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

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

До захисту допущено:

Завідувач кафедри

_____ Олександр РОЛІК

«___»_____20__ р.

Дипломний проєкт

на здобуття ступеня бакалавра
за освітньо-професійною програмою «Інформаційне забезпечення робототехнічних систем»»
спеціальності 126 «Інформаційні системи та технології»
на тему: «Рекомендаційна система на базі алгоритму колаборативної фільтрації»

Виконав:
студент IV курсу, групи ІК-93
Петренко Владислав Вікторович _____

Керівник:
доцент кафедри ІСТ, к.т.н
Галушко Дмитро Олександрович _____

Рецензент:
асистент кафедри ОТ
Алещенко Олексій Вадимович _____

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

Київ – 2023 року

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

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 126 «Інформаційні системи та технології»

Освітньо-професійна програма «Інформаційне забезпечення робототехнічних систем»

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

ЗАВДАННЯ

на дипломний проєкт студенту

Петренку Владиславу Вікторовичу

1. Тема проєкту «Рекомендаційна система на базі алгоритму колаборативної фільтрації», керівник проєкту Галушко Дмитро Олександрович, доцент кафедри ІСТ, к.т.н, затверджені наказом по університету від «31» травня 2023 р. № 2101-с

2. Термін подання студентом проєкту: 12 червня 2023 року

3. Вихідні дані до проєкту: файли з наборами даних anime.csv та rating.csv

4. Зміст пояснювальної записки:
1. Дослідження предметної області: огляд існуючих рішень, які використовують рекомендаційні системи, наявні підходи створення систем колаборативної фільтрації, специфіка предметної області.
2. Аналіз методів надання рекомендацій: змістовна постановка задачі, математична постановка задачі, способи розрахунку схожості між елементами, метод прогнозування оцінки елемента, порівняльний аналіз способів знаходження подібності елементів, обгрунтування методів вирішення.
3. Проектування програмного забезпечення: інформаційне забезпечення, засоби розробки, проектування системи.

4. Реалізація програмного рішення: попередня нормалізація та обробка даних, специфікація функцій застосунку, керівництво користувача, випробовування розробленого програмного продукту, тестування рекомендаційної системи.

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслеників, плакатів, презентацій тощо)
1. Діаграма компонентів
2. Діаграма послідовності
3. Діаграма станів
4. Діаграма потоків даних
5. Структурна схема

6. Дата видачі завдання 1 грудня 2022 року

Календарний план

№ з/п	Назва етапів виконання проєкту	Термін виконання етапів проєкту	Примітка
1.	Затвердження теми роботи	11.02.23	
2.	Аналіз наявних рекомендаційних систем у вигляді компоненту в існуючих рішеннях	17.04.23 – 23.04.23	
3.	Аналіз алгоритмів фільтрації та способів знаходження схожих елементів	24.04.23 – 30.04.23	
4.	Проектування програмного забезпечення	01.05.23 – 07.05.23	
5.	Розробка програмного продукту	08.05.23 – 14.05.23	
6.	Впровадження системи та тестування продукту	15.05.23 – 21.05.23	
7.	Оформлення текстової документації	22.05.23 – 06.06.23	

Студент Владислав ПЕТРЕНКО

Керівник Дмитро ГАЛУШКО

АНОТАЦІЯ

Ключові слова: середньоквадратична помилка, середня абсолютна помилка, рекомендаційна система, матриця, рейтинг, користувач, метрика, алгоритм колаборативної фільтрації.

Пояснювальна записка дипломного проєкту містить 4 розділи, 18 рисунки, 12 таблиць, 19 джерел.

Метою роботи є полегшення пошуку контенту користувачам зі схожими вподобаннями, для досягнення якої виконується завдання з реалізації рекомендаційної системи на базі алгоримту колаборативної філтьтрації.

Для реалізації системи було використано: мова програмування Python, бібліотеки Numpy, Pandas, Scikit-learn, фреймворк Pyforms-GUI, середовище розробки Visual Studio Code.

У дипломному проєкті було розроблено рекомендаційну систему на базі алгоримту колаборативної фільтрації, що прогнозує оцінку елемента для визначеного користувача, знаходить схожі на визначений елементи, формує вибірку з рекомендованих елементів використовуючи коефіцієнти подібності.

Дана розробка в подальшому може бути використана для побудови повноцінного сервісу чи застосунку потокового відтворення відео, або бути впровадженою у вже існуюче рішення, чи розширена за рахунок моделі машинного навчання і власної бази даних.

ANNOTATION

Keywords: mean square error, mean absolute error, recommendation system, matrix, rating, user, metric, collaborative filtering algorithm.

The explanatory note of the thesis project contains 4 chapters, 18 figures, 11 tables and 19 sources.

The aim of the work is to facilitate the search for content for users with similar preferences, to achieve which the task of implementing a recommendation system based on the collaborative filtering algorithm is performed.

To implement the system, we used the Python programming language, Numpy, Pandas, Scikit-learn libraries, Pyforms-GUI framework, and Visual Studio Code development environment.

In the diploma project, a recommendation system based on a collaborative filtering algorithm was developed that predicts the rating of an item for a specific user, finds items similar to a specific one, and forms a sample of recommended items using similarity coefficients.

This development can be further used to build a full-fledged video streaming service or application, or be implemented in an existing solution, or expanded with a machine learning model and its own database.

Номер рядка	Формат	Позначення	Найменування	Кільк. аркушів	Номер екзем.	Примітка
1			Документація загальна			
2						
3			Знову розроблена			
4						
5	А4	ІК93.150БАК.005 ПЗ	Пояснювальна записка	67		
6	А3	ІК93.150БАК.005 Д1	Діаграма компонентів	1		
7						
8	А3	ІК93.150БАК.005 Д2	Діаграма послідовності	1		
9						
10	А3	ІК93.150БАК.005 Д3	Діаграма станів	1		
11						
12	А3	ІК93.150БАК.005 Д4	Діаграма потоків даних	1		
13						
14	А3	ІК93.150БАК.005 Д5	Структурна схема	1		
15						
16						
17						
18						
19						
20						
21						
22						
23						
24						
25						
26						
27						
28						

Зм.	Аркуш	№ докум.	Підпис	Дата	ІК93.150БАК.005 ТП		
Розроб.		Петренко В.В.			Рекомендаційна система на базі алгоритму колаборативної фільтрації. Відомість проєкту	Літ.	Аркуш 1 / Аркушів 1
Керівн.		Галушко Д.О.				Т	
						КПІ ім. Ігоря Сікорського	
						Група ІК-93	
Затв.							

Пояснювальна записка
до дипломного проєкту
на тему: «Рекомендаційна система на базі алгоритму колаборативної фільтрації»

Київ – 2023 року

ЗМІСТ

ВСТУП

З теоретичної точки зору слід зазначити, що люди завжди споживали величезні об’єми контенту та платили за різноманітні послуги, товари тощо. Переглядаючи фільми, серіали, мультфільми, читаючи новини, прослуховуючи музичні композиції, купуючи що-небудь - у кожного складається своє враження від отриманої взаємодії. Таке особистісне сприйняття називають користувацьким досвідом. До появи веб-сервісів ним ділились між собою, порівнювали його, вступали в дискусії, обговорюючи деталі, після чого робили висновки і приблизно знали, які у співрозмовників спільні вподобання й могли порадити один одному одиницю контенту або товару для вживання у майбутньому. Ця модель людської поведінки пізніше лягла в основу механізму рекомендаційних систем.

З розвитком міжнародної комп’ютерної мережі, вона почала стрімко наповнюватись різноманітними веб-сервісами для перегляду, торгівельними інтернет-майданчиками, новинними стрічками, стрімінговими сервісами тощо. Якщо раніше споживачі не мали великого вибору де брати контент та купувати послуги, то наразі вони отримали все в одному місці – в інтернеті. Як відомо, чим більший попит, тим швидше зростає конкуренція та кількість пропозицій для потенційних глядачів, читачів, клієнтів та слухачів. Такі сервіси почали надавати можливість залишати рейтинг предмету зацікавленості, шляхом виставлення оцінок, написання відгуків, коментарів, рецензій та всього іншого, що значно полегшувало іншим пошуки бажаного, та підвищувало якість подальшої взаємодії.

Але був великий недолік, який полягав у тому, що користувач фізично не був у змозі швидко проаналізувати такий великий обсяг інформації, який, до того ж, постійно зростав. Ця проблема стала потужним поштовхом для розробки рекомендаційних систем, які дозволили автоматизувати процес обробки величезних наборів даних та прогнозування найкращих варіантів, враховуючи індивідуальні вподобання кожного.

Отож, рекомендаційна система – це система, що використовується для прогнозування особистих інтересів людей, шляхом аналізу зібраної про них

інформації, для створення їм рекомендацій з метою збільшення загальної зацікавленості до певних об’єктів.

Крім того, що таким чином значно полегшується пошук контенту користувачам, рекомендаційні системи та їх розвиток є важливим в рамках економічного та бізнес-секторів, формуванні та підтримуванні бізнес-процесів. Наприклад, компанії можуть забезпечити більш ефективний маркетинг та продажі, якщо вони зможуть пропонувати свої продукти та послуги відповідно до інтересів та вподобань своїх клієнтів. В цьому контексті системи надання рекомендацій є надзвичайно вагомим інструментом для підвищення конкурентноспроможності компаній та покращення їх бізнес-стратегій.

У сучасному світі, майже будь-який великий сервіс з надання послуг клієнтам має свою рекомендаційну систему. Сьогодні недостатньо просто створити гарний дизайн сайту чи надати широкий спектр для вибору. Головним є користувацький досвід, який напряму залежить від того, наскільки якісно система взаємодіє зі споживачем. Отже, можна стверджувати, що наявність потужної рекомендаційної системи надасть величезну перевагу перед більшістю існуючих рішень, а їх актуальність впровадження кожного дня тільки зростає.

За останні роки, завдяки стрімкому розвитку галузі машинного навчання та штучного інтелекту, був розроблений новий вид рекомендаційної системи, яка використовує модель нейронної мережі у своїй архітектурі, що значно покращує якість рекомендованих вибірок [1]. Але також, відкритою залишається проблема ефективності та використання таких систем у реальних умовах. В цілому, вивчення та використання рекомендаційних систем на основі нейронних мереж має великий потенціал та може привести до розвитку нових технологій та ринків, що забезпечить важливий внесок у галузь інформаційних технологій та економіки в цілому. Це демонструє, що рекомендаційні системи є порівняно новим об’єктом для досліджень.

На сьогоднішній день існують різні типи та види рекомендаційних систем з використанням різних алгоритмів фільтрації даних, які будуть розглянуті у даній роботі. У дипломному проєкті було розроблено систему, базовану на підході

колаборативної фільтрації на основі елементів. Тобто рекомендації користувачу створюються на основі схожості між предметами, які він раніше споживав.

Мета і завдання дослідження. Метою роботи є полегшення процесу пошуку контенту користувачам зі схожими вподобаннями.

Для досягнення поставленої мети необхідно вирішити ряд важливих задач:

– проаналізувати існуючі рекомендаційні системи. Проаналізувати такі системи, як частина готових рішень;

– провести аналіз методів надання рекомендацій, для реалізації системи. Дослідити алгоритми для фільтрації контенту;

– реалізувати рекомендаційну систему на базі обраного алгоритму колаборативної фільтрації;

– оцінити якість розробленої системи.

Об'єктом дослідження є медіаконтент, що переглядається користувачем.

Предметом дослідження є алгоритми колаборативної фільрації та метрики для обрахунку подібності між елементами.

Дипломний проєкт складається з наступних розділів: вступ, основні розділи, висновки, список використаних джерел із 19 найменувань. Графічна частина включає 5 креслеників формату А3. Загальний обсяг 67 сторінок.

1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

Системи надання рекомендацій є потужним інструментом, який впроваджують у свої сервіси такі гіганти веб-ринку та електронної комерції, як Amazon Incorporated, Netflix, Spotify Technology S.A., Google Inc., Alibaba group та багато інших. Також слід зазначити вітчизняні сервіси, які використовують рекомендаційні системи. У сфері інтернет-продажів таким, безумовно, є Rozetka, а у розрізі потокового відтворення відео лідером є Megogo.

Megogo 2 рази проводив свій конкурс, схожий на легендарний конкурс від Netflix – Netlix Prize, який має назву MEGOGO Media Hackaton – MEGOGO Kaggle Challenge, і в останнє відбувся у 2019 році. Компанія надала доступ до анонімізованих реальних даних про активність користувачів за 3 місяці. Учасники повинні були створити високоточне відтворюване рішення, яке було б в змозі на основі цих даних передбачити, що користувачі Megogo будуть дивитись у наступному місяці. Команди й учасники, розроблені моделі яких зайняли перші три місця, отримали, відповідно, 2000$, 1000$ та 500$. Хоча призовий фонд набагато менший від того, який був у Neflix, а часу надали всього 48 годин, конкуренція була не меншою. Це демонструє, що об’єкт та предмет дослідження на сьогодні, безперечно, є актуальними як за кордоном, так і в Україні.

З одного боку, рекомендаційні системи можуть зробити життя користувачів простішим і зручнішим, надаючи персоналізовані пропозиції товарів, послуг, музики та відео. Науковці з компанії Netflix розробили дослідження, в якому з'ясували, що більш ніж 80% відомих на Netflix фільмів та телешоу були знайдені за допомогою рекомендаційної системи. Це означає, що рекомендаційні алгоритми можуть допомогти користувачам знайти контент, який їм сподобається, і збільшити їхню вірогідність перегляду цього контенту.

З іншого боку, рекомендаційні системи можуть допомогти бізнесу підвищити продажі та прибуток. Наприклад, дослідження, проведене компанією Amazon, показало, що рекомендації товарів на їхньому сайті допомогли збільшити продажі

на 29%. Це означає, що рекомендаційні системи можуть допомогти бізнесу ефективніше пропонувати свої товари та послуги й збільшувати свої прибутки [2].

Отже, рекомендаційні системи мають великий потенціал у розвитку бізнесу та підвищенні задоволення користувачів, тому проведення досліджень з цієї теми є дуже важливим та релевантним на сьогодні.

1.1 Огляд існуючих рішень, які використовують рекомендаційні системи

Як правило, сьогодні рекомендаційні системи впроваджені майже в кожне рішення у веб-середовищі, а кількість таких рішень зростає кожного дня в геометричній прогресії через фактор зацікавленості користувачів у них. В залежності від контентної частини, системи фільтрації інформації можуть відрізнятися своїми особливостями реалізації та багатьма іншими ключовими аспектами, наприклад, такими вагомими, як ціна. У більшості випадків найпоширеніші сервіси надають рекомендації таких типів елементів, як:

- відео (фільми, серіали, мультфільми і т. д.);
- музичні композиції;
- послуги (страхування, туризм тощо);
- новини;
- товари.

Для повноти дослідження слід оглянути декілька існуючих сервісів, які рекомендують одиницю контенту, або вибірку з елементів, типи яких зазначені у переліку вище (відео, музичні композиції та товари). На основі цього, буде обрано сферу та вузьковизначене направлення (якщо це відео, то буде обраний конкретний підвид) для впровадження власної підсистеми.

1.1.1 Spotify Web Player

Не так давно зайшовший на український ринок, всесвітньо відомий сервіс стрімінгової музики – Spotify. Він надає можливість користувачам прослуховувати

музику онлайн та зберігати її у своїй бібліотеці. Системи, що надають рекомендації в Spotify допомагають користувачам знайти нові музичні композиції, які можуть їм сподобатись, на основі їхніх прослуховувань та вподобань.

Основна рекомендаційна система Spotify називається “Discover Weekly”. Кожного тижня Spotify створює персоналізований плейлист для кожного користувача на основі його прослуховувань та поведінки на платформі. Для їх створення використовується аналіз даних та машинне навчання, що оцінюють музичні вподобання та формують рекомендаційну підбірку, що складається з нових пісень та виконавців. Окрім цього, такі системи також використовуються для створення інших плейлистів та функцій. Наприклад, “Daily Mix” – це функція, яка автоматично формує підбірку музики, яка вже є в бібліотеці користувача та доповнює її новими піснями, що можуть сподобатись. Також Spotify рекомендує елементи, які схожі на ті, які вже були додані до бібліотеки. Крім того, використовує ці системи для пропонування користувачам концертів та подій з улюбленими виконавцями, що є величезною перевагою перед іншими сервісами стрімінгової музики. Типовими явними даними для збору про користувачів є проставлення ними оцінок “подобається” чи “не подобається”. Ці дані використовуються для півищення ефективності рекомендаційних систем у додатку та забезпечення більш персоналізованого досвіду користувача [3].

Усі ці рекомендаційні системи в Spotify забезпечують користувачів більш індивідуальним досвідом прослуховування музики, дозволяють знайти нову музику та виконавців, які можуть сподобатись, та допомагають зберегти час, який користувачі витрачають на пошук нової музики самостійно.

1.1.2 Медіасервіс Megogo

Найбільш поширений онлайн-сервіс для перегляду відео контенту на теренах України на сьогодні - платформа Megogo. Вона пропонує широкий вибір фільмів, серіалів, телешоу, анімаційних фільмів та іншого контенту, надає користувачам

можливість переглядати вміст на різних пристроях, включаючи смартфони, планшети, телевізори та інші.

У Megogo рекомендаційні системи використовуються для пропонування користувачам контенту, що відповідає їнім інтересам та попередньо переглянутому вмісту. Рекомендації базуються на різних факторах, включаючи явні та неявні дані, такі як історія перегляду, вподобання, рейтинги, відгуки інших користувачів. Також система враховує популярність та новизну тієї чи іншої одиниці контенту.

За допомогою цих систем, Megogo формує вибірку з даних, та формує різні категорії відео-контенту, які можуть зацікавити потенційного глядача, наприклад “Рекомендовані”, “Популярні”, “Новинки”, “Вам може сподобатись” та інші. Крім того, в системі враховуються такі критерії, як мова, жанр та тип вмісту. Це дозволяє знайти найбільш релевантний варіант користувачу [4].

Також, системи фільтрації інформації використовуються для розподілу контенту відповідно до регіону користувача. До прикладу, якщо користувач знаходиться в Англії, то йому будуть запропоновані фільми з англійською мовою озвучки та локалізації. До того ж, використовуєтсья персоналізоване прогнозування, що базується на даних про користувача – вікова категорія, стать та інтереси, що дає змогу більш точно рекомендувати відеовміст, що відповідатиме потребам кожного.

Слід врахувати, що рекомендаційна система, яку використовує Megogo, має складну і ємкісну архітектуру, в якій поєднуються багато типів фільтрації інформації. З вище написаного, можна зробити висновок, що Megogo має багато переваг, які сподобаються потенційному глядачу.

1.1.3 E-commerce Amazon.com

Amazon.com є одним з найбільших інтернет-магазинів у світі, який продає різноманітні товари, такі як книги, електроніка, побутова техніка, одяг, косметика та інші товари. Був заснований у 1994 році та швидко став одним з найбільших та найуспішніших онлайн-ретейлерів у світі. Одним з ключових елементів успіху є

використання рекомендаційних систем для підбору товарів для своїх користувачів. Amazon використовує безліч алгоритмів та методів машинного навчання для аналізу поведінки своїх користувачів, зокрема їхніх пошукових запитів, переглядів товарів та покупок.

На основі цієї інформації Amazon.com здійснює персоналізовані рекомендації користувачам. Наприклад, користувач може побачити блок з пропозиціями товарів, які можуть бути йому цікавими, на основі його попередніх переглядів та покупок. Крім того, Amazon також використовує рекомендації при підборі товарів для різних категорій, зокрема категорій бестселерів, новинок та топових продажів.

Крім рекомендацій товарів, також використовуються рекомендаційні системи для інших послуг, таких як пропозиції постачальників послуг, рекомендації відгуків користувачів та інші. В цілому, рекомендаційні системи є ключовим елементом успіху Amazon.com [5].

Вони допомагають збільшити продажі, залучити нових користувачів та збільшити лояльність існуючих. Однак, також важливо зберігати баланс між персоналізованими рекомендаціями та приватністю користувачів, щоб зберегти довіру своїх клієнтів.

Також слід зазначити, що саме Amazon Inc. винайшов рекомендаційну систему колаборативної фільтрації підходу “item-item” чи “item based” у 1998 році, яка зробила революцію у сфері електронної комерції, та підвищила продажі компанії, буде описана нижче в роботі і надалі використана як основа для побудови власного рішення.

1.1.4 Обґрунтування вибору сфери застосування

Оглянувши існуючі рішення, які використовують рекомендаційні системи, слід виділити, що у кожне з них впроваджено рекомендаційну систему на базі алгоритму колаборативної фільтрації задля вирішення схожих завдань, але є

глибоке переконання в тому, що сфера потокового відтворення відео виглядає як найбільш краща для використання у ній системи колаборативної фільтрації.

Вищевикладене твердження можна довести наступними чинниками:

– виразніше уподобання: у порівнянні з музикою та інтернет-товарами, відеоконтент часто вимагає більше часу та уваги користувачів. У зв'язку з цим, вони мають виразніші уподобання та перегляди, що дозволяє рекомендаційним системам прогнозувати вподобання точніше й ефективніше;

– спільноти: потокові відеосервіси часто сприяють створенню спільнот, які обговорюють інтереси, рейтинги та відгуки до різних видів контенту, що надає можливість використовувати ці дані для покращення якості рекомендованих вибірок;

– взаємодія з контеном: у сфері потокового відтворення відео відбувається більше видів взаємодії, таких як перегляд, відгуки, рейтинги, додавання до списку "дивитись пізніше" тощо. Ці дані також можуть бути використані для покращення алгоритмів колаборативної фільтрації;

– змістовні метадані: відеоплатформи часто мають багато метаданих, пов'язаних з відео, таких як жанри, режисери, актори, країни виробництва, рейтинги та анотації. Ці метадані використовують для поліпшення алгоритмів колаборативної фільтрації, що сприяє більш точним рекомендаціям для користувачів;

– ефект мережі: потокові відеоплатформи мають велику кількість активних користувачів. За рахунок цього, рекомендаційні системи збирають більше даних про уподобання та поведінку перегляду. Це в свою чергу може покращити якість рекоменлдацій, оскільки система може враховувати досвід широкого кола користувачів;

– довготривалість контенту: відеоконтент має більш тривалий час життя, ніж товари в інтернет-магазинах, що дозволяє рекомендаційним системам працювати з більш стабільним набором елементів. Це може покращити ефективність алгоритмів колаборативної фільтрації, особливо для нових користувачів.

Враховуючи ці аспекти, сфера потокового відтворення відео може надавати кращі умови для впровадження рекомендаційних систем на базі алгоритму колаборативної фільтрації, ніж галузь прослуховування музики чи продажу інтернет-товарів.

Отже, було обрано галузь потокового відтворення відео. А вузьковизначеним видом рекомендованих елементів обрано анімаційні фільми, оскільки за останні роки вони стали більш популярними серед української молоді, що підтверджують не тільки різноманітні фестивалі та івенти, але й збільшення кількості переглядів анімаційних фільмів на потокових відеоплатформах, вони охоплюють широкий спектр жанрів та тематик, від фентезі й наукової фантастики до драми та комедії. Це робить цю категорію відеоконтенту відмінною для впровадження рекомендаційної системи колаборативної фільтрації, тому що користувачам буде пропонуватись широкий вибір фільмів та серіалів. Також, до сьогодні, лише мала кількість наборів даних відгуків користувачів про такі фільми була використана для створення рекомендаційної системи суспільної фільтрації.

1.2 Наявні підходи створення систем колаборативної фільтрації

Рекомендаційні системи можна класифікувати за різними критеріями. Зазвичай цим критерієм є стратегія створення, яка показує, яким чином треба буде використувати алгоритми фільтрації інформації у системі. Стратегій на сьогодні існує доволі багато, але основними виділяють наступні [6]:

– фільтрація на основі змісту (Content-Based Filtering) – цей метод базується на аналізі вмісту елементів та профілю користувача. Рекомендації генеруються на основі схожості між елементами, які користувач уже оцінив позитивно, та елементами, які він ще не переглядав. Спрощений вигляд роботи цього підходу полягає у тому, що коли система отримує дані про раніше вподобані елементи, починає шукати схожі елементи за його критеріями, наприклад метаданими, і таким чином формує вибірку рекомендацій;

– колаборативна фільтрація (Collaborative filteriing) – цей метод базується на аналізі поведінки та взаємодії користувачів із продуктами або елементами. Він використовує відгуки, оцінки або історію перегляду користувачів для виявлення шаблонів та відповідно визначення схожості між користувачами або елементами. Колаборативна фільтрація ділиться на два підходи, що будуть розглянуті нижче;

– гібридні рекомендаційні системи – цей метод комбінує два або більше різних методів рекомендаційних систем, таких як колаборативна фільтрація та фільтрація на основі вмісту, для покращення рекомендацій та усунення обмежень окремих методів [7].

Також, для повноти викладення, слід згадати такі методи, як навчання з підкріпленням, стратегія базована на знаннях, демографічні та соціальні рекомендаційні системи, групові, та контекстно залежні. Вони не користуються такою популярністю, як ті, що наведені в переліку вище, але також використовуються.

Хоча колаборативна фільтрація має недоліки у вигляді проблеми “холодного старту” та рідкості даних, цей метод не є таким затратним як гібридний підхід і має набагато точніші та ефективніші рекомендації, ніж контентний підхід, а зазначені проблеми легко вирішуються пропозицією новим користувачам оцінити найбільш популярні елементи.

1.2.1 На основі користувачів (User-based Collaborative filtering)

Підхід на основі користувачів (User-Based Collaborative Filtering) є одним з ключових методів колаборативної фільтрації. Він зосереджений на аналізі схожості між користувачами, враховуючи їх оцінки та взаємодію з елементами [8]. Основна ідея полягає в тому, що користувачі зі схожими вподобаннями та інтересами мають схожі оцінки і вподобання щодо елементів.

У більшості алгоритм колаборативної фільтрації на основі користувачів виконує перелік наступних кроків:

– зберігає інформацію про оцінки, відгуки та взаємодію користувачів з елементами. Зазвичай це представлено у вигляді матриці користувачів та елементів, де рядки відповідають користувачам, а стовпці - елементам. Кожна комірка матриці містить оцінку користувача для відповідного елемента.

– використовує певну метрику схожості для обчислення ступеня подібності між користувачами на основі їх оцінок;

– для кожного користувача вибираються найбільш схожі користувачі (сусіди) на основі раніше обчисленої схожості. Вибір може бути зроблений за допомогою порогового значення схожості або фіксованої кількості сусідів;

– для кожного користувача й неоцінених елементів обчислюється прогнозована оцінка, використовуючи ваги схожості й відомі оцінки сусідів;

– на основі прогнозованих оцінок для неоцінених елементів створюється список рекомендацій для користувача. Елементи з найвищими прогнозованими оцінками рекомендуються користувачеві.

Розглянемо даний алгоритм у роботі на прикладі примітивного рисунку 1.1, що зображено нижче.

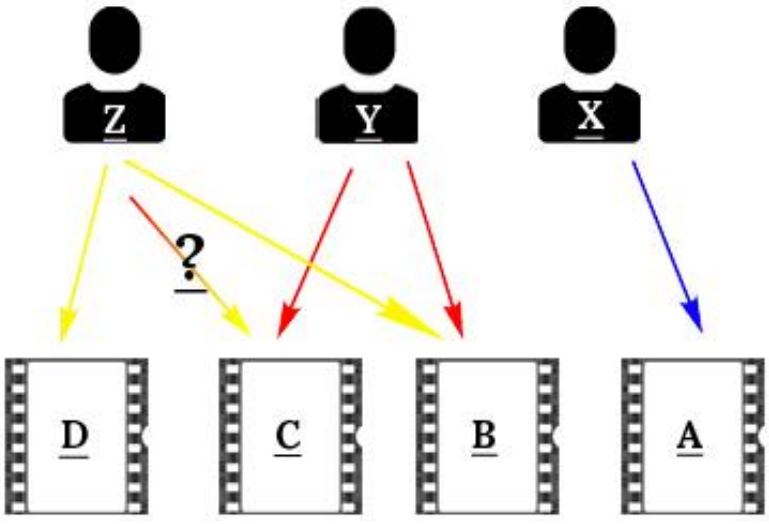


Рисунок 1.1 – Принцип роботи підходу на основі користувачів

Користувач *X* вподобав елемент *A*, користувач *Y* елементи *B* та *C*, а користувач *Z* елементи *B* та *D*. Оскільки користувачі *Y* та *Z* вподобали один і то й

же елемент *B*, вони (користувачі) є схожими. Отже, через те, що схожий користувач - *Y* вподобав елемент *C*, цільовий для нас користувач *Z* не надавав відгук щодо нього, елемент *C* буде йому рекомендований.

Такий підхід має як переваги так і недоліки, які будуть проаналізовані нижче в роботі.

1.2.2 На основі елементів (Item-Based Collaborative Filtering)

Підхід на основі елементів (Item-Based Collaborative Filtering) на відміну від попередньо розглянутого використовує схожість між елементами, щоб зробити рекомендації користувачам [9]. Це метод рекомендаційної системи, який базується на відносинах між елементами, замість схожості між користувачами. Цей підхід аналізує оцінки та поведінку користувачів щодо елементів (наприклад, товарів або фільмів) і використовує ці дані для визначення схожості між елементами. Рекомендації формуються шляхом знаходження найбільш схожих елементів на ті, які вже оцінив або споживав користувач, а потім прогнозуванням оцінки користувача для цих схожих елементів.

Принцип його роботи від user-based загалом відмінний тим, що обчислюється схожість між елементами, а не користувачами і саме на основі цієї метрики прогнозуються оцінки. У формулах, наведених у наступних розділах, наглядно продемонстровано, де саме використовується коефіцієнт подібності і те, на що він впливає при обрахунку оцінок.

Розглянемо даний алгоритм у роботі також на прикладі примітивного рисунку 1.2 нижче.

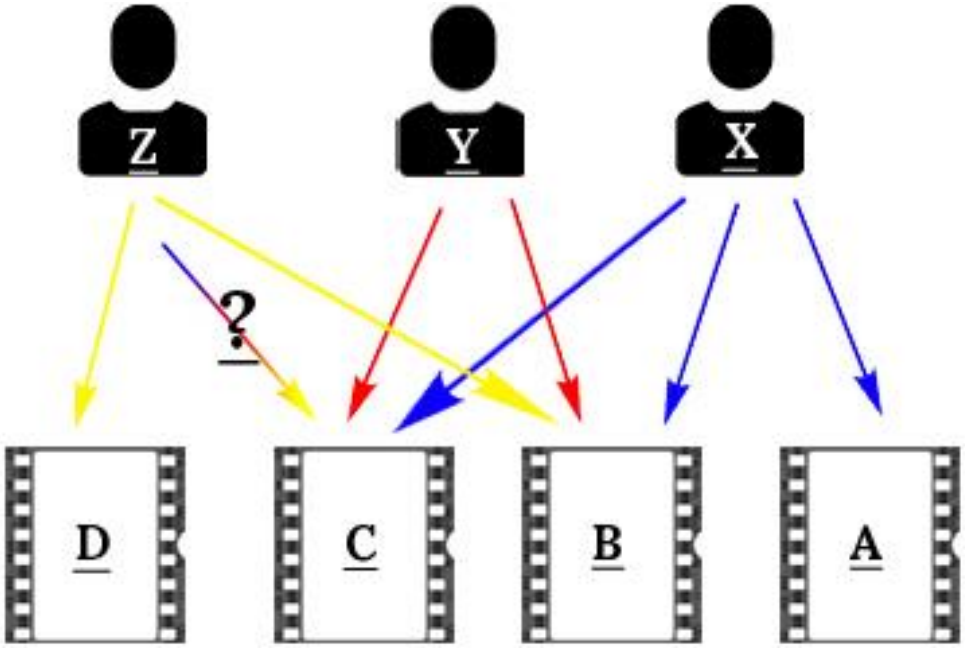


Рисунок 1.2 – Принцип роботи підходу на основі елементів

Користувач *X* вподобав елементи *A, B, C*. Користувач *Y* вподобав елементи *B* та *C*. Користувач *Z* вподобав елементи *B* та *D*. Оскільки елемент *B* та *C* були вподобані і користувачем *X*, і користувачем *Y*, вони рахуються схожими елементами. Так як користувач *Z* вподобав елемент *B*, який у свою чергу схожий з елементом *C*, який даний користувач ще не споживав – йому буде рекомендовано спробувати саме елемент *C*.

Такий підхід також має свої недоліки та переваги, які будуть проаналізовані нижче в роботі.

1.2.3 Порівняльний аналіз підходів надання рекомендацій

User-Based Collaborative Filtering та Item-Based Collaborative Filtering мають ряд переваг та недоліків.

До переваг підходу на основі користувачів відносять:

– простота реалізації: цей підхід досить легко реалізувати та зрозуміти;

– персоналізація: рекомендації засновані на схожості між користувачами надають можливість створювати більш індивідуальні рекомендації для кожного споживача;

До недоліків підходу на основі користувачів відносять:

– масштабованість: обчислення схожості між усіма парами користувачів може бути обчислювально важким для великих систем;

– проблема холодного старту для нових користувачів: важко робити точні рекомендації для користувачів без історії взаємодії з елементами.

До переваг підходу на основі елементів відносять:

– масштабованість: цей підхід має менші вимоги до обчислень, так як схожість між елементами змінюється рідше, ніж між користувачами;

– стабільність: рекомендації, засновані на схожості між елементами, як правило, менш схильні до змін порівняно з схожістю між користувачами.

До недоліків підходу на основі елементів відносять обмежену персоналізацію.

На основі вищезазначених тверджень виконано порівняльний аналіз, що наведено в таблиці 1.1 нижче.

Таблиця 1.1 – Порівняльний аналіз підходів надання рекомендацій

Критерій	User-based	Item-based
Масштабованість	Відносно низька	Висока
Персоналізація	Висока	Середня
Проблема холодного старту	Висока	Низька
Стабільність	Відносно низька	Висока
Простота реалізації	Висока	Висока

Підхід на основі елементів вважається кращим, оскільки даний підхід працює ефективніше з великими наборами даних, тому що обчислювальні вимоги менші, і

схожість між елементами змінюється рідше, ніж схожість між користувачами. Також рекомендації на основі схожості між елементами стабільніші в часі, що забезпечує більш консистенті рекомендації для користувачів. Підхід item-based краще впорається з проблемою холодного старту для нових користувачів, оскільки необхідно лише декілька оцінок від користувача, щоб алгоритм працював. У випадку ж user-based підходу, нові користувачі можуть не мати достатньої кількості оцінок для надійного порівняння з іншими користувачами.

Item-based підхід також може впоратися з новими елементами, які мають небагато оцінок, оскільки схожість між елементами буде визначатись за допомогою доступних оцінок. Без достатньої кількості оцінок від користувачів, у випадку user-based підходу, нові елементи можуть бути нерекомендованими.

На основі проаналізованих чинників, безперечно, був обраний item-based підхід, чи підхід на основі схожості елементів для подальшої реалізації у власному рішенні.

1.3 Специфіка предметної області

Звернемо увагу на те, що системи колаборативної фільтрації, які надають рекомендації вимагають аналізу та обробки великої кількості даних про користувачів та елементи, зокрема їх взаємодії, що допомагає виявити закономірності та відтворити інтереси користувачів для надання персоналізованих рекомендацій. Необхідно враховувати різноманітності смаків та інтересів користувачів, а також характеристик елементів та важливість застосовуваних методів вимірювання схожості для визначення подібності між користувачами або елементами. Слід зазначити також, що необхідна стратегія для вирішення проблеми холодного старту, пов’язаної з новими користувачами або елементами.

Не менш важливим є забезпечення прозорості, яким чином надаються рекомендації. Користувачі повинні бути повідомлені, які неявні дані (наприклад поведінка при перегляді фільму, чи історія переглядів) збираються системою без їх відома задля отримання довіри до сервісу. Системи збору інформації повинні

відповідати вимогам щодо захисту даних та забезпечувати безпеку інформації користувачів. Також системи повинні мати змогу масштабуватись. Це означає, що системи повинні вміти оброблювати велику кількість користувацьких оцінок та елементів швидко, задля оптимізованої роботи сервісу.

Висновок до розділу 1

У даному розділі було проаналізовано актуальність рекомендаційних систем у сучасному світі, зазначено яким чином вони полегшують пошук бажаного контенту користувачам. Оглянуто існуючі рішення, показано де саме і як у них використовуються системи фільтрації інформації. Були зазначені переваги використання таких систем для підвищення ефективності сервісів, на прикладі Megogo, Spotify та Amazon.

Вони є платними, з закритим кодом, що заваджує подальшому їх розширенню спільнотою, мають складну архітектуру та багаторівневість. Через це потребують величезних обчислювальних спроможностей. Також вузьконаправлені на визначені елементи.

Також було обгрунтовано вибір сфери – потокове відтворення відео, та тип елементів для рекомендацій – анімаційні фільми, задля впровадження рекомендаційної системи. Оглянуто підходи надання рекомендацій з використанням колаборативної фільтрації, а саме:

- підхід на основі користувачів;
- підхід на основі елементів.

Виконано їх порівняльний аналіз, на основі чого обрано стратегію для створення свого рішення.

На останок, було виділено особливості рекомендаційних систем колаборативної фільтрації, перелік яких складає: велика кількість даних; різноманітність користувачів та елементів; способи вимірювання схожості; проблема холодного старту; забезпечення прозорості та приватності; масштабування.

2 АНАЛІЗ МЕТОДІВ НАДАННЯ РЕКОМЕНДАЦІЙ

Оскільки обраним підходом до надання рекомендацій є колаборативна фільтрація на основі елементів, слід проаналізувати математичні засади для його подальшої реалізації. Розглянемо наявні способи знаходження коефіцієнту схожості та як він використовується у розрахунку прогнозованої оцінки.

1.2 Змістовна постановка задачі

Користувачам сервісів потокового відтворення відео зазвичай важко швидко знайти контент, який буде максимально відповідний їх вподобанням.

Для полегшення пошуку та покращення користувацького досвіду, слід розробити рекомендаційну систему на базі алгоритму колаборативної фільтрації з підходом на основі елементів.

1.3 Математична постановка задачі

Дано датасет у вигляді матриць, що містить дані про елементи та відгуки про них користувачами. Після попередньої обробки даних (буде розглянуто нижче в роботі) необхідно розрахувати схожість між елементами, можливу оцінку користувача для раніше неоціненого елементу, та на основі цього надати вибірку рекомендацій для обраного користувача.

2.3 Способи розрахунку схожості між елементами

Існує велика кількість способів та технік, які можуть бути використані для оцінки схожості між елементами. Такі методи можуть включати статистичні показники, геометричні міри, інформаційні метрики чи, навіть, машинне навчання. Аналіз цих методів надасть можливість обрати оптимальний підхід для нашої

конкретної задачі та набору даних, що допоможе покращити якість рекомендацій системи [10].

2.3.1 Косинусна подібність

Косинусна міра подібності є одним з поширених способів розрахунку схожості між двома елементами у векторному просторі, зокрема в контексті колаборативної фільтрації на основі елементів. Даний метод базується на вимірюванні кута між двома векторами, які представляють елементами. І чим менший кут між цими векторами, тим більш схожими вважаються елементи.

Для розрахунку косинусної подібності між двома елементами і та j спочатку створюються вектори оцінок користувачів для цих елементів. Нехай r_{ui} та r_{uj} – оцінки користувача u для елементів i та j відповідно. $\sum u$ – сума за всіма користувачами u, які оцінили обидва елементи i та j. Важливо враховувати лише спільних користувачів, які оцінили обидва елементи, оскільки метрика базується на порівнянні оцінок цих користувачів для кожного елемента.

Тоді формула для розрахунку косинусної міри подібності матиме наступний вигляд:

$$\cos(i,j) = \frac{\sum_u (r_{ui} * r_{uj})}{\sqrt{\sum_u {r_{ui}}^2} * \sqrt{\sum_u {r_{uj}}^2}}, \quad (2.1)$$

де результат розрахунку буде знаходитись у діапазоні від -1 до 1, де -1 відповідає повній негативній схожості (косинус 180 градусів), 0 – відсутності схожості (косинус 90 градусів), 1 – повній позитивній схожості (косинус 0 градусів).

Цей метод враховує відсутність оцінок користувачів, тому що його значення не залежить від довжини векторів, а лише від кута між ними. Це означає, що він може коректно працювати навіть у випадках, коли користувачі оцінили лише частину елементів. Косинусна подібність відносно добре масштабується для

великих наборів даних, оскільки вона вимагає розрахунку лише тих оцінок, які перетинаються між двома елементами.

Різні користувачі можуть мати різні шкали оцінювання. Деякі користувачі можуть бути схильними до вищих оцінок, тоді як інші – до нижчих. Косинусна міра подібності не залежить від абсолютних значень оцінок, а лише від кута між векторами оцінок, тому вона може забезпечити кращі результати, ніж інші метрики схожості, які залежать від абсолютних значень оцінок.

Також, необхідно зазначити відмінність від коефіцієнта кореляції Пірсона (буде описаний в наступному підрозділі). Косинусна подібність не вимагає додаткової нормалізації даних перед розрахунком, що робить її більш швидкою для обчислень.

Для прикладу на рисунку 2.1 зображено випадок, коли на основі оцінок для двох елементів(векторів) обраховано їх подібність.

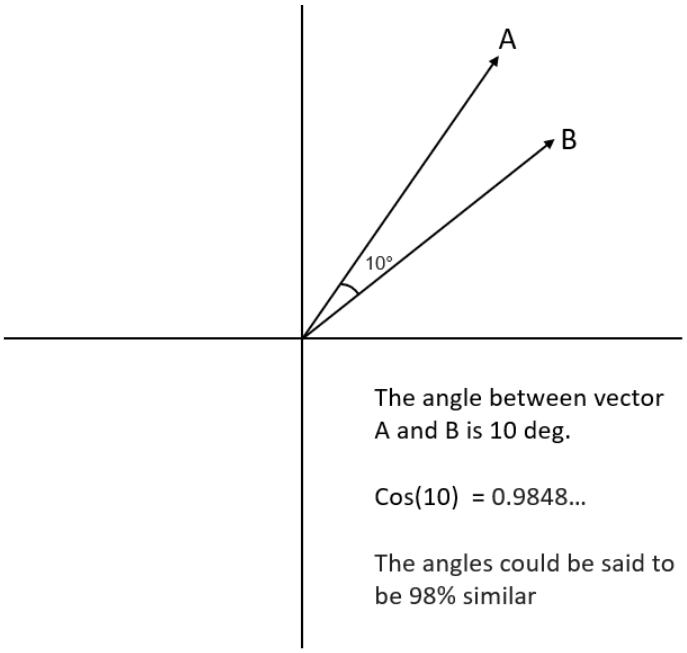


Рисунок 2.1 – Подібність двох векторів

Кут між векторами A та B 10 градусів. За формулою отримуємо результат, що приблизно рівний значенню 0.9848. Отже, можна стверджувати, що дані вектори є подібними.

2.3.2 Лінійний коефіцієнт кореляції Пірсона

Наступним розглянемо спосіб, що базується на кореляції Пірсона. Це статистична міра між двома змінними, яка відображається у діапазоні від -1 до 1, де -1 вказує на повну негативну лінійну залежність, значення 1 – на повну позитивну лінійну залежність, тоді як 0 свідчить про відсутність лінійної залежності.

У контексті ж підходу колаборативної фільтрації на основі елементів, коефіцієнт кореляції Пірсона може бути використаний для вимірювання схожості між елементами на основі оцінок користувачів та може допомогти виявити скриті закономірності. Формула для розрахунку коефіцієнта кореляції між двома елементами і та j матиме наступний вигляд:

$$p_{i,j} = \frac{\sum_u (r_{ui} - r'_i)(r_{uj} - r'_j)}{\sqrt{\sum_u (r_{ui} - r'_i)^2 \sum_u (r_{uj} - r'_j)^2}}, \quad (2.2)$$

де $p_{i,j}$ – коефіцієнт кореляції Пірсона між елементами *i* та *j*; r_{ui} та r_{uj} – оцінки користувачів u для елементів і та j відповідно; r'_i та r'_j – середні оцінки для елементів *i* та *j* відповідно; розраховані за всіма користувачами, які оцінили відповідні елементи; $\sum u$ сума за всіма користувачами u, які оцінили обидва елементи *i* та *j*.

На рисунку 2.2 нижче наглядно продемонстровані можливі результати кореляції між змінними.

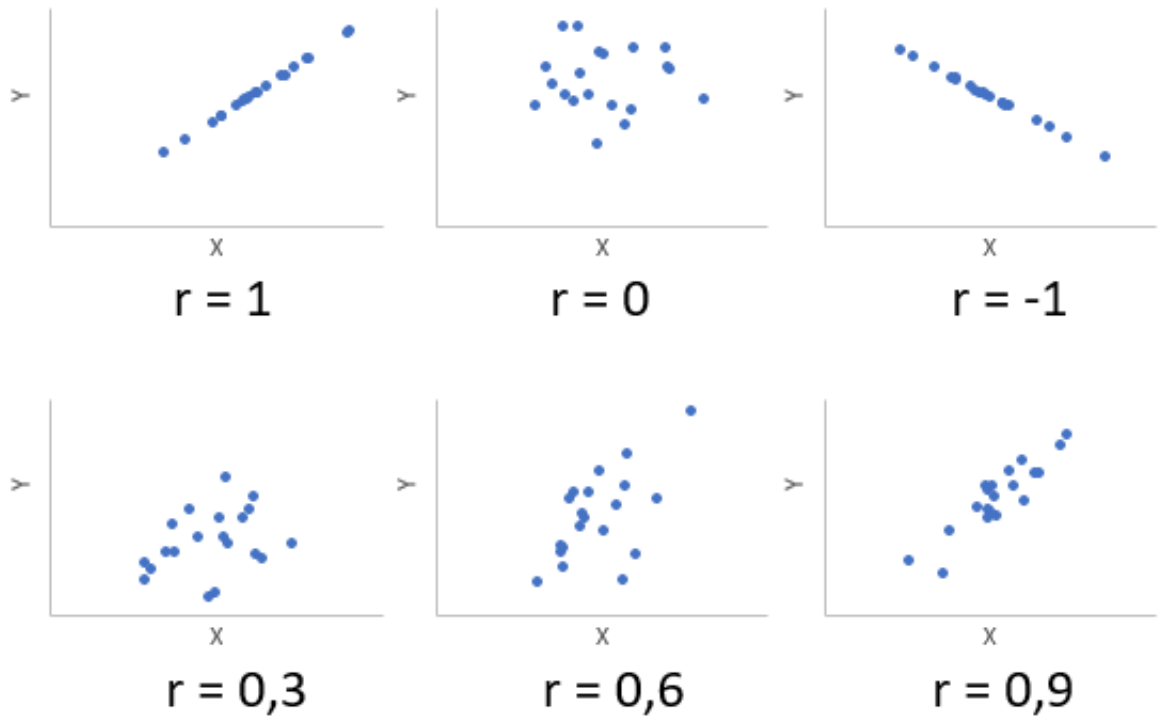


Рисунок 2.2 – Можливі результати кореляції

Тобто, для того, щоб два елементи були схожими, повинна виконуватись пряма лінійна залежність, або значення мають бути близькими до цієї прямої. Іншими словами, точки на діаграмах мають бути якнайближче до прямої залежності, інакше даний метод не зможе правильно ідентифікувати подібність. Це означає, що на реальному прикладі, якщо залежність матиме нелінійний характер, алгоритм буде працювати неправильно. А також відсутність властивості розпізнавати пусті дані та нейтральні може призвезти до некоректних результатів.

2.3.3 Індекс Жаккара (Jaccard Index)

Ще один популярний спосіб – знаходження індексу Жаккара. Ця міра подібності враховує тільки спільні та унікальні характеристики двох об’єктів. В контексті рекомендаційних систем колаборативної фільтрації на основі елементів, індекс Жаккара може бути використаний для вимірювання схожості між елементами на основі спільних користувачів, які взаємодіяли з ними.

Формула, для розрахунку індексу Жаккара між двома елементами *i* та *j* має наступний вигляд:

$$J(i,j) = \frac{|U_i \cap U_j|}{|U_i \cup U_j|}, \quad (2.3)$$

де *J(i, j)* - індекс Жаккара між елементами *i* та *j*; *U(i)* та *U(j)* - множини користувачів, які взаємодіяли з елементами *i* та *j* відповідно; $|U(i) \cap U(j)|$ - кількість спільних користувачів, які взаємодіяли з обома елементами *i* та *j*; $|U(i) \cup U(j)|$ - кількість унікальних користувачів, які взаємодіяли з будь-яким з елементів *i* або *j*.

Індекс, або коефіцієнт Жаккара приймає значення від 0 до 1. Значення, близькі до 1, свідчать про високу подібність між елементами (більше спільних користувачів), а значення, близькі до 0, вказують на низьку подібність (менше спільних користувачів).

Однак, варто зазначити, що індекс Жаккара не враховує самі оцінки, які користувачі виставляють елементам, а лише фокусується на спільних користувачах. Тому, у випадку, коли необхідно врахувати оцінки, краще використовувати описані раніше методи.

2.3.4 Евклідова відстань

Даний спосіб, Евклідова відстань, є класичною мірою відстані між двома точками у просторі і може бути використана для вимірювання схожості між елементами в рекомендаційних системах спільної фільтрації. У контексті таких систем, Евклідова відстань враховує оцінки, які користувачі виставляють елементам, та вимірює “відстань” між двома елементами на основі різниці в оцінках.

Формула розрахунку Евклідової відстані між двома елементами і та j має наступний вигляд:

$$d(i,j) = \sqrt{\sum_u (r_{ui} - r_{uj})^2}\,, \tag{2.4}$$

де *d(i, ,j)* Евклідова відстань між елементами *i* та *j*; r_{ui} та r_{uj} оцінки, які користувач u виставив елементам *i* та *j* відповідно; $\sum u$ сума по всім користувачам u, які оцінили обидва елементи *i* та *j*.

Однак, цей спосіб вимірює відстань, а не подібність. Тому, для перетворення відстані на міру подібності, можна використати наступну формулу:

$$s(i,j) = \frac{1}{1+d(i,j)}\,, \tag{2.5}$$

де *s(i,j)* – міра подібності між елементами *i* та *j*, яка змінюється від 0 (найменш схожі) до 1 (найбільш схожі).

Евклідова відстань працює добре у випадку, коли оцінки мають однаковий масштаб, але може бути чутливою до різниці в масштабах оцінок між користувачами. Тому, у системах колаборативної фільтрації доцільніше використовувати методи, які враховують ці різниці (наприклад, косинусну подібність).

2.3.5 Манхеттенська відстань

Ще одним методом вимірювання відстані між двома точками у просторі є міська блокова відстань, який сумує абсолютні значення різниць між координатами. Цей алгоритм також відомий як L1-норма чи таксометрова відстань. В контексті систем колаборативної фільтрації може бути використаним для вимірювання відстані між двома елементами на основі різниці в оцінках, виставлених користувачами.

Формула для розрахунку такої відстані між двома елементами і та j має наступний вигляд:

$$d(i,j) = \sum_u |r_{ui} - r_{uj}|, \quad (2.6)$$

де *d(i,j)* манхетенська відстань між елементами *i* та *j*; r_{ui} та r_{uj} оцінки, які користувач u виставив елементам *i* та *j* відповідно; $\sum u$ сума по всім користувачам *u*, які оцінили обидва елементи *i* та *j*.

Подібно до евклідової відстані, манхеттенську відстань можна перетворити на міру подібності, що приймає значення від 0 до 1, за допомогою аналогічної формули 2.5.

Така метрика працює добре, у випадку коли оцінки мають однаковий масштаб, але, подібно до Евклідової відстані, може бути чутливою до різниці в масштабах оцінок між користувачами.

2.4 Порівняльний аналіз способів знаходження подібності елементів

Для вибору доцільного способу, був проведений порівняльний аналіз способів обчислення схожості, результати якого занесено до таблиці 2.1 та наведено нижче.

Таблиця 2.1 – Порівняльний аналіз метрик для обчислення подібності

Метод	Переваги	Недоліки
Косинусна подібність	Нормалізує оцінки користувачів, незалежний від масштабу, ефективний при розрідженості даних	Не враховує глобальний рейтинг або популярність елементу

Продовження до таблиці 2.1

Метод	Переваги	Недоліки
Коефіцієнт кореляції Пірсона	Враховує глобальний рейтинг елементу, незалежний від масштабу	Неточний при розрідженост даних, чултивий до викидів
Індекс Жаккара	Простий для обчислень	Ігнорує оцінку елемента, неточний для розріджених даних
Евклідова відстань	Простий для обчислень	Неточний для розріджених даних, не нормалізується, залежний від масштабу
Манхеттенська відстань	Чутливий до нюансів в рейтингах, простий для обчислень	Неточний для розріджених даних, не нормалізується, залежний від масштабу

На основі аналізу способів ідентифікації подібностей між елементами, безперечно, був обраний спосіб косунусної подібності, оскільки він нівелює недоліки інших методів та має переваги, такі як відсутність необхідності попередньої нормалізації даних, незалежність від абсолютних значень, здатність масштабуватись та враховувати відсутність оцінок.

2.5 Метод прогнозування оцінки елемента

Для прогнозування оцінки користувача, на основі попереднього аналізу, була обрана метрика – косинусна подібність. З підходом колаборативної фільтрації на основі елементів, ми можемо скористатись наступною математичною формулою:

$$p(u,i) = \frac{\sum_j (sim(i,j) * r_{uj})}{\sum_j |sim(i,j)|}, \quad (2.6)$$

де $p(u,i)$ прогнозована оцінка користувача *u* для елемента *i*; *sim(i, j)* косинусна міра подібності мжі елементами *i* та *j*, r_{uj} оцінка, яку користувач u виставив елементу *j*; $\sum j$ сума по всім елементам j, які належать множині елементів, які користувач u оцінив; |*sim(i, j)*| абсолютне значення косинусної міри подібності між елементами *i* та *j*.

За формулою 2.6 ми знаходимо суму добутків косинусної подібності між елементом *i*(той, для якого ми хочемо зробити прогноз) та всіма іншими елементами *j*, які користувач u оцінив, помножених на оцінки користувача u для цих елементів. В результаті ми отримуємо вагову суму оцінок користувача, де ваги відображають подібність між елементами. Сума ділиться на суму абсолютних значень косинусної подібності для нормалізації значень.

Тобто, ця формула дозволяє розраховувати прогнозовані оцінки, враховуючи схожість між елементами та оцінки, які користувач виставив іншим елементам.

2.6 Обгрунтування методу вирішення

Для виконання поставленої задачі, при реалізації рекомендаційної системи буде розроблено наступний функціонал:

– обчислення схожості між елементами на основі формули 2.1 (косинусна подібність);

– прогнозування оцінки елемента користувачем на основі формули 2.6. Це покаже міру зацікавленості користувача;

– формування підбірки рекомендацій з 10 елементів, вибір найкращих варіантів з-поміж схожих елементів, для яких прогнозуватиметься оцінка.

Реалізація даного функціоналу буде продемонстрована та описана нижче у дипломній роботі.

Висновок до розділу 2

У даному розділі був проведений аналіз наступних способів обрахунку коефіцієнту подібності між елементами для підходу колаборативної фільтрації на основі елементів:

- косинусна міра подібності;
- лінійний коефіцієнт кореляції Пірсона;
- індекс Жаккара;
- Евклідова відстань;
- манхеттенська відстань.

На основі аналізу переваг та недоліків був обраний за міру схожості елементів – спосіб косинусної подібності. Також слід зауважити, що існує велика кількість інших методів, таких як PMI, Баєсівський класифікатор, моделі багатовимірного шкалювання, техніка локально-чутливого хешування, автоенкодери тощо. Але, оскільки вони є досить дорогими та складними в реалізації, мають свою специфіку використання, тому в рамках даного дипломного проєкту не розглядались як варіанти для реалізації.

Також було проаналізовано метод прогнозування оцінки користувачем з математичної точки зору. Даний метод стане основою для реалізації функціоналу програмного забезпечення.

Виконана змістовна та математична постановка задачі, обгрунтовано метод її вирішення.

3 ПРОЕКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Оскільки метою дипломної роботи є полегшення пошуку контенту користувачам зі схожими вподобаннями, для її досягнення необхідно виконати завдання з розробки рекомендаційної системи анімаційних фільмів колаборативної фільтрації. Для проектування системи необхідно обрати набір даних, середовище розробки, мову програмування, набір біліотек та представити роботу такої системи у схемах для подальшої її розробки.

Слід враховувати, що обраним раніше підходом з надання рекомендацій є суспільна фільтрація на основі елементів. Також, для реалізації такої системи необхідна величезна вибірка даних, яка міститиме інформацію про елементи та відгуки користувачів про них релевантні обраному напрямку – анімаційні фільми. Вирішено не створювати власну базу даних для системи, тому що існує велика кількість анонімізованих датасетів зі вже існуючих сервісів, що дозволить продемонструвати роботу системи на реальних даних.

Розроблене рішення буде вирішувати такі задачі, як:

– надання списку з 10 рекомендованих елементів (фільмів) для визначеного користувача;

– прогнозування оцінки елементу для визначеного користувача;

– знаходження схожих елементів на попередньо переглянутий.

3.1 Інформаційне забезпечення

3.1.1 Сервіс для пошуку набору даних Kaggle

Для пошуку набору даних був обраний сервіс Kaggle, оскільки він має ряд переваг, таких як велика кількість даних, якість даних, професійна спільнота та зручність. Kaggle має величезну кількість датасетів з різних доменів, що включає анімаційні фільми, фільми, товари, серіали, музику та інші. Це дозволяє нам швидко знайти відповідний набір даних для нашої рекомендаційної системи.

Слід зазначити, що датасети на Kaggle мають високу якість, оскільки вони часто використовуються для проведення конкурсів і наукових досліджень. Це означає, що дані зазвичай чисті, структуровані та корисні для нашого завдання.

Також Kaggle має велику спільноту дослідників та розробників, які обговорюють різні аспекти даних, а також рішення на основі даних. Це може допомогти знайти корисні поради та техніки, які стосуються нашої задачі.

З практичної точки зору, даний сервіс пропонує зручний інтерфейс для завантаження та візуалізації даних та надає вбудовану середу розробки Jupyter Notebook для аналізу даних та розробки моделей прямо на платформі.

З урахуванням вищенаведеного, сервіс Kaggle є відмінним вибором для пошуку наборів даних для розробки рекомендаціїної системи анімаційних фільмів.

3.1.2 Набір даних Anime Recommendations Database

Обраним датасетом для використання у системі є Anime Recommendations Database, який містить інформацію про користувацькі вподобання 73516 користувачів щодо 12294 анімаційних фільмів з сервісу потокового перегляду відео myanimelist.net. Кожен користувач був у змозі додати фільм до свого списку і поставити йому оцінку, і цей набір даних є компіляцією цих оцінок [11].

Набори даних знаходяться у двох файлах Anime та Rating з розширенням .csv. CSV, або Comma Separeted Values є розширенням файлів, що містять дані, які розділяються комами або іншими розділювачами (наприклад, табуляцією). Вони часто використовуються для зберігання табличних даних, наприклад баз даних або електронних таблиць. Оскільки файли з таким розширенням мають табличну структуру, яка зручно представляє рейтинги користувачів, описи елементів та інші атрибути, вони широко використовуються при створенні рекомендаційних систем. Слід врахувати, що csv файли сумісні з більшістю мов програмування та бібліотеками обробки даних, такими як pandas в Python, readr в R або CSV у JavaScript. Це значно полегшує імпорт, експорт та обробку даних. Файли CSV мають відносно малі розміри порівняно з іншими форматами даних, такими як

XML чи JSON, що забезпечує швидше завантаження даних та менші вимоги до сховища.

Вищезгадані файли містять інформацію про ідентифікатор анімаційного фільму – anime_id, ім’я анімаційного фільму - name, жанр – genre, тип – type, кількість епізодів – eposides, середній рейтинг елементу – rating, кількість користувачів, які переглянули фільм – members, ідентифікатор користувача – user_id та оцінка, виставлена конкретним користувачем визначеному елементу, яка варіюється від 0 до 10, та набуває значення -1, якщо воно було ним переглянуте, але не оцінене.

У таблицях 3.1 та 3.2 нижче наведено фрагменти з перших десяти рядків даних, які знаходяться у файлах Anime.csv та Rating.csv відповідно.

Таблиця 3.1 – Фрагмент даних Anime.csv файлу

anime_id	name	genre	type	episodes	rating	members
32281	Kimi no Na wa.	Drama, Romance, School, Supernatural	Movie	1	9.37	200630
5114	Fullmetal Alchemist: Brotherhood	Action, Adventure, Drama, Historical	TV	64	9.26	793665
28977	Gintama°	Action, Comedy, Historical,	TV	51	9.25	114262
9253	Steins;Gate	Sci-Fi, Thriller Military, Sci-Fi, Space	TV	24	9.17	673572

Продовження таблиці 3.1

anime_id	name	genre	type	episodes	rating	members
9969	Gintama'	Action, Comedy, Historical	TV	51	9.16	151266
32935	Haikyuu!!: Karasuno	Comedy, Drama	TV	10	9.15	93351
11061	Hunter x Hunter (2011)	Action, Adventure	TV	148	9.13	425855
820	Ginga Eiyuu Densetsu	Drama, Military, Sci-Fi, Space	OVA	110	9.11	80679
15335	Gintama Movie: Kanketsu	Action, Comedy	Movie	1	9.10	72534
15417	Gintama': Enchousen	Action, Comedy, Historical	TV	13	9.11	81109

Таблиця 3.2 – Фрагмент даних Rating.csv файлу

user_id	anime_id	rating
1	2787	-1
1	2001	-1
1	8074	10
1	15451	8
2	11771	10
2	12189	-1
2	16417	7
3	341	6
3	430	7
3	527	7

Даний набір даних містить відгуки для кожного елементу, що забезпечує вирішення проблеми “холодного старту”, та низький рівень рідкості даних. Також, слід враховувати, що він має відкриту ліцензцію CC0: Public Domain, яка надає можливість його вільного використання без порушення політики конфеденційності або прав на інтелектуальну власність. Ще одним критерієм, який повпливав на вибір цього набору даних є шкала оцінки від користувачів, яка знаходиться в діпазазоні від 0 до 10. Це означає, що він чудово підійде для знаходження схожих елементів при роботі алгоритму колаборативної фільтрації.

При проектуванні системи слід враховувати, що необхідний модуль попередньої обробки даних, для того, щоб привести їх до вигляду, який буде використувавитсь для створення таблиці типу елемент-елемент алгоритмом.

На вхід до розроблюваної системи будуть подаватись ідентифікатор користувача у цілочисельному форматі для надання йому списку рекомендацій, ідентифікатор користувача та ідентифікатор елементу для прогнозування оцінки та назву елементу для знаходження схожих йому. На вихід розроблюваної системи будуть надходити наступні масиви інформації:

– масив рекомендованих системою анімаційних фільмів визначеному користувачу;

– масив схожих на цільовий елементів;

– прогнозована оцінка для визначеного елементу.

3.2 Засоби розробки

Правильний вибір засобів для розробки системи є одним з ключових аспектів при проектуванні. Вони можуть полегшити процес розробки та підвищити ефективність роботи. Окрім того, від них залежить сумісність, стабільність та надійність системи. Ці фактори впливають на безпроблемне функціонування та можливість легкої інтеграції системи з іншими компонентами.

3.2.1 Середовище розробки Visual Studio Code

Visual Studio Code (VSCode) є вільним, кросплатформенним текстовим редактором з відкритим кодом від Microsoft. Він підтримує різні мови програмування та фреймворки, пропонує розширення, налаштування, інтеграцію з системами контролю версій (наприклад, GIT), а також зручний інтерфейс для написання, тестування та відлагодження коду.

Дане середовище для розробки є повністю безкоштовним як студентів, так і для розробників. До того ж, кожен бажаючий має змогу написати своє розширення і додати його до бібліотеки застосунку, що робить його лідером на ринку IDE (Integrated Development Environment) [12].

3.2.2 Мова програмування Python

Python є високорівневою, інтерпретованою мовою програмування з сильною підтримкою об’єктно-орієнтованого програмування. Він відрізняється своєю простотою, читабельністю коду та широким спектром застосувань, від веб-розробки до наукових та аналітичних обчислень.

Вибір даної мови програмування для написання рекомендаційної системи колаборативної фільтрації на основі елементів визначався наступним переліком факторів:

– велика кількість бібліотек: Python має велику кількість бібліотек, які спеціалізуються на обробці даних, машинному навчанні та рекомендаційних системах, таких як pandas, NumPy, scikit-learn та інші. Ці бібліотеки спрощують розробку, аналіз даних та реалізацію алгоритмів;

– спільнота та підтримка: Python має велику та активну спільноту розробників, що забезпечує доступ до багатої документації, прикладів коду та порад. Це допомагає при розробці рекомендаційних систем;

– простота та читабельність: Python відомий своєю простотою синтаксису та читабельністю коду, що сприяє швидкому розробленню та легкій підтримці коду.

Це дозволяє командам розробників працювати продуктивніше та забезпечує стабільність системи;

– гнучкість: Python дозволяє легко інтегрувати код з іншими мовами програмування або системами, що робить його гнучким для розробки рекомендаційних систем, які можуть потребувати інтеграції з існуючими або зовнішніми сервісами. Ця гнучкість полегшує розробку складних рекомендаційних систем, які можуть включати різні компоненти та залежності;

– масштабованість: Python підтримує різні підходи до масштабування, такі як багатопоточність, асинхронність та розподілені обчислення, що дозволяє вам побудувати рекомендаційні системи, які можуть легко масштабуватися відповідно до потреб і навантаження.

Враховуючи ці переваги, Python є ідеальним вибором для розробки власного рішення.

3.2.3 Використані бібліотеки

NumPy, що розшифровується як Numerical Python, це фундаментальна бібліотека для наукових обчислень у Python. Вона надає підтримку високопродуктивних багатовимірних масивів та матриць, а також містить математичні функції для роботи з цими структурами даних. Дана бібліотека пропонує ефективні операції з масивами завдяки внутрішній реалізації на C. Це робить обчислення набагато швидше, ніж при використанні вбудованих списків Python. NumPy містить ряд функцій для виконання операцій з лінійної алгебри, таких як множення матриць, обернення, знаходження рангу, вирішення лінійних рівнянь тощо, а також надає базові статистичні функції, такі як стандартне відхилення, кореляція, медіана та інші. Слід враховувати також елементні операції без використання циклів.

Це програмне розширення має властивість інтероперабельності, тобто воно є сумісним з різними джерелами даних та середовищами розробки, та має широку підтримку, оскільки є основою для багатьох інших наукових та аналітичних

біліотек, таких як pandas, SciPy, scikit-learn, TensorFlow та PyTorch. Таким чином, NumPy стає необхідним інструментом для розробки та реалізації рекомендаційної системи.

Наступною використаною бібліотекою стала Pandas, яка є відкритою бібліотекою для Python і забезпечує високопродуктивні структури даних та інструменти для аналізу даних. Pandas побудовано на основі NumPy та спроектовано, щоб зробити обробку даних швидкою та простою. Основною структурою даних є DataFrame, яка є двовимірною таблицею з мітками рядків та стовпців і дозволяє зберігати та обробляти різні типи даних у структурованому форматі. Має велику кількість переваг, такі як легке зчитування та запис даних, широкий спектр операцій обробки даних, можливість швидкої обробки пропущених даних, наявність спеціальних функцій та класів для аналізу часових рядів, можливість візуалізації даних і високу продуктивність.

Також, для знаходження схожості між елементами на основі користувацьких оцінок була задіяна бібліотека Scikit-learn(скорочено sklearn), яка є популярною та потужною бібліотекою машинного навчання для Python і надає широкий спектр алгоритмів, інструментів та модулів для роботи з даними. В контексті власної рекомендаційної системи буде використана вбудована у цю бібліотеку метрика cosine_similarity, що є вже написаною функцією для розрахунку косинусної подібності між елементами.

3.2.4 Фреймворк Pyforms-GUI

Для розробки користувацього інтерфейсу був обраний фреймворк Pyforms GUI. Це легкий інструмент, для створення графічних користувацьких інтерфейсів(GUI) у Python, заснований на бібліотеках PyQt та Qt. Він надає високорівневий API для створення вікон, вкладок, кнопок, текстових полів, списків та інших елементів інтерфейсу. Фреймворк розроблений з метою зробити процес створення GUI простим, швидким та ефективним, зосереджуючись на кінцевому користувачеві, забезпечуючи гнучкість та інтуїтивність роботи [13].

Слід зазначити перелік переваг, на основі яких він був обраний:

– модульність: Фреймворк підтримує модульний підхід до створення інтерфейсів, що дозволяє легко додавати, змінювати та видаляти елементи інтерфейсу;

– кросплатформеність: Завдяки використанню бібліотек PyQt та Qt, PyForms GUI може працювати на різних операційних системах, таких як Windows, macOS та Linux;

– інтеграція з Python: PyForms GUI легко інтегрується з існуючим кодом на Python, що дозволяє використовувати ваші алгоритми рекомендаційної системи без необхідності розробки додаткових адаптерів чи обгорток;

– підтримка подій: PyForms GUI дозволяє обробляти події, такі як натискання кнопок, зміна значень полів введення або вибір елементів зі списку. Це дозволяє створити інтерактивний інтерфейс, який реагує на дії користувача та надає рекомендації в реальному часі;

– налаштування вигляду: Засоби стилізації PyForms GUI дозволяють налаштовувати вигляд елементів інтерфейсу, щоб він відповідав вашим потребам та стилі. Ви можете змінювати колір, шрифт, розмір, відступи та інші візуальні параметри;

– документація: PyForms GUI має гарну документацію, яка допомагає швидко розібратися в особливостях фреймворка. Також, завдяки активній спільноті розробників, ви можете знайти відповіді на питання та отримати допомогу при вирішенні проблем.

Для наочності представлено приклад користувацього інтерфейсу, створеного за допомогою Python GUI, на рисунку 3.1 нижче.

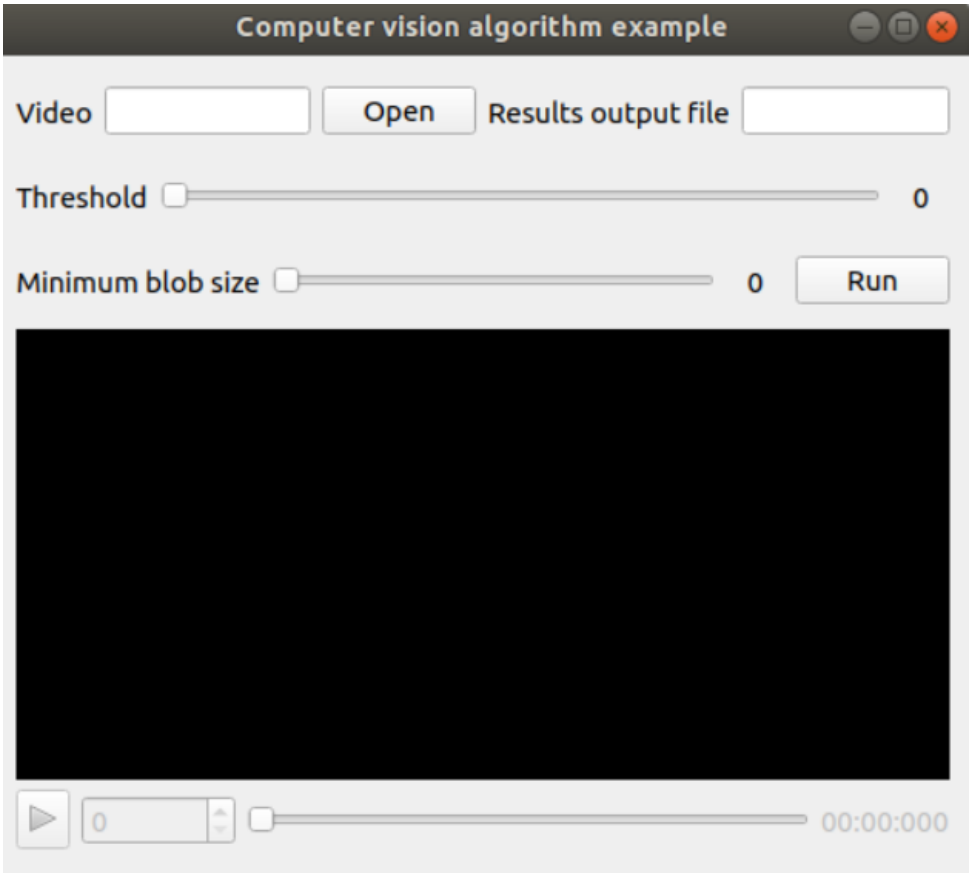


Рисунок 3.1 – Приклад користувацього інтерфейсу Pyforms GUI

Отже, використовуючи PyForms GUI для реалізації користувацького інтерфейсу рекомендаційної системи, буде отримано гнучкий, потужний та легкий у використанні інструмент, що дозволить швидко реалізувати проект та збезпечить зручний інтерфейс для кінцевих користувачів.

3.3 Проектування системи

Для розробки рекомендаційної системи необхідно описати її майбутню структуру, архітектуру та поведінку. Також слід візуалізувати потоки даних для розуміння їх обігу між компонентами системи. Для цього була обрана мова UML(Unified Modeling Language), яка є стандартною мовою моделювання, що використовується для специфікації, візуалізації, побудови та документації програмного забезпечення.

Слід врахувати, що для візуалізації потоків даних між компонентами системи, було використано діаграму потоків даних (Data Flow Diagram), яка є окремим інструментом структурного аналізу та проектування систем. Вона корисна для моделювання процесів та даних на високому рівні.

3.3.1 Архітектура застосунку

Архітектура застосунку спроєктована таким чином, що він імпортує набори даних зі сторонніх сервісів та попередньо нормалізує їх, приводячи дані до необхідного виду. Після цього, з ними взаємодіє компонент обробки даних, що відповідає за створення двох важливих структур даних, які включать інформацію про користувацькі оцінки елементів і подібність елементів між собою, та зберігає результат у сховищі даних.

З іншого боку, запити від користувача надходять до API, що в свою чергу надсилає їх до модулів формування рекомендацій, прогнозування оцінок та пошуку схожих елементів (в залежності від обраної дії користувачем). Вони ж, взаємодіють з компонентом роботи з рекомендаційним алгоритмом. Він бере збережені до цього дані з сховища, та, в залежності від потреб, розраховує й повертає необхідний користувачу результат.

Така структура дозволяє забезпечити модульність, ефективну обробку та нормалізацію даних, легке розширення окремих компонентів та модулів в подальшому, гнучкість та можливість працювати з будь-якими датасетами.

Архітектура застосунку зображена у вигляді структурної схеми, що наведена у графічному матеріалі(кресленику) з шифром ІК93.150БАК.005 Д5.

3.3.2 Діаграма компонентів

Діаграма компонентів є одним з видів діаграм Unified Modeling Language і використовується для візуалізації організації та взаємодії компонентів програмного забезпечення. Вона фокусується на модульній структурі системи, показуючи її

аспекти, такі як компоненти, інтерфейси, залежності між ними тощо. Компоненти ж можуть включати різні елементи системи, такі як класи, об’єкти, модулі, фреймворки чи бібліотеки. Вони відображають частини функціональності, що пов’язані з іншими компонентами через залежності або інтерфейси [14].

Було використано 3 компоненти. User Interface (користувацький інтерфейс), який відповідає за взаємодію між користувачем та рекомендаційною системою. Recommandation algorithm (Алгоритм надання рекомендацій), який генерує рекомендації та Data processor (Обробка даних), що оброблює та нормалізує дані. Кожен з них містить інші компененти, такі як фреймворки та бібліотеки, використані у функціоналі даних компонентів. UI має двосторонній асоціативний зв’язок з RA-компонентом, бо вони між собою однаково залежні, що описано в нотатках на діаграмі. Також DP-компонент надає послугу (оброблені дані) RA, що показано штрих-пунктирною направленою стрілкою (звичайна залежність).

Діаграму компонентів наведено у графічному матеріалі(кресленику) з шифром ІК93.150БАК.005 Д1.

3.3.3 Діаграма послідовності

Діаграма послідовності є ще одним з видів діаграм UML (Unified Modeling Language) і використовується для візуалізації взаємодій між об'єктами або компонентами системи відповідно до часу. Вона фокусується на представленні послідовності повідомлень, які обмінюються між учасниками, та на відображенні порядку, в якому ці взаємодії відбуваються.

Тобто, вона є абстрактною формою блок-схеми та покроково вказує дії системи від початку до кінця.

Діаграму послідовності та обмін повідомленнями між учасниками наведено у графічному матеріалі(кресленику) з шифром ІК93.150БАК.005 Д2.

3.3.4 Діаграма станів

Ще одна діграма уніфікованої мови моделювання (UML) – діаграма станів, що дозволяє візуалізувати можливі стани системи, тобто її поведінку, та дії, за допомогою яких вона переходить від одного стану до іншого.

Стани та дії, за допомогою яких спроєктовано поведінку системи що наведено у графічному матеріалі(кресленику) з шифром ІК93.150БАК.005 Д3.

3.3.5 Діаграма потоків даних

DFD (Data Flow Diagram) є графічним інструментом, який використовується для представлення потоків даних у системі. Він допомагає аналізувати, моделювати та документувати процеси, що відбуваються у системі. DFD використовується для візуалізації взаємодії між різними компонентами системи та потоку даних між ними.

Діаграму потоків даних наведено у графічному матеріалі(кресленику) з шифром ІК93.150БАК.005 Д4.

Висновок до розділу 3

У даному розділі було обгрунтовано вибір набору даних, який буде задіяний для реалізації рекомендаційної системи, та проведено аналіз його переваг і вмісту. Перевагами обраного датасету Anime Recommendations Database є:

- вирішення проблеми холодного старту;
- відкрита ліцензія для використання;
- відповідний діапазон оцінок;
- задовільний рівень рідкості даних.

Також, було обрано засоби для розробки системи та описано їх переваги. Середовищем (IDE) розробки – Vusial Studio Code, мовою програмування – Python,

фреймворком для реалізації інтерфейсу користувача – PyformsGUI, та певний набір бібліотек.

Слід враховувати, що для комерційної розробки сервісу перегляду відеоконтенту та його рекомендації необхідно буде залучити спеціалістів для створення презентаційного рівня застосунку. Тобто дану систему в подальшому можна буде використовувати для створення комерційних проектів, а у контексті дипломної роботи даний GUI - гарне рішення для демонстрації роботи рекомендаційної системи.

Було спроектовано майбутнє програмне рішення. Для цього візуалізовано структуру системи – діаграма компонентів (мова UML), показано поведінку системи – діаграма послідовності та діаграма станів (мова UML) та потоки даних – діаграма DFD (окремий інструмент). Та спроектовано архітектуру застосунку.

4 РЕАЛІЗАЦІЯ ПРОГРАМНОГО РІШЕННЯ

4.1 Попередня нормалізація та обробка даних

Для роботи алгоритму рекомендаційної системи, попередньо був нормалізований датасет та оброблено дані для подальшого використання.

Після імпорту файлу Anime.csv за допомогою команди read_csv бібліотеки pandas ми отримали структуру даних формату DataFrame розмірністю 12294 (кількість анімаційних фільмів) на 7, перші рядки якої зображені на рисунку 4.1 нижче.

	anime_id	name	genre	type	episodes	rating	members
0	32281	Kimi no Na wa.	Drama, Romance, School, Supernatural	Movie	1	9.37	200630
1	5114	Fullmetal Alchemist: Brotherhood	Action, Adventure, Drama, Fantasy, Magic, Mili...	TV	64	9.26	793665
2	28977	Gintama°	Action, Comedy, Historical, Parody, Samurai, S...	TV	51	9.25	114262
3	9253	Steins;Gate	Sci-Fi, Thriller	TV	24	9.17	673572
4	9969	Gintama'	Action, Comedy, Historical, Parody, Samurai, S...	TV	51	9.16	151266

Рисунок 4.1 – Перші рядки імпортованого файлу Anime.csv

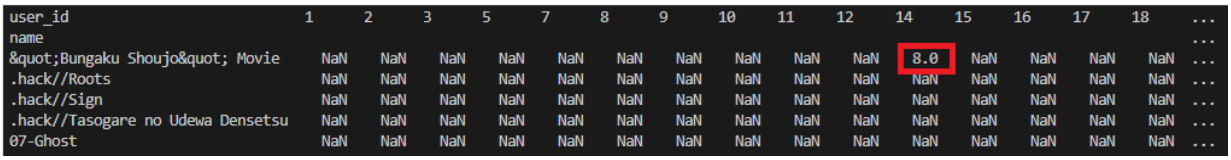
Для оптимізації роботи алгоритму, було вирішено обрати елементи, що мають типи Movie та TV відповідно, та найбільш переглянуті користувачами шляхом використання команди quantile(0.75) бібліотеки pandas. Це дозволило зменшити відсоток розрідженості даних, та скоротити кількість елементів до 1534. Таким же чином було імпортовано файл Rating.csv, перші рядки DataFrame якого зображено на рисунку 4.2 нижче.

	user_id	anime_id	rating
0	1	20	-1
1	1	24	-1
2	1	79	-1
3	1	226	-1
4	1	241	-1

Рисунок 4.2 – Перші рядки імпортованого файлу Rating.csv

Значення -1 показують те, що користувач не оцінював відповідний елемент, його було замінено на NaN для того, щоб надалі правильно конвертувати оцінки. Після нормалізації обох структур даних, було виконано їх злиття, та побудовано

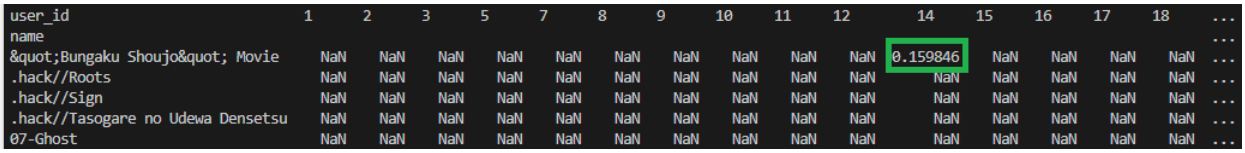
таблицю item-user, що вказує, який користувач якому елементу яку оцінку проставив. Частина таблиці продемонстрована на рисунку 4.3 нижче.



user_id	1	2	3	5	7	8	9	10	11	12	14	15	16	17	18	...
name																...
"Bungaku Shoujo" Movie	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	8.0	NaN	NaN	NaN	NaN	...
.hack//Roots	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...
.hack//Sign	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...
.hack//Tasogare no Udewa Densetsu	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...
07-Ghost	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...

Рисунок 4.3 – Частина таблиці item-user

Для обчислення подібності між елементами на основі item-user таблиці, необхідно центрувати дані навколо 0. Це виконано шляхом віднімання середнього значення кожного рядка від кожного значення в рядку з використанням функції nanmean бібліотеки numpy, що ігнорує NaN. В результаті отримано оцінки, які підходять для виміру центрованої косинусