проте у 2021, український необанк “Monobank” розробив додаток для інвестицій у фондовий ринок США, а саме компаній з списку S&P500 та ETF фондами.

1.4 Постановка задачі

Метою цієї дипломної роботи є формування та створення системи комплексного розуміння взаємозв’язків у фінансовій сфері щодо оцінки явищ та процесів, що відбуваються у кредитно-фінансовій сфері, у нерозривному зв’язку з їх кількісними та якісними характеристиками, та виявлення основної тенденції та закономірності їх розвитку для користувачів системи, які не володіють знаннями фінансової грамоти.

Згідно з аналітичного огляду предметної області, фондовий ринок залишається найкращим інвестиційним інстументом 2022 в Україні. Система повинна непреривно взаємодіяти з фондовими ринками, та аналізувати їх у реальному часі.

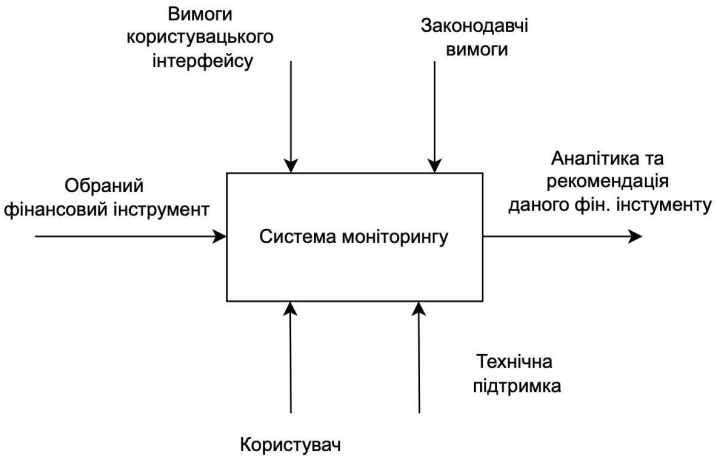

Рис. 3 Контекстна діаграма IDEF0 системи.

На вхід системи подається фінансовий інструмент фондового ринку, який закріплюється за користувачем. У чорній скриньці відбувається збір свіжих даних по даному фінансовому інструменту та згідно з інструментами інтелектуального аналізу відбувається повноцінний аналіз фінансового інструменту. Вихід системи передбачає повноцінний звіт, що надходить користувачеві щоденно. Елемент управління «Технічна підтримка», необхідний для коректної роботи системи.

Об'єкт дослідження – засоби автоматизованого аналізу фінансового інструмента.

Предмет дослідження – методи автоматизованого аналізу фінансового інструмента, їх аналіз та дослідження за домогою обчислювальних методів та статистики.

Методи дослідження – в дипломному проекті використовуються методи обробки числових даних та текстових документів, їх представлення у числовому форматі, для подання на вхід моделі глибокого навчання, а також методи багатоміткової класифікації тексту – для вирішення проблеми сентиментального аналізу.

1.5 Визначення профілів зацікавлених сторін

Основна ціль даної системи - спростити поняття "інвестиції" для звичайних людей, в яких немає базової фінансової грамотності. Отже, такою системою будуть користуватися багато користувачів, як і простих ,так і професіоналів, оскільки система буде наділена широким спектром інструментів та інтелектуальних алгоритмів, які набагато глибше можуть аналізувати дані, чим професіонал у сфері фінансів.

Також, дуже важливою стороною є об'єкт, який надає послуги купівлі акцій на фондовому ринку, а саме - Брокер.

Брокер — посередник при укладанні угод між продавцем і покупцем, страхувальником і страховиком, судновласником і фрахтувальником.

Брокерські компанії — це брокерські фірми, контори, представництва, асоціації, які створені для спільного надання посередницьких послуг при проведенні торгових, біржових операцій. Завдання брокерських компаній:

* зводити разом покупців і продавців;
* діяти як агенти продавця і покупця;
* здійснювати операції купівлі-продажу;
* отримувати за свої послуги плату — комісійну або брокерську винагороду.

Величина брокерської комісії звичайно пропорційна сумі угоди і регламентується біржовим комітетом (радою)

Отже, якщо система набуває великої популярності в інформаційному полі, та має юридичні підстави співпраці з брокером, обидві сторони залишаються в “плюсі”.
До прикладу, партнерство з брокером виглядає таким чином:

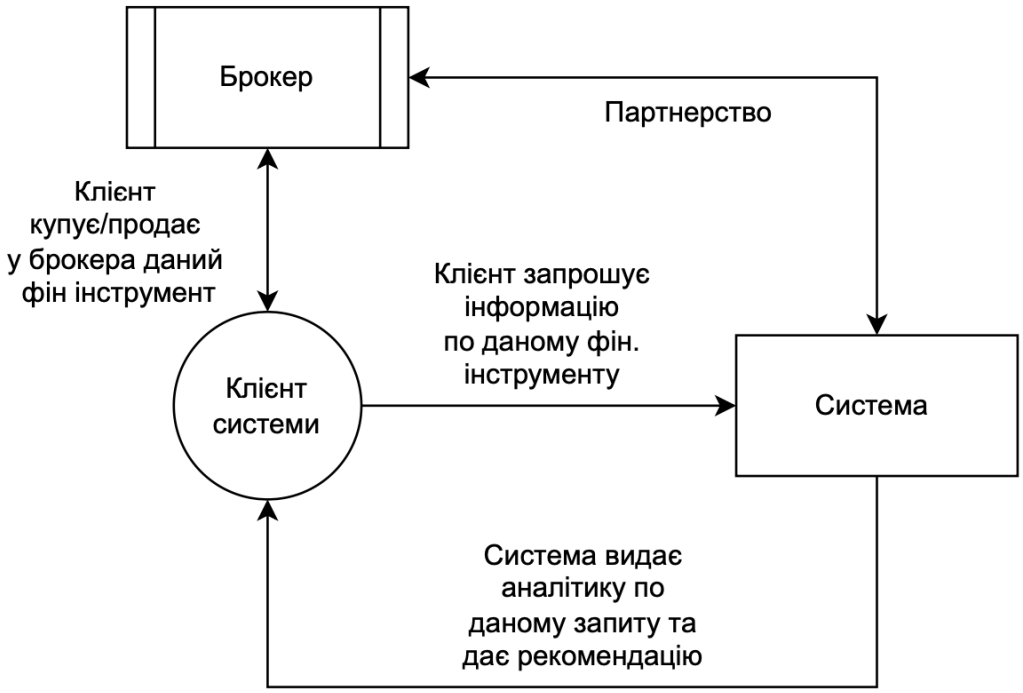

Рис. 4 Схема колаборації даної системи

Даний рисунок описує “ідеальний” випадок, коли клієнт запрошує у системи аналітику по даному фін. Інструменту. Клієнт системи остаточно приймає за собою рішення на рахунок купівлі/продажі даного фін. Актива та переходить на

сервіс брокера, де брокер надає свої послуги, а саме купівлі/продажі фін. Актива.

Відповідно, така схема задовільняє усі сторони. Брокер отримує комісію від активності клієнта, клієнт задоволений рекомендаціями від системи, а керівництво системи отримує прибуток з клієнта, та невелику частину від комісії, яку заплатив клієнт брокеру.

Нещодавно, український необанк “Monobank” створив додаток для інвестицій у фондовий ринок США. Команда “monobank” 24 2022 січня почала закрите тестування свого додатку для торгівлі цінними паперами mono invest. З його допомогою можна купувати/продавати акції компаній та індексні фонди.

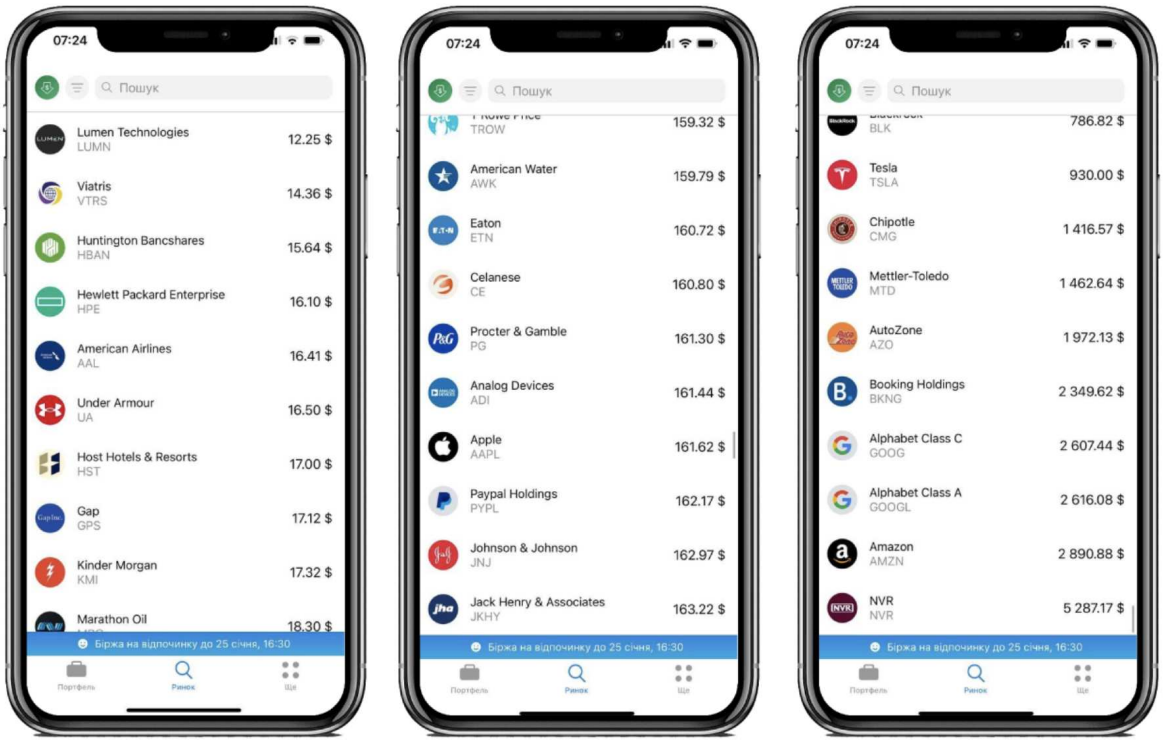

Рис. 5 Мобільний додаток “mono invest”

"mono invest” — окремий додаток для торгівлі акціями. Він тісно пов’язаний з “monobank”, має той же візуальний стиль. Команда вирішила зробити окремий додаток, оскільки торгівля акціями – опція, що цікава окремій групі людей.

Після завантаження “mono invest” система попросить пройти невелике опитування. Це вимога НБУ, таке ж опитування є і у інших сервісів. Таким чином сервіс намагається попередити про те, що торгівля цінними паперами — це також ризик: можна як заробити, так і втратити кошти. А головне — що відповідальність на кожному клієнті, а не банку чи брокері.

В сервісі заявили, що для купівлі доступні акції з індексу S&P 500 та декілька найпопулярніших ETF. Це американські компанії та індекси. Для навігації по п’яти сотням компаніям є пошук, розбивка по секторам та фільтри.

Поки невідомо, чи буде збільшуватись число доступних акцій (у брокера теоретично є доступ до більшого числа ринків, але чи їх підключать — невідомо).

1.7 Висновки до розділу

У даному розділі розглянуто предметну область. В ході аналізу предметної області, виявлено, що Україна майже не володіє інвестиційними компаніями, що призводить до самостійних рішень у сфері інвестування для фізичного лиця. Для цього було поставлено задачу у створені системи моніторингу фінансових ринків, щоб забезпечити громадян достовірною та якісною інформацією, які є зацікавленими у сфері фінансових інвестицій. Також було обрано фондовий ринок, як найкращий спосіб інвестиції та розглянуто зацікавлені сторони у вигляді колаборацій з брокерами та іншими компаніями даної області.

Розділ 2. Аналіз проектних рішень

2.1 Аналіз існуючих рішень

Рішення компанії Bloomberg

Bloomberg "торговельний термінал" - словосполучення, яке існує вже 30 років. Коли з'явився Bloomberg ще не було PC. Раніше термінал був hardware продуктом, а сьогодні це програмно-апаратний комплекс. Терміналом Bloomberg користуються виключно професіонали – передплата платформи коштує $24 тисячі (за даними Investopedia). Програма надає доступ до величезної кількості даних, тому зазвичай її використовують у комбінації з кількома моніторами. За офіційною статистикою на сайті виробника цим інструментом користуються 325 тисяч професіоналів фінансового ринку у світі.

Рішення компанії Thomson Reuters Eikon

Головний конкурент терміналу Bloomberg – це Eikon від Thomson Reuters. Це також професійна система для моніторингу та аналізу фінансової інформації. Інвестори використовують Eikon для доступу до даних у реальному часі з фінансових ринків та її аналітики. Одна з найцікавіших функцій терміналу – вбудований аналіз настроїв інвесторів. Система аналізує повідомлення в Twitter на задані теми і шукає індикатори позитивного або негативного настрою аудиторії. На основі цієї інформації інвестори можуть будувати гіпотези щодо подальших рухів на біржі.

Розробники Eikon використовували модульний підхід, тому вартість залежить від кінцевої функціональності терміналу. Повноцінна версія коштує $22 тисяч, а базова доступна від $3600 за рік.
Хоча ці системи є доволі різні та мають широкий та не схожий до іншого функціонал, основне що об’єднює цих двох терміналів - це професійна система для професіоналів у фінансах. На просторах інтернету, завжди можна знайти різні, не зовсім успішні інформаціййні джерела, які пропонують сервіс аналітики та моніторингу без залучень інтелектуальних алгоритмів, чи будь-яких алгоритмів.

Подача фінансової інформації є досить різною. Великі компанії застосовують професіональних підхід з дотриманням усіх правил аналізу та рекомендацій, а звичайні інформаційні джерела нічим не відрізняються від звичайних ЗМІ

інформаційних веб сторінок, не завжди можна довіряти таким джерелам. Отже, згідно огляду та застосування існуючих рішень прийнято скомбінувати два важливих та фундаментальних фактора - професіоналізм у обробці фін. даних та простота у подачі.

2.2 Аналіз вхідних даних для фінансових систем

Фінансові дані мають багато структур та форм. У таблиці показано чотири основних типи фінансових даних, упорядкованих зліва направо з точки зору збільшення різноманітності(Табл. 2).

