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

Інститут прикладного системного аналізу

Кафедра математичних методів системного аналізу

До захисту допущено:

Завідувач кафедри

_____ Оксана ТИМОЩУК

«___»_____20__ р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Системний аналіз і управління» спеціальності 124 «Системний аналіз»

на тему: «Методи та підходи до категоризації коментарів користувачів»

Виконала:
студентка IV курсу, групи КА-77
П’ятецька Анна Андріївна _____

Керівник:
завідувач кафедри Тимощук О. Л. _____

Консультант з економічного розділу:
доцент, к.е.н. Рощина Н. В. _____

Консультант з нормоконтролю:
доцент, к.т.н. Коваленко А. Є. _____

Рецензент:
доцент, к.т.н. Безносик О.Ю _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2021 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Інститут прикладного системного аналізу
Кафедра математичних методів системного аналізу

Рівень вищої освіти – перший (бакалаврський)
Спеціальність – 124 «Системний аналіз»
Освітньо-професійна програма «Системний аналіз і управління»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Оксана ТИМОЩУК
«25» травня 2021 р.

ЗАВДАННЯ
на дипломну роботу студенту
П'ятецькій Анні Андріївні

1. Тема роботи «Методи та підходи до категоризації коментарів користувачів», керівник роботи Тимощук Оксана Леонідівна, завідувач кафедри ММСА, затверджені наказом по університету від «26» травня 2021 р. №1344-с

2. Термін подання студентом роботи 7 червня 2021р.

3. Вихідні дані до роботи: Дані були зібрані з ресурсу IMDB та є загальнодоступними та були агреговані в великий тренувальний та набір.

4. Зміст роботи: Порівняльний аналіз класичних методів аналізу даних у вирішенні задачі класифікації текстових даних.

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо)

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Рощина Н.В., доцент кафедри ТПЕ		

7. Дата видачі завдання

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1.	Затвердження теми ДР	02.03.2021-05.03.2021	виконано
2.	Ознайомлення зі структурою БДР згідно з Положенням про державну атестацію студентів НТУУ «КПІ ім. І. Сікорського»	06.03.2021-19.03.2021	виконано
3.	Ознайомлення з ДСТУ 3008-95 та стандарти ЄСПД	20.03.2021-27.03.2021	виконано
4.	Проведення дослідження за темою БДР під керівництвом керівника	27.03.2021-03.04.2021	виконано
5.	Завершення роботи над першим варіантом частини БДР	04.04.2021-18.04.2021	виконано
6.	Проведення роботи над експериментальною частиною БДР	19.04.2021-26.04.2021	виконано
7.	Проведення роботи над програмним продуктом	27.04.2021-04.05.2021	виконано
8.	Оформлення БДР та аналіз отриманих результатів	05.05.2021-10.05.2021	виконано
9	Оформлення пояснювальної записки у цілому	11.05.2021-24.05.2021	виконано
10	Підготовка презентації для захисту	25.05.2021-30.05.2021	виконано

11	Попередній захист дипломної роботи	31.05.2021	виконано
12	Захист дипломної роботи	18.06.2020	

Студент Анна П’ЯТЕЦЬКА

Керівник Оксана ТИМОЩУК

РЕФЕРАТ

Дипломна робота: 85 с., 7 табл., 28 рис., 1 додаток, 26 джерел.

АНАЛІЗ ТЕКСТОВИХ ДАНИХ, КЛАСИФІКАЦІЯ, КОМЕНТАРІ КОРИСТУВАЧІВ, ТОНАЛЬНІСТЬ ТЕКСТУ, НЕЙРОННА МЕРЕЖА, НЕСТРУКТУРОВАНІ ДАНІ.

Тема: Методи та підходи до категоризації коментарів користувачів.

Мета роботи: дослідити методи та підходи до категоризації текстових даних, розробити систему для визначення тональності тексту за допомогою методів машинного навчання, розробити методи для зчитування та обробки даних для навчання системи та її подальшої коректної роботи.

Результатом даної роботи є програмний продукт у вигляді прикладного програмного інтерфейсу, який аналізує відгуки користувачів до медіаконтенту, класифікує їх за настроєм та відображає статистику у реальному часі.

ABSTRACT

Thesis: 85 p., 7 tabl., 28 fig., 1 appendices, 26 sources.

TEXT MINING, CLASSIFICATION, USER COMMENTS, SENTIMENT ANALYSIS, NEURAL NETWORK, UNSTRUCTURED DATA.

Theme : Methods and approaches to categorization of user comments

Objective: To explore the methods and approaches to categorizing text data, develop a system to determine the tone of the text using machine learning techniques to develop methods for reading and processing data to study the system and its work correctly.

The result of this work is a software product in the form of an application software interface that analyzes user feedback on media content, classifies them by mood and displays real-time statistics.

ЗМІСТ

ВСТУП

Коментарі користувачів - це є текстове відображення їх думок. Якщо навчитись їх аналізувати - це допоможе зрозуміти настрій відвідувачів сторінки, знайти основні характеристики в яких вони зацікавлені та багато інших критеріїв.

На основі цієї інформації буде можливо зробити висновки, та використовувати їх під час наступних стадій аналізу і обробки інформації, Для зразку, коментарі користувачів інтернету дають змогу зрозуміти їх потреби та сфери їх зацікавленості. Адже, перед тим, як купити якийсь товар чи послугу в інтернеті, у сучасного суспільства сформувалася звичка читати коментарі. Довіру до них викликає те, що їх писали такі ж люди і якщо відгуки позитивні, то шанс, що даний товар буде куплений набагато більший. Та навпаки, якщо вони у більшості негативні - шанси знижуються. Організаціям як великим, так і малим варто звернути на коментарі увагу, адже за допомогою аналізу цих даних, отриманих на їх основі можна легко виявити що хвилює їх аудиторію, та як можна покращити продукти компанії, а за допомогою цього, можна значно збільшити прибуток та створювати більш якісні товари та послуги в подальшому.

РОЗДІЛ 1
ЗАДАЧІ АНАЛІЗУ ДАНИХ У СУЧАСНОМУ ІНФОРМАЦІЙНОМУ ПРОСТОРІ

1.1 Необхідність аналізу даних

За період свого існування людство створило надзвичайно великі масиви інформації, що оточують нас всюди. Це все дані про наші пошукові запити, телефонні розмови, поведінку в соціальних мережах, діяльність будь-яких підприємств, візити до лікаря і ще багато іншого. Та у зв'язку з удосконаленням технологій запису та зберігання інформації з кожним днем кількість даних, які нас оточують невпинно зростає. Це спричиняє хаос, тому став очевидним факт того, що без аналізу потоки необроблених даних нікому не потрібні. З середини 90-х років в інформаційній галузі зростає інтерес до технологій аналізу даних**.** Щоб вирішити цю проблему існують обробки технології аналізу великих даних, що дозволяє отримати вичерпний погляд на прикладні проблеми. Data Science, або робота з даними - це наука про аналіз даних, за допомогою якої можна якісно передбачити розвиток певної галузі чи конкретної сфери. DS пов’язана з Machine Learning (машинним навчанням), Cognitive Science (когнітивною наукою) та Big Data, що є підрозділом науки про дані (DS).

Ці технології сприяють економії часу та ресурсів. Читати, впорядковувати та аналізувати величезні обсяги інформації - це обтяжливий та трудомісткий процес, який не має сенсу у 21-му столітті. До того ж людську похибку ніхто не відміняв, та за допомогою навчених моделей машинного навчання можна з набагато більшою точністю досягнути мети та знайти залежності між елементами системи, сенс в суперечливих даних,

дізнатись багато корисного про поведінку кожної людини s і багато чого іншого.

Також можна зменшити ризики, так як базуючись на інформації, яка з'являється після аналізу та яку ніхто не брав до уваги до нього можна прийняти більш виважені рішення. Наприклад, всесвітньо відома компанія Google почала співпрацювати з Центром по контролю захворювань та щоб передбачити осередки епідемії грипу у США відслідковує користувацькі пошукові запити на рахунок симптомів.

Ще однією проблемою в відсутності аналізу великих даних була відсутність зв'язку з користувачем. Компанії не мали зворотного зв'язку від чого користувачі мали менш якісні продукти. Наприклад, стрімінгова платформа Netflix після створення великої клієнтської бази формує медіаконтент, який охоплює велику частку аудиторії глядачів на основі інформації про користувацькі дані та аналітику про міжнародні звички перегляду, отримані від досвіду клієнтів. Це сприяє виправленню недоліків та уникненню помилок, яких вже припустились конкуренти.

За рахунок цього, відбувається збільшення прибутків, наприклад з джерела [1] відомо, що компанія Evolv, яка допомагає великим світовим компаніям приймати кращі рішення щодо прийняття управлінських рішень використовуючи прогнозну аналітику. Evolv аналізує більше 500 змінних, по типу таких як дані про ціни на газ, використання соціальних мереж та рівень безробіття, допомагає клієнтам, прогнозувати наскільки скоро працівник покине своє місце роботи. Наприклад, компанії Xerox, AT&T та Kelly Services використовують цей сервіс, та спостерігають заощадження своїх ресурсів в середньому на суму 10 мільйонів доларів. В період з третього кварталу 2012 року до третього кварталу 2013 року продажі послуг Evolv зросли на 150% .

1.2 Аналіз текстових даних

Щоб досягти поставленої мети, основною складовою роботи є аналіз коментарів користувачів до фільмів, а це є текстові дані. Останнім часом, прогрес з точки зору обробки даних був достатньо великий. Були розроблені аналітичні рішення та алгоритми, які використовують користувацькі дані, більшість з яких є структурованими, наприклад дати, числа, статистики, тощо. Для них було розроблено достатньо ефективних засобів аналізу та типовими є задачі такі як побудова діаграм, графіків залежностей, виділення аномалій.

Але значно більша частина текстових даних відноситься до неструктурованих. Розвиток методів запису і зберігання даних привів до бурхливого зростання обсягів збираної та аналізованої інформації. Обсяги даних настільки великі, що людина просто невзмозі проаналізувати їх самостійно, хоча необхідність проведення такого аналізу цілком очевидна, адже в цих “сирих даних” є знання, які можуть бути використані для прийняття рішень [2]**.** Відгуки або коментарі користувачів якраз відносяться до такого типу даних, для обробки яких потрібно використовувати так званий інтелектуальний аналіз даних.

Інтелектуальний аналіз даних (Data Mining) - це сучасна концепція аналізу даних, яка припускає, що дані можуть бути неточними, неповними (містити пропуски), суперечливими, різнорідними, непрямими, і при цьому мати гігантські обсяги[3]. Зокрема для обробки неструктурованих текстових даних існує термін Інтелектуальний аналіз текстових даних (Text mining). Ключовими завданнями ІАТ є:

- категоризація текстів,
- пошук інформації,

- обробка змін в колекціях текстів,
- розробка засобів представлення інформації для користувача.[4]

Його основною метою є отримання інформації з текстових документів, використовуючи ефективні методи машинного навчання та обробки природної мови.

Обробка природної мови (Natural Language Processing). — це комп'ютеризований підхід до аналізу тексту, що базується на низці теорій та наборі технологій. Ця галузь не має одного загальноприйнятого визначення, адже вона перебуває у стані постійних досліджень та розробок. Однак, існують певні аспекти, які б об'єднували усі існуючі визначення [5].

Методи обробки природної мови аналізують текст на лінгвістичному рівні, чого достатньо для значної частини задач. Аде для того, щоб визначити чи є коментар до фільму позитивним, чи негативним необхідно враховувати семантику. Для вирішення цього існує сентимент-аналіз, або аналіз тональності тексту. Він розроблений для того, щоб в автоматизованому порядку знаходити в текстах емоційно забарвлену лексику, а також емоційну оцінку автора щодо об’єктів про які йдеться в тексті.

Тональність всього тексту в цілому можна визначити як функцію (в найпростішому випадку суму) лексичних тональностей складових його одиниць і правил їх поєднання . Полярність, тобто чи зміст вираженої думки у текстовому вигляді є позитивним, нейтральним чи негативним, визначають за:

1. бінарною шкалою
2. багатосмуговою шкалою (рейтинг за 3-х або 4-бальною шкалою)
3. системою шкалювання (від -10 до 10 балів, тобто від найнегативнішого до найпозитивнішого)
4. ідентифікацією суб'єктивності/об'єктивності

На сьогоднішній день, обробка природної мови вирішує достатньо велику кількість задач аналізу текстових даних, а саме знаходить та виокремлює змістовні фрагменти тексту.

1.3 Актуальність проблеми

Час - це найцінніший ресурс у кожної людини, який невпинно йде і його завжди не вистачає. Тому питання його економії та раціонального використання стає все більш актуальним. Але наскільки б не хотілося вивчити все на світі та досягати неймовірних успіхів в роботі, люди не роботи і потребують відпочинку. Тож перед тим, як подивитись якийсь медіаконтент, суспільство часто звертає увагу на коментарі та відгуки до нього. Більше того, в основному на основі цих відгуків люди вирішують переглядати їм дану кінострічку чи ні. Але зазвичай коментарів досить велика кількість біля кожної картини. З одного боку, коментарі читають для того, щоб зекономити свій час та не витрачати близько двох годин на медіаконтент, який не прийде до вподоби, але з іншого боку, коментарів іноді настільки багато, що часу на те, щоб їх всі прочитати витрачається більше, ніж сама кінострічка б тривала. Це схоже на замкнуте коло, та з нього можна знайти вихід. В ідеальному світі було б корисно зразу бачити статистику коментарів біля кожного фільму, та знаючи скільки відсотків відгуків є позитивних, а скільки негативних зразу вирішувати чи витрачати свій час на фільм.

До того ж, з кожним днем кіноіндустрія розвивається все швидшими темпами, тож така функція має гарну перспективу. На рисунку 1.1 та 1.2 можна спостерігати, що за даними “A comprehensive analysis and survey of the theatrical and home/mobile entertainment market environment” за 2019 рік світовий ринок касових зборів усіх фільмів, випущених у кожній країні по

всьому світу, у 2019 році досяг 42,2 мільярда доларів, що на один відсоток більше, ніж у 2018 році [6].

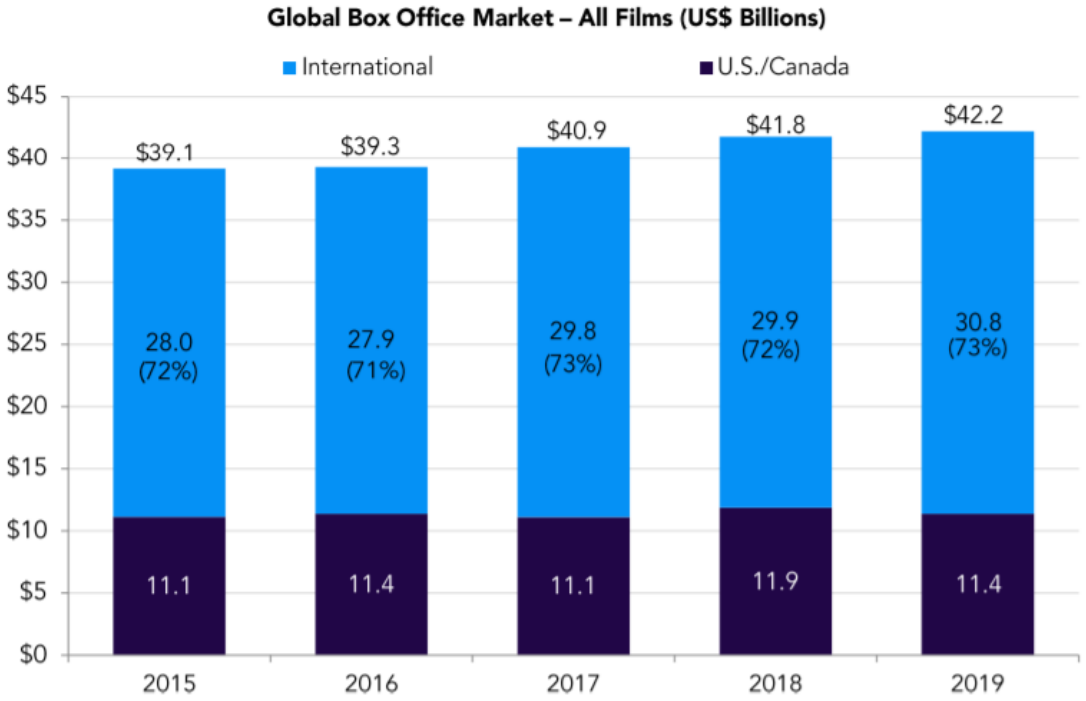


Рисунок 1.1 - Графік світового ринку касових зборів усіх фільмів за 2019-й рік

Міжнародний ринок кас (30,8 млрд. доларів) зріс на 3 відсотки, вперше перевищивши 30,0 млрд. Доларів. Ринок кас США / Канади (11,4 млрд доларів) зменшився на 4% порівняно з рекордним рівнем 2018 року. Міжнародний ринок кас становив 73% від загального ринку кас у 2019 році, що на один відсотковий пункт більше ніж у 2018 році. Міжнародний ринок кас зріс на 10% порівняно з п'ятьма роками тому. Світовий ринок каси зріс на 8% за той самий період.

	2015	2016	2017	2018	2019	% Change[6] 19 vs. 18	% Change[6] 19 vs. 15
U.S./Canada[7]	$11.1	$11.4	$11.1	$11.9	$11.4	-4%	2%
International[8]	$28.0	$27.9	$29.8	$29.9	$30.8	3%	10%
Total	$39.1	$39.3	$40.9	$41.8	$42.2	1%	8%

Рисунок 1.2 - Порівняльні дані світового ринку касових зборів усіх фільмів за 2019-й рік

А ось за даними того ж видання, світовий ринок касових зборів усіх фільмів, випущених у кожній країні по всьому світу, становив 12,0 млрд доларів у 2020 році, що на 72% менше порівняно з 2019 роком через глобальну пандемію COVID-19. На рисунку 1.3 та 1.4 можна побачити, що міжнародний ринок кас (9,8 млрд доларів) зменшився на 68%, тоді як ринок кас США і Канади (2,2 млрд доларів) зменшився на 80% порівняно з 2019 роком [7].

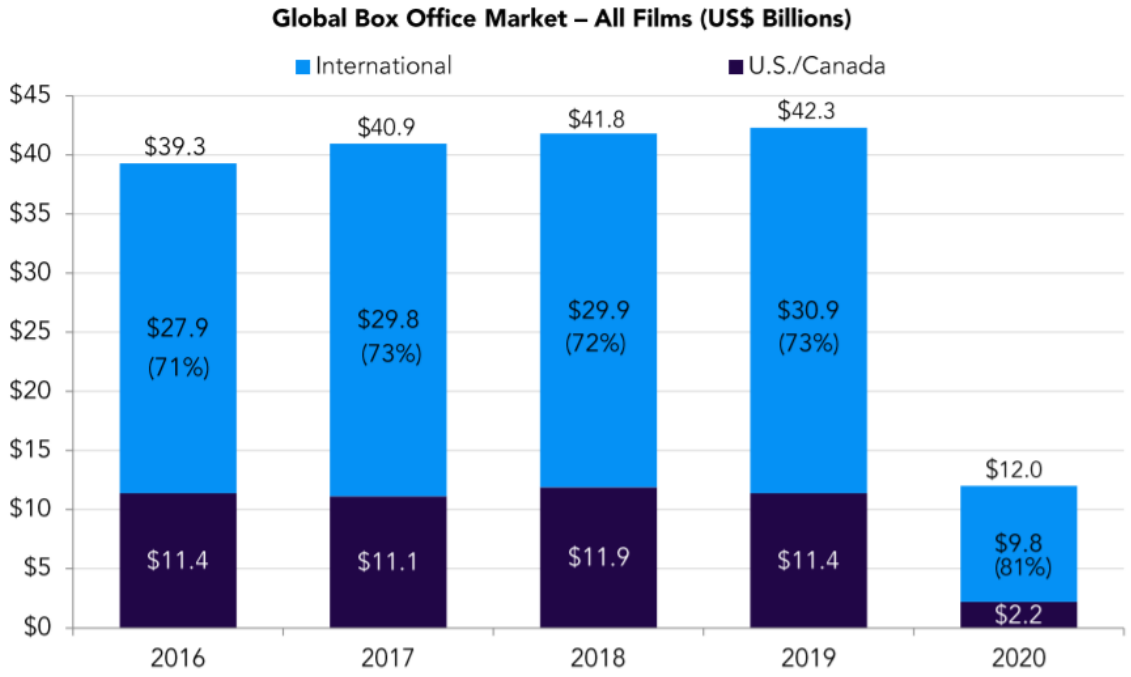


Рисунок 1.3 - Графік світового ринку касових зборів усіх фільмів за 2020-й рік

Міжнародний ринок кас становив 81% від загального ринку кас у 2020 році, до восьми відсоткових пунктів порівняно з 2019 роком.

	2016	2017	2018	2019	2020	% Change[30] 20 vs. 19	% Change[30] 20 vs. 16
U.S./Canada[31]	$11.4	$11.1	$11.9	$11.4	$2.2	-80%	-80%
International[32]	$27.9	$29.8	$29.9	$30.9	$9.8	-68%	-65%
Total	$39.3	$40.9	$41.8	$42.3	$12.0	-72%	-69%

Рисунок 1.4 - Порівняльні дані світового ринку касових зборів усіх фільмів за 2020-й рік

Але не дивлячись на те, що касові збори зазнали значних втрат, пандемія була частково компенсована збільшенням цифрового ринку на 33% у порівнянні з 2019м роком, шо видно на рисунку 1.5, який наведено нижче.

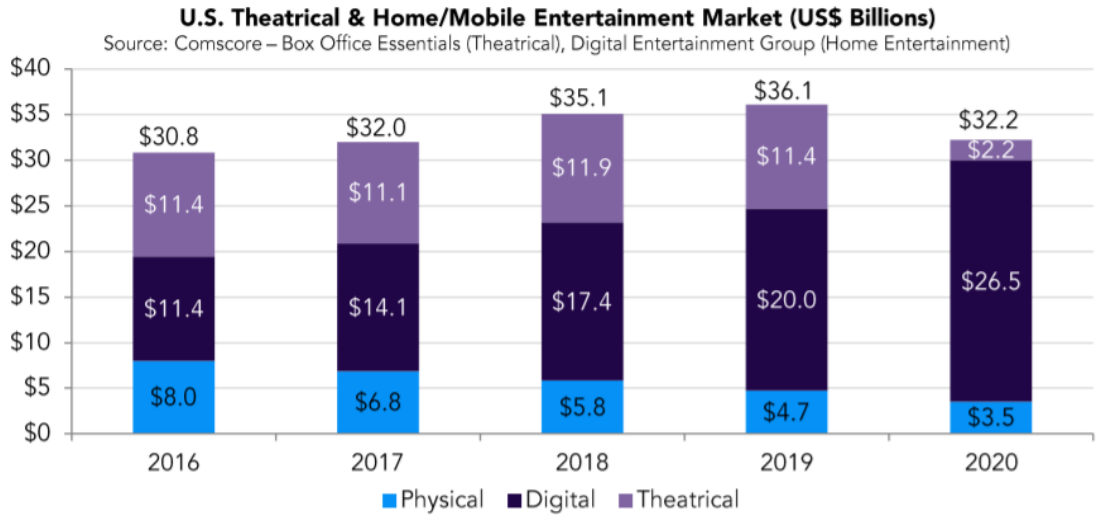


Рисунок 1.5. - Графік розвитку цифрового ринку за 2020-й рік

Виходячи з усього вищесказаного, можна зробити висновок, що не дивлячись на пандемію та зниження прибутку підприємств в інших сферах, ця галузь тримається на плаву, на відміну від ресторанної сфери, сфери туризму та багатьох інших. Крім цього, люди звертають ще більше уваги на кіноіндустрію, адже навіть в такий складний час для людей - це спосіб відволіктись та розслабитись під час буденних справ.

1.4 Висновки до розділу 1

У цьому розділі даної бакалаврської роботи була описана сутність обраної теми, проблеми та способи їх вирішення. Також було проведено аналіз актуальності теми та обраної сфери для аналізу коментарів користувачів. З огляду на все наведене вище можна зробити висновок, що аналіз великих даних здатний допомогти вирішити як локальні проблеми на рівні компаній та організацій, так і глобальні - на рівні освіти, медицини та покращення стану країн в цілому. Він допомагає оптимізувати ресурси, та збільшити прибутки, що свідчить про доцільність та актуальність даної дипломної роботи.

РОЗДІЛ 2
МЕТОДИ АНАЛІЗУ ТЕКСТОВИХ ДАНИХ У ЗАДАЧАХ КАТЕГОРИЗАЦІЇ

2.1 Загальний огляд методів класифікації настрою

У сучасних системах автоматичного визначення емоційної оцінки тексту найчастіше використовується одновимірний емоційний простір: позитивне або негативне (хороший чи поганий). Однак є вдалі випадки використання і багатовимірних просторів. Основне завдання при аналізі тональності - класифікувати полярність документа, тобто визначити, чи є думка, виражена в документі або пропозиції, позитивним, негативним або нейтральним. У більш широкому сенсі “позаполярна” класифікація тональності виражається, наприклад, такими емоційними станами, як «зло», «смуток» і «щастя».

Методи класифікації настрою, або визначення тональності тексту поділяється на такі підходи:

- за допомогою машинного навчання
- лексичний
- гібридний

Методи за допомогою машинного навчання поділяються на так звані метоти під наглядом (Supervised Learning) та без нагляду (Unsupervised Learning). В свою чергу SL розрізняють на Дерева рішень (Decision tree), Класифікатори на основі правил (Rule-based classifiers), Ймовірнісні класифікатори (Probabilistic Classifiers) та Лінійні (Linear). До лінійних належить метод опорних векторів (Support based method) та Нейронні мереж (Neural Networks). Мережі Байєса (Bayesian Network), Наївний Байєсівський

метод (Naive Bayes) та Метод максимальних ентропій (Maximum Entropy) належать до Ймовірнісних класифікаторів.

Лексичний підхід можна поділити на методи, що засновані на словниках (Dictionary-based) та корпусний підхід (Corpus-based approach). Останній розділяють на статичний та семантичний.

Гібридний підхід включає в себе два попередніх підходи та є надзвичайно поширеним , адже базується на лексиконах, які у більшості методів відіграють ключову роль. Для наглядності усього вищенаведеного було зображено блок-схему нижче на рисунку 2.1

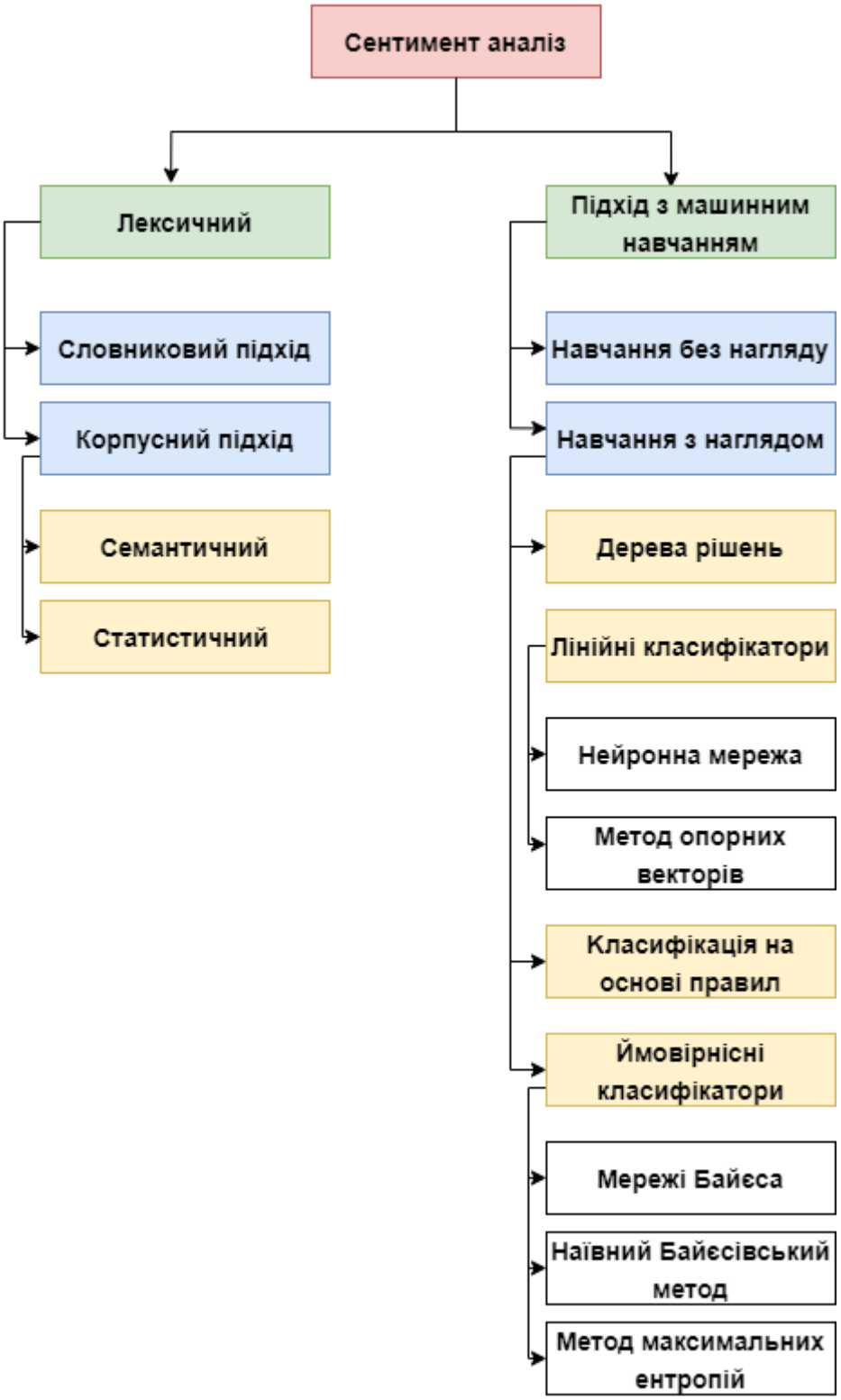


Рисунок 2.1 - Методи класифікації настрою

У наступних підрозділах описано деякі з найбільш часто використовуваних методів.

2.2 Методи на основі машинного навчання

Як вже було раніше зазначено, метод Machine Learning поділяється на методи з наглядом та без. Перші беруть за основу марковані документи у великій кількості. Методи без нагляду застосовуються, у випадку відсутності класифікованих навчальних даних, або якщо їх зовсім не багато. Далі розглянемо деякі з них детальніше.

2.2.1 Метод дерева рішень (Decision trees)

Дерева рішень розбивають дані на групи на основі значень змінних простору ознак, внаслідок чого виникає ієрархія операторів “якщо-то”, які класифікують дані [8]. Щоб визначити до якої категорії даний документ віднести, необхідно починаючи з кореня звернути увагу на вузли цього дерева. І так покроково задавати питання по типу чи значення змінної $x_i >$ порога b_i, якщо так, то відбувається перехід у правий вузол, а якщо ні - то у лівий. Наступне за чергою питання виходить з обраного вузла і так далі. Було розроблено ряд алгоритмів для автоматичної побудови дерев рішень за допомогою навчання на прикладах. Одним з них є CLS алгоритм. Відповідно до змінної, що має найбільшу класифікуючу силу, він розбиває навчальні дані на класи циклічно. Далі кожна наступна підмножина даних, що виокремлюється такою змінною, розбивається на класи з використанням змінної з найбільшою класифікуючою здатністю і т. д. Коли в підмножині залишаються елементи тільки з одного класу, розбиття завершується. Таким чином створюється дерево рішень. Щоб визначити яка змінnaa має найбільшу класифікуючу силу, використовують критерій інформаційної ваги слова в рубриці. Щоб зберегти баланс між складністю дерева, кількістю вузлів і

якістю навчання, після того, як дерево побудоване, його усікають та перетворюють різними способами. Наприклад, алгоритм C4.5 та ID3 є класичними підходами для модифікації дерев рішень. Існують програми для наочного графічного відображення дерев рішень на основі того, що результат роботи алгоритму можна інтерпретувати в наочних термінах, адже побудоване дерево добре піддається аналізу, що безперечно є значною перевагою. Але суттєвим недоліком можна вважати те, що даний алгоритм надає позитивним і негативним розгалуженням у вузлах однакову вагу. а занадто велика кількість гілок негативного характеру в описі рубрики може призводити перенавчання алгоритму класифікації.

2.2.2 Нейронна мережа (Neural Network)

Даний підхід належить до методу лінійних класифікаторів. Нейронні мережі представляють собою великий клас систем, побудова яких схожа на структуру нервової тканини з нейронами. Штучні нейронні мережі почали широко застосовуватись в галузі штучного інтелекту для аналізу даних ще з 1996 року. В одній з найпоширеніших архітектур – багатошаровому перцептроні зі зворотним поширенням помилки – імітується робота нейронів у складі ієрархічної мережі, де кожен нейрон вищого рівня з'єднаний своїми входами з виходами нейронів нижнього шару[9]. Нейрони нижчого шару приймають значення вхідних параметрів для подальшого прийняття рішення, щоб прогнозувати розвиток поточної ситуації. Значення сприймаються у вигляді сигналів, які надаються в наступний шар. Вони послаблюються або підсилюються в залежності від ваг, які приписуються міжнейронним зв’язкам. Після цього на виході з верхнього шару нейрону розраховується деяке значення, що сприймається як На виході нейрону верхнього шару з’являється значення - реакція мережі на вхідні значення параметрів. Для

застосування цієї мережі її необхідно натренувати за допомогою отриманих даних для яких вже є відомі вхідні значення параметрів та правильні відповіді. Суть в тому, щоб підібрати ваги для зв'язків між нейронами та відповідей з найбільшою близькістю мережі до відповідей. Основний недолік - необхідність великого обсягу навчальної вибірки. Інший істотний недолік полягає у тому, що навіть натренована нейронна мережа являє собою “чорну скриньку”. Знання, що зафіксовані як ваги кількох сотень міжнейронних зв'язків, зовсім не піддаються аналізу й інтерпретації людиною (відомі спроби дати інтерпретацію структурі налаштованої нейромережі виглядають непереконливими – система “ KINOsuite-PR”)[9].

Для ефективного проведення класифікації текстів необхідно визначити раціональну структуру і топологією нейронної мережі. Основні топології класифікуючих нейронних мереж це одно- і багатошаровий персептрон, нейромережевий Гаусів класифікатор, мережа Кохонена, мережа вбудованого розповсюдження, каскадна мережа [13]. Всі дані топології показують високу точність в питанні обробки як лінійних, так і нелінійних прикладів, але у випадку прийняття рішень класифікації важко формалізуються у зв’язку з природою організації нейронної мережі та являють собою нетривіальну задачу в якій слід врахувати масштабність і з обмежені обчислювальні ресурси. До прикладів систем нейронних мереж можна віднести: BrainMaker (CSS), NeuroShell (Ward Systems Group), OWL (HyperLogic).

2.2.3 Метод опорних векторів (Support Vector Machine)

Цей метод також належить до лінійних класифікаторів та у ньому навчальні документи представлені як вектори. У текстових даних іноді деякі характеристики корелюються один з одним і організовані в категорії, як правило, що можна лінійно розділити. Шляхом нелінійного відображення

зразкових даних до внутрішнього простору продукту, цей метод використовує розділення класів лінійно гіперплощиною, що може розділити позитивні і негативні приклади документів. Для класифікації текстів Торстен Джохімс вперше застосував SVM метод, який В. Вапник розробив в 1995 році. В своєму початковому вигляді алгоритм вирішував завдання приналежності об’єктів двох класів. Підхід, що запропонував В. Вапник заснований на принципі структурної мінімізації ризику для визначення того, до якого з двох заздалегідь визначених класів повинен належати текстовий зразок, що аналізується. У порівнянні з іншими методами машинного навчання, результати класифікації текстів за допомогою методу опорних векторів є одними з найкращих [10]. Проте, SVM метод вимагає достатньо великого обсягу пам’яті і витрат машинного часу на навчання, через що його масштабність є нижчою. Якщо опорних векторів буде порівняно небагато, то даний метод буде працювати ефективно і може вважатись еталоном з точки зору якості класифікації, проте якщо кількість опорних векторів зростає, через значно збільшену складність SVM метод стає малоефективний.

2.2.4 Логістична регресія (Logistic regression)

Логістична регресія є представником лінійних регресій, основною задачею яких є розбити ознаковий простір гіперплощиною і спрогнозувати значення цільового класу. Один з найпростіших і найпоширеніших методів машинного навчання, який використовується спеціально для вирішення задач класифікації. Основна ідея алгоритму - знайти зв’язок між ознаками та визначити ймовірність певного результату. Ця ймовірність використовується для визначення класу, до якого належить досліджуваний об'єкт. Для задач, у яких є лише два класи (проблема класифікації спаму), застосовується біноміальна логістична регресія, а для задач з більш ніж двома класами - поліноміальна регресія.

Найпростіша лінійна модель бінарного класифікатора має вигляд:

$$y(x) = sign(\sum_{i=1}^{n} w_i * x_i) = sign(\overline{w}\overline{x})$$

Ця модель передбачає належність об’єкта до відповідного класу, визначаючи знак лінійної комбінації, ваги моделі $\overline{w}$ та вхідний вектор ознак об’єкта. Процес навчання цієї моделі - це пошук тих самих ваг. Основна відмінність між логістичною та лінійною регресією полягає в тому, що логістична модель передбачає не клас, до якого належить об’єкт, а ймовірність належності об’єкта $\overline{x}$ до певного класу. Основне припущення логістичної регресії - чи належить вона до відповідного класу, тобто це визначення ймовірності. З її допомогою можливо побудувати прогноз, але лінійна функція може приймати будь-які значення, а ймовірність може бути лише в межах від 0 до 1. З огляду на це необхідно скористатися якоюсь нелінійною функцією, яка обмежить отримане значення. Їх існує достатньо багато, але найбільш вживаною є функція сигмоїди, що обчислюється за формулою:

$$f(x) = \frac{1}{1+e^{-x}}$$

Дана функція зображена на рисунку 2.2.

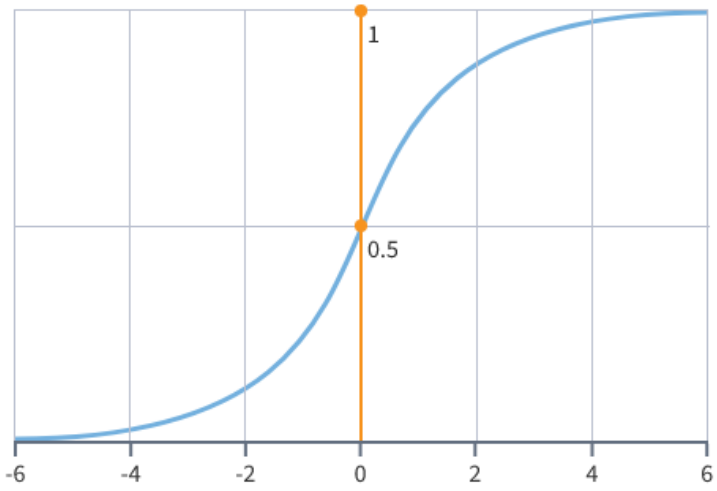


Рисунок 2.2 – Графік функції сигмоїд

Розглянемо основні переваги та недоліки методу класифікації за допомогою логістичної регресії. Головними перевагами даного методу є те, що логістична регресія проста в реалізації, інтерпретації і дуже ефективна для навчання, її достатньо легко розширити до декількох класів (поліноміальна регресія), вона швидко в порівнянні з іншими методами класифікує невідомі записи. Крім цього, логістична регресія в порівнянні з іншими методами менше схильна до перенавчання, але у випадку багатовимірного набору даних - це не гарантовано. До недоліків можна віднести обмеження, що має логістична регресія, а саме факт того, що незалежні і залежні змінні лінійно залежні. Але для коректного застосування цього методу необхідно, щоб незалежні змінні були лінійно пов'язані з логарифмічними коефіцієнтами. Також недоліком є за допомогою логістичної регресії складно отримати складні зв'язку. Нейронні мережі є більш потужним методом та меншим в написанні, що набагато актуальніше.

2.2.5 Класифікатори на основі правил (Rule-Based classifiers)

Виходячи з назви можна зрозуміти, що даний метод моделюється за допомогою набору правил. Умовно його можна розділити на дві сторони: ліва і права. Перша має вигляд набір характеристик, що визначені диз'юнктнивній нормальній формі. Друга - це позначка класу. Основною умовою є наявність деякого терму. Рідко використовується випадок відсутності терму, так як у розріджених даних вона не інформована.

Головними критеріями для формування правил є підтримка та впевненість. Етап навчання залежно від критеріїв контролює правила. Підтримка - це абсолютна кількість примірників набору навчальних даних, що відносяться до правил. Впевненість - умовна ймовірність того, що права і ліва частини виконуються. Правила прийняття рішень та дерева рішень дещо схожі адже обидва методи переносять правила на характеристичний простір. Тільки дерево рішень використовують ієрархічний підхід, тобто має строгий ієрархічний розподіл простору даних, а класифікатори на основі правил дозволяють в просторі рішень деякі перетини.

2.2.5 Метод Байєсівської класифікації (Naive Bayes)

Це порівняно простий класифікатор, що має за основу імовірнісну модель, характерним припущенням якої є незалежність компонент вектора ознак. В основному дане припущення не відповідає дійсності, тому назва Naive Bayes є досить іронічною.

В основі лежить формула Байєса з обчислення апостеріорної ймовірності гіпотез. Якщо використовувати її для задачі класифікації текстів, ймовірність того, що документ належить категорії можна обчислити за формулою:

$$P(y = c_r|E) = P(x_1 = c_p^{1}|y = c_r) \times P(x_2 = c_d^{21}|y = c_r) \times \ldots \times$$

$$\times P(x_m = c_b^{m}|y = c_r) = P(y = c_r)/P(E),$$

де y- залежна змінна, що вказує на приналежність документа до категорії c_r;

E - подія, яка характеризує наявність в текстовому документі ознак, що притаманні категорії c_r.

Наївний класифікатор Байєса має кілька властивостей, що роблять його дуже корисним на практиці, незважаючи на те, що існують вагомі припущення незалежність часто порушується. Цей метод показує високі показники швидкості роботи і досить високі показники якості класифікації. Якщо існують Його можна рекомендувати для будівництва класифікатор, коли існують суворі обмеження щодо часу підрахунку.

Даний метод є однією з популярних методик для класифікації тексту. Багато дослідників, зокрема МакКаллум і Нігам у 1998 році та Льюїс і Рінгетт у 1994 році показали, що він працює надзвичайно ефективно[11].

2.2.6 Мережі Байєса (Bayesian Network)

У 1985 році термін “байєсівська мережа” був запропонований американським вченим Джуді Перелом [12].

Байєсівські мережі – перспективний ймовірнісний інструментарій для моделювання складних ієрархічних процесів (статичних і динамічних) з невизначеностями довільного характеру. Мережа будується у вигляді спрямованого ациклічного графа, який відображає причинні зв’язки між вузлами (змінними) процесу, що досліджується. Вузли - це випадкові величини, а ребра - умовні залежностями. Мережі Байєса вважається повною

моделлю для визначення взаємозв'язків між змінними. Головне припущення класифікатора на основі Байесівських мереж є незалежність характеристик. А інше - те, що вони взаємозалежні. У обробці текстової інформації обчислювальна складність мереж Байєса достатньо дорога, тому не часто використовується.

2.2.7 Метод максимальних ентропій (Maximum Entropy)

Метод максимальних ентропій тісно пов’язаний з іншим поширеним алгоритмом машинного навчання - лінійною регресією. На етапі класифікації даний метод дуже схожий на байєсівський, адже він ґрунтується на такій же таблиці ваг і експоненційної моделі класифікації, яка використовується в Байєса для формування ймовірнісного простору з логарифмічних оцінок. Звернемо увагу, що класифікаційні індикатори нумеруються по наскрізного принципу. Індикатори, що дають одиницю як залишок від ділення на кількість класів, спрацьовують (повертають істину) тільки на перший клас, кратні двійці на другий і т.д. Такий підхід не є обов'язковим при реалізації класифікатора, але для усвідомлення теорії важливо розуміти різницю між ознакою і індикатором, а також різницею в їх нумерації.

Безпосередньо класифікація відбувається за формулою:

$$p(c|d,\lambda) = \frac{exp \sum_{i}^{n\times k} \lambda_i f_i(c,d)}{\sum_{\bar{c} \in C} exp \sum_{i}^{n\times k} \lambda_i f_i(\bar{c},d)},$$

де f_i - і-й класифікаційний ідентифікатор, що приймає значення 0 або 1;

λ_i- вага і-го класифікаційного індикатора f_i;

c - клас-гіпотеза;

C- множина всіх можливих класів;

d - класифікаційний документ.

У кожного індикатора f_i є своя вага λ_i, який описує взаємозв'язок між відповідним кваліфікаційним ознакою і класом. Чим більше вага, тим сильніше зв'язок. Таким чином, чисельник дробу описує експоненту ваг для класу-гіпотези, а знаменник нормує значення по одиниці. Найскладніша частина цієї формули - набір ваг λ, який доводиться шляхом чисельної оптимізації. Результатом цього відношення є не просто класифікаційне рішення, а значення ймовірності для заданого класу. Один з плюсів-класифікації методом максимальної ентропії полягає в тому, що вона набагато більш точно моделює імовірнісний розподіл класів. На цьому схожість закінчується і починаються відмінності. Даний класифікатор має кілька суттєвих відмінностей від байєсівського:

- він грунтується на взаємозв'язку ентропії імовірнісного розподілу і його рівномірності;
- він використовує ітеративні алгоритми підбору параметрів моделі, які складніше в реалізації і більш ресурсомісткі.

2.2.7 Метод k-найближчого сусіда (k-nearest neighbor).

В 1952 році цей метод був запропонований вперше для вирішень задач дискримінантного аналізу. Метод k-найближчого сусіда є один найбільш застосовуваних та вивчених алгоритмів в області створення автоматичних класифікаторів**.** Цей метод демонструє одні з найкращих результатів в дослідженнях, що присвячені аналізу роботи різних алгоритмів машинного навчання для задачі класифікації текстів[13].

Ідея, що лежить в основі цього алгоритму полягає в тому, щоб знаходити та розподіляти за рубриками колекції, які найбільш схожі на текст, що аналізується, спираючись на попередні знання про їх категоріальну приналежність класифікувати невідомий документ.

Документ *d* порівнюється з усіма документами навчальної вибірки щоб знайти рубрики, які релевантні даному. З навчальної вибірки для кожного документу *d* знаходиться вибірка, що дорівнює косинусу кута між векторами ознак:

$$\rho(d, e) = cos(d, e)$$

Потім з навчальної вибірки вибираються *k* документів, що є найближчими до *d* (k-параметри). Для кожної рубрики обчислюється релевантність за формулою:

$$s(c_j, d) = \sum_{e \in \{k \text{ найближчих сусідівЇ} \wedge c_j \in Rub(e)} cos(d, e)$$

Надалі задається поріг та рубрики, які мають більшу релевантність за цей поріг, вважаються відповідними *d*. Параметр *k* обирається в інтервалі від 0 до 10. Якщо категоризація монотематична. тобто документ причислений до однієї теми, то рубрика вибирається з максимальним значенням. У випадку мультитематичної категорізації класи є відповідними, коли значення є більшим за заданий напередодні поріг. У цього методу відсутня стадія навчання, що виділяє його з-поміж інших, тобто приналежність документа до рубрик визначається без побудови класифікуючої функції. Тобто, якщо

навчальну вибірку потрібно оновлювати, то це можна зробити без перенавчання класифікатора.

Цей метод може бути особливо корисним у разі, якщо навчальна вибірка часто поповнюється новими документами, та перенавчання забирає занадто часу. Основний недолік методу k-найближчого сусіда полягає в тривалості часу роботи рубрикатора на етапі класифікації [14].

Наприклад, на рисунку 2.3 видно, що зелене коло (тестовий зразок) повинен належати до одного з двох класів, або до класу синіх квадратів, або до червоних трикутників.

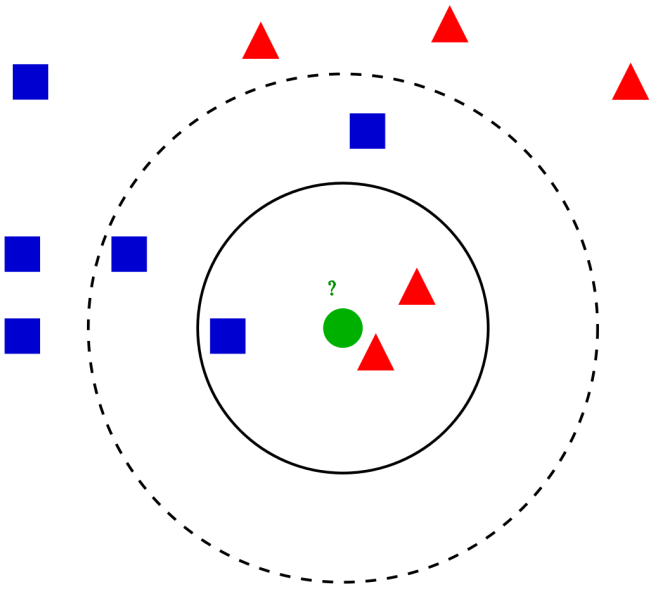


Рисунок 2.3 - Наглядний приклад застосування методу k-найближчих сусідів

Якщо k = 3, то зразок класифікується як клас червоних трикутників , так як всередині меншого кола 2 трикутника і 1 квадрат. Якщо k = 5, то він буде класифікований як клас синіх квадратів, адже маємо всередині більшого кола 3 квадрата проти 2-ох трикутників всередині більшого кола.

2.3 Лексичні методи

Лексичний метод полягає в пошуку словникової зв'язку для аналізу тексту. Підхід залежить від знаходження лексикону. Загалом існує два основних методи: словниковий та корпусний, або метод, що базується на ядрі. Останній ще поділяється на статистичні або семантичні методи, про які детальніше йдеться нижче.

2.3.1 Словниковий підхід (Dictionary approach)

Виходячи з назви, можна зрозуміти, що даний метод базується на невеликому наборі слів з лексики, що збирається вручну, а далі його обсяг збільшують шляхом пошуку в відомих словниках WorldNet або словниках з синонімами та антонімами. Всі знайдені нові слова поповнюють основний список. Далі повторюється та сама процедура. Ітерації завершуються у тому випадку, коли в сторонніх ресурсах нових слів вже не знайдено. Цей підхід має суттєвий недолік в тому, що він не здатний знаходити лексикон з контекстними орієнтаціями, а також необхідно вручну перевіряти сформований список слів задля уникнення помилок.

2.3.2 Корпусний підхід (Corpus-based Method)

Даний підхід базується на конкретних випадках, допомагає вирішити проблему пошуку лексикону думок з контекстно орієнтованими залежностями. Методи залежать від синтаксичних моделей або моделей, які складаються з списку базових словникових думок, щоб знайти інші слова в великому тілі корпусу. Один із методів починається зі списку прикметників базису та використовував їх поряд із переліком лінгвістичних обмежень для того, щоб визначити додаткові слова. Але існують обмеження,

що застосовуються до таких зв’язків, як “і”, “або”, “але” тощо, наприклад “і”, говорить, що зазвичай поєднані прикметники
мають однакову орієнтацію. Існують також протилежні вирази, такі як "але", "проте" які називаються змінами в думках. Для того, щоб визначити, однакові чи різні прикметники орієнтації пов'язані, навчання застосовується до великого корпусу. Далі зв'язки між прикметниками утворюють граф, на якому виконується кластеризація для визначення позитивних і негативних наборів слів.

Проте, словниковий підхід є більш ефективним так як достатньо важко створювати для охоплення всіх англійських слів величезний корпус. Але до переваг цього методу можна віднести факт того, що він може за допомогою доменного корпусу знайти думки у конкретному контексті та їх орієнтацію. Корпусний підхід відбувається за допомогою семантичного, або статистичного підходів.

Статистичні методи базуються на великій кількості даних про частоту зустрічання слова в тексті. Щоб визначити полярність слова можна визначити за допомогою ідентифікації частоти зустрічання слова у анотованому корпусі текстів. Якщо обране слово зустрічається частіше серед текстів позитивного контексту - його полярність позитивна. Якщо ж серед негативних текстів - його полярність відповідно є негативною. У випадку, коли це слово воно має рівні частоти, то воно вважається нейтральним.

Класичним підходом можна вважати використання метрики TF-IDF , а також її модифікації (для виділення ключових слів), і аналіз колокацій (для виділення словосполучень). TF-IDF - статистична міра, яка використовується для оцінки важливості слова, як в контексті документа (TF), так і в контексті корпусу документів (IDF). Метрика використовується з різними варіантами обчислення TF і IDF. Відомі різні розширення моделі TF-IDF, наприклад Okapi BM25[15]. Важливою особливістю даного методу є те, що набір даних

не повинен під час розрахунку змінюватись. Це достатньо сильно ускладнює процес обрахування якщо вони необхідні в реальному часі.

Оцінка важливості слова визначається за такими формулами:
TF (term frequency) - частота слова:

$$tf(t,d) = \frac{n_i}{\sum_k n_k},$$

де n_i- кількість входжень слова в документ

$\sum_k n_k$ - загальна кількість слів в документі

IDF (inverse document frequency) - обернена частота документа:

$$idf(t,D) = log\frac{|D|}{|d_i \ni t_i|},$$

де |D| - кількість документів в корпусі;

$|d_i \ni t_i|$ – кількість документів, у яких зустрічається термін t_i, $n_i \neq 0$

Тобто міра TF-IDF - це добуток двох попередніх оцінок:

$$tf - idf(t,d,D) = tf(t,d) \times idf(t,D)$$

Семантичний підхід спирається на різні принципи обчислення подібності слів і визначає безпосереднє значення настрою. Він визначає подібні почуття для семантично близьких слів. Для прикладу, для отримання списку сентиментальних слів може бути використаний WordNet. Початковий

набір слів з синонімами та антонімами ітераційно розширюється, після чого визначається полярність почуття для невідомого слова способом підрахунку синонімів до нього, як позитивних, так і негативни.

2.4 Постановка задачі класифікації текстових даних

Основною задачею, яку вирішує дана дипломна робота є бінарне розбиття текстових даних на два класи:

- позитивні;
- негативні.

Текстові дані у нашому випадку - це набір коментарів користувачів до деякого медіаконтенту. Необхідно знайти значення функції у вигляді оцінки від 0 до 1, якщо коментар негативний та позитивний відповідно. Варто зазначити, що всі вхідні дані (тексти) за результатами роботи мають належати відповідному класу, тобто необроблених текстів бути не може. Класифікація відбувається за правилом, якщо значення шуканої функції менше 0.5 - текстовий зразок вважається негативним коментарем, а якщо значення більше 0.5 - позитивним.

Кожному текстовому зразку відповідає тільки одна оцінка, тобто коментар не може мати двох чи більше значень функції.

2.5 Висновки до розділу 2

Даний розділ був присвячений огляду найпоширеніших методів класифікації текстових документів для визначення тональності тексту. Було розглянуто загальну структуру, а також детально кожний підхід. На основі

отриманої інформації було сформовано постановку задачі для розробки власної моделі.

РОЗДІЛ 3
ПРОГРАМНА РЕАЛІЗАЦІЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ МОДЕЛЮВАННЯ

3.1 Обґрунтування вибору платформи та мови програмування

Для розв’язку поставленого завдання, а саме категоризації відгуків користувачів до фільмів необхідно скористатись методами аналізу текстової інформації, а саме аспекту настрою за допомогою методів машинного навчання. Після ґрунтовного огляду методів та підходів аналізу тональності тексту було вирішено скористатись найбільш популярним, доступним та ефективним методом - нейромережа. Отже основною ідеєю є нейромережі, що на вхід отримує користувацький коментар до фільму, а на виході каже чи він є позитивний, чи негативний. На основі отриманої інформації у подальшому буде можливо розробляти програмний продукт для користувача.

Мовою програмування було обрано Python, що має значну кількість суттєвих переваг, в першу чергу це структурованість, читабельність та компактність коду, також різні операційні системи підтримують її. Крім цього, в області машинного навчання Python не має рівних по ефективності та інтелектуального аналізу даних.

Також наявна велика кількість доступних бібліотек. Для реалізації програмного продукту було використано ряд вбудованих бібліотек, таких як Keras, Numpy, Pyplot, String, bs4 та Requests

Keras представляє з себе високорівневу API, написаний на Python і здатний працювати поверх TensorFlow,Theano або CNTK. Він був розроблений з розрахунком на швидке навчання. Здатність переходити від гіпотез до результати з найменшими тимчасовими витратами є ключем до проведення успішних досліджень[16].

Головні міркування якими я керувалася при виборі було те, що дана бібліотека ставить користувальницький досвід в основу всього. Keras використовує передові методи зниження когнітивної навантаження: він пропонує узгоджений і простий API, мінімізує кількість дій користувача, що необхідні для вирішення поширених завдань, а також надає чіткий і дієвий зворотний зв'язок в разі виникнення помилок. Крім цього, модульність є величезною перевагою, адже під моделлю мається на увазі послідовність або граф автономних та повністю сконфігурованих модулів, що можуть бути підключені без будь-яких додаткових обмежень. Наприклад нейронні шари, функції помилки, оптимізатори, схеми ініціалізації, функції активації і схеми регуляризації. Це все окремі модулі, які можна комбінувати для створення власної моделі, що представляє найбільший інтерес.Також можливе розширення, тобто нові модулі легко додавати (так само як нові класи та функції), а існуючу модулі надають достатню кількість подібних прикладів. Можливість додавати нові модулі робить Keras максимально привабливим засобом для проведення передових досліджень.

А також, всі моделі написані на мові програмування Python, завдяки чому код компактний, легко читається і налагоджували, а також легко розширюється і що робить його простим та зрозумілим у використанні. Базова структура даних цієї бібліотеки - це модель, що описує спосіб організації шарів. Найпростішим типом моделі є модель Sequential, що представляє собою лінійний стек шарів. Для більш складних архітектур необхідно використовувати функціональне програмування, яке дозволяє побудувати довільні графи шарів.

NumPy - це бібліотека мови Python, що додає підтримку великих багатовимірних масивів і матриць, разом з великою бібліотекою високорівневих (і дуже швидких) математичних функцій для операцій з цими масивами.

За допомогою бібліотеки Pyplot можна будувати графіки та діаграми для наглядності, а бібліотека String розширює функціонал роботи з рядками даних. Requests дає змогу робити http запити, а bs4 допомагає збирати необхідні текстові дані з обраного сайту.

3.2. Підготовка та аналіз даних та навчання моделі.

Для тренування та тестування моделі було взято дані користувацьких відгуків до фільмів з сайту IMDB [17]. Так як в кіноіндустрії це найбільша база даних всіх фільмів, що доступна користувачам у вигляді веб сайту. На даний момент, вона начислює інформацію про більше ніж 7.7 млн фільмів з усього світу [18]

Набір готових даних було знайдено в статті [19], що складається з 50 000 відгуків з IMDB, враховуючи не більше 30 відгуків за фільм. Створений набір даних містить парну кількість позитивних та негативних відгуків, тому випадкове відгадування дає 50% точності. Тобто 25 000 коментарів - абсолютно позитивні з оцінками від 7 до 10, та ще 25 000 коментарів - негативні з оцінками від 1 до 10. Полярність відгуку визначається кількістю зірочок, що ставили самі користувачі біля свого коментаря. Нейтральні огляди не включаються до набору даних. Так як ключовим завданням є якісно натренувати модель з їх допомогою.

Також 50 000 даних було порівну розподілено на тестовий набір та тренувальний, для якісного тренування нейронної мережі буде достатньо 25 000 екземплярів. Кожен відгук має правильну відповідь у вигляді оцінки 0 або 1, так як з точки зору машинного навчання це задача бінарної класифікації.

Для набору даних використано лише 10 000 найпопулярніших слів, адже при більшій кількості швидкість роботи програми значно знизиться, а даної кількості достатньо для отриманя якісного результату.

Детальніше структуру даних можна переглянути в таблиці 3.1, що розташована нижче.

Таблиця 3.1 - Змінні даних

Назва	Опис
x_train	Тренувальні дані у вигляді вектору чисел-кодів кожного слова.
y_test	Оцінка відгуку 0 або 1.
x_test	Тестові дані у вигляді вектору чисел-кодів кожного слова.
y_test	Оцінка відгуку 0 або 1.

У наборі даних IMDB використовується частотне кодування слів. Тобто в тексті кожне слово замінюється на частоту його застосування. На рисунку 3.1 частково показано словник.

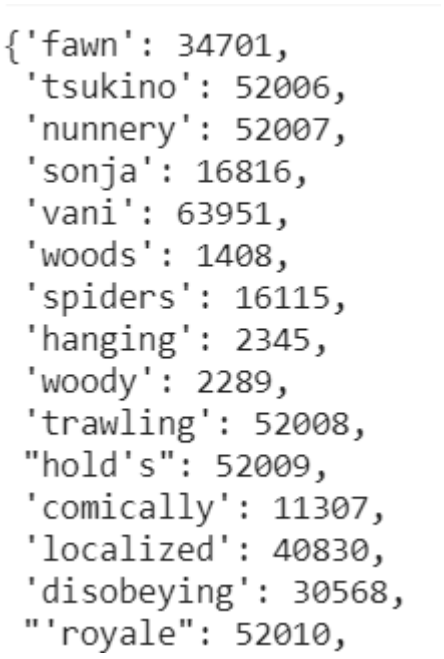


Рисунок 3.1 - Словник слів та їх кодів

Він використовувався для кодування, де ключ - це слово, а значення - це частота його застосування у відгуках. Для прикладу, на рисунку 3.2 зображено 20 найпопулярніших слів.

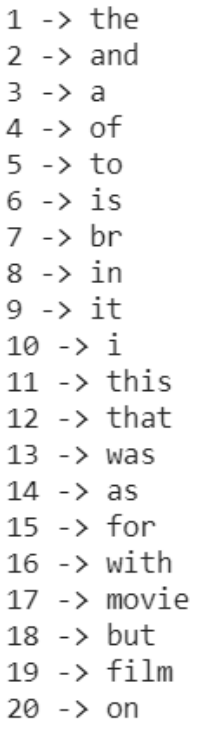


Рисунок 3.2 - 10 найвживаніших слів

В бібліотеці Keras при підготовці набору даних використовувалось кодування вигляду:

0 - символ заповнювач,

1 - початок послідовності,

2 - слово, якого немає в словнику.

Нижче на рисунках 3.3 , 3.4 можна побачити приклади вигляду текстових даних у закодованому та розкодованому вигляді.

```
[1, 778, 128, 74, 12, 630, 163, 15, 4, 1766, 7982, 1051, 2, 32, 85, 156, 45,
40, 148, 139, 121, 664, 665, 10, 10, 1361, 173, 4, 749, 2, 16, 3804, 8, 4, 226,
 65, 12, 43, 127, 24, 2, 10, 10]
```

Рисунок 3.3 - Закодований вигляд відгуку

```
'? begins better than it ends funny t
hat the russian submarine crew ? all
other actors it's like those scenes w
here documentary shots br br spoiler
part the message ? was contrary to th
e whole story it just does not ? br b
r'
```

Рисунок 3.4 - Розкодований вигляд відгуку

“?” заміняє одне з зазначених вище кодувань. Нижче зображено функцію, що розкодовує відгук на рисунку 3.5

```
def get_review_text(review_int_vector):
    word_index = imdb.get_word_index()
    reverse_word_index = dict([(value, key) for (key, value) in word_index.items()])
    decoded_review = ' '.join([reverse_word_index.get(i - 3, '?') for i in review_int_vector])
    return decoded review
```

Рисунок 3.5 - Функція для розкодування відгуку

Якщо передати дані у такому вигляді, то нейронна мережа буде показувати вихідні значення майже навмання, адже ймовірність коливається близько 0.5, тому для результативності нейронної мережі вхідні дані необхідно було векторизувати. Для цього було порівняно два формати вхідних даних для нейронної мережі.

Формат One Hot Encoding представляє собою вектор зі значеннями 0 та 1, де 1 означає, що потрібне слово є в вхідному переліку, а 0 - його відсутність. Наприклад, якщо є якась функція "квітка", яка може приймати значення "нарцис", "лілія" та "троянда". One Hot Encoding кодування перетворює функцію "квітка" у три функції: "is daffodil", "is_lily" та "is_rose", які всі є двійковими [18].

В даному методі послідовність слів не грає ніякої ролі, важливий лиш факт “зустрічання” цього слова у вибірці як спрощений варіант. Недоліком є те, що ми втрачаємо кількість “зустрічання” даного слова в тексті. Було реалізовано функцію для векторизації даних, що зображено на рисунку 3.6.

```
def vectorize_sequences(sequences, dimension=10000):
    results = np.zeros((len(sequences), dimension))
    for i, sequence in enumerate(sequences):
        results[i, sequence] = 1.
    return results
```

Рисунок 3.6 - Функція кодування методом One Hot Encoding

Приклад векторизації методом One Hot Encoding можна побачити на рисунку 3.7 нижче.

```
array([1., 1., 1., 0., 1., 0., 0., 0., 1., 0., 1., 0., 1., 0., 0., 1., 1.,
       0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 1., 0.,
       0., 0., 0., 0., 0., 0., 1., 0., 0., 1., 0., 1., 0., 0., 0., 0.])
```

Рисунок 3.7 - Закодований відгук методом One Hot Encoding

За допомогою функціоналу бібліотеки Keras, модель побудови нейронної мережі можна побачити на рисунку 3.8

```
model = Sequential()
model.add(Dense(16, activation='relu', input_shape=(max_words,))
model.add(Dense(16, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
```

```
model.compile(optimizer='rmsprop',
              loss='binary_crossentropy',
              metrics=['accuracy'])
```

Рисунок 3.8 - Вигляд нейронної мережі

За закодованими даними методом One Hot Encoding було навчено дану нейронну мережу, за допомогою функції fit з бібліотеки Keras, що показано на рисунку 3.9

```
history = model.fit(x_train[1000:],
                    y_train[1000:],
                    epochs=20,
                    batch_size=512,
                    validation_split=0.1)
```

Рисунок 3.9 - Навчання нейронної мережі

За отриманими даними можна було визначити оптимальну кількість епох щоб уникнути перенавчання

З графіків на рисунках 3.10, 3.11 видно, що почало відбуватись перенавчання мережі, тому необхідно зменшити кількість епох до 4.

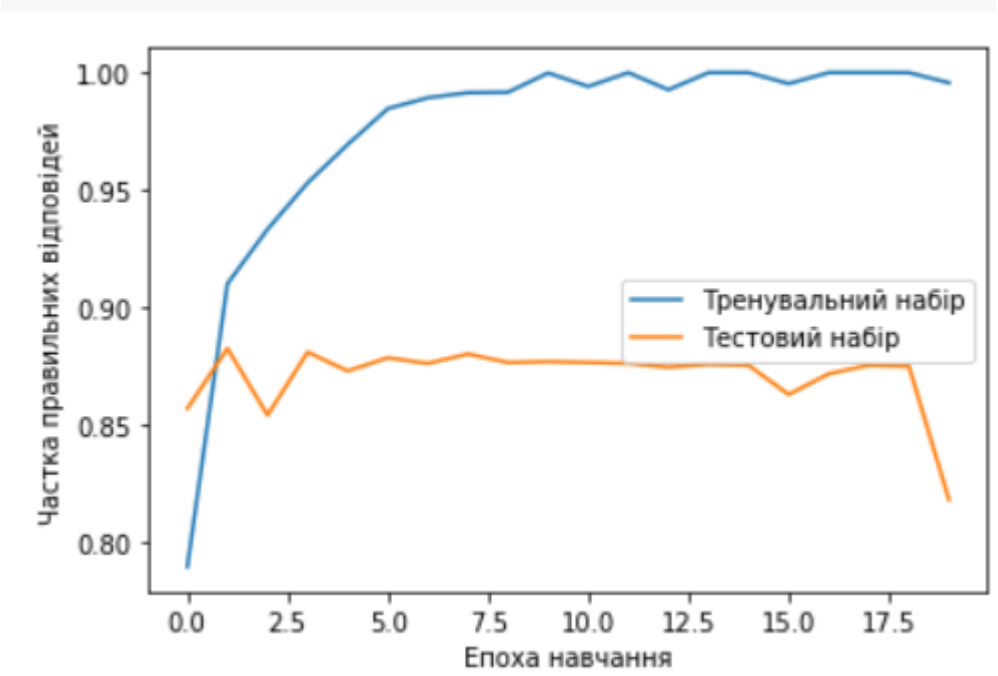


Рисунок 3.10 - Графік частоти правльних відповідей

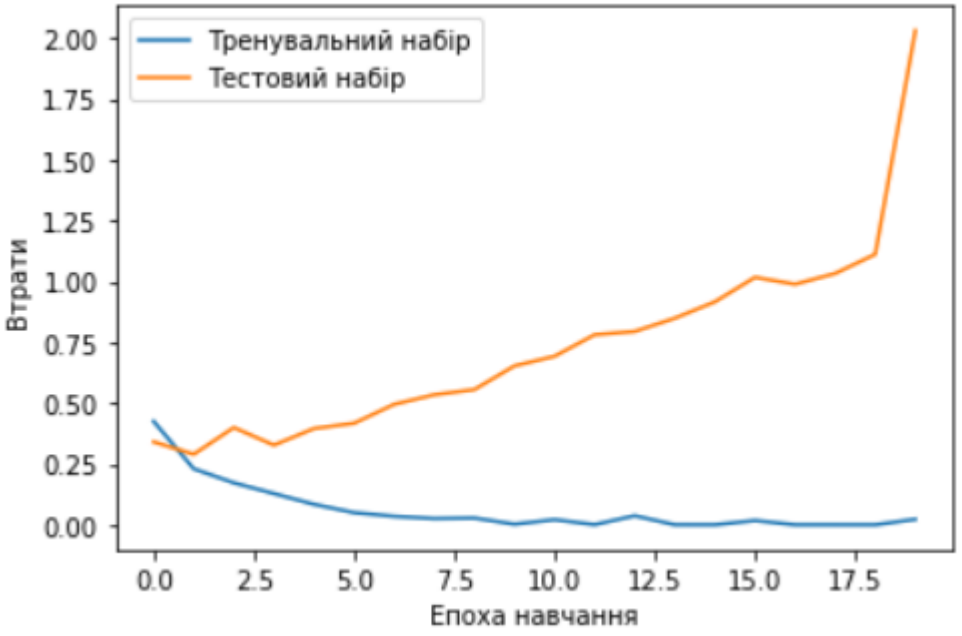


Рисунок 3.10 - Графік частоти втрат

Даний метод показує досить непогані результати класифікації даних, а саме 85.828% правильних відповідей на тестових даних.

Наступний формат для підготовки даних та їх подальшої обробки - це так званий embedding, що являють собою числові вектори, які отримані з слів або інших мовних сутностей. Дані для обробки нейронною мережею мають вигляд як на рисунку 3.3, але нейронна мережа має дещо іншу структуру, що зображена на рисунку 3.11.

```
model = Sequential()
model.add(Embedding(max_words, 2, input_length=maxlen))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(1, activation='sigmoid'))
```

```
model.compile(optimizer='adam',
              loss='binary_crossentropy',
              metrics=['accuracy'])
```

Рисунок 3.11 - Вигляд нейронної мережі

Після навчання даної мережі маємо такі показники з кількістю епох навчання, що дорівнює 15, щоб запопігти перенавчанню. Результати зображено на рисунках 3.12, 3.13.

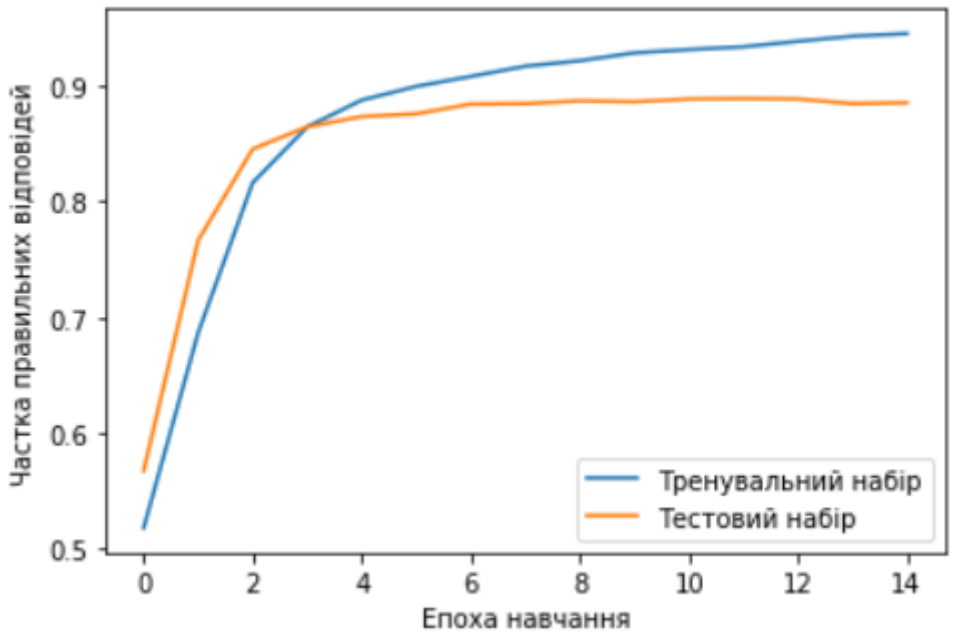


Рисунок 3.12 - Графік частоти правльних відповідей

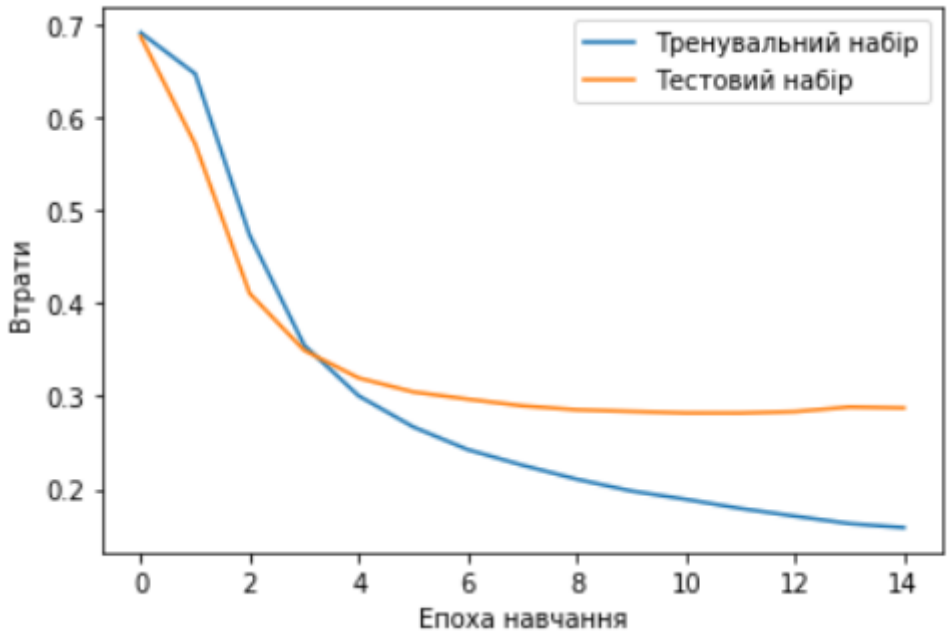


Рисунок 3.13 - Графік частоти втрат

З графіків видно кращі результати класифікації даних, а саме 87.324% правильних відповідей на тестових даних.

Тому подальша реалізація програмного продукту буде проводитись на основі другої нейронної мережі.

3.3. Опис кінцевого програмного продукту

Під програмним продуктом розуміється система, що аналізує користувацькі відгуки до заданої кінострічки та показує статистику позитивних, негативних та нейтральних коментарів на основі нейронної мережі, що була описана вище.

Програма розроблена в онлайн середовищі Google Colab, що доступний усім користувачам та є сам по собі непоганим та зрозумілим користувацьким інтерфейсом для першої версії програмного продукту.

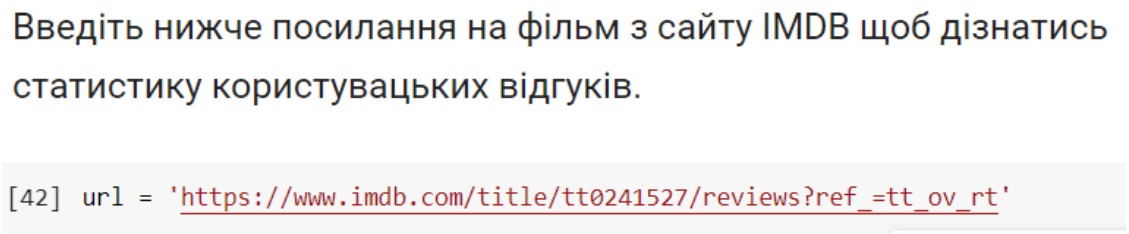


Рисунок 3.14 - Поле для вводу посилання на фільм

На цьому етапі програма збирає текстові дані з заданого посилання, зберігає їх в масив, а далі за допомогою функцій готує дані для передачі в нейронну мережу. Метод find_word_code, що зображено на рисунку 3.15 приймає на вхід слово та знаходить в словнику його числове значення.

```
def find_word_code(word):
  word_index = imdb.get_word_index()
  word_index = {k: v for k, v in word_index.items() if v < 10000}
  code = word_index.get(word)
  if type(code) == int:
    return code + 3
  else:
    return 0
```

Рисунок 3.15 - Функція find_word_code

Метод create_reviewWordsArray, що зображено на рисунку 3.16 приймає на вхід текст та повертає масив закодованих текстів.

```
def create_reviewWordsArray(text_example):
  words = text_example.lower().split()
  table = str.maketrans('', '', string.punctuation)
  reviewWordsArray = [w.translate(table) for w in words[:200]]
  codeArray = map(lambda x: find_word_code(x), reviewWordsArray)
  return list(codeArray)
```

Рисунок 3.16 - Функція create_reviewWordsArray

Отриманий набір даних передається до нейронної мережі, що визначає тональність кожного тексту, Так, як даних багато, фрагмент результату роботи нейронної мережі показано на рисунку 3.17.

```
array([[0.64740694],
       [0.57226187],
       [0.1697506 ],
       [0.9108463 ],
       [0.32150325],
       [0.02472082],
       [0.9930382 ],
       [0.03493306],
       [0.9973676 ],
       [0.0235239 ],
       [0.14306194],
       [0.18972954],
       [0.8541038 ],
       [0.22364104],
       [0.9909985 ],
```

Рисунок 3.17 - Фрагмент результату роботи нейронної мережі

Далі ці дані структуризуються та відображаються у вигляді діаграми, що наведено на рисунку 3.18.

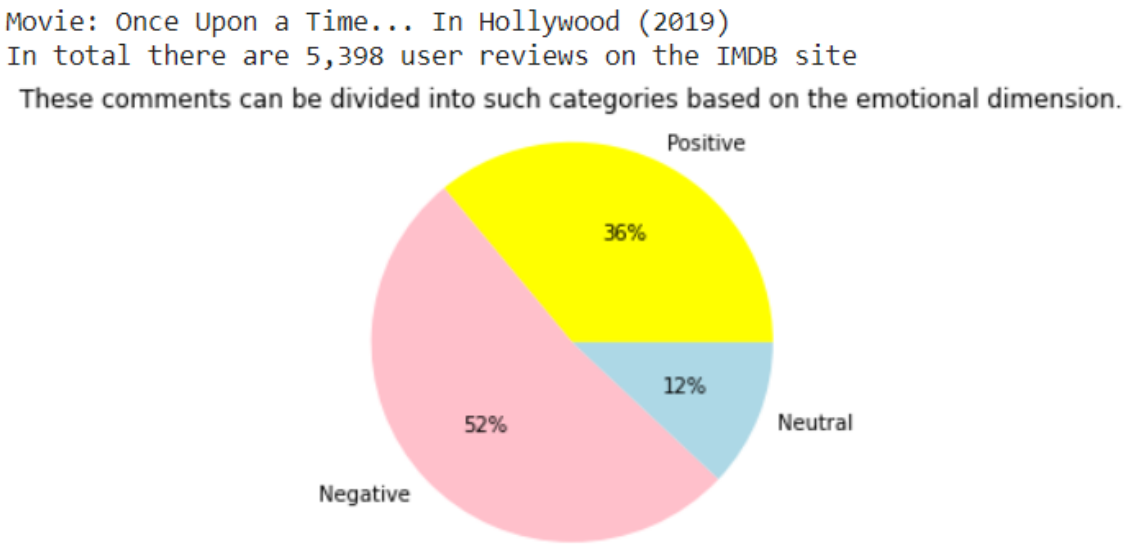


Рисунок 3.18 - Результат роботи програми

Надалі можливе вдосконалення розробленого програмного продукту у вигляді оптимізації коду для збільшення швидкості, а також розробки повноцінного додатку для різних пристроїв. Також можлива імплементація у вже існуючі сервіси з великою кількістю відгуків, що потребують аналізу.

3.4 Висновки до розділу 3

У даному розділі було обґрунтовано вибір мови програмування, відповідних бібліотек, середовища розробки та методів реалізації програмного продукту. Також, було проведено порівняння двох нейронних мереж та на основі кращої з них було реалізовано систему по визначенню тональності відгуків користувачів з сайту IMDB. Крім цього отримані результати було опрацьовано та структуровано в зрозумілий вигляд для користувача у вигляді статистичної діаграми. Також було наведено інструкцію для користувача даного програмного продукту та інтерфейс застосунку.

Також були запропоновані варіанти наступних версій програмного продукту та окреслено область вдосконалення.

РОЗДІЛ 4
ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ

У даному розділі проводиться оцінка основних характеристик програмного продукту, розробленого для категоризації коментарів користувачів в інтернет мережі.

Нижче наведено аналіз різних варіантів реалізації модулю з метою вибору оптимальної, з огляду при цьому як на економічні фактори, так і на характеристики продукту, що впливають на продуктивність роботи і на його сумісність з апаратним забезпеченням. Для цього було використано апарат функціонально-вартісного аналізу.

Функціонально-вартісний аналіз (ФВА) – це технологія, яка дозволяє оцінити реальну вартість продукту або послуги незалежно від організаційної структури компанії. ФВА проводиться з метою виявлення резервів зниження витрат за рахунок ефективніших варіантів виробництва, кращого співвідношення між споживчою вартістю виробу та витратами на його виготовлення. Для проведення аналізу використовується економічна, технічна та конструкторська інформація.

Алгоритм функціонально-вартісного аналізу включає в себе визначення послідовності етапів розробки продукту, визначення повних витрат (річних) та кількості робочих часів, визначення джерел витрат та кінцевий розрахунок вартості програмного продукту.

4.1 Постановка задачі проектування

У роботі застосовується метод ФВА для проведення техніко-економічного аналізу розробки категоризації коментарів

користувачів. Оскільки рішення стосовно проектування та реалізації компонентів, що розробляється, впливають на всю систему, кожна окрема підсистема має її задовольняти. Тому фактичний аналіз представляє собою аналіз функцій програмного продукту, призначеного для збору, обробки та проведення аналізу даних.

Технічні вимоги до програмного продукту є наступні:

- функціонування на персональних комп'ютерах із стандартним набором компонентів;
- зручність та зрозумілість для користувача;
- швидкість обробки даних та доступ до інформації в реальному часі;
- можливість зручного масштабування та обслуговування;
- мінімальні витрати на впровадження програмного продукту.

4.2 Обґрунтування функцій та параметрів програмного продукту

Головна функція F_0 – розробка програмного продукту, який вирішує задачу категоризації неструктурованих даних, а саме користувацьких коментарів. Беручи за основу цю функцію, можна виділити наступні:

F_1 – вибір мови програмування;

F_2 – вибір середовища розробки.

F_3 – вибір методу категоризації

Кожна з цих функцій має декілька варіантів реалізації:

Функція F_1:

A. Python

B. C++

Функція F_2:

A. Jupyter Notebook;

B. Visual Studio.

C. Google Collaboratory

D. PyCharm IDE

Функція F_3:

A. Машинне навчання;

B. Лексичний метод.

Варіанти реалізації основних функцій наведені у морфологічній карті системи (рис. 4.1).

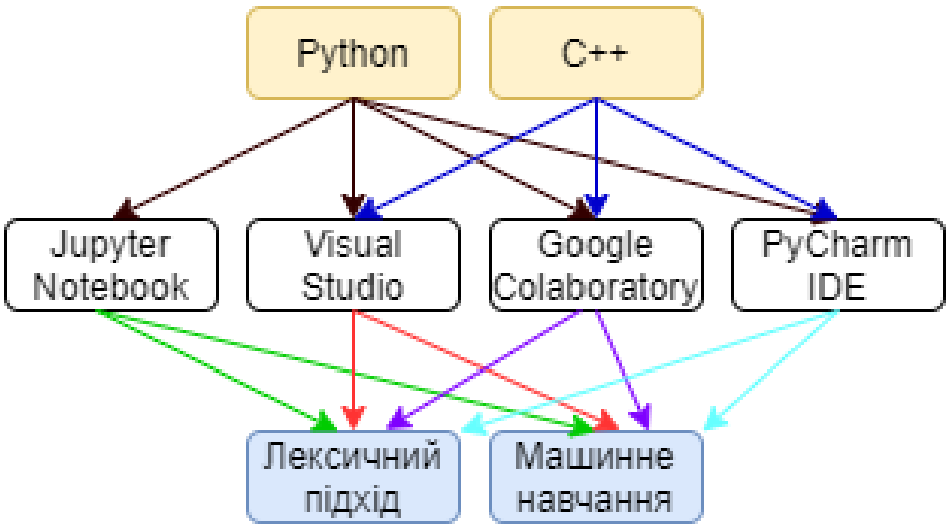


Рисунок 4.1 – Морфологічна карта

Морфологічна карта відображає множину всіх можливих варіантів основних функцій.

Таблиця 4.1 - Позитивно-негативна матриця

Функції	Варіанти реалізації	Переваги	Недоліки

F_1	*А*	Швидка розробка програми, доступність великої кількості бібліотек, читабельність та компактність коду, кросплатформеність	Нижча швидкість роботи у випадку обробки великої кількості даних
	Б	Висока швидкість виконання коду	Витрачається багато часу на розробку програмного продукту, код громіздкий та складний
F_2	*А*	Підтримується багатьма мовами програмування, легко запускається на будь-якому сервері	Відсутня можливість роботи без інтернету
	Б	Багато інструментів, безпечна	Підтримує одночасно лише одну мову програмування
	В	Не потребує встановлення програмного забезпечення, хмарне зберігання	Можлива втрата даних через тимчасову робочу сесію

	Г	Зручний процес дебагу, доступність засобів інтегрування	Займає великий обсяг пам’яті, іноді повільно спрацьовує
F_3	*А*	Ефективний та показує точні результати	Необхідно якісно обробляти дані для навчання моделі
	Б	Швидко знаходить словниковий зв’язок	Не враховує контекст,, неможливо обробляти в реальному часі

На основі цієї карти будуємо позитивно-негативну матрицю варіантів основних функцій (Таб.4.1), з чого можна зробити висновок, що при розробці програмного продукту деякі варіанти реалізації функцій варто відкинути, тому що вони не відповідають поставленим перед програмним продуктом задачам. Ці варіанти відзначені у морфологічній карті.

Функція F_1:

Перевагу було надано швидкості вивчення, простоті використання та наявності стандартних бібліотек. Для спрощення роботи з написання коду варіант Б має бути відкинутий.

Функція F_2:

Віддаємо перевагу варіанту В в разі вибору мови програмування Python та оскільки є можливість редагувати код та розробляти програмний продукт одночасно з різних пристроїв та не потребує встановлення програмного забезпечення, а також використовує хмарне зберігання даних.

Функція F_3:

Віддаємо перевагу варіанту А, адже він ефективний та показує точні результат. Для оцінювання якості розглянутих функцій обрана система параметрів, описана нижче.

4.3 Обґрунтування системи параметрів ПП

На основі даних, розглянутих вище, визначаються основні параметри вибору, які будуть використані для розрахунку коефіцієнта технічного рівня.

Для того, щоб охарактеризувати програмний продукт, будемо використовувати наступні параметри:

- *X1* – швидкодія мови програмування;
- *X2* – об’єм пам’яті для обчислень та збереження даних;
- *X3* – потенційний об’єм програмного коду.

Гірші, середні і кращі значення параметрів вибираються на основі вимог замовника й умов, що характеризують експлуатацію ПП як показано у таблиці 4.2.

Таблиця 4.2 - Основні параметри ПП

Назва Параметра	Умовні позначення	Одиниці виміру	Значення параметра		
			гірші	середні	кращі
Швидкодія мови програмування	X1	оп/мс	10000	14000	19000
Об’єм пам’яті	X2	Мб	420	128	64
Час навчання моделі	X3	с	50	120	210

Потенційний об’єм програмного коду	X4	кількість рядків коду	500	400	300

За даними таблиці 4.2 будуються графічні характеристики параметрів – рис. 4.2 – рис. 4.4.

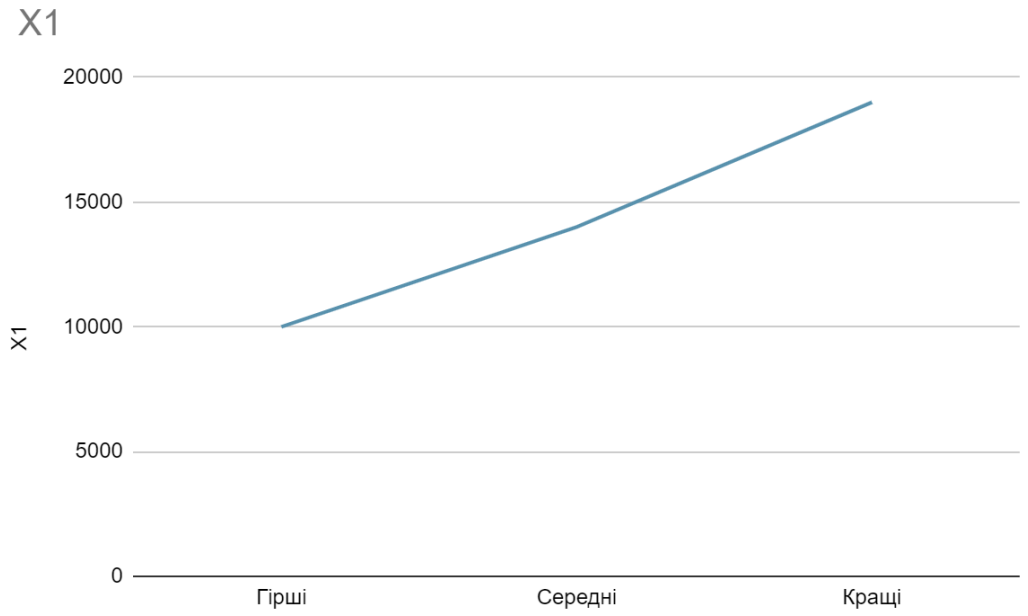


Рисунок 4.1 - Значення параметра X1

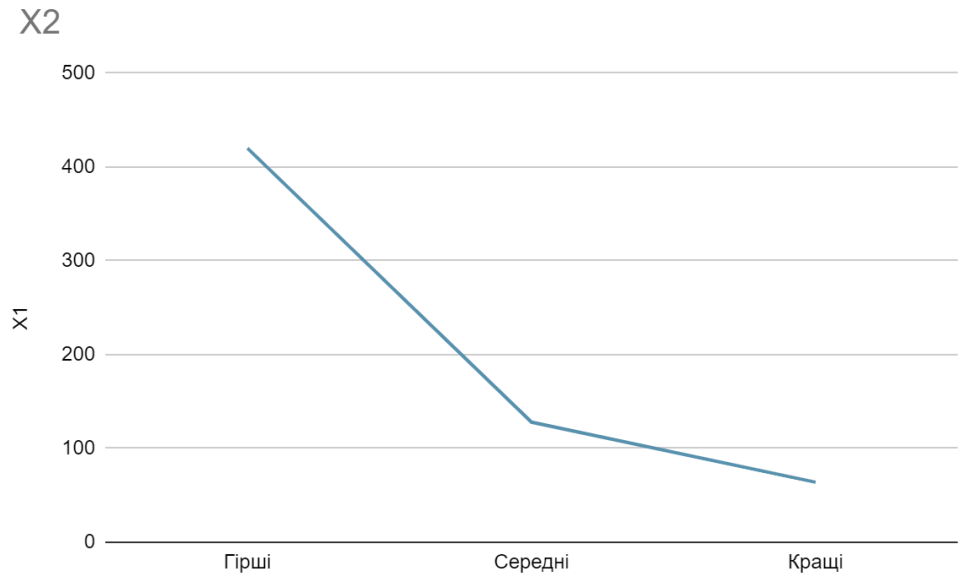


Рисунок 4.2 - Значення параметра X2

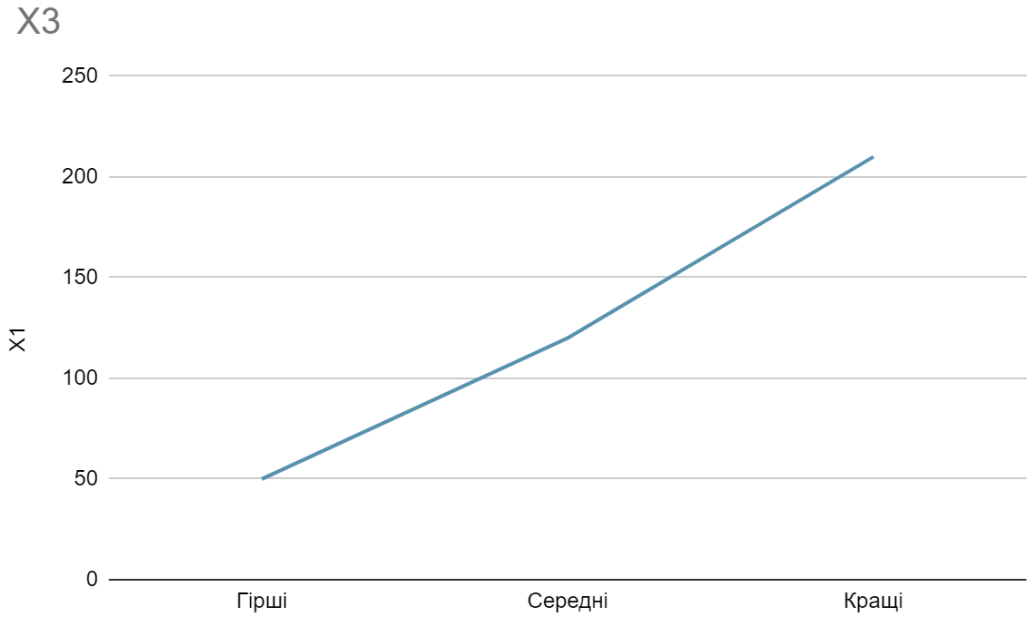


Рисунок . 4.3 - Значення параметра X3

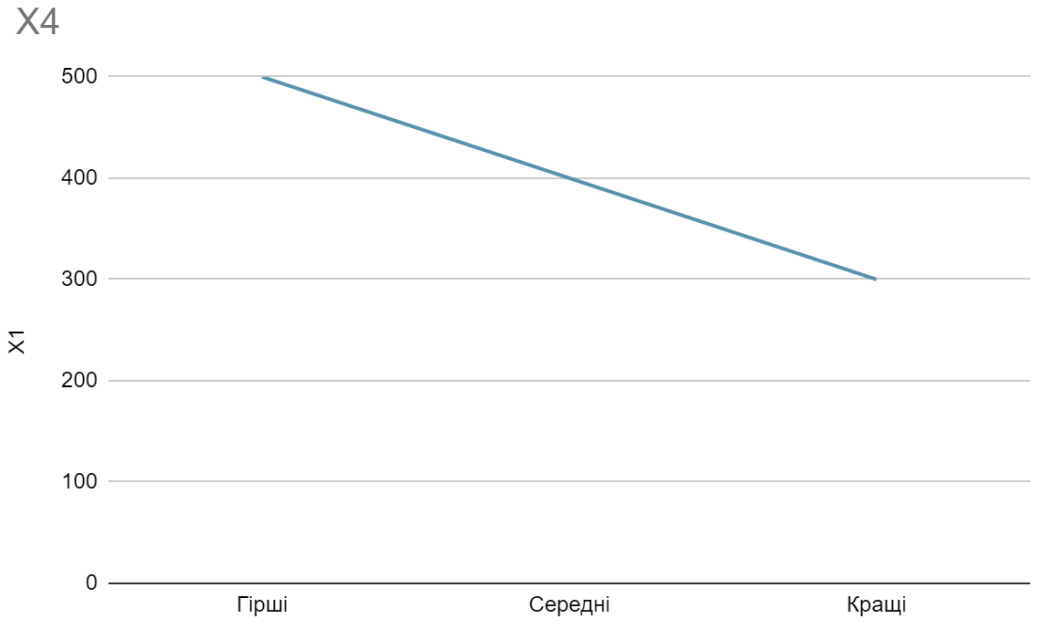


Рисунок 4.4 - Значення параметра X4

4.4 Аналіз експертного оцінювання параметрів

Після детального обговорення й аналізу кожний експерт оцінює ступінь важливості кожного параметру для конкретно поставленої цілі – розробка програмного продукту, який дає найбільш точні результати при знаходженні параметрів моделей адаптивного прогнозування і обчислення прогнозних значень.

Значимість кожного параметра визначається методом попарного порівняння. Оцінку проводить експертна комісія із 7 людей. Визначення коефіцієнтів значущості передбачає:

- визначення рівня значущості параметра шляхом присвоєння різних рангів;
- перевірку придатності експертних оцінок для подальшого використання;
- визначення оцінки попарного пріоритету параметрів;

– обробку результатів та визначення коефіцієнту значимості.

Вагомість параметрів оцінюється за допомогою методів попарного зрівняння. Ранги варіюються від 1 до 5, де 1 – найменший ранг, 5 – найбільший. Результати наведені в табл. 4.3-4.4.

Для перевірки степені достовірності експертних оцінок, визначимо наступні параметри:

а) сума рангів кожного з параметрів і загальна сума рангів:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 70,$$

де N – число експертів;

n – кількість параметрів.

б) середня сума рангів:

$$T = \frac{1}{n} R_{ij} = 17,5,$$

в) відхилення суми рангів кожного параметра від середньої суми рангів:

$$\Delta_i = R_i - T,$$

Сума відхилень по всім параметрам повинна дорівнювати 0;

г)загальна сума квадратів відхилення:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 173,$$

Порахуємо коефіцієнт узгодженості:

$$W = \frac{12S}{N^2\left(n^3-n\right)} = \frac{12\cdot 173}{7^2\left(4^3-4\right)} = 0,71 > W_k = 0,67,$$

Ранжування можна вважати достовірним, тому що знайдений коефіцієнт узгодженості перевищує нормативний, котрий дорівнює 0,67.

Скориставшись результатами ранжирування, проведемо попарне порівняння всіх параметрів і результати занесемо у таблицю 4.3.

Таблиця 4.3 - Результати ранжування параметрів

Позначення параметра	Назва параметра	Одиниці виміру	Ранг параметра за оцінкою експерта							Сумарангів R_i	Відхилення Δ_i	Δ_i^2
			1	2	3	4	5	6	7			
X1	Швидкодія мови програмування	Оп/мс	5	4	2	5	2	5	4	27	9.5	90.25
X2	Об’єм пам’яті	Mb	1	2	4	1	2	2	1	13	-4.5	20.25

X3	Час навчання моделі	с	2	3	1	2	4	1	1	14	-3.5	-12.25
X4	Потенційний об’єм програмного коду	Кількість рядків коду	2	1	3	2	2	3	4	17	-0.5	0.25
	Разом		10	10	10	10	10	10	10	70	0	173

Скориставшись результатами ранжування, проведемо попарне порівняння всіх параметрів і результати занесемо у таблицю 4.4.

Таблиця 4.4 - Попарне порівняння параметрів.

Параметри	Експерти							Кінцева оцінка	Числове значення
	1	2	3	4	5	6	7		
X1 та X2	>	<	<	>	=	>	>	>	1.5
X1 та X3	>	>	>	>	<	>	>	>	1.5
X1 та X4	>	>	<	>	<	>	>	>	1.5
X2 та X3	<	<	>	>	<	<	>	<	0.5
X2 та X4	<	>	>	<	=	<	<	<	0.5
X3 та X4	=	>	<	=	>	<	<	<	0.5

Числове значення, що визначає ступінь переваги *i*–го параметра над *j*–тим, a_{ij} визначається по формулі:

$$a_{ij} = \{1.5 \text{ при } X_i > X_j, 1.0 \text{ при } X_i = X_j, \ 0.5 \text{ при } X_i < X_j$$

З отриманих числових оцінок переваги складемо матрицю $A=\|a_{ij}\|$.

Для кожного параметра зробимо розрахунок вагомості $K_{вi}$ за наступними формулами:

$$K_{\text{ві}} = \frac{b_i}{\sum_{i=1}^{n} b_i}$$

$$b_i = \sum_{i=1}^{N} a_{ij}$$

Відносні оцінки розраховуються декілька разів доти, поки наступні значення не будуть незначно відрізнятися від попередніх (менше 5%). На другому і наступних кроках відносні оцінки розраховуються за наступними формулами:

$$K_{\text{ві}} = \frac{b_i'}{\sum_{i=1}^{n} b_i'}$$

$$b_i' = \sum_{i=1}^{N} a_{ij} b_j$$

Таблиця 4.6 - Розрахунок вагомості параметрів

Параметри	Параметри	Перша ітер.	Друга ітер.

	X1	X2	X3	X4	b_i	$K_{\text{ві}}$	b_i^1	$K_{\text{ві}}^1$
X1	1,0	1,5	1,5	1,5	5,5	0,344	21,25	0,360
X2	0,5	1,0	0,5	0,5	2,5	0,156	9,25	0,157
X3	0,5	1,5	1,0	0,5	3,5	0,219	12.25	0,207
X4	0,5	1,5	1,5	1,0	4,5	0,281	16.25	0,275
Всього:					16	1	59	1

Як видно з таблиці 4.6, різниця значень коефіцієнтів вагомості не перевищує 5%, тому більшої кількості ітерацій не потрібно.

4.5 Аналіз рівня якості варіантів реалізації функцій

Визначаємо рівень якості кожного варіанту виконання основних функцій окремо.

Абсолютні значення параметрів *X2* (Об’єм пам’яті) та *X4* (час попередньої обробки даних) відповідають технічним вимогам умов функціонування даного ПП.

Абсолютне значення параметра *X1* (швидкість роботи мови програмування) обрано не найгіршим.

Коефіцієнт технічного рівня для кожного варіанта реалізації ПП розраховується так (таблиця 4.7):

$$K_k(j) = \sum_{i=1}^{n} K_{\text{в}i,j} B_{i,j},$$

де n – кількість параметрів;

$K_{\text{в}i}$ – коефіцієнт вагомості i–го параметра;

B_i – оцінка i–го параметра в балах.

Таблиця 4.7 - Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основні функції	Варіант реалізації функції	Параметри	Абсолютне значення параметра	Бальна оцінка параметра	Коефіцієнт вагомості параметра	Коефіцієнт рівня якості
F1	А	Х1	10000	6	0,360	2,76
F2	В	Х3	300	5	0,32	1,6
	Г	Х3	400	4	0,32	1,28
F3	А	Х3	120	5	0,157	0,39

За даними з таблиці 4.7 за формулою:

$$K_K = K_{TY}[F_{1k}] + K_{TY}[F_{2k}] + \ldots + K_{TY}[F_{zk}]$$

визначаємо рівень якості кожного з варіантів:

$$К_{К1} = 2,76+0,39+1,6=4,75,$$
$$К_{К2} = 2,76+0,39+1,28 = 4,43.$$

Як видно з розрахунків, кращим є перший варіант, для якого коефіцієнт технічного рівня має найбільше значення.

4.6 Економічний аналіз варіантів розробки ПП

Для визначення вартості розробки ПП спочатку проведемо розрахунок трудомісткості.

Всі варіанти включають в себе два окремих завдання:

1. Розробка проекту програмного продукту;

2. Розробка програмної оболонки;

Завдання 1 за ступенем новизни відноситься до групи А, завдання 2 – до групи Б. За складністю алгоритми, які використовуються в завданні 1 належать до групи 1; а в завданні 2 – до групи 3.

Для реалізації завдання 1 використовується довідкова інформація, а завдання 2 використовує інформацію у вигляді даних.

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань.

Загальна трудомісткість обчислюється як

$$Т_О = Т_Р \cdot К_П \cdot К_{СК} \cdot К_М \cdot К_{СТ} \cdot К_{СТ.М},$$

де $Т_Р$ – трудомісткість розробки ПП;

$К_П$ – поправочний коефіцієнт;

$К_{СК}$ – коефіцієнт на складність вхідної інформації;

$К_М$ – коефіцієнт рівня мови програмування;

$К_{СТ}$ – коефіцієнт використання стандартних модулів і прикладних програм;

$К_{СТ.М}$ – коефіцієнт стандартного математичного забезпечення

Для першого завдання, виходячи із норм часу для завдань розрахункового характеру ступеню новизни А та групи складності алгоритму 1, трудомісткість дорівнює: $Т_Р$ =90 людино-днів. Поправочний коефіцієнт, який враховує вид нормативно-довідкової інформації для першого завдання: $К_П$ = 1.7. Поправочний коефіцієнт, який враховує складність контролю вхідної та вихідної інформації для всіх семи завдань рівний 1: $К_{СК}$ = 1. Оскільки при розробці першого завдання використовуються стандартні модулі, врахуємо це за допомогою коефіцієнта $К_{СТ}$ = 0.8. Тоді загальна трудомісткість програмування першого завдання дорівнює:

$$Т_1 = 90 \cdot 1.7 \cdot 0.8 = 122.4 \text{ людино-днів.}$$

Проведемо аналогічні розрахунки для подальших завдань.

Для другого завдання (використовується алгоритм третьої групи складності, степінь новизни Б), тобто $Т_Р$ =27 людино-днів, $К_П$ =0.9,$К_{СК}$ = 1,$К_{СТ}$ =0.8:

$$Т_2 = 27 \cdot 0.9 \cdot 0.8 = 19.44 \text{ людино-днів.}$$

Складаємо трудомісткість відповідних завдань для кожного з обраних варіантів реалізації програми, щоб отримати їх трудомісткість:

$Т_I$ = (122.4 + 19.44 + 4.8 + 19.44) · 8 = 1328,64 людино-годин.

$Т_{II}$ = (122.4 + 19.44 + 6.91 + 19.44) · 8 = 1345.52 людино-годин.

Найбільш високу трудомісткість має варіант II.

В розробці беруть участь два програмісти з окладом 13000 грн., один аналітик в області даних з окладом 14500. Визначимо середню зарплату за годину за формулою:

$$С_{ч} = \frac{M}{T_m \cdot t} \text{ грн},$$

де М – місячний оклад працівників;

T_m – кількість робочих днів тиждень;

t – кількість робочих годин в день.

$$С_{ч} = \frac{2*13000 + 14500}{3 \cdot 21 \cdot 8} = 80,35$$

Тоді, розрахуємо заробітну плату за формулою:

$$С_{ЗП} = С_{ч} \cdot T_i \cdot К_{Д},$$

де $С_{ч}$– величина погодинної оплати праці програміста;

T_i – трудомісткість відповідного завдання;

$К_{Д}$ – норматив, який враховує додаткову заробітну плату.

Зарплата розробників за варіантами становить:

I. $С_{ЗП}$ =80, 35 · 1328.64 · 1.2 =128107,46 грн.

II. $С_{ЗП}$ = 80, 35 · 1345.52 · 1.2 =129735,04 грн.

Відрахування на єдиний соціальний внесок становить 22%:

I. $С_{ВІД} = С_{ЗП} \cdot 0.22 = 128107,46 \cdot 0.22 = 28183,64$ грн.

II. $С_{ВІД} = С_{ЗП} \cdot 0.22 = 129735,04 \cdot 0.22 = 28541,70$ грн.

Визначаємо витрати на оплату однієї машино-години. ($С_М$)

Так як одна ЕОМ обслуговує одного програміста з окладом 13000 грн., з коефіцієнтом зайнятості 0,2 то для однієї машини отримаємо:

$$С_Г = 12 \cdot М \cdot К_З = 12 \cdot 13000 \cdot 0,2 = 31200 \text{ грн.}$$

З урахуванням додаткової заробітної плати:

$$С_{ЗП} = С_Г \cdot (1 + К_З) = 31200 \cdot (1 + 0.2) = 37440 \text{ грн.}$$

Відрахування на соціальний внесок:

$$С_{ВІД} = С_{ЗП} \cdot 0.22 = 37440 \cdot 0,22 = 82368.8 \text{ грн.}$$

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 20000 грн.

$$С_А = К_{ТМ} \cdot К_А \cdot Ц_{ПР} = 1.15 \cdot 0.25 \cdot 20000 = 5750 \text{ грн.},$$

де $К_{ТМ}$– коефіцієнт, який враховує витрати на транспортування та монтаж приладу у користувача;

$К_А$– річна норма амортизації;

$Ц_{ПР}$– договірна ціна приладу.

Витрати на ремонт та профілактику розраховуємо як:

$$С_Р = К_{ТМ} \cdot Ц_{ПР} \cdot К_Р = 1.15 \cdot 20000 \cdot 0.05 = 1150 \text{ грн.},$$

де $К_Р$– відсоток витрат на поточні ремонти.

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$$Т_{ЕФ} =(Д_К – Д_В – Д_С – Д_Р) \cdot t_З \cdot К_В = (365 – 104 – 12 – 16) \cdot 8 \cdot 0.9 = \\ = 1677.6 \text{ годин},$$

де $Д_К$ – календарна кількість днів у році;

$Д_В$, $Д_С$ – відповідно кількість вихідних та святкових днів;

$Д_Р$ – кількість днів планових ремонтів устаткування;

t –кількість робочих годин в день;

$К_В$– коефіцієнт використання приладу у часі протягом зміни.

Витрати на оплату електроенергії розраховуємо за формулою:

$$С_{ЕЛ} = Т_{ЕФ} \cdot N_С \cdot К_З \cdot Ц_{ЕН} =1677.6 \cdot 0,3 \cdot 0,2 \cdot 3,52 = 354,3 \text{ грн.},$$

де $N_С$ – середньо-споживча потужність приладу;

$К_З$– коефіцієнтом зайнятості приладу;

$Ц_{ЕН}$ – тариф за 1 КВт-годин електроенергії.

Накладні витрати розраховуємо за формулою:

$$С_Н = Ц_{ПР} \cdot 0.67 = 20000 \cdot 0,67 =13400 \text{ грн.}$$

Тоді, річні експлуатаційні витрати будуть:

$$С_{ЕКС} = С_{ЗП} + С_{ВІД} + С_А + С_Р + С_{ЕЛ} + С_Н$$

$$С_{ЕКС} = 37440 + 7603.2 + 5750 + 1150 + 354,3 + 13400= 65697,5 \text{ грн.}$$

Собівартість однієї машино-години ЕОМ дорівнюватиме:

$$С_{М\text{-}Г} = С_{ЕКС} / Т_{ЕФ} = 65697,5/1677.6 = 39,16 \text{ грн/год.}$$

Оскільки в даному випадку всі роботи, які пов‘язані з розробкою програмного продукту ведуться на ЕОМ, витрати на оплату машинного часу, в залежності від обраного варіанта реалізації, складає:

$$С_М = С_{М-Г} \cdot Т$$

I. $С_М = 39,16 \cdot 1328,64 = 52029,54$ грн.

II. $С_М = 39,16 \cdot 1345.52 = 52690,56$ грн.

Накладні витрати складають 67% від заробітної плати:

$$С_Н = С_{ЗП} \cdot 0,67$$

I. $С_Н = 128107,46 \cdot 0,67 = 85831,90$ грн.

II. $С_Н = 129735,04 \cdot 0,67 = 86922,47$ грн.

Отже, вартість розробки ПП за варіантами становить:

$$С_{ПП} = С_{ЗП} + С_{ВІД} + С_М + С_Н$$

I. $С_{ПП} = 128107,46 + 28183,64 + 85831,90 = 242123,90$ грн.

II. $С_{ПП} = 129735,04 + 28541,70 + 86922,47 = 245198,21$ грн.

4.7 Вибір кращого варіанту ПП техніко-економічного рівня

Розрахуємо коефіцієнт техніко-економічного рівня за формулою:

$$K_{TEPj} = \frac{K_{Kj}}{C_{\Phi j}}$$

$$К_{ТЕР1} = 4{,}75/\ 242123{,}90 = 1{,}96 \cdot 10^{-5},$$

$$К_{ТЕР2} = 4{,}43/\ 245198{,}21 = 1{,}80 \cdot 10^{-5}.$$

Як бачимо, найбільш ефективним є перший варіант реалізації програми з коефіцієнтом техніко-економічного рівня $К_{ТЕР1} = 1{,}96 \cdot 10^{-5}$.

Після виконання функціонально-вартісного аналізу програмного комплексу що розробляється, можна зробити висновок, що з альтернатив, що залишились після першого відбору двох варіантів виконання програмного комплексу оптимальним є перший варіант реалізації програмного продукту. У нього виявився найкращий показник техніко-економічного рівня якості

$$К_{ТЕР} = 1{,}96 \cdot 10^{-5}.$$

Цей варіант реалізації програмного продукту має такі параметри:

- мова програмування – Python;
- Використання методів машинного навчання
- Використання стандартного інтерфейсу візуалізації, швидкість розробки

Даний варіант виконання програмного комплексу дає користувачу зручний інтерфейс, непоганий функціонал і швидкодію.

4.8 Висновки до розділу

У даному розділі було проведено повний функціонально-вартісний аналіз програмного продукту. Визначено та проведено оцінку основних функцій програмного продукту. Визначено параметри, які характеризують

програмний продукт. Проведено експертне оцінювання параметрів та аналіз якості варіантів реалізації функцій. На основі аналізу було обрано найкращий варіант реалізації програмного продукту.

ВИСНОВКИ

У даній дипломній роботі було розглянуто актуальність аналізу великих у різних сферах життя. Особливу увагу було звернено на аналіз тональності текстових даних. Для прикладу реалізації було обрано сферу кіноіндустрії та аналіз великих неструктурованих даних у вигляді відгуків користувачів.

Крім цього, для вирішення поставленої задачі було проведено аналіз існуючих методів категоризації настрою, зокрема нейронних мереж, які були використані в подальшому для бінарної класифікації коментарів користувачів на позитивні та негативні.

У результаті було реалізовано методи для аналізу тональності тексту, натреновано модель та реалізовано інтелектуальну систему, що автоматизує процес розпізнавання настрою коментаря.

Також було проведено функціонально-вартісний аналіз розробленого продукту, а також було проаналізовано роботу розробленого продукту, завдяки чому було визначено які аспекти можна вдосконалити та які є способи інтеграції даної системи в уже існуючі додатки.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Optimize the utility of Unstructured Text with Text Analytics. Datafloq: Driving Innovation through Data. URL: https://datafloq.com/read/optimize-utility-unstructured-text-text-analytics/9815
2. Чубукова И.А. Data Mining: учебн. пособ. – М.: Интернет-университет информационных технологий БИНОМ: Л (date of access: 17.05.2021).аборатория знаний, 2006. – 382 с.
3. Data-mining-for-project-management URL: http://iasa.kpi.ua/education-uk/bachelor/data-mining-for-project-management-uk
4. Survey of Text Mining I: Clustering, Classification, and Retrieval / Ed. by M. W. Berry. — 2004. — Springer, 2003. — 261 с. — ISBN 0387955631.
5. Drake J. E. Encyclopedia of Library and Information Science, Second Edition (Online Version). Marcel Dekker, 2003. 3500 p.
6. A comprehensive analysis and survey of the theatrical and home/mobile entertainment market environment for2019.
   URL:
   https://www.mpa-apac.org/wp-content/uploads/2020/03/MPA-THEME-2019.pdf
7. A comprehensive analysis and survey of the theatrical and home/mobile entertainment market environment for2020.
   URL:
   https://www.mpa-apac.org/wp-content/uploads/2020/03/MPA-THEME-2020.pdf
8. Вагин В. Н. Достоверный и правдоподобный вывод в интеллектуальных системах / В. Н. Вагин, Е. Ю. Головина, А. А. Загорянская, М. В. Фомина. Москва : Физматлит, 2004. 704 с.37 Big

9. Inc B. Bitstream 1. Cambridge, Mass : Bitstream Inc.
10. CJC Burges. A Tutorial on Support Vector Machines for Pattern Recognition [Електронний ресурс] // – Режим доступу : http://www.music.mcgill.ca/rfergu/adamTex/references/Burges98.pdf
11. Барсегян А. А. Анализ данных и процессов: учеб. пособие / А. А. Барсегян, М. С. Куприянов, И. И. Холод, М. Д. Тесс, С. И. Елизаров. 3-е изд., перераб. и доп. Санкт-Петербург : БХВ-Петербург, 2009. 512 с.
12. БАЙЄСІВСЬКІ МЕРЕЖІ В СИСТЕМАХ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ / М. З. Згуровський та ін. URL: https://ela.kpi.ua/bitstream/123456789/19582/1/SPPR_01072015.pdf.
13. Sebastiani F. Machine learning in automated text categorization / F. Sebastiani // ACM Comput. Surv. – March 2010. – Vol. 34, No. 1. P. 1-47
14. Барсегян А. А. Технология анализа данных: Data Mining, Visual Mining, Text Mining, OLAP / А. А. Барсегян. – Санкт-Петербург : БХВ-Петербург, 2007. – 384 с. 10.
15. Chitumba H. O., Chinduma A. T. M. https://www.rfbeditora.com/catalogo. RFB Editora, 2020. URL: https://doi.org/10.46898/rfb.9786558890638
16. Keras documentation: About Keras. Keras: the Python deep learning API. URL: https://keras.io/about/
17. IMDb: Ratings, Reviews, and Where to Watch the Best Movies & TV Shows. IMDb: Ratings, Reviews, and Where to Watch the Best Movies & TV Shows. URL: https://www.imdb.com/ (date of access: 25.05.2021).
18. Учасники проектів Вікімедіа. Internet Movie Database – Вікіпедія. Вікіпедія. URL: https://uk.wikipedia.org/wiki/Internet_Movie_Database
19. https://konferencefzsplzen.files.wordpress.com/2020/11/sbornik-2020.pdf / ed. by J. Frei. Západočeská univerzita, 2020. URL: https://doi.org/10.24132/cpvo.2020.09.17-135.26109631

20. https://konferencefzsplzen.files.wordpress.com/2020/11/sbornik-2020.pdf / ed. by J. Frei. Západočeská univerzita, 2020. URL: https://doi.org/10.24132/cpvo.2020.09.17-135.26109631 (date of access: 17.05.2021).
21. Computer Science | University of Illinois at Chicago. URL: https://www.cs.uic.edu/~liub/S-EM/unlabelled.pdf
22. Data Case Studies with Big Results. Schaefer Marketing Solutions: We Help Businesses {grow}. URL: https://businessesgrow.com/2016/12/06/big-data-case-studies/
23. Yang Y. A re-examination of text categorization methods / Y. Yang, X. Liu // Proc. SIGIR'2012, 22nd ACM International Conference on Research and Development in Information Retrieval, 2012. P. 42-49.
24. Baeza-Yates, B. Ribeiro-Neto, "Modern Information Retrieval", ACM, Press, Page 64, Year 1999.
25. Vasudev. What is One Hot Encoding? Why and When Do You Have to Use it? | Hacker Noon. Hacker Noon. URL: https://hackernoon.com/what-is-one-hot-encoding-why-and-when-do-you-have-to-use-it-e3c6186d008f
26. Что такое эмбеддинги и как они помогают искусственному интеллекту понять мир людей. Наука и жизнь. URL: https://www.nkj.ru/open/36052/

ДОДАТОК А

```
from tensorflow.keras.datasets import imdb
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Embedding, Flatten, Dropout
from tensorflow.keras import utils
from tensorflow.keras.preprocessing.sequence import pad_sequences
import numpy as np
import matplotlib.pyplot as plt
from google.colab import files
import string
from requests import get
from bs4 import BeautifulSoup as Soup

max_words=10000
(x_train, y_train), (x_test, y_test) = imdb.load_data(num_words=max_words)

word_index = imdb.get_word_index()

def get_review_text(review_int_vector):
    word_index = imdb.get_word_index()
            reverse_word_index = dict([(value, key) for (key, value) in word_index.items()])
        decoded_review = ' '.join([reverse_word_index.get(i - 3, '?') for i in review_int_vector])
    return decoded_review
```

```
maxlen = 200
x_train = pad_sequences(x_train, maxlen=maxlen, padding='post')
x_test = pad_sequences(x_test, maxlen=maxlen, padding='post')

model = Sequential()
model.add(Embedding(max_words, 2, input_length=maxlen))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='binary_crossentropy',
              metrics=['accuracy'])
history = model.fit(x_train, y_train, epochs=15, batch_size=128,
validation_split=0.1)


history_dict = history.history
history_dict.keys()


loss = history.history['loss']
val_loss = history.history['val_loss']
accuracy = history.history['accuracy']
val_accuracy = history.history['val_accuracy']


plt.plot(accuracy, label= 'Тренувальний набір')
```

```
plt.plot(val_accuracy, label='Тестовий набір')
plt.xlabel('Епоха навчання')
plt.ylabel('Частка правильних відповідей')
plt.legend()
plt.show()


plt.plot(loss, label= 'Тренувальний набір')
plt.plot(val_loss, label='Тестовий набір')
plt.xlabel('Епоха навчання')
plt.ylabel('Втрати')
plt.legend()
plt.show()

scores = model.evaluate(x_test, y_test, verbose=1)
print("Частка правильних відповідей на тестових даних:", round(scores[1] * 100, 4))

def find_word_code(word):
  word_index = imdb.get_word_index()
  word_index = {k: v for k, v in word_index.items() if v < 10000}
  code = word_index.get(word)
  if type(code) == int:
    return code + 3
  else:
    return 0
def create_reviewWordsArray(text_example):
  words = text_example.lower().split()
```

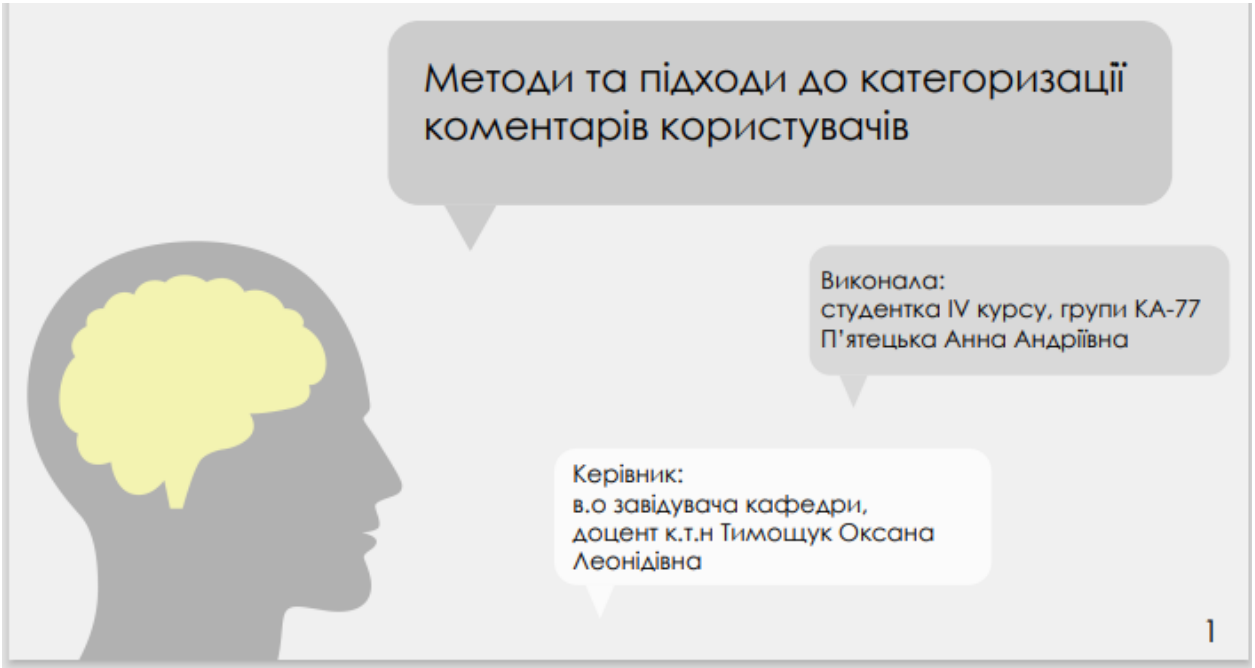
```
  table = str.maketrans('', '', string.punctuation)
  reviewWordsArray = [w.translate(table) for w in words[:200]]
  codeArray = map(lambda x: find_word_code(x), reviewWordsArray)
  return list(codeArray)
```

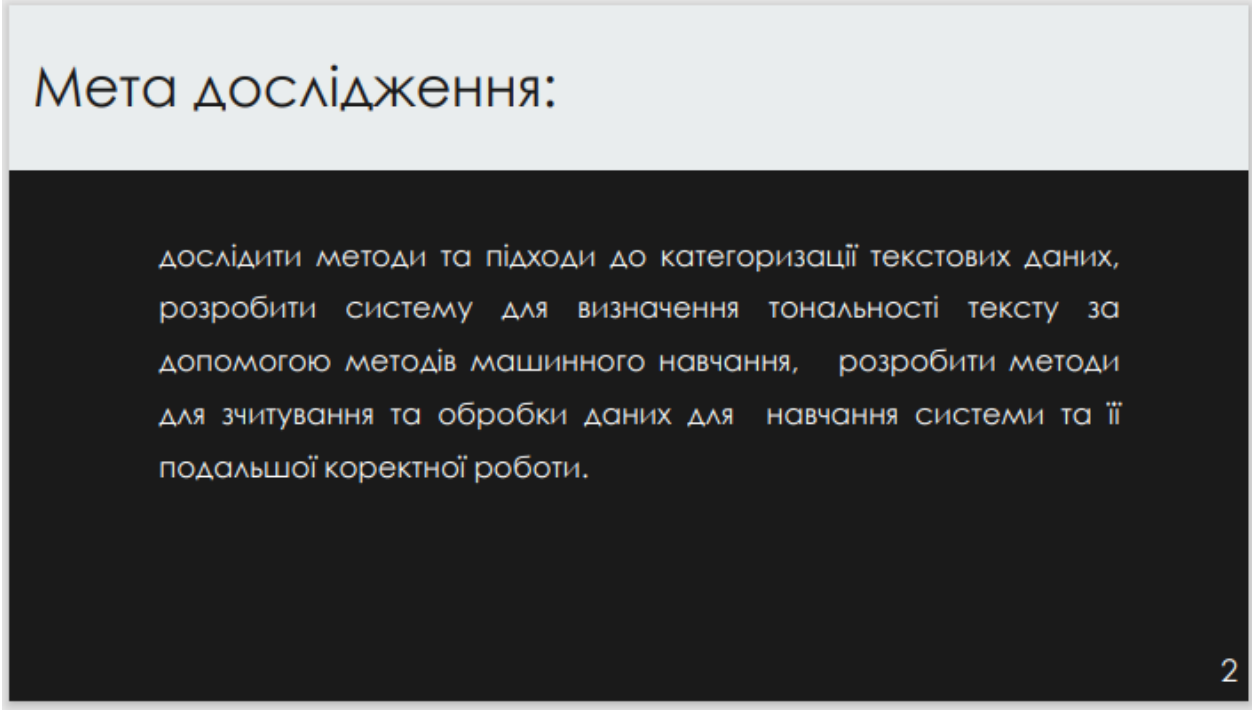
Введіть нижче посилання на фільм з сайту IMDB щоб дізнатись статистику користувацьких відгуків.

```
url = get("https://www.imdb.com/title/tt7131622/reviews?ref_=tt_urv")
request = url.text
from bs4 import BeautifulSoup as Soup
soup_data = Soup(request, 'html.parser')
soup_data.title.text
reviews = soup_data.findAll('div', {'class':'text show-more__control'})
commentsList = []
i = 0
for i in range(len(reviews)):#len(reviews)
  currentText = create_reviewWordsArray(reviews[i].text)
  commentsList.append(currentText)
commentsList

commentsList = pad_sequences(commentsList, maxlen=maxlen, padding='post')
x_testik = np.array(commentsList).astype('int32')
x_testik

result = model.predict(x_testik)
result

values = [0,0,0]
for i in result:
```

```
  if float(i) >= 0.7: values[0] = values[0] + 1
  elif float(i) <= 0.4: values[1] = values[1] + 1
  else: values[2] = values[2] + 1
labels = ['Positive','Negative','Neutral']
colors = ['yellow','pink','lightblue']
plt.pie(values,labels=labels,autopct='%1.0f%%', colors=colors)
plt.axis('equal')
print('Movie: '+ soup_data.title.text.split('-')[0])
print('In total there are ' + soup_data.find('div', {'class':'header'}).text[1:7] + 'user reviews on the IMDB site')
plt.title('These comments can be divided into such categories based on the emotional dimension.')
plt.show()
```

ДОДАТОК Б


Методи та підходи до категоризації коментарів користувачів
Виконала:
студентка IV курсу, групи КА-77
П'ятецька Анна Андріївна
Керівник:
в.о завідувача кафедри,
доцент к.т.н Тимощук Оксана Леонідівна
1

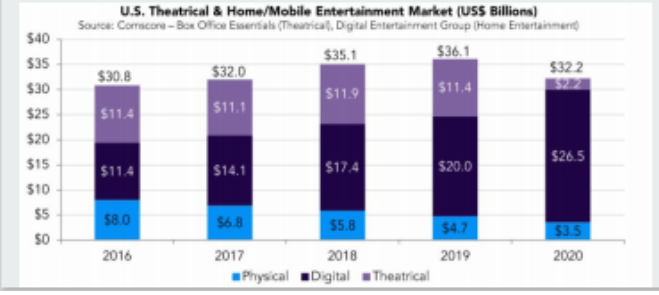

Мета дослідження:
дослідити методи та підходи до категоризації текстових даних, розробити систему для визначення тональності тексту за допомогою методів машинного навчання, розробити методи для зчитування та обробки даних для навчання системи та її подальшої коректної роботи.
2

Актуальність

- ➔ за період свого існування людство створило надзвичайно великі масиви інформації, яка потребує аналізу.
- ➔ з середини 90-х років в інформаційній галузі зростає інтерес до технологій аналізу даних
- ➔ коментарі користувачів - важлива складова даних, проаналізувавши які можна зрозуміти настрій відвідувачів сторінки та знайти основні характеристики в яких вони зацікавлені

3

Кіноіндустрія



не дивлячись на пандемію та зниження прибутку підприємств в інших сферах, ця галузь тримається на плаву, на відміну від ресторанної сфери, сфери туризму та багатьох інших. Крім цього, люди звертають ще більше уваги на кіноіндустрію, адже навіть в такий складний час для людей - це спосіб відволіктись та розслабитись під час буденних справ.

4

Об'єкт дослідження:

Категоризація неструктурованих текстових даних

Предмет дослідження:

Визначення тональності коментарів користувачів до кінострічок за допомогою методів нейронних мереж

5

МЕТОДИ АНАЛІЗУ ТЕКСТОВИХ ДАНИХ

6

Методи обробки природної мови аналізують текст на лінгвістичному рівні, чого достатньо для значної частини задач. Але для того, щоб визначити чи є коментар до фільму позитивним, чи негативним необхідно враховувати семантику. Для вирішення цього існує сентимент-аналіз, або аналіз тональності тексту. Він розроблений для того, щоб в автоматизованому порядку знаходити в текстах емоційно забарвлену лексику, а також емоційну оцінку автора щодо об'єктів про які йдеться в тексті.

7

Основне завдання при аналізі тональності - класифікувати полярність документа, тобто визначити, чи є думка, виражена в документі або пропозиції, позитивним, негативним або нейтральним. У більш широкому сенсі "позаполярна" класифікація тональності виражається, наприклад, такими емоційними станами, як «зло», «смуток» і «щастя».

8

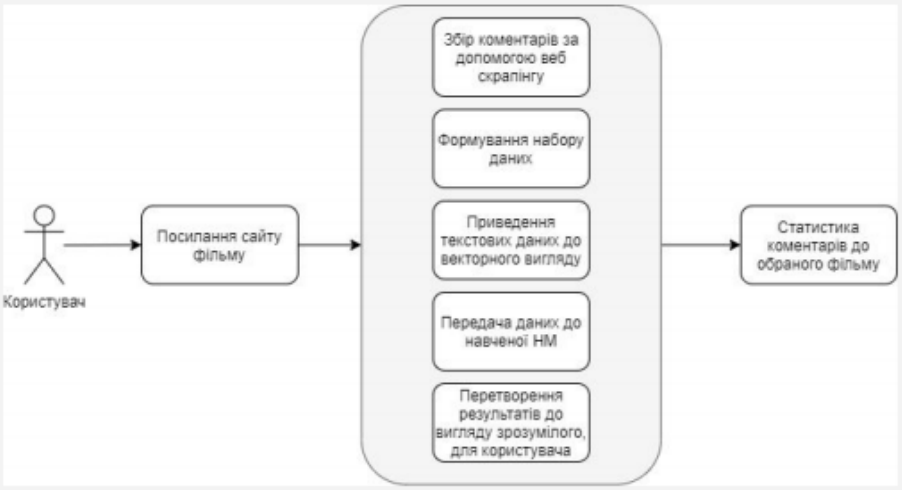
Лексичний підхід	Підхід з машинним навчанням
полягає в пошуку словникової зв'язку для аналізу тексту. Підхід залежить від знаходження лексикону. Загалом існує два основних методи: словниковий та корпусний, або метод, що базується на ядрі.	поділяється на методи з наглядом та без. Перші беруть за основу марковані документи у великій кількості. Методи без нагляду застосовуються, у випадку відсутності класифікованих навчальних даних, або якщо їх зовсім не багато.

9

Програмна реалізація

10

Архітектура програмного продукту



11

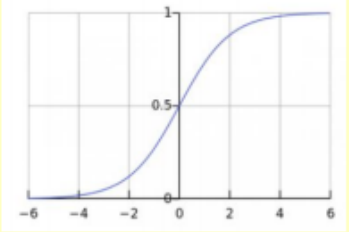
Нейронна мережа

Метрика Accuracy - це показник, який описує загальну точність передбачення моделі по всіх класах. Це особливо корисно, коли кожен клас однаково важливий. Він розраховується як відношення кількості правильних прогнозів до їх загальної кількості

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Функція активації: сігмоїд

$$A = \frac{1}{1+e^{-x}}$$



Оптимізатор: Adam

Правило оновлення:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t + \epsilon}}\hat{m}_t$$

12

Засоби розроблення

python™

Keras

colab

13

База даних IMDB

```
{'fawn': 34701,
 'tsukino': 52006,
 'nunnery': 52007,
 'sonja': 16816,
 'vani': 63951,
 'woods': 1408,
 'spiders': 16115,
 'hanging': 2345,
 'woody': 2289,
 'trawling': 52008,
 "hold's": 52009,
 'comically': 11307,
 'localized': 40830,
 'disobeying': 30568,
 "'royale": 52010,
```

У наборі даних IMDB використовується частотне кодування слів

```
1 -> the
2 -> and
3 -> a
4 -> of
5 -> to
6 -> is
7 -> br
8 -> in
9 -> it
10 -> i
11 -> this
12 -> that
13 -> was
14 -> as
15 -> for
16 -> with
17 -> movie
18 -> but
19 -> film
20 -> on
```

20 найвживаніших слів

14

Приклад закодованого і розкодованого відгуку

[1, 778, 128, 74, 12, 630, 163, 15, 4, 1766, 7982, 1051, 2, 32, 85, 156, 45, 40, 148, 139, 121, 664, 665, 10, 10, 1361, 173, 4, 749, 2, 16, 3804, 8, 4, 226, 65, 12, 43, 127, 24, 2, 10, 10]

При підготовці набору даних використовувалось кодування вигляду:

0 - символ заповнювач,

1 - початок послідовності,

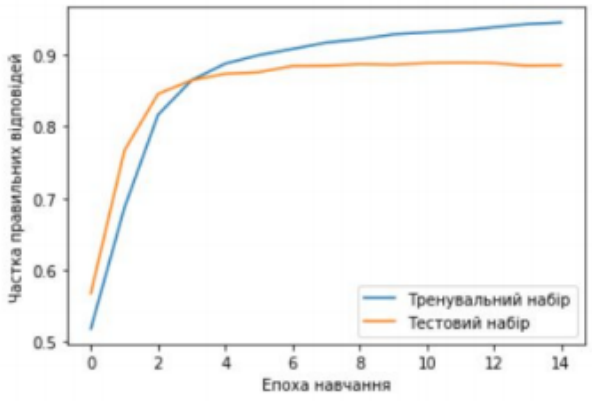
2 - слово, якого немає в словнику.

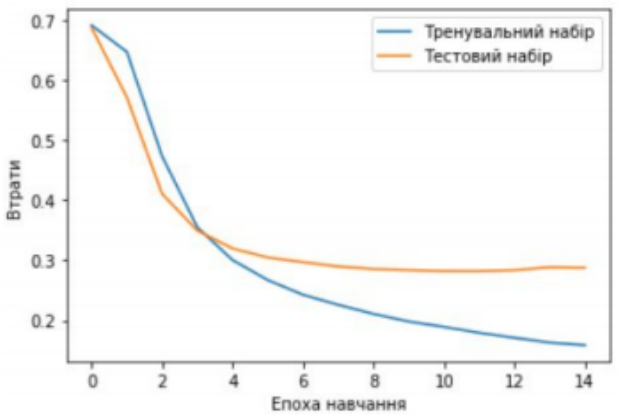
```
'? begins better than it ends funny t
hat the russian submarine crew ? all
other actors it's like those scenes w
here documentary shots br br spoiler
part the message ? was contrary to th
e whole story it just does not ? br b
r'
```

"?" заміняє одне з зазначених кодувань

15

Після навчання даної мережі маємо такі показники з кількістю епох навчання, що дорівнює 15

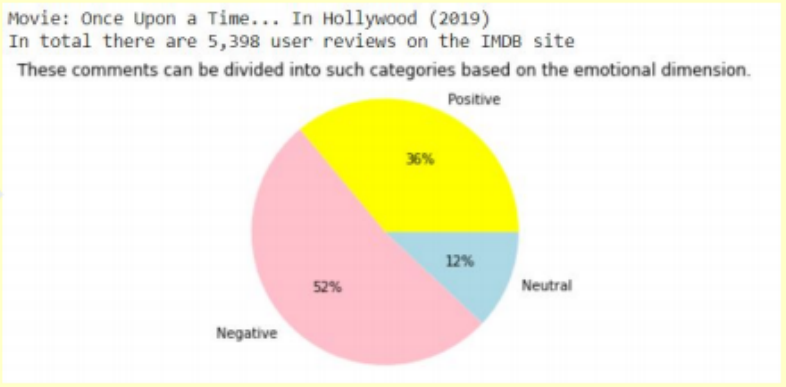




16

```
array([[0.64740694],
       [0.57226187],
       [0.1697506 ],
       [0.9108463 ],
       [0.32150325],
       [0.02472082],
       [0.9930382 ],
       [0.03493306],
       [0.9973676 ],
       [0.0235239 ],
       [0.14306194],
       [0.18972954],
       [0.8541038 ],
       [0.22364104],
       [0.9909985 ],
```

Фрагмент результату роботи Нейронної мережі



Результат для користувача

17

Вдосконалення розробленого програмного продукту

- ➔ оптимізація коду для збільшення швидкості
- ➔ розробка повноцінного додатку для різних пристроїв
- ➔ імплементація у вже існуючі сервіси з великою кількістю відгуків, що потребують аналізу

18

Висновки:

- було розглянуто актуальність аналізу великих у різних сферах життя
- особливу увагу було звернено на аналіз тональності текстових даних.
- для прикладу реалізації було обрано сферу кіноіндустрії та аналіз великих неструктурованих даних у вигляді відгуків користувачів
- було проведено аналіз існуючих методів категоризації настрою, зокрема нейронних мереж, які були використані в подальшому для бінарної класифікації коментарів користувачів на позитивні та негативні
- було натреновано модель
- реалізовано інтелектуальну систему, що автоматизує процес розпізнавання настрою коментаря

19

Дякую за увагу ☺

20