Таблиця 2 Основні типи фінансових даних

Фундаментальні дані	**Ринкові дані**	**Аналітика**	**Альтернативні**
Активи	Ціна/волатильність	Рекомендації	Фото/аудіо/відео
Зобов'язання	Об’єм	Очікувана дохідність	Пошуки в Google
Продажі	Відкритий інтерес	Новини	Соціальні мережі
вартість/прибуток	Квоти	Кредитна ставка	Метадані

2.2.1 Фундаментальний та технічний аналіз

Фундаментальні дані охоплюють інформацію, яку можна знайти в регуляторних документах і бізнес-аналітиці компанії. Здебільшого це бухгалтерські дані, які звітуються щоквартально. Особливим аспектом цих даних є те, що вони повідомляються зі затримкою. Важливо точно розуміти, коли кожний звіт буде опублікований, щоб аналізувати цю інформацію завчасно, а не лише після того, як вона стала загальнодоступною. Поширеною помилкою фінансистів є припущення, що ці дані були опубліковані на кінець звітного періоду. Такого ніколи не буває. Наприклад, фундаментальні дані, опубліковані Bloomberg, індексуються за останньою датою, включеною у звіт, яка передує даті випуску (часто на 1,5 місяці). Іншими словами, Bloomberg призначає ці значення даті, коли вони не були відомі. Де коли неможливо повірити, скільки статей публікується щороку з неузгодженими фундаментальними даними, особливо в літературі щодо інвестування.

Другий аспект фундаментальних даних заключається у тому, що їх часто доповнюють, або відновлюють. "Доповнення" означає, що відсутнім даним надається значення, навіть якщо ці значення на той час були невідомі. "Відновлене значення" - це виправлене значення, яке виправляє неправильний початковий статус. Компанія може вносити кілька виправлень для результатів за минулий квартал після багато часу після першої публікації, а постачальники даних можуть перезаписувати початкові значення своїми виправленнями. Проблема в тому, що виправлені значення не були відомі того першого дня випуску. Деякі постачальники даних обходять цю проблему, зберігаючи кілька дат випуску та значень для кожної змінної. Наприклад, зазвичай є три значення для одного квартального випуску ВВП: вихідне опубліковане значення та два щомісячні перегляди. Тим не менш, дуже часто можна знайти дослідження, в яких використовується остаточне випущене значення та приписується час першого випуску або навіть останній день звітного періоду.

Фундаментальні дані надзвичайно прості та малочастотні. Будучи настільки доступним на ринку, малоймовірно, що у них залишається багато цінності.

Ринкові дані включають всю торгівельну діяльність, що відбувається на біржі (наприклад, CME) або на торговому майданчику (наприклад, MarketAxess). В ідеалі постачальник даних надав необроблений формат даних з різноманітною неструктурованою інформацією, такою як повідомлення FIX, які дозволяють повністю реконструювати торговий стакан, або повний набір відповідей BWIC (лімітовані заявки у стакані). Кожен учасник ринку залишає характерний слід у торгових записах, і за допомогою достатнього терпіння можна знайти спосіб передбачити наступний крок конкурента. Наприклад, алгоритми TWAP залишають дуже специфічний слід, який використовується хеджувальними алгоритмами для випередження своєї торгової діяльності наприкінці дня. Трейдери з графічним інтерфейсом часто торгують круглими лотами, і це можна використати як факт, щоб оцінити, який відсоток обсягу припадає на них у даний момент часу, а потім пов'язати це з конкретною поведінкою ринку.
Одним із привабливих аспектів даних FIX є те, що їх не так просто обробляти, на відміну від фундаментальних даних. Їх дуже багато: щоденно генерується понад 10 ТБ.

2.2.3 Формування аналітичних даних

Аналітичні дані - похідні дані, засновані на вихідному джерелі, яке може бути фундаментальним, ринковим, альтернативним або навіть набором іншої аналітики. Що характеризує аналітику, так це не зміст інформації, а те, що вона недоступна з першоджерела та оброблена певним чином. Інвестиційні банки та дослідницькі фірми продають цінну інформацію, отриману в результаті глибокого аналізу бізнес-моделей компаній, діяльності, конкуренції, перспектив тощо. Позитивним аспектом аналітики є те, що сигнал був витягнутий із

необробленого джерела. Негативні сторони полягають у тому, що аналітика може бути дорогою, методологія, що використовується при її виробництві, може бути упередженою чи непрозорою, і вона ніколи не буде унікальною.

2.2.4 Альтернативні дані

Альтернативні дані, створені окремими особами (соціальні мережі, новини, веб-пошук тощо), бізнес-процесами (транзакції, корпоративні дані, державні установи тощо) та датчиками (супутники, геолокація, погода, відеоспостереження та т. д.). Деякі популярні супутникові зображення або відеопотоки включають моніторинг танкерів, рух транспорту в тунелях або парковки.
Що дійсно характеризує альтернативні дані, то це те, що це первинна інформація, тобто інформація, яка не потрапила в інші джерела. До того, як Exxon Mobile повідомила про збільшення прибутку, до того, як ринкова ціна злетіла, до того, як аналітики написали свої коментарі до своїх останніх документів, до цього були рухи танкерів і бурильників, а також рух трубопроводів. Вони відбулися кілька місяців дотри, як ці дії було відображено в інших типах даних. Двома проблемними аспектами альтернативних даних є їхня вартість та конфіденційність. Весь цей шпигунський прийом коштує дорого, і слідчий комітет може надати певні претензії про такому виду діяльності.

Отже, згідно описаних вище типів фінансових даних, аналітичні дані є найкращим типом фінансових даних, так як вони містять у собі достатньо глибокі метрики фундаментальних та ринкових даних. Фундаментальний аналіз та технічний аналіз є основними інструментами для формування такого типу даних. У наступному пункті - ці два принципи аналізу буде розглянуто набагато глибше.

2.3 Методи інтелектуального аналізу фінансових даних

Інтелектуальний аналіз даних - це процес вилучення та виявлення закономірностей у великих наборах даних з використанням методів на стику машинного навчання, статистики. Інтелектуальний аналіз даних — це міждисциплінарна область комп'ютерних наук та статистики, загальною метою якої є вилучення інформації (за допомогою інтелектуальних методів) з набору даних та перетворення інформації на зрозумілу структуру для подальшого використання. Інтелектуальний аналіз даних - це етап аналізу процесу "виявлення знань у базах даних", або KDD. Крім етапу необробленого аналізу, він також включає аспекти бази даних та управління даними, попередню обробку даних, розгляд моделей і висновків, показники інтересу, міркування складності, постобробку виявлених структур, візуалізацію та онлайн-оновлення.

Термін «інтелектуальний аналіз даних» є неправильним, оскільки метою є вилучення шаблонів та знань з великих обсягів даних, а не вилучення (вилучення) самих даних. Це також є модним словом і часто застосовується до будь-якої форми великомасштабної обробки даних або інформації (збирання, вилучення, зберігання, аналіз та статистика), а також будь-якого застосування комп'ютерної системи підтримки прийняття рішень, включаючи штучний інтелект (наприклад, машинне навчання).) та бізнес-аналітики. Часто більш підходящими є загальні терміни (великомасштабний) аналіз даних та аналітика, або, коли йдеться про реальні методи, штучний інтелект та машинне навчання.

Фактична задача інтелектуального аналізу даних - це напівавтоматичний або автоматичний аналіз великих обсягів даних для отримання раніше невідомих цікавих шаблонів, таких як групи записів даних (кластерний аналіз), незвичайні записи (виявлення аномалій) та залежності (аналіз правил асоціації, послідовне вилучення патернів). Зазвичай це пов'язано з використанням методів баз даних, таких як просторові індекси. Ці шаблони можна розглядати як свого роду зведення вхідних даних і використовувати в подальшому аналізі або, наприклад, в машинному навчанні і прогнозній аналітиці. Наприклад, на етапі інтелектуального аналізу даних можна визначити кілька груп даних, які можна використовувати для отримання більш точних результатів прогнозування за допомогою системи підтримки прийняття рішень. Ні збір даних, ні підготовка даних, ні інтерпретація результатів і звітність є частиною етапу інтелектуального аналізу даних, хоча вони відносяться до загального процесу KDD як додаткові кроки.
Різниця між аналізом даних та інтелектуальним аналізом даних полягає в тому, що аналіз даних використовується для перевірки моделей та гіпотез на наборі даних, наприклад, для аналізу ефективності маркетингової кампанії незалежно від обсягу даних. Навпаки, інтелектуальний аналіз даних використовує машинне навчання та статистичні моделі для виявлення таємних чи прихованих закономірностей у великому обсязі даних.
Пов'язані терміни «вилучення даних», «вилов даних» та «відстеження даних» відносяться до використання методів інтелектуального аналізу даних для вибірки частин більшого набору даних про населення, які надто малі (або можуть бути) для того, щоб можна було зробити надійні статистичні висновки щодо достовірності будь-яких даних. виявлено закономірності. Однак ці методи можна використовувати для створення нових гіпотез для перевірки великих масивах даних.

Інтелектуальний аналіз фінансових даних в де чому відрізняється від класичних. Цьому сприяє великий рівень ентропії фінансових даних та те що науковці у сфері фін. моделювання приймають те, що фін ринок схиляється до гіпотези випадкового блукання. ВБ - це фінансова теорія, яка стверджує, що

ціни на фондовому ринку розвиваються відповідно до випадкового блукання (тому зміни цін є випадковими) і, отже, не можуть бути передбачені.

2.3.1 Проблема випадкового блукання

Випадкове блукання у фінансових ринках не дає змоги класичного застосування методів інтелектуального аналізу даних. Аналіз випадкових блукань припадає на більш глибший предмет, такий як теорія стохастичних процесів.

Якщо ринки є ефективними, то (технічний) аналіз закономірностей минулого динаміки цін для прогнозування майбутнього буде марний, оскільки будь-яка інформація, яку можна отримати з такого аналізу, вже відображена в поточних ринкових цінах. Припустимо, що учасники ринку впевнені, що наступного тижня ціни на якийсь товар зростуть вдвічі. У цьому випадку ціна не змінюватиметься поступово, повільно наближаючись до свого нового рівноважного значення, а виросте негайно і стрибкоподібно. Справді, якби ціна не змінилася негайно і стрибкоподібно, виникла б можливість здійснення вигідних арбітражних угод, і якщо ринок ефективний, такою можливістю негайно скористалися б усі. Точно так само і з курсом цінних паперів: якщо існує якась новина і що обіцяє вигоду сезонна закономірність у динаміці курсів акцій (наприклад, значне зростання курсів перед Різдвом), учасники ринку почнуть набавляти ціни задовго до Різдва, щоб не залишилося ніяких невикористаних можливостей для отримання арбітражної прибутку. Переконливо довели, що якщо потоку інформації ніщо не перешкоджає і трансакційних витрат не існує, то завтрашня зміна цін на спекулятивних ринках буде відображати тільки завтрашні «новини» і не буде залежати від сьогоднішніх змін цін.
Але «новини» тому й новини, що вони непередбачувані, і тому виникає в результаті їх зміну цін теж буде непередбачуваним і випадковим. Термін «випадкове блукання» зазвичай використовується у фінансовій літературі для характеристики таких рядів цін, де всі зміни являють собою випадкові відхилення від попередніх цін. Таким чином, «випадкове блукання» цін означає, що їхні майбутні зміни ніяк не пов'язані з минулими змінами. (Більш суворе визначення формулюється так: модель випадкових блукань припускає, що доходи від інвестицій в цінні папери не мають серійної кореляції і розподілу їх ймовірностей інваріантні в часі.

Отже, з даного матеріалу про випадкове блукання та з матеріалу про типи фінансових даних, можна зробити висновок, що простий часовий ряд зміни ціни фінансового інструменту не несе за собою ніякої інформативності, а якщо і несе то потребує обробці на макро рівні, аналізуючи десятки терабайт інформації. Що до аналітичних даних, які вже є підготовленні та несуть у собі якусь інформативність, що до прийняття рішень, було прийнято рішення використовувати лише їх, для того щоб отримувати максимальну оцінку аналізу

показників для даної системи. В аналітичний тип даних - входять фінансові новини, фундаментальні новини компаній.

2.3.2 Сентиментальний аналіз фінансових даних

Сентиментальний аналіз фінансових ринків в де-чому відрізняється від класичного аналізу емоційних забарвлень тексту наприклад. Щоб перейти більш глибше до фінансових даних, потрібно розглянути що загалом представляє у собі сентиментальний аналіз.

Аналіз емоцій (або аналіз думок) — це метод обробки природного мови (НЛП), який використовується для визначення того, є даними позитивними, негативними або нейтральними. Аналіз емоцій часто виконується на текстових даних, щоб, наприклад, допомогти компаніям відстежувати ставлення до бренду та продукту в відгуках клієнтів та потребувати клієнтів.

Аналіз емоцій зосереджується на полярності тексту (позитивний, негативний, нейтральний), але він також виходить за межі полярності, щоб виявити конкретні почуття та емоції (злий, радісний, сумний тощо), терміновість (терміново, не терміново) і навіть наміри (зацікавлені). v. не зацікавлений). Залежно від того, що потрібно інтерпретувати, можна визначити та налаштувати категорії відповідно до потреб у аналізі настроїв.

Так як у аналітичному типі фінансових даних, який буде використовуватися у подальшому прийнятті рішень, існують лише фінансові новини, які найкраще підходять під задачу оцінювання через сентиментальний аналіз фінансових новин.

Аналіз фінансових новин є складним завданням через спеціальну мову та відсутність розмічених даних у цій галузі. Моделі загального призначення недостатньо ефективні через спеціалізовану мову, яка використовується у фінансовому контексті. Можна припустити, що попередньо навчені мовні моделі можуть допомогти вирішити цю проблему, тому що вони вимагають менше розмічених прикладів і можуть бути навчені на предметно-орієнтованих корпусах.

Основна різниця між класичним методом є те що, у випадку з фінансовими новинами сентиментальний аналіз не тільки оцінює фон новини, але і вираховує, як дана фінансова новина вплине на майбутню поведінку даного фін. інструмента.

2.4 Оцінка фінансових даних на основі сентиментального аналізу

Дана задача входить до класу задач обробки природньої мови. Саме Обробка Природньої мови є основним інтрументом при аналізу новин(тексту). Слід зазначити що, сентиментальний аналіз - це один зі способів обробки природньої мови. Обробляти тексти можна через класичні інтелектуальні алгоритми. Особливість таких задач полягає в тому, що будь якому інтелектуальному алгоритму на вхід подається Марковський ланцюг зі слів. Тобто Марковський ланцюг описує зв’язки між словами своїми станами.

На сьогоднішній момент, найкращими алгоритмами для обробки природньої мови є алгоритми глибинного навчання. Вже декілька років поспіль, глибокі нейронні мережі б’ють усі рекорди по аналізу тексту.

Глибине навчання (также відомо, як глибоко структуроване навчання) є частиною більш широкого роду методів машинного навчання, заснованих на штучних нейронних наборах з репрезентативним навчанням. Навчанн може бути контрольованим, напівконтрольованим або неконтрольованим.

Архітектура моделі глибиного навчання, такі як глибокі нейронні мережі, глибокі мережі переконання, глибоке навчання з підкріпленням, рекуррентні нейронні мережі та згорткові нейронні мережі, застосовуються в таких областях, як комп'ютерний зір, розпізнавання речей, обробка природного мови, машинний переклад, біоінформатика, розробка лікарів, медицина. аналіз зображень, наука про клімат, перевірка матеріалів та програми настільних ігор, у яких вони давали неймовірні результати.

Штучні нейронні мережі (ШНМ) були надихнуті обробкою інформації та розподіленими комунікаційними вузлами в біологічних системах. ШНМ мають різні відмінності від біологічного мозку. Зокрема, штучні нейронні мережі, зазвичай, статичні і символічні, тоді як біологічний мозок більшості живих організмів динамічний (пластичний).

Прикметник «глибокий» у глибокому навчанні відноситься до використання кількох шарів у мережі. Ранні роботи показали, що лінійний персептрон не може бути універсальним класифікатором, але мережа з неполіноміальною функцією активації з одним прихованим шаром необмеженої ширини - може. Глибоке навчання - це сучасна варіація, пов'язана з необмеженим числом шарів обмеженого розміру, що допускає практичне застосування та оптимізовану реалізацію, зберігаючи при цьому теоретичну універсальність у м'яких умовах. У глибокому навчанні шарам також дозволяється бути гетерогенними і сильно

відхилятися від біологічно обґрунтованих моделей заради ефективності, навченості та зрозумілості.

Сенс застосування глибинного навчання для обробки природних мов у тому, що глибинні нейронні мережі виконують роботу, реалізації якої у протягом прийнятного проміжку часу треба було застосовувати десятки і навіть сотні команд професійних лінгвістів. Традиційні нейронні мережі не можуть ухвалювати поточні рішення на основі своїх попередніх суджень. Багато завдань, вирішуваних при машинної обробці природних мов, вимагає поетапного аналізу даних з урахуванням попередніх результатів. Нейронна мережа повинна «читати» речення слово за словом, «осмислюючи» його значення, виходячи з контексту. Рекурентні нейронні мережі містять зворотні зв'язки і дозволяють короткочасно зберігати інформацію, завдяки чому вони найкраще підходять для обробки послідовностей слів і символів, якою є пропозиція природної мови.

Рекурентні нейронні мережі (англ. Recurrent Neural Networks, RNN) – це мережі, що містять зворотні зв’язки і дозволяють зберігати інформацію (рис. 6)

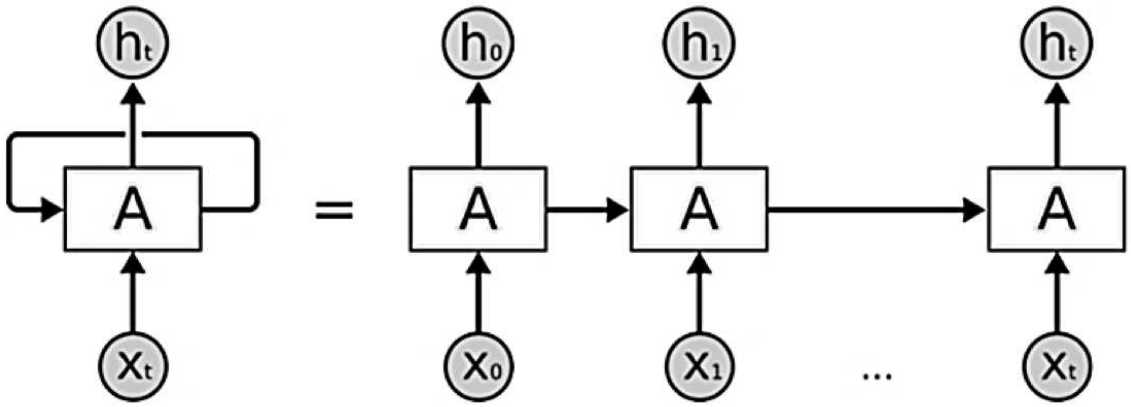

Рис. 6. Приклад RNN

На схемі вище фрагмент нейронної мережі A набуває вхідного значення xt і повертає значення ht. Наявність зворотного зв'язку дозволяє передавати інформацію від одного кроку навчання до іншого.

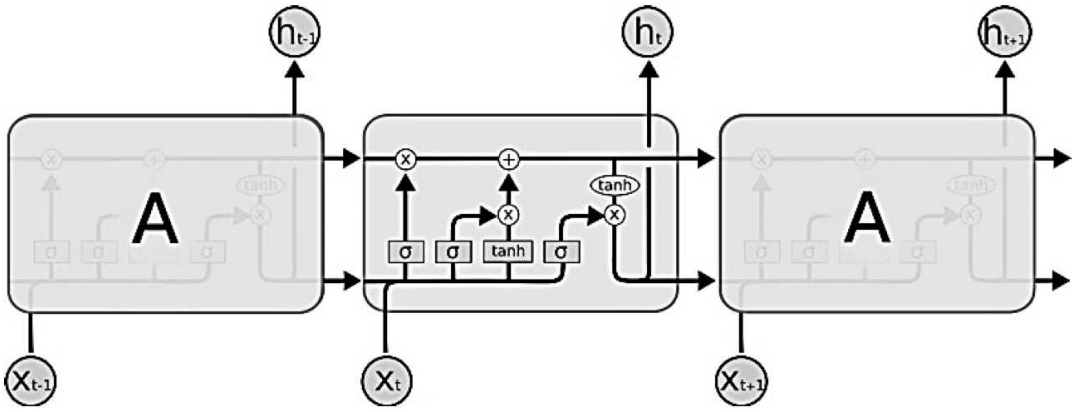

Рис. 7. Розгорнутий LSTM модуль

Одним з різновидів RNN є LSTM-мережі. LSTM (Long Short Term Memory). RNN здатні до навчання довгострокових залежностей. LSTM-мережі складаються з частин, що повторюються. Кожен такий елемент містить чотири шари і відрізняється тим, що має осередок довгої короткочасної пам'яті (рис. 7).

Ідея Sequence-to-sequence, полягає в тому, щоб використовувати одну LSTM-мережу для читання вхідної послідовності крок за кроком, щоб отримати векторний простір даних фіксованої розмірності, а потім використовувати іншу LSTM-мережу для формування вихідної послідовності цього вектора (рис. 8). Можливість LSTM-мереж успішно вивчати дані з тривалими залежностями робить їх найкращим вибором для вирішення задач, у яких як вхідна, і вихідна інформація представляються у вигляді послідовностей деяких елементів (наприклад, букв, слів, речень) тобто. - морківський ланцюг.

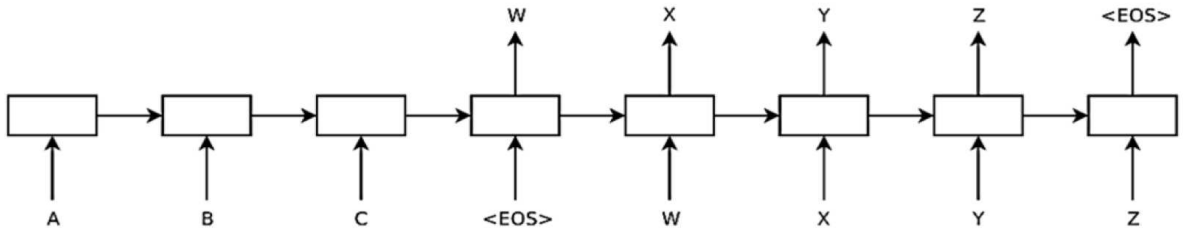

Рис. 8. Принцип послідовність до послідовності

Саме концепція sequence-to-sequence дозволяє отримати формат даних для задач емоційного забарвлення фінансових даних.

Сімейство рекурентних нейронних мереж є доволі серйозним інструментом, але без належного корпусу даних, який де коли може складатись із 10 терабайт даних, модель не буде видавати точну та правильну оцінку щодо емоційного

забарвлення фінансових даних. В даному випадку, на допомогу приходить принцип донавчання нейронної мережі, який буде розглянутий у наступному пункті.

2.4.2 Принцип донавчання нейронної мережі

Донавчання — це процес у глибокому навчанні, зосереджений на зберіганні знань, отриманих при вирішенні однієї задачі, і застосуванню їх до іншої пов'язаної задачі. Наприклад, знання, отримані під час навчання розпізнаванню емоцій, можна застосувати під час спроби розпізнавати сентимент фінансової новини. Цей напрямок досліджень має певне відношення до тривалої історії психологічної літератури щодо передачі навчання, хоча практичні зв'язки між цими двома напрямками обмежені.

Визначення донавчання задають у термінах областей визначення та задач. Область визначення D складається з простору ознак X та відособленого розподілу ймовірності P(X), де X = { x1, x2, …, xn } Є X. Для даної конкретної області визначення D = {X, P(X)} задача складається з двох складових: простору міток Y та цільової функції f(x). Функцію f використовують для передбачення відповідної мітки f(x) нового примірника x. Цієї задачі, позначуваної через T = {Y, f(x)}, навчаються з тренувальних даних, що складаються з пар {xi, yi}, де xi Є X та yi Є Y. Донавчання має на меті покращити, або добавити результати попереднього навчання завдяки новим тренувальним даним.

Найкращим прикладом навчання сентиментального аналізу є глибока нейронна мережа компанії Google. BERT – це модель машинного навчання, що базується на трансформері, для попереднього тренування обробки природної мови (NLP). BERT була створена та опублікована у 2018 році Джейкобом Девліним та його колегами з Google. Станом на 2022 рік Google застосовує BERT, щоб краще розуміти пошуки користувачів.

Оригінальна англомовна модель BERT поставляється у двох заздалегідь натренованих варіантах:
(1) модель BERTBASE, нейромережева архітектура з 12 шарами, 768 прихованими, 12 головами, 110 мільйонами параметрів, та (2) модель BERTLARGE, нейромережева архітектура з 24 шарами, 1024 прихованими, 16 головами, 340 мільйонами параметрів; обидві треновані на BooksCorpus з 800 мільйонами слів та однією з версій англійської Вікіпедії з 2500 мільйонами слів. BERT виграла нагороду за найкращу роботу на щорічній конференції

Північноамериканського відділення Асоціації з обчислювальної лінгвістики 2019 року.

Щоб перейти докладніше до моделі BERT та її навчання, потрібно розглянути, що таке трансформер і як трансформер відрізняється від звичайних рекурентних нейронних мереж, описаних вище.

Трансформер – це модель глибинного навчання, що переймає механізм уваги, окремо зважуючи значимість кожної частини даних входу. Її використовують переважно у галузі обробки природної мови (ОПМ). Як і рекурентні нейронні мережі (РНМ), трансформери призначені для обробки послідовних даних входу, таких як обробка природної мови, як переклад і сентиментальний аналіз тексту. Однак, на відміну від РНМ, трансформерам не потрібно обробляти дані послідовно. Швидше, механізм уваги забезпечує контекст для будь-якого положення в послідовності входу. Наприклад, якщо дані входу є реченням природної мови, трансформеру не потрібно обробляти його початок, перш ніж приступити до обробки кінця. Він визначає контекст, який надає значення кожному слову у цій послідовності. Ця властивість дозволяє набагато пришвидшити процес навчання, ніж РНМ, і тому знижує тривалість тренування.

Також важливою відмінністю трансформера від РНМ є два шари. Кодувальник та декодувальник. Кожен кодувальник складається з двох головних складових: механізму самоповаги та нейронної мережі прямого поширення. Механізм самоповаги приймає кодування входу з попереднього кодувальника і зважує їхню релевантність один одному, щоб згенерувати кодування виходу. Нейронна мережа прямого поширення здійснює подальшу обробку кожного кодування виходу окремо. Ці кодування виходу потім передають наступному кодувальнику як його вхід, як і декодувальнику. Перший кодувальник отримує як вхід не кодування, а інформацію про положення та вкладення послідовності входу. Інформація про положення необхідна трансформеру, щоб використовувати порядок послідовності, оскільки жодна друга частина трансформера не використовується. Кожен декодувальник складається з трьох головних складових: механізму самоуваги, механізму уваги над кодуванням та нейронної мережі прямого поширення. Декодувальник працює подібно до кодувальника, крім вставленого додаткового механізму уваги, який натомість отримує релевантну інформацію з кодувань, породжених кодувальниками. Подібно до першого кодувальника, перший декодувальник бере в якості свого входу не кодування, а інформацію про положення та вкладення послідовності виходу. Трансформер не повинен використовувати для передбачення виходу поточний або майбутній вихід, тому послідовність виходу повинна бути прихована, щоб запобігти цьому зворотному потоку інформації. За крайнім декодувальником слід завершальне лінійне перетворення і шар softmax, щоб робити можливість ймовірну приналежність виходу над словником.

Згідно з проаналізованого, сімейство моделей трансформер є набагато гнучкішим у порівнянні з класичними рекурентними мережами. Трансформер перемагає, як і по швидкості, так і по моделюванню.

2.5 Висновки до розділу

У даному розділі проаналізовано проектні рішення даної системи. Оглянуто існуючі рішення та прийнято рішення, максимально спростити інформацію, яка буде надходити кінцевому користувачеві у вигляді аналітики фінансового інструмента.

По-друге, розглянуто усі типи фінансових даних, враховані усі за і проти та прийнято рішення використовувати лише аналітичний тип фінансових даних на вхід у систему для оцінок метрик фінансових інструментів.

По-третє, обрано сентементальний аналіз, як найкращий інструмент для оцінки фінансового інструмента, який відноситься до обробки природньої мови. Прийнято рішення використовувати глибинне навчання, в якості основного алгоритма для оцінки вхідних даних системи.

Розділ 3. Розробка системи

3.1 Python, як інструмент для реалізації системи

Один із способів оцінки популярності мови програмування – індекс TIOBE. Він розраховується на основі кількості пошукових запитів у Google та інших пошукових системах. Враховуються запити, що включають назву програмних мов. Згідно з індексом TIOBE, у серпні 2021 року Python посідає друге місце у списку найпопулярніших мов програмування. Він випереджає JavaScript, PHP, Swift та інші поширені мови, поступаючись лише C. У рейтингу GitHub Octoverse за 2020 рік Python посідає друге місце, поступаючись лише JavaScript. Рейтинг Github Octoverse відбиває популярність мови серед користувачів GitHub. У рейтингу RedMonk "пайтон" також посідає друге місце. Співзасновник RedMonk Джеймс Гавернер зазначає, що Python вже став основною мовою для інтелектуального аналізу даних. Тим не менш, Гавернер не виключає, що зараз Python досяг піка популярності.

Python — одна з основних мов програмування, які застосовують у галузі машинного навчання та штучного інтелекту (Machine Learning та Artificial Intelligence). Наприклад, бібліотека з відкритим вихідним кодом TensorFlow, створена дослідницькою командою Google Brain, написана за допомогою Python. Google використовує цю бібліотеку для програмування та навчання нейронних мереж, які використовуються для вивчення штучного інтелекту.

Ще одна відома бібліотека – scikit-learn. Вона написана на Python з включеннями Cython - статично типізованого підмножини Python, що компілюється. Бібліотека scikit-learn застосовується у дослідженнях штучного інтелекту, на навчання інженерів machine learning, на управління промисловими системами.

У Python є кілька потужних та популярних бібліотек, які призначені для роботи з великими даними: аналізу, візуалізації, прогнозування тенденцій. Наприклад, бібліотека з відкритим вихідним кодом SciPy включає модулі для математичних, інженерних та наукових обчислень. Matplotlib – одна з найпопулярніших бібліотек для візуалізації даних. Бібліотека PANDAS використовується для аналізу інформації.

У веб-розробці пайтон застосовується для серверного програмування. Програмісти працюють з бекендом веб-застосунків, використовуючи нативний Python або популярні фреймворки, наприклад, Django, Pyramid або Flask. «Пайтон» однаково зручно використовувати як для створення прототипів або невеликих додатків, так і для великих проектів, що масштабуються, наприклад, порталів, веб-сервісів, інтернет-магазинів.

З описаної вище інформації, Python є найкращим інструментом для реалізації даної системи. На мові Python можна розробляти високонавантажені веб-системи, так і доповнювати їх інтелектуальними алгоритмами, маючи все під рукою.

3.2 Розробка моделі глибиного навчання

Найважливішою частиною даної системи є нейронна мережа. Без неї система не зможе оцінювати будь-який фін. інструмент. Тому, дуже важливо оприділити, який саме тип донавчання повинен бути. Як правильно обробляти дані для такого донавчання та як правильно приймати рішення про отримані результати.

3.2.1 Попередня обробка даних

Одним з основних етапів донавчання даної НН є попередня обробка даних. Фінансові новини є доволі складні дані, так як їх досить багато і у них є досить багато шуму. Зазвичай, для побудови сентиментальної моделі для аналізу фінансових даних - використовують великий набір даних по принципу “фін. заголовок статті”- “Фон”.

Для донавчання BERT був обраний набір даних, який знаходиться на популярній мережі для аналітиків даних - “kaggle”. Цей набір даних містить 40837 унікальних записів заголовків новин та їхній фон, який оцінюється у три класи: Негативний, Позитивний, Нейтральний.

Після завантаження даного набору даних, було виявлено, що набір був оброблений, перед тим як був опублікований.

Таблиця 3. Вхідні дані для донавчання

	Headline	Positive	Negative	Neutral
0	Top Five Weekend Stock Stories - Jan. 20	0.027219	0.085570	0.887211
1	Top Research Reports for McDonald's, CVS Healt...	0.047733	0.021920	0.930347
2	Turtle Beach (HEAR) to Report Q1 Earnings: Wha...	0.041581	0.065403	0.893016
3	Bullish Quintet	0.072750	0.046990	0.880260
4	Agilent Technologies (A) Earnings Expected to ...	0.220927	0.013274	0.765799

3.2.2 Імплементація нейронної мережі

Імплементацію нейронної мережі часто розбивають на декілька об’єктно-орієнтованих класів. Клас для тренувальної та тестової вибірки, клас для навчання нейронної мережі, клас для валідації та подальшого використання.

Клас для вибірок - створюється для того щоб можна було 2D формат даних перевести у тензор, тобто у ND, де N простір ознак у вибірці. Також у даному класі добавляють декілька методів щодо попередньої обробки даних. На прикладі системи у цей клас у конструкторі ініціалізує токенайзер слів, для подальшої обробки НН.

Клас для тренування - містить лише метод для тренування та оптимізації гіперпараметрів моделі. Конструктор даного класу містить усі потрібні гіперпараметри моделі. Після завершення навчання, метод оптимізації починає оптимізовувати дані гіперпараметри, максимізуючи точність нейронної мережі.

Параметри конфігурації оптимізаційного алгоритма:

1. learning_rate=0.00001 – швидкість навчання або розмір кроку. Пропорція, в якій ваги оновлюються.
2. beta_1=0.6858488 – експоненційна оцінка швидкості розпаду для першого моменту .
3. 3. beta_2=0.94444 – експоненційна оцінка швидкості розпаду для другого моменту.
4. 4. epsilon=1e-09 – дуже мале число, близьке до нуля, щоб запобігти ділення на нуль у реалізації.

Клас валідації потрібен для перевірки результатів на “свіжих” даних, які не попадали у тренувальну вибірку, та для оцінки моделі.

Таблиця 4. Результати на критеріях якості

	precision	recall	f1-score
0	0.72	0.87	0.78
1	0.78	0.89	0.83
2	0.92	0.80	0.85
accuracy			0.83
macro avg	0.80	0.85	0.82
weighted avg	0.84	0.83	0.83

Таблиця 4 показує базові метрики для задач класифікації. У таблиці видно, що дисбалансу метрик немає, середнє зважуване говорить про те, що усі метрики лежать у одному інтевалі та практично одинаково описують дану точність та задачу класифікації.

3.2.3 Оцінка результатів та формування аналітики

Важливо розуміти, що дана точність у 83-85% не прямо описує аналіз новини та не дає цілої картини про прийняття якогось рішення. Якщо наприклад, на вхід подати новину Z та отримати результат 0.83493 ймовірність, що дана новина належить негативному класу, то це не означає, що потрібно приймати рішення про продаж фінансового актива, оскільки це не гарантує, що дана новина погано вплине на подальшу зміну ціни фін. Інструмента.

Таблиця 5. Інсайдерські дані про купівлі/продажі

	Ticker	Owner	Relationship	Date	Transaction	Cost	#Shares	Value ($)	#Shares Total	SEC Form 4
0	NRIM	MARUSHACK JOSEPH	Director	May 31	Buy	41.68	605	25214.0	1580	Jun 01 03:09 PM
1	KLIC	Wong Nelson MunPun	Senior Vice President	Jun 01	Sale	53.78	5000	268883.0	166167	Jun 02 10:12 AM
2	MORN	Mansueto Joseph D	Executive Chairman	Jun 01	Sale	258.94	2072	536526.0	12480590	Jun 02 12:47 PM
3	POSH	Tung Hans	Director	May 27	Sale	11.53	10900	125670.0	126643	Jun 01 05:42 PM
4	SITM	Assaderaghi Fariborz	See Remarks	May 27	Sale	213.53	171	36514.0	122303	Jun 01 08:42 PM

Для того щоб описати повний глибинний аналіз ринку через аналітичний тип інформації, потрібно не лише оператись на результат оцінки нейронної мережі, а і водити процеси фільтрації, які будуть мінімізувати шанс помилитися.

Таблиця 6. Фундаментальні показники на прикладі “Airbnb”

0	Index	Market Cap	Income	Sales	Book/sh	Cash/sh	Dividend	Dividend %	Employees	Optionable	Shortable	Recom
1	-	77.18B	801.40M	6.61B	7.46	14.47	-	-	6132	Yes	Yes	2.50
2	P/E	Forward P/E	PEG	P/S	P/B	P/C	P/FCF	Quick Ratio	Current Ratio	Debt/Eq	LT Debt/Eq	SMA20
3	102.59	47.80	1.60	11.67	16.06	8.28	27.95	1.60	1.60	0.00	0.42	3.47%
4	EPS (ttm)	EPS next Y	EPS next Q	EPS this Y	EPS next Y	EPS next 5Y	EPS past 5Y	Sales past 5Y	Sales Q/Q	EPS Q/Q	Earnings	SMA50
5	1.17	2.51	0.45	92.40%	26.87%	64.10%	-7.30%	29.30%	70.10%	98.50%	May 03 AMC	-16.42%
6	Insider Own	Insider Trans	Inst Own	Inst Trans	ROA	ROE	ROI	Gross Margin	Oper. Margin	Profit Margin	Payout	SMA200
7	3.50%	-17.55%	62.70%	3.16%	5.40%	18.50%	0.00%	67.40%	13.20%	12.10%	0.00%	-24.98%
8	Shs Outstand	Shs Float	Short Float	Short Ratio	Target Price	52W Range	52W High	52W Low	RSI (14)	Rel Volume	Avg Volume	Volume
9	635.31M	291.02M	3.65%	1.63	183.55	103.74 - 212.58	-43.18%	16.44%	45.60	0.67	6.53M	2597791
10	Perf Week	Perf Month	Perf Quarter	Perf Half Y	Perf Year	Perf YTD	Beta	ATR	Volatility	Prev Close	Price	Change
11	4.84%	-23.27%	-16.03%	-29.35%	-20.50%	-28.03%	-	8.05	4.72% 6.82%	119.83	120.80	0.81%

З таблиць 5 та 6 можна витягнути декілька інформативних факторів та добавляти у загальну модель, у яку будуть входити оцінка нейроної мережі та оцінка фільтрів. Дана модель виглядатиме так:

$$Y = a1*b1 + a2*b2 + ... + an*bn,$$

де параметр *an* вплив фактора на результат, а параметр *bn* результат фільтації(булева змінна).До *bn* параметрів належать:

- оцінка фону новини через нейронну мережу
- наплив капіталу через інсайдерську торгівлю
- волатильність
- тренд 200 дневна ковзана середня

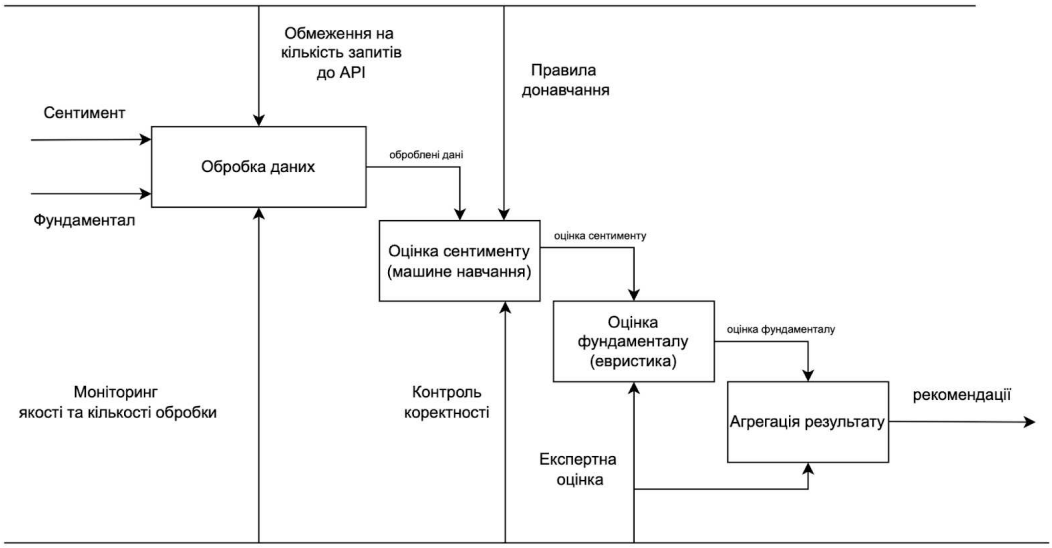

Рис. 9 Контекстна діаграма формування рекомендації

З даною фільтрацією, система здатна мінімізувати свої хибні рекомендації, що до прийняття рішення для якогось фін. Активу. Також, дуже важливою ознакою при аналізі фундаментальних новин є час. Важливо досить серйозно підійти до задачі обробки даних у реальному часі, щоб подавати завжди свіжу інформацію.

3.3 Архітектура програмного забезпечення

Архітектура програмного забезпечення – спосіб структурування програмної чи обчислювальної системи, абстракція елементів системи певної фази. Система може складатися з кількох рівнів абстракції та великою кількістю фаз роботи, кожна з яких може мати окрему програмну реалізацію та принципи проектування. Процес побудови архітектури програмного забезпечення намагається визначити як краще зробити декомпозицію системи, щоб всі

частини декомпозиції могли безперешкодно спілкуватися одна з одною. Найкраще описували процеси через неформальну нотацію.

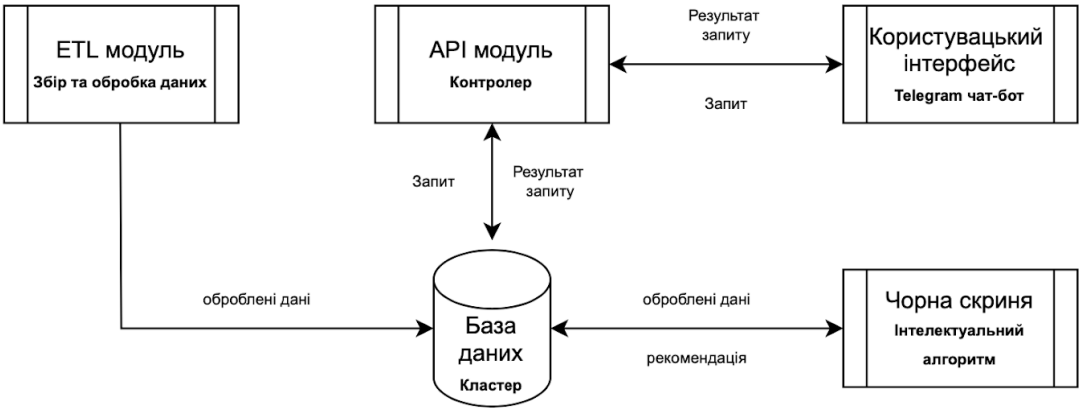

Рис. 10 Архітектура системи

На рисунку 10 представлена архітектура системи моніторингу фінансових метрик.

3.3.1 ETL модуль; збір та обробка даних

Extract, Transform, Load (ETL) або Витяг, Перетворення та Завантаження — процес, який використовується в базах даних та, особливо, у сховищах даних та у засобах Business Intelligence для забезпечення їх роботи для підтримки прийняття рішень. Він охоплює наступні етапи обробки даних:

- Виймання даних із зовнішніх джерел,
- Перетворення даних, для зберігання даних у відповідній структурі або форматі, з можливого аналізу.
- Завантаження даних у кінцеву базу даних. Більш точно, це може бути вітрина даних або сховище даних.

Поняття ETL може застосовуватися в процесі завантаження будь-якої бази даних. Оскільки виведення даних займає багато часу, то для скорочення загального часу обробки, поширення є одночасною роботою всіх трьох етапів ETL. Поки виймаються, процес перетворення одержує інші дані і готує їх для завантаження, щоб уникнути очікування виконання цих етапів.

В даній системі ETL процес потрібен для постійного витягу, обробки та завантаження фінансових даних у базу даних. Процес витягу робить паралельно декілька запитів на сторінки популярних фінансових ресурсів кожних 15 секунд та передає у процес перетворення. Процес перетворення відповідає за очистку даних та формування єдиної структури даних(Таблиця 7).

Таблиця 7. Результат процесу перетворення у ETL модулі

	news_url	date	tickers	systemSector
0	https://www.cnbc.com/2022/06/02/options-trader...	2022-06-02 12:21:37-04:00	[LULU]	MAGNIFICENCE
1	https://seekingalpha.com/article/4516006-7-upc...	2022-06-02 10:56:09-04:00	[CNO, DCI, GLPI, NDAQ, NSP, TRV, UNTY]	INDUSTRIAL
2	https://www.youtube.com/watch?v=38gaExF99ZQ	2022-06-02 12:05:44-04:00	[HPE]	IT
3	https://investorplace.com/2022/06/6-undervalue...	2022-06-02 10:54:20-04:00	[AVT, CSCO, HPQ, KBH, ORCL, QCOM]	INDUSTRIAL
4	https://www.zacks.com/stock/news/1932931/crane...	2022-06-02 11:18:35-04:00	[CR]	INDUSTRIAL

Процес створює структуру у вигляді таблиці, та задає необхідні типи даних кожній колонці у таблиці. Результат процесу перетворення попадає у процес завантаження. Процес завантаження надсилає розподіленій базі даних дану структуру.

3.3.2 База даних; розподілений кластер

Так, як архітектура системи розбита на декілька модулів, які спілкуються через мережу, та базу даних повинні опрацьовувати незалежно декілька модулів, тому було прийнято рішення створити розподілену базу даних на обчислювальному кластері.

Станом на 2022 рік, найпопулярнішим рішення надає компанія MongoDB. MongoDB - документо орієнтована система управління базами даних, яка не вимагає опису схеми таблиць. Вважається одним із класичних прикладів NoSQL-систем, використовує JSON-подібні документи та схему бази даних. Написана мовою C++. Застосовується у веб-розробці, зокрема, в рамках JavaScript-орієнтованого стека MEAN.

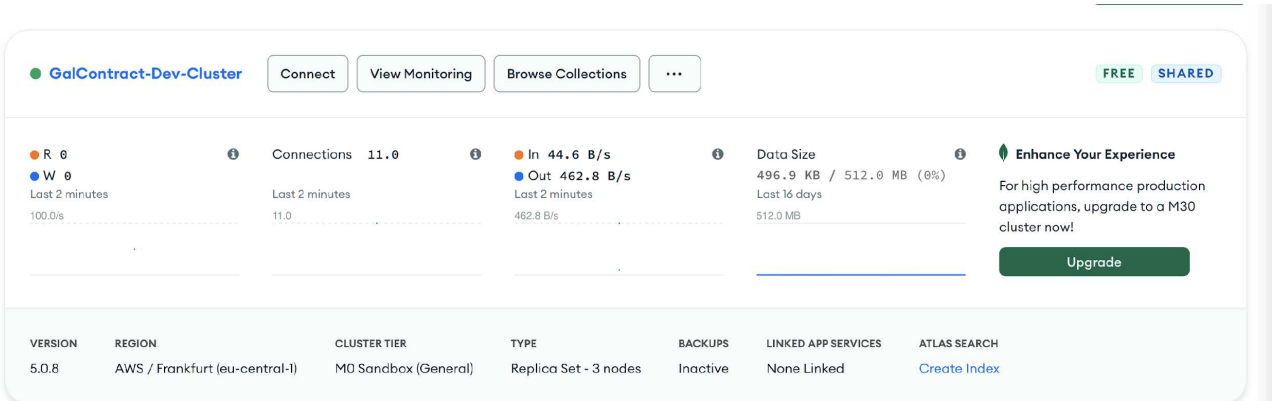

Рис. 11 Управління обчислювальним кластером

Система підтримує ad-hoc-запити: вони можуть повертати конкретні поля документів та користувацькі JavaScript-функції. Підтримується пошук за регулярними виразами. Також можна настроїти запит на повернення випадкового набору результатів.Є підтримка індексів. Система може працювати з набором реплік, тобто містити дві або більше копій даних на різних вузлах. Кожен екземпляр набору реплік може будь-якої миті виступати у ролі основної чи допоміжної репліки. Усі операції запису та читання за замовчуванням здійснюються з основною реплікою. Допоміжні репліки підтримують копію даних у актуальному стані. У разі коли основна репліка дає збій, набір реплік проводить вибір, яка з реплік повинна стати основною. Другорядні репліки можуть додатково бути джерелом для операцій читання. Система масштабується горизонтально, використовуючи техніку сегментування об'єктів баз даних - розподіл їх частин на різних вузлах кластера. Адміністратор вибирає ключ сегментування, який визначає, за яким критерієм дані будуть рознесені вузлами (залежно від значень хеша ключа сегментування)(Рис. 11). Завдяки тому, що кожен вузол кластера може приймати запити забезпечується балансування навантаження.

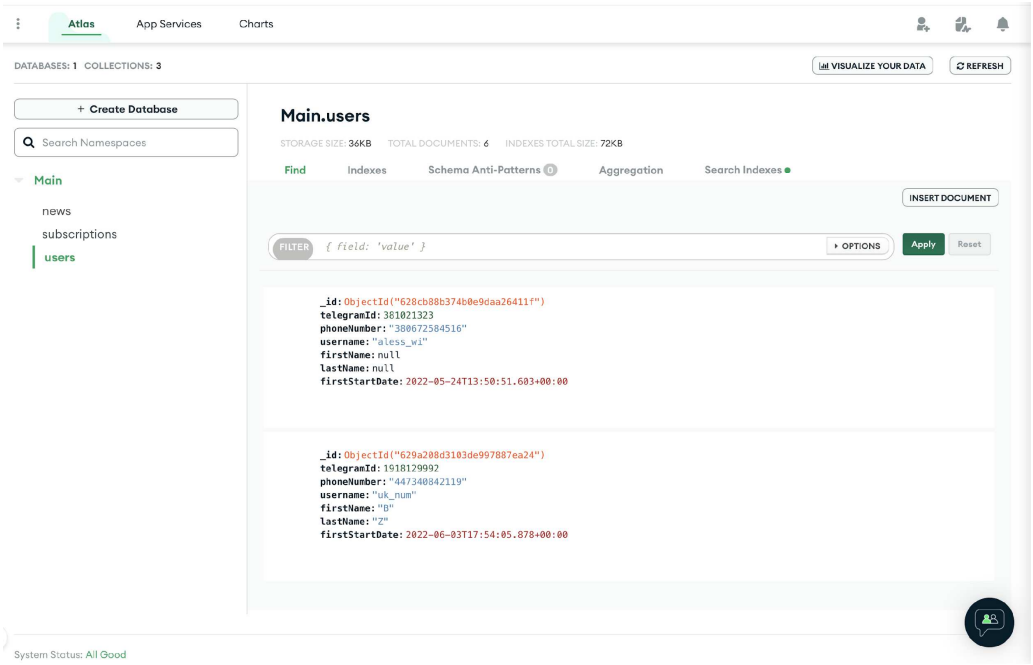

Рис. 12 Огляд колекцій у базі даних

У веб додатку MongoDB Atlas можна створити безоплатний кластер, з низькими конфігураціями. Але навіть низькі конфігурації, дозволяють працювати з даними досить швидко та надійно. У рисунку 12 продемонстровано панель керування кластером.

Найкраща особливість MongoDB - створення любих різноманітних зв’язків та логічних структур. Хоча MongoDB це NoSQL база даних, не складає жодної проблеми створити SQL-based зв’язки та структури.

До прикладу, колекція users має поле telegram_id. Для того, щоб дане поле зробити як у реляційних базах primary key, достатньо оголосити, що дане поле є індексом та воно є унікальним.

У рисунку 13 наведено приклад запису інформації про оформлення підписки користувачем. Даний запис також містить поле telegram_id (foreign key). Якщо пройтись MongoDB запитом по двом колекціям, то можна спокійно об’єднати ці дві сутності(колекції) та отримати очікуваний результат операції.

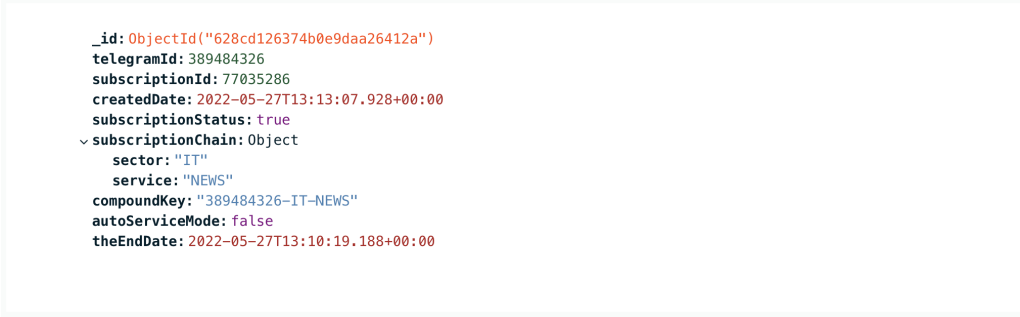

Рис. 13 Запис у колекції subscriptions

3.3.3 API модуль; контролер

Контролер є посередником між усіма розподіленими модулями у системі, окрім ETL модуля. Основна задача контролера - брокер повідомлень між базою даних, користувацьким інтерфейсом, та алгоритмами інтелектуального аналізу. Контролер послуговується архітектурі REST API та принципу проектування MVC.

REST – передача репрезентативного стану, або передача самоописуваного - архітектурний стиль взаємодії компонентів розподіленого додатка в мережі. Іншими словами, REST - це набір правил того, як програмісту організувати написання коду серверного додатка, щоб усі системи легко обмінювалися даними та програму можна було масштабувати. REST є узгодженим набором обмежень, що враховуються при проектуванні розподіленої гіпермедіа-системи. У певних випадках (інтернет-магазини, пошукові системи, інші системи, засновані на даних) це призводить до підвищення продуктивності та спрощення архітектури. У широкому сенсі компоненти в REST взаємодіють на зразок взаємодії клієнтів і серверів у Всесвітній павутині. REST є альтернативою RPC.

В інтернеті виклик віддаленої процедури може бути звичайним HTTP-запитом (зазвичай GET або POST; такий запит називають «REST-запит»), а необхідні дані передаються як параметри запиту.
Для веб-служб, побудованих з урахуванням REST (тобто не порушують обмежень, що накладаються ним), застосовують термін «RESTful».
На відміну від веб-сервісів (веб-служб) на основі SOAP, не існує офіційного стандарту для RESTful веб-API. Справа в тому, що REST є архітектурним стилем, тоді як SOAP є протоколом. Незважаючи на те, що REST не є стандартом сам по собі, більшість RESTful-реалізацій використовують такі стандарти як HTTP, URL, JSON і, рідше, XML.

Модель–вигляд–контролер (MVC, Модель–представлення–контролер, англ. Model-view-controller) — архітектурний шаблон, який використовується під час проектування та розробки програмного забезпечення.
Цей шаблон розділяє системи на трьох взаємопов'язаних частинах: модель даних, вигляд (інтерфейс користувача) та модуль керування. Застосовується відокремлення даних (вигляду) від інтерфейсу користувача (вигляду) так, щоб зміна інтерфейсу користувача мінімально вплинула на роботу із змінами моделей даних, щоб могти використовувати без змін інтерфейсу користувача(Рис. 14).

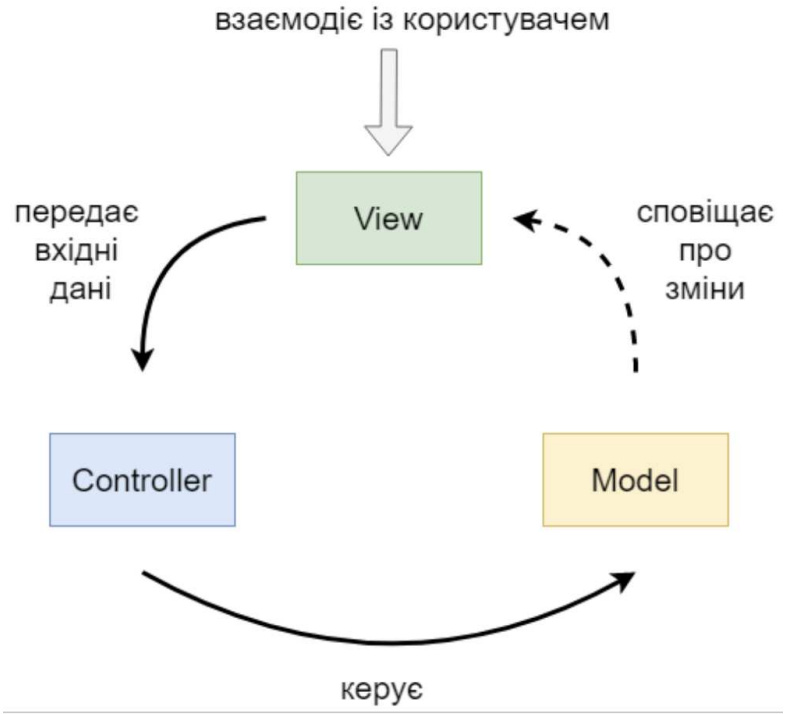

Рис. 14 Діаграма взаємодії між компонентами шаблону MVC

Отже, мета такої декомпозиційної архітектури — дизайн програмного забезпечення, який повинен посилювати подальші зміни чи розширення системи, а також надавати можливість повторного використання шаблонів окремих програм компонентів. Крім того, що використовується загально визнані конструкції у системній упорядкованості, їх структури роблять їх більш зрозумілими для зменшення складності.

3.5 Інтерфейс кінцевого користувача

Для інтерфейсу кінцевого користувача, було обрано використовувати соціальну мережу Telegram. На сьогоднішній день, дана система набирає шалених оборотів по кількості відвідування. Все більше бізнесів, пробують інтегрувати свої продукти у телеграм за допомогою чат-ботів. Чат-бот (англ. chatbot) — це програма, яка імітує реальний розмову з користувачем. Чат-боти дозволяють спілкуватися за допомогою текстових, чи аудіо-повідомлень у Telegram мережі з високоякісним комфотом та простотою.

Переваги чат-ботів для онлайн бізнесу:

1. Ефективне взаємодія з клієнтами. Чат-боти допомагають отримати як потенційних покупців, так і клієнтів, що ведуть до збільшення продажу. У відмінності від компаній, які користуються традиційними методами обслуговування, чат-боти не перегружають аудиторську інформацію, а кожен раз надають лише дані, які відповідають запитам користувачів. Тому клієнти регулярно отримують виключно релевантну інформацію. Такий підхід допомагає краще підтримувати зацікавленість аудиторії завдяки автоматизованому ланцюгу повідомлень.

2. Економність. Власникам бізнесу необхідно платити співробітникам за сервісне обслуговування клієнтів. А з збільшенням компанії ростуть і витрати. Чат-боти — це одноразова інвестиція, яка допомагає брендам зменшити витрати на персонал. Компанії можуть легко інтегрувати чат-ботів, щоб відповідати на прості питання потенційних покупців і передавати більш складні питання менеджерам з обслуговування.

3. Відстеження доставки контенту та даних про споживачів. Чат-боти збирають відгуки клієнтів, які допомагають брендам покращити свої послуги та оптимізувати сторінки з низьким рівнем конверсії. Понад те, виходячи з виконаних клієнтом дій можна сегментувати аудиторію.

4. Генерація, кваліфікація та вирощування лідів. Чат-боти отримують інформацію про користувачів, яка дозволяє персоналізувати розсилку

повідомлень клієнтам на різних етапах вирви продажів. Боти можуть ставити релевантні питання, генерувати ліди, переконувати потенційних покупців. Крім того, вони допомагають компаніям знаходити некваліфікованих лідів за допомогою ключових показників результативності. Такий підхід позбавляє взаємодії з лідами, які просто забирають час.

5. Легкість в експлуатації. Чат-боти допомагають компаніям якісно обслуговувати клієнтів кількома мовами. Це дає змогу розширювати діяльність бренду на нових ринках.

3.6 Реалізація чат-боту

Чат-боти переважно використовують штучний інтелект для спілкування з користувачами, тому надають релевантний контент та актуальні пропозиції. Вони працюють на основі набору інструкцій або використовують машинне навчання. Функціонал чат-бота, який працює на основі вказівок, досить обмежений. Найчастіше він призначений відповіді на фіксовані питання. Таким чином, якщо людина ставить питання не так, як передбачено програмою, бот не зможе відповісти.

Але для коректної роботи, чат-бот має бути запрограмований під конкретні стани спілкування з користувачем. До приклади, функціонал для допомоги, може бути реалізований через алгоритми машинного навчання, но процес вибору фін. Ісінструмента, підписка та рекомендація - усе це є детерміновані стани, які запрограмовані де яким ланцюгом станів.

В електроніці, та й у програмуванні, у найпростішому випадку усі мають справу з так званими «перемикачами». Це де-яка примітивна абстракція яка не має власної пам'яті: на вхід подається аргумент, а на вихід якийсь значення. Вихідне значення завжди залежить лише від вхідного. А якщо потрібно, щоб наступне значення функції залежало від попереднього? Від кількох попередніх? В такому випадку потрібно переходити до алгоритмів з деякою пам'яттю. Це і буде детермінованим автоматом. Вихідне значення в автоматі залежить від значення на вході та поточного стану автомата.

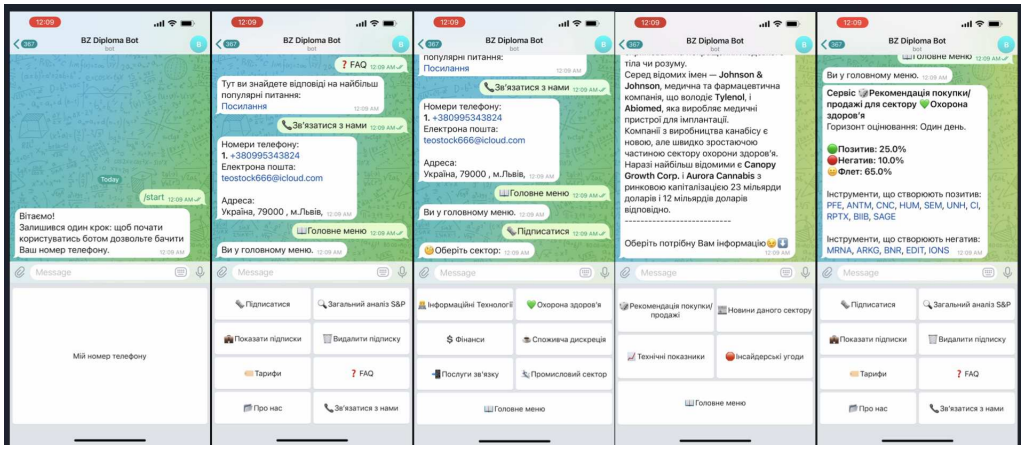

Рис. 15 Приклад роботи чат-боту на базі оформлення підписки

Кінцевий автомат відповідно тому називається, що його кількість внутрішніх станів кінчене. А, напевно, найпростішим з кінцевих автоматів є детермінований: для кожного вхідного сигналу існує лише один стан, в який автомат може перейти з поточного.

Таким чином, автомат визначається наступним чином:
- початковим станом
- вхідним алфавітом (набір з всіх можливих вхідних сигналів)
- множиною станів
- правилами переходів

Для даної системи описано об'єктно-орієнтований клас, в якому міститься реалізація детермінованого автомату, відповідно до вхідних та вихідних даних.

На рисунку 15 добре видно роботу детермінованого автомату. Кожна наступна дія залежить від попередньої.

Головне меню чат-бота складається з восьми кнопок, три з яких є динамічними. Кнопка "підписатися" перекидає користувача в меню для вибору фін. інструмента. У чат-боті фін. інструменти поділені на предметні сектори по стандарту композитного індекса S&P500.

Після попадання у меню вибора сектору, користувач зустрічається з шести секторами. При нажатті на кожен сектор, користувач може ознайомитися з даним сектором, зрозуміти, які найпопулярніші акції входять у нього, та перейти до ще одного меню - меню вибора сервісу для даного сектору.

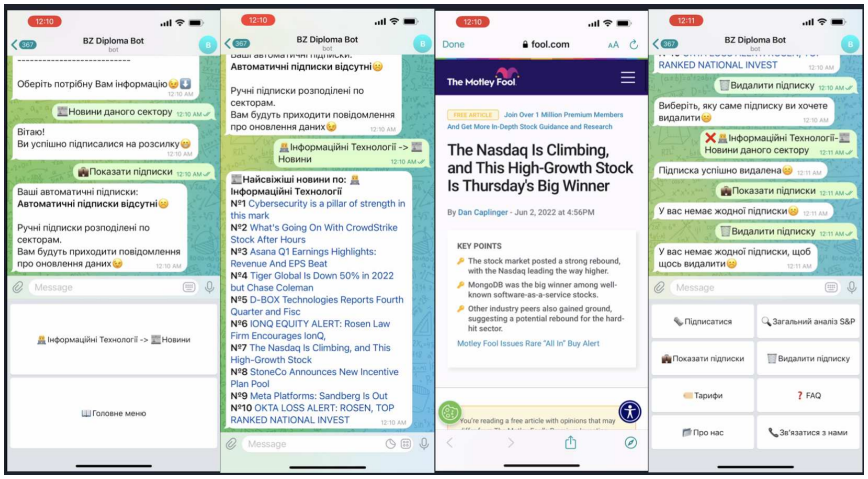

Рис. 17 Приклад роботи чат-боту на базі додаткового сервісу

Основна кнопка у даному меню - “Рекомендації покупки/продажі”. У даній кнопці міститься уся логіка, на яку припадає увесь акцент у даній роботі.

При успішному підписані, користувачу буде приходити рекомендація та свого роду звіт про зарашню картину по даному сектору. У повідомленні міститься новиний фон за допомогою сентиментального аналізу, де описано три показника : позитив, негатив, та “флет”(нейтральна позиція на ринку, пояснює про утримання до будь яких операцій на ринку). Також, у повідомленні міститься розподіл по фін. інструментам, які створюють позитив та негатив на ринку. При нажатті на фін. актив, користувач попадає на всесвітньо відомий сервіс “yahoo.finance” де він може детальніше ознайомитися з активом.

Чат-бот не обмежується лише функціоналом рекомендацій. До прикладу, користувача може цікавити лише новини, без оцінки алгоритмів. У меню вибора сервісу, користувач може обрати сервіс “новини даного сектору”. Ця підписка є свого роду ручною, щоб отримати новини по даному сектору, потрібно самому нажати на кнопку “мої підписки”, та обрати що саме потрібно вивести. Так, як новини отримуються та обробляються, фактично щохвилини, було прийнято рішення реалізувати саме таким чином даний сервіс, щоб уникати спаму, та негативних відгуків від клієнтів даної системи.

3.7 Висновки до розділу

У даному розділі було розглянуто проектну реалізацію системи. Створено декомпозицію системи та розбито на декілька модулів. Розглянуто побудову кожного модуля, та запропоновано найактуальнішу реалізацію усіх модулів за допомогою визнаних шаблонів проектування, та проектних рішень.

По-друге, навчено алгоритм глибинного навчання на базі фінансових новин, та розроблено процес фільтрації для мінімізації хибних тверджень моделі. Реалізовано архітектуру для підтримки моделі глибинного навчання.

По-третє, обрано чат-бот, як найкращий графічний інтерфейс для кінцевого користувача. Через високу популярність соціальної мережі Telegram та просту та зрозумілу форму відображення інформації для кінцевого користувача.

Висновки

У даній роботі було розроблено систему моніторингу фінансових метрик, яка успішно обробляє фінансові дані у реальному часі, паралельно оцінює фінансові дані за допомогою сентиментального аналізу і методів глибинного навчання та надає рекомендації по обраному фінансовому інструменту(сектору) за допомогою чат-боту, який служить кінцевим користувацьким інтерфейсом.

У першому розділі було описано повну предметну область. Обрано фондовий ринок, як найкращий фінансовий інструмент для фізичної особи в Україні. В ході цього було поставлено задачу розробити дану систему, щоб забезпечити громадян якісною та цінною інформацією, з якою можна приймати подальші стратегічні рішення, що до інвестицій у фондовий ринок. Розглянуто спосіб колаборації даної системи з компаніями, які надають брокерські послуги для фондових ринків.

У другому розділі було описано про проектні рішення системи. Було розглянуто існуючі рішення. Згідно проаналізованих існуючих рішень, було прийнято надавати інформативні дані користувачу системи у спрощеній та більш зрозумілій формі представлення. Аналізуючи плюси та мінуси фінансових даних для аналізу, обрано аналітичний тип даних, який найкраще підходить для оцінки фінансового інструменту. Розглянуто архітектуру глибинної нейронної мережі сентиментального аналізу, яка підходить під аналітичний тип даних.

У третьому розділі було описано про програмну розробку системи. Обрано Python, як найкращий інструмент для реалізації даної системи. Донавченно нейронну мережу на фінансових новинах та отримано результат 83% точності. Добавлено процеси фільтрації, для мінімізації хибних результатів моделі глибинного навчання. Розроблено архітектуру програмного забезпечення, з дотриманням усіх норм та шаблонів сучасного проектування програмного забезпечення. Створений зручний та простий у користуванні чат-бот на базі соціальної мережі Telegram.

Списки використаних джерел

1. У що вкласти у 2021 році?; Економічна правда: https://www.epravda.com.ua/publications/2021/01/21/670205/
2. Інвестиційний банк; Вікіпедія: https://uk.wikipedia.org/wiki/Інвестиційний_банк
3. Фондовий ринок; Вікіпедія: https://uk.wikipedia.org/wiki/Фондовий_ринок
4. Способи захисту прав інвесторів; Шостий апеляційний адміністративний суд: https://6aas.gov.ua/ua/proekty/articles/h/1504-sposobi-zakhistu-prav-investoriv-teoritichni-ta-praktichni-pitannya.html
5. Платформи для торгівлі або торгові термінали; top10stockbroker: https://top10stockbroker.com/trading-platforms/
6. Особливості у машиному навчані для фінансів, Розділ 2 Фінансові структури даних: Marcos López de Prado
7. Приклади використання інтелектуальних алгоритмів для фінансових даних; Stephen Langdell, PhD, Numerical Algorithms Group: https://www.nag.com/industryarticles/dminfinancialapps.pdf
8. Випадкове блукання; Вікіпедія: https://uk.wikipedia.org/wiki/Випадкове_блукання
9. Сентементальний аналіз; Вікіпедія: https://uk.wikipedia.org/wiki/Аналіз_тональності_тексту
10. Модель глибиного навчання BERT; Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova: https://doi.org/10.48550/arXiv.1810.04805
11. Приклади донавчання для моделей глибиного навчання: machine-learning-mastery: https://machinelearningmastery.com/transfer-learning-for-deep-learning/
12. Опис донавчання BERT для фінансових даних; Dogu Araci: https://arxiv.org/pdf/1908.10063.pdf
13. Топ мов програмування для інтлектуального аналізу; datacamp: https://www.datacamp.com/blog/top-programming-languages-for-data-scientists-in-2022
14. Імплементація донавчання BERT через Pytorch; medium: https://medium.com/geekculture/implement-bert-using-pytorch-40e3068639e6
15. ETL процес; Вікіпедія: https://uk.wikipedia.org/wiki/ETL
16. Найкращі бази даних 2021 року; MongoDB: https://www.mongodb.com/nosql-explained/best-nosql-database
17. Набори даних про фінансові новини; Kaggle: https://www.kaggle.com/search?q=financial+news
18. Модель-вид-контролер; Вікіпедія: https://uk.wikipedia.org/wiki/Модель-вид-контролер

19. 5 причин чому чат-бот кращий за веб застосунок; medium: https://medium.com/hijiffy/5-reasons-why-a-chatbot-is-better-than-a-website-75477cee7341

Додатки

Додаток А, збір, обробка та оцінка фін даних.

```
import time

from json import (
    loads,
    dumps,
    JSONDecodeError,
)

from serverless.inputs import validate
from loguru import logger

GENERAL_HEADERS = \
    {
        "Content-Type": "application/json"
    }


def execute(event, context):
    root = None

    output = \
        {
            'body':
                {
                    'timestamp': int(time.time())
                },

            'code': 200
        }

    if not isinstance(event, dict):
        output['body']['result'] = "Event variable have to be dict type."
        output['code'] = 400
```

```
elif 'body' not in event:
    output['body']['result'] = "Event variable must have 'body' field"
    output['code'] = 400

elif not isinstance(event['body'], dict):
    logger.warning('Decoding body field...')
    try:
        _body = loads(event['body'])

    except JSONDecodeError as decode_err:
        logger.error(f"Can't decode body field. {decode_err}")
        output['body']['result'] = "Can't decode 'body' field"
        output['code'] = 400

    else:
        data = validate(**_body)

        if 'result' in data:
            output['body'].update(data)

        else:
            if data['event'] == 'insider':
                import serverless.insider.etl as root

            elif data['event'] == 'news':
                import serverless.news.etl as root

            extracted = root.extract()
            transformed = root.transform(extracted)
            root.load(transformed)

            output['body']['result'] = 'Success'
return \
    {
        "statusCode": output['code'],
        "body": dumps(output['body']),
        "headers": GENERAL_HEADERS
    }
```

```
from pydantic import (
    BaseModel,
    ValidationError,
    validator,
)

from serverless.config import EVENTS


class ServerlessInput(BaseModel):
    event: str

    @validator('event')
    def check_event(cls, value):
        if value not in EVENTS:
            raise ValueError(f'Invalid event name. Can be only: {EVENTS}')

        return value


def validate(**kwargs) -> dict:
    try:
        data = ServerlessInput(**kwargs).dict()

    except ValidationError as val_error:
        return \
            {
                "result": val_error.errors()
            }

    return data


EVENTS = [
    'insider'
]
```

```
    ETL_RULES = {
       "insider":
          {
             "extract":
                {
                   "url": "https://finviz.com/insidertrading.ashx",
                },

             "transform":
                {
                   "processingEngine": "pandas",
                },

             "load":
                {
                   "collection": "Name"
                }
          }
    }
```

```
import json
import pytest

from serverless import handler


@pytest.fixture()
def main_apigw_event():
    """ Generates API GW Event just with body field """

    return \
        {
            "body": '{"event": "insidr"}',
        }


@pytest.fixture()
def invalid_apigw_event():
    """ Generates Invalid API GW Event just with body field """

    return \
        {
```

```
        "body": '{"1", "2"}'
    }


def test_general_input(main_apigw_event):
    response = handler.execute(main_apigw_event, None)
    data = json.loads(response["body"])
    print(data)

    assert response["statusCode"] == 200
    assert 'body' in response
    assert isinstance(data, dict)
    assert 'result' in data


def test_invalid_input(invalid_apigw_event):
    response = handler.execute(invalid_apigw_event, None)

    data = json.loads(response["body"])
    print(data)

    assert response["statusCode"] == 400
    assert 'body' in response
    assert isinstance(data, dict)
    assert 'result' in data


'''

{
    "resource": "/{proxy+}",
    "requestContext": {
    "resourceId": "123456",
    "apiId": "1234567890",
    "resourcePath": "/{proxy+}",
    "httpMethod": "POST",
    "requestId": "c6af9ac6-7b61-11e6-9a41-93e8deadbeef",
    "accountId": "123456789012",
    "identity": {
    "apiKey": "",
    "userArn": "",
    "cognitoAuthenticationType": "",
    "caller": "",
    "userAgent": "Custom User Agent String",
    "user": "",
```

```
    "cognitoIdentityPoolId": "",
    "cognitoIdentityId": "",
    "cognitoAuthenticationProvider": "",
    "sourceIp": "127.0.0.1",
    "accountId": "",
    },
    "stage": "prod",
    },
    "queryStringParameters": {"foo": "bar"},
    "headers": {
    "Via": "1.1 08f323deadbeefa7af34d5feb414ce27.cloudfront.net (CloudFront)",
    "Accept-Language": "en-US,en;q=0.8",
    "CloudFront-Is-Desktop-Viewer": "true",
    "CloudFront-Is-SmartTV-Viewer": "false",
    "CloudFront-Is-Mobile-Viewer": "false",
    "X-Forwarded-For": "127.0.0.1, 127.0.0.2",
    "CloudFront-Viewer-Country": "US",
     "Accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/
*;q=0.8",
    "Upgrade-Insecure-Requests": "1",
    "X-Forwarded-Port": "443",
    "Host": "1234567890.execute-api.us-east-1.amazonaws.com",
    "X-Forwarded-Proto": "https",
                " X - A m z - C f - I d " :   " a a a a a a a a a a e 3 V Y Q b 9 j d - n v C d -
de396Uhbp027Y2JvkCPNLmGJHqlaA==",
    "CloudFront-Is-Tablet-Viewer": "false",
    "Cache-Control": "max-age=0",
    "User-Agent": "Custom User Agent String",
    "CloudFront-Forwarded-Proto": "https",
    "Accept-Encoding": "gzip, deflate, sdch",
    },
    "pathParameters": {"proxy": "/examplepath"},
    "httpMethod": "POST",
    "stageVariables": {"baz": "qux"},
    "path": "/examplepath",
}

'''

import os
import json
import asyncio
import requests
import requests_async
import cloudscraper
```

```
import pandas as pd

from datetime import datetime

from functools import partial
from pydash import flatten

import multiprocessing
import multiprocess as mp

from pymongo import MongoClient

import seaborn as sns
import matplotlib.pyplot as plt

%config InlineBackend.figure_format = 'retina'
sns.set(color_codes=True, rc={'figure.figsize':(15, 10)})
sns.set_palette(sns.color_palette('muted'))

from bs4 import BeautifulSoup
from tqdm.notebook import tqdm

news_streaming = 'https://finviz.com/news.ashx'
news_source = ‘https://finviz.com/news.ashx?v=2'

def self_request(url) -> str:
  scraper = cloudscraper.create_scraper()
  response = scraper.get(url).content

  return response

def by_pandas(response) -> tuple[pd.DataFrame, pd.DataFrame]:
  parsed = pd.read_html(response)

  return parsed[-1], parsed[5]

def process(dfs: tuple[pd.DataFrame, pd.DataFrame]):
  for df in dfs:
    df.columns = ['ignore', 'time', 'title']
    df.drop(['ignore'], axis=1, inplace=True)

  return pd.concat(dfs)

for i in by_pandas(self_request(news_streaming)):
```

```
  if isinstance(i, pd.DataFrame):
    if len(i.columns) > 1:
      print(i)

%%time

url = "http://192.168.1.125:5000/score"
api_key = 123456789

runtime_now = datetime.now()

results = [requests.get(url=url, json={"api_key": api_key,
                                       "input_string": i}).json()
  for i in df['title']]

runtime_end = datetime.now()

scores = pd.DataFrame(results)

# new = []
# for i in scores['result']:
#   for j in i:


parsed_result = [
  {
     'selfIndex': ii,
     str(j['class']): float(j['score']),
  }
  for i, ii in zip(scores['result'], range(scores.shape[0])) for j in i
]

parsed = pd.DataFrame(parsed_result).fillna(0)
all_columns = list(parsed.columns[1:])
agg = {col: "max" for col in all_columns}
grouped = parsed.groupby([‘selfIndex']).agg(agg).reset_index(drop=True)

scored_df = pd.concat([df, grouped], axis=1)
scored_df.head()

scored_df['neutral'].mean() + scored_df['negative'].mean() +
scored_df['positive'].mean()

scored_df['negative'].mean()
```

```
scored_df['positive'].mean()
```

Додаток Б API модуль, контролер

```
import certifi
import time
import functools

from pymongo import MongoClient
from pymongo.errors import AutoReconnect

from loguru import logger

MAX_AUTO_RECONNECT_RETRY_COUNT = 5
MAX_FIND_ASYNC_LEN = 1000000


def reconnect(f):
    """

    :param f:
    :return:
    """

    @functools.wraps(f)
    def wrapper(*args, **kwargs):
        """

        :param args:
        :param kwargs:
        :return:
        """
        attempts = 0
        while True:
            try:
                return f(*args, **kwargs)
            except AutoReconnect as e:
                logger.warning("Handled auto-reconnect!")
                attempts += 1
                if attempts >= MAX_AUTO_RECONNECT_RETRY_COUNT:
                    raise e
                time.sleep(0.3)

    return wrapper
```

```
class AbstractRepository:
    """
        Pass
    """
    def __init__(self,
                 db_name: str = None,
                 col_name: str = None,
                 mongo_uri: str = None):

        ca = certifi.where()

        try:
                        self.client = MongoClient(mongo_uri, retryWrites=True, ssl=True,
tlsCAFile=ca)
            self.db = self.client[db_name]
            self.collection = self.db[col_name]

        except Exception as e:
            logger.error(e)

    @reconnect
    def find(self, filter_, projection):
        return list(self.collection.find(filter_, projection))

    @reconnect
    def find_one(self, filter_, projection):
        return self.collection.find_one(filter_, projection)

    @reconnect
    def insert_one(self, doc):
        return self.collection.insert_one(doc)

    @reconnect
    def insert_many(self, docs: list):
        return self.collection.insert_many(documents=docs)

    @reconnect
    def update_one(self, filter_, update, upsert=False,
                   bypass_document_validation=False,
                   collation=None, array_filters=None, hint=None,
                   session=None):

        return self.collection.update_one(filter_, update, upsert,
                                          bypass_document_validation,
```

```
                                collation, array_filters, hint,
                                session)

    @reconnect
    def delete_one(self, filter_, collation=None, hint=None, session=None):
        return list(self.collection.delete_one(filter_, collation, hint, session))

    @reconnect
    def delete_many(self, filter_, collation=None, hint=None, session=None):
        return list(self.collection.delete_many(filter_, collation, hint, session))

    @reconnect
    def aggregate(self, pipeline):
        return self.collection.aggregate(pipeline)

    @reconnect
    def get_last_record_by_sort(self, filter_, projection, sort_params):
        pipeline = [
            {
                '$match': filter_
            },
            {
                '$project': projection
            },
            {
                '$sort': sort_params
            },
            {
                '$limit': 1
            },
        ]

        cursor = self.aggregate(pipeline)

        return list(cursor)[0]

    def __del__(self):
        logger.debug("Connection closed")
        self.client.close()
import os

from common.database.abstact import AbstractRepository
from versions.sectors import SECTORS_CODES
```

```
class News(AbstractRepository):
    """
        Pass
    """

    def __init__(self):
        super().__init__(db_name='Main', col_name='news',
                         mongo_uri=os.environ["MONGO_URI"])

    def get_news(self, refs: list):
        """

        :param refs:
        """
        data = list(
            self.find(
                filter_=
                {
                    "news_url": {"$in": refs}
                },
                projection=
                {
                    "_id": 0,
                    "news_url": 1
                }
            )
        )

        return data

    def get_last_tenth(self, sector):
        pipeline = [
            {
              "$match":
                {
                   "systemSector": SECTORS_CODES[sector]
                }
            },
            {
                "$project":
                  {
                     "news_url": 1,
                     "title": 1,
                     "date": 1,
                     "_id": 0,
```

```
                },

            },
            {
                "$sort":
                    {
                        "date": -1
                    }
            },
            {
                "$limit": 10
            }
        ]

        data = list(self.aggregate(pipeline))
        return data

import os
import gc

import pymongo

import pandas as pd

from common.database.abstact import AbstractRepository

from random import randint
from datetime import datetime

from versions.sectors import SECTORS_CODES, SERVICES, AUTO_SERVICES
from common.utils.tools import en_to_ua


def sub_id(n: int) -> int:
    """
        Pass
    """

    return 1111 + int(''.join(str(randint(0, n)) for _ in range(n)))


def parse_sub(sub: str) -> dict:
    """

    :param sub:
```

```
    """
    sub = sub.replace('❌', '').split('-')
    return {"sector": sub[0], "service": sub[1]}


class Subscriptions(AbstractRepository):
    """
        Pass
    """

    def __init__(self):
        super().__init__(db_name='Main', col_name='subscriptions',
                         mongo_uri=os.environ["MONGO_URI"])

    def insert_subscription(self, sub: dict) -> bool:
        """

        :param sub:
        """

        sub_sector = SECTORS_CODES[sub['sector']]
        sub_service = SERVICES[sub['service']]
        compound = f"{sub['telegramId']}-{sub_sector}-{sub_service}"

        try:
            doc = {
                "telegramId": sub["telegramId"],
                "subscriptionId": sub_id(8),
                "createdDate": datetime.now(),
                "subscriptionStatus": True,
                "subscriptionChain":
                    {
                        "sector": sub_sector,
                        "service": sub_service
                    },
                "compoundKey": compound,
                "autoServiceMode": AUTO_SERVICES[sub_service]
            }
            self.insert_one(doc=doc)

            return True

        # noinspection PyUnresolvedReferences
        except pymongo.errors.DuplicateKeyError:
            deactivated = self.check_sub_status(compound)
```

```
        if deactivated:
            return True

        return False

    def check_sub_status(self, compound: str):
        """

        :param compound:
        """

        data = self.find_one(filter_={
            "compoundKey": compound
        }, projection={"subscriptionStatus": 1, "_id": 0})

        if not data:
            return False

        if not data['subscriptionStatus']:
            self.update_sub_status(compound)
            return True

        return False

    def update_sub_status(self, compound: str):
        """

        :param compound:
        """

        return self.update_one(
            filter_={
                "compoundKey": compound,
            },
            update={
                "$set": {
                    "subscriptionStatus": True,
                    "createdDate": datetime.now()
                }
            }
        )

    def select_subscriptions(self, telegram_id: int) -> list:
        """
```

```
    :param telegram_id:
"""

data = list(
    self.find(
        filter_=
        {
            "telegramId": telegram_id,
            "subscriptionStatus": True,
        },
        projection=
        {
            "_id": 0,
            "createdDate": 1,
            "subscriptionChain": 1,
            "autoServiceMode": 1,
        }
    )
)

if not data:
    return data

services = en_to_ua(SERVICES)
_sectors = en_to_ua(SECTORS_CODES)

parsed_services = [services[j['subscriptionChain']['service']] for j in data]
parsed_sectors = [_sectors[i['subscriptionChain']['sector']] for i in data]
dates = [i['createdDate'] for i in data]
modes = [i['autoServiceMode'] for i in data]

if len(parsed_services) != len(parsed_sectors):
    return list()

translated = [
    {
        "sector": parsed_sectors[i],
        "service": parsed_services[i],
        "autoServiceMode": modes[i],
        "date": dates[i]
    }
    for i in range(len(parsed_services))
]

del parsed_sectors, parsed_services, services, _sectors, data, dates
```

```
        gc.collect()

        return translated

    def deactivate_subscription(self, telegram_id: int, sub: str):
        """

        :param telegram_id:
        :param sub:
        """

        chain = parse_sub(sub=sub)

        return self.update_one(
            filter_={
                "telegramId": telegram_id,
                "subscriptionChain.sector": SECTORS_CODES[chain["sector"]],
                "subscriptionChain.service": SERVICES[chain["service"]]
            },
                                update={"$set": {"subscriptionStatus": False, "theEndDate":
datetime.now()}}
        )

    def get_all_uniques(self) -> pd.DataFrame:
        """
            Pass
        """

        subs = pd.DataFrame(self.find(
            filter_={
                "subscriptionStatus": True
            },
            projection={
                "telegramId": 1,
                "subscriptionChain": 1,
                "_id": 0
            }
        ))

            subs['subscriptionChain-type'] = subs['subscriptionChain'].apply(lambda x:
x['sector'])
          subs['subscriptionChain-location'] = subs['subscriptionChain'].apply(lambda x:
x['service'])
        subs = subs.drop(['subscriptionChain'], axis=1).drop_duplicates(
            ['subscriptionChain-sector', 'subscriptionChain-service'], keep='first'
```

```
        ).reset_index(drop=True)

        return subs

import os
from common.database.abstact import AbstractRepository

from datetime import datetime


class Users(AbstractRepository):
    """
        Pass
    """

    def __init__(self):
        super().__init__(db_name='Main', col_name='users',
                         mongo_uri=os.environ["MONGO_URI"])

    def get_users_id(self) -> set:
        """

        :return:
        """

        users = list(self.find(
            projection={"telegramId": 1, "_id": 0}, filter_=None
        ))

        return set([i['telegramId'] for i in users])

    def insert_new_user(self, contact: dict, username):
        """
            :return
            :doc
        """

        if 'first_name' not in contact or 'last_name' not in contact:
            contact['first_name'] = None
            contact['last_name'] = None

        doc = {
            "telegramId": contact['user_id'],
            "phoneNumber": contact['phone_number'],
            "username": username,
```

```
            "firstName": contact['first_name'],
            "lastName": contact['last_name'],
            "firstStartDate": datetime.now()
        }

        return self.insert_one(doc=doc)


BOARD_IDS = [
    389484326,
]

def en_to_ua(enum: dict):
    """

    :param enum:
    :return:
    """

    return dict([(value, key) for key, value in enum.items()])

const env = {
    // env can be here
}

module.exports = {
    apps:
        [
            {
                name: "Diploma-Bot",
                script: "main.py",
                args: "-u",
                interpreter: "python3",
                autorestart: true,
                error_file: "logs/Diploma-Bot/err.log",
                out_file: "logs/Diploma-Bot/out.log",
                time: true,
                env // can be from .env via python logic
            },
        ]
}

#jupyter serverextension enable --py jupyter_http_over_ws

#source venv/bin/activate
```

```
jupyter notebook \
  --NotebookApp.allow_origin='https://colab.research.google.com' \
  --port=8888 \
  --NotebookApp.port_retries=0 \
  —no-browser
```

Додаток В. Чат-Бот

```
import os
import types
import gc

import aiogram
from aiogram import Bot
from aiogram import Dispatcher
from aiogram import executor

from loguru import logger

from versions.demo import DemoDiplomaVersion

from common.database.users import Users
from common.database.subscrptions import Subscriptions

from dotenv import load_dotenv

load_dotenv()

bot = Bot(token=os.environ["TG_TOKEN"])
dp = Dispatcher(bot=bot)

subscription_temporary_enum = {}


@dp.message_handler(commands=["start"])
async def start(message: aiogram.types.Message) -> None:
    """

    :param message:
    """

    ui = DemoDiplomaVersion()
    all_users = Users().get_users_id()
```

```
    if message.from_user.id not in all_users:
        await bot.send_message(message.from_user.id,
                               text=ui.chain_states[message.text]['textResponse'],
                               reply_markup=ui.chain_states[message.text]['markupResponse'],
                               disable_web_page_preview=True, parse_mode='Markdown')

    else:
            await bot.send_message(message.from_user.id, "Раді бачити Вас знову!\n📖
Головне меню",
                                   reply_markup=ui.chain_states['📖Головне меню']
['markupResponse'],
                               disable_web_page_preview=True)

    del ui
    gc.collect()

    return


@dp.message_handler(content_types=['contact'])
async def contact(message: aiogram.types.Message) -> None:
    ui = DemoDiplomaVersion()

    if message.contact is not None:
        await bot.send_message(message.from_user.id,
                               'Ви успішно відправили номер телефону!\n📖Головне меню',
                               reply_markup=ui.chain_states['📖Головне меню']
['markupResponse'],
                               disable_web_page_preview=True)
                                    Users().insert_new_user(contact=dict(message.contact),
username=message.from_user.username)

    else:
        logger.error("Contact is None")
        pass  # TODO Need to send that contact is None

    logger.info(f"User {message.from_user.id} successfully send his phone number.")

    del ui
    gc.collect()

@dp.message_handler()
```

```
async def handler(message: aiogram.types.Message) -> None:
    """
    :param message:
    :return:

    """
    keyboard = None
    ui = DemoDiplomaVersion()

    try:
        response = ui.chain_states[message["text"]]
        logger.debug(response)

        if isinstance(response, dict):
            # noinspection PyTypeChecker
            if isinstance(response['textResponse'], types.LambdaType):
                text = response['textResponse'](message.from_user.id)

            else:
                text = response['textResponse']

            if 'markupResponse' in response:

                # noinspection PyTypeChecker
                if isinstance(response["markupResponse"], types.LambdaType):
                    keyboard = response['markupResponse'](message.from_user.id)

                else:
                    keyboard = response['markupResponse']

            await bot.send_message(chat_id=message.from_user.id,
                                   text=text,
                                   reply_markup=keyboard,
                                   parse_mode='Markdown',
                                   disable_web_page_preview=True)

        # noinspection PyTypeChecker
        if isinstance(response, types.LambdaType):
            answer = response(message.from_user.id)

            if 'markupResponse' in answer:
                keyboard = answer['markupResponse']

            await bot.send_message(chat_id=message.from_user.id,
                                   text=answer['textResponse'],
```

```
                        reply_markup=keyboard,
                        parse_mode='Markdown',
                        disable_web_page_preview=True)

        else:
            logger.debug("Unresolved data type in version type")

        return

    except KeyError as unresolved:
        logger.debug(f"Unresolved: {unresolved}")

        if message["text"] in ui.categories:
            logger.debug("User want to make subscription")
            subscription_temporary_enum['telegramId'] = message.from_user.id
            subscription_temporary_enum['sector'] = message['text']

                                    await  bot.send_message(chat_id=message.from_user.id,
text=ui.prepare_sector_description(message['text']),
                                                reply_markup=ui.sector_service_menu,
disable_web_page_preview=True,
                        parse_mode='Markdown')
            return

        elif message["text"] in ui.all_services:
            subscription_temporary_enum['service'] = message['text']
            logger.info(f"User sub: {subscription_temporary_enum}")

                                                                insert_result  =
Subscriptions().insert_subscription(sub=subscription_temporary_enum)

            if insert_result:
                text = "Вітаю!\nВи успішно підписалися на розсилку😁"

            else:
                text = "У вас є така підписка🤪"

            await bot.send_message(chat_id=message.from_user.id, text=text,
                                        reply_markup=ui.chain_states['📖Головне меню']
['markupResponse'],
                        disable_web_page_preview=True)

            return
```

```
        elif '❌' in message["text"]:
            Subscriptions().deactivate_subscription(
                telegram_id=message.from_user.id, sub=message['text']
            )

                await bot.send_message(chat_id=message.from_user.id, text="Підписка успішно видалена☹️",
                                       reply_markup=ui.chain_states['📖Головне меню']['markupResponse'],
                                disable_web_page_preview=True)

            return

        elif '->' in message["text"]:

            if '📰' in message["text"]:
                                    await  bot.send_message(chat_id=message.from_user.id, text=ui.send_news(message["text"]),
                                       reply_markup=ui.chain_states['📖Головне меню']['markupResponse'],
                                    parse_mode="Markdown",
                                    disable_web_page_preview=True)

            elif '📈' in message["text"]:
                print("Send Indicztorr")

            else:
                await bot.send_message(chat_id=message.from_user.id, text="Я не можу вас зрозуміти☹️",
                                       reply_markup=ui.chain_states['📖Головне меню']['markupResponse'],
                                    disable_web_page_preview=True)

    del ui
    gc.collect()


if __name__ == '__main__':
    executor.start_polling(dispatcher=dp)

from aiogram.types import ReplyKeyboardMarkup, KeyboardButton
```

```
from common.utils.tools import en_to_ua

from common.database.subscrptions import Subscriptions
from common.database.news import News

from versions.sectors import SECTORS, SERVICES, SIMPLIFIED_SERVICES


class DemoDiplomaVersion:
    """
        data types class for responses on ukrainian
    """

    general_buttons = {
        "mainMenu": KeyboardButton('📖Головне меню')
    }

    sign_up_text = 'Ви успішно відправили номер телефону!\n' \
                   '📖Головне меню'

    general_menu = ReplyKeyboardMarkup(row_width=2).add(
        KeyboardButton("📡Підписатися"),
        KeyboardButton("🔍Загальний аналіз S&P"),
        KeyboardButton("💼Показати підписки"),
        KeyboardButton("🗑Видалити підписку"),
        KeyboardButton("🏷Тарифи"),
        KeyboardButton(" ❓ FAQ"),
        KeyboardButton("📂Про нас"),
        KeyboardButton("📞Зв’язатися з нами")
    )

    sector_service_menu = ReplyKeyboardMarkup(row_width=2).add(
        KeyboardButton("🎲Рекомендація покупки/продажі"),
        KeyboardButton("📰Новини даного сектору"),
        # KeyboardButton("📈Технічні показники"),
        # KeyboardButton("🛑Інсайдерські угоди"),
        KeyboardButton("📖Головне меню"),
```

```
    )

    all_sectors = SECTORS
    all_services = SERVICES

    categories = list(SECTORS.keys())
    contact_us = "Залишити заявку на дзвінок"

    chain_states = \
        {
            "/start":
                {
                    "textResponse":
                      "Вітаємо!\n"
                                "Залишився один крок: щоб почати користуватись ботом дозвольте бачити Ваш номер телефону.",

                    "markupResponse": ReplyKeyboardMarkup(
                        row_width=1
                              ).add(KeyboardButton(request_contact=True, text='Мій номер телефону'))
                },

            "📑Підписатися":
                {
                    "textResponse": "🧐Оберіть сектор:",
                    "markupResponse": ReplyKeyboardMarkup(
                        row_width=2
                    ).add(*[{"text": i} for i in list(categories)]).add('📖Головне меню')
                },

            "📖Головне меню":
                {
                    "textResponse":
                      "Ви у головному меню.",
                    "markupResponse": general_menu
                },

            " ❓ FAQ":
                {
                     "textResponse": "Тут ви знайдете відповіді на найбільш популярні питання:\n"
```

```
                "[Посилання](https://www.google.com/)",
        },

    "🏷️Тарифи":
        {
            "textResponse":
                "Інформація про наші тарифи:\n"
                "[Посилання](https://www.google.com/)",
        },

    "📂Про нас":
        {
            "textResponse":
                "[Посилання](https://www.google.com/)"
        },

    "📞Зв’язатися з нами":
        {
            "textResponse":
                "Номери телефону:\n"
                "*1.* [+380995343824](tel:+380995343824)\n"
                "Електрона пошта:\n"
                "teostock666@icloud.com\n\n"
                "Адреса:\n"
                "Україна, 79000 , м.Львів,\n",

            "markupResponse":
                ReplyKeyboardMarkup().add(contact_us).add('📖Головне меню')
        },

    "🔍Загальний аналіз S&P":
        {
            "textResponse": "Потім"  # lambda x: elastic_search(x),
        },

    "🗑️Видалити підписку": lambda x: delete_subscription(x),

                "💼Показати підписки": lambda x: show_subscriptions(x),   #
"textResponse":

    }
```

```
@staticmethod
    def prepare_sector_description(sector: str):
        header = "*Інформація про сектор:*\n--------------------------\n"
        sector_description = SECTORS[sector]['description']
            footer = "\n--------------------------\n\nОберіть потрібну Вам інформацію😉
⬇️"

        return header + sector_description + footer

    @staticmethod
    def send_news(message: str):
        index = message.find('->')
        non_sys_sector = message[:index-1]
        last_news = News().get_last_tenth(non_sys_sector)

        message = f"📰*Найсвіжіші новини по: {non_sys_sector}*\n"
        for _unique, k in zip(last_news, range(len(last_news))):
            message += f'*№{k + 1}* [{_unique["title"][:50]}]({_unique["news_url"]})
\n'

        return message


def show_subscriptions(telegram_id):
    """

    :param telegram_id:
    """
    data = Subscriptions().select_subscriptions(telegram_id)
    auto = [i for i in data if i['autoServiceMode']]
    not_auto = [i for i in data if not i['autoServiceMode']]

    if not data:
        return \
            {
                "textResponse": "У вас немає жодної підписки☹️",
                "replayMarkup": None
            }

    text = 'Ваші автоматичні підписки:\n'

    if auto:
```

```
        for subscription, i in zip(auto, range(len(auto))):
            temp = f'----------------------\n' \
                   f'*№ {i + 1}*\n' \
                   f'Сектор: {subscription["sector"]}\n' \
                   f'Сервіс: {subscription["service"]}\n' \
                   f'Дата створення: {str(subscription["date"])[:16]}\n'

            text += temp

            text += f'----------------------\n'

    else:
        text += "*Автоматичні підписки відсутні*☹\n"

        text += "\nРучні підписки розподілені по секторам.\n" \
                "Вам будуть приходити повідомлення про оновлення даних😉\n"


    return {
        "textResponse": text,
        "markupResponse": ReplyKeyboardMarkup(row_width=2).add(
            *[{"text": i['sector'] + " -> " + SIMPLIFIED_SERVICES[i['service']]} for i in
not_auto]).add(
            '📖Головне меню'
        )
    }


def delete_subscription(telegram_id):
    """

    :param telegram_id:
    """
    data = Subscriptions().select_subscriptions(telegram_id)

    if not data:
        return \
            {
                "textResponse": "У вас немає жодної підписки, щоб щось видалити☹"
            }

    subs = [f'{i["sector"]}-{i["service"]}' for i in data]
```

```
    return \
        {
            "textResponse": "Виберіть, яку саме підписку ви хочете видалити☹️",
                "markupResponse": ReplyKeyboardMarkup(row_width=2).add(*[{"text":
"❌" + i} for i in subs]).add(
                '📖Головне меню'
            )
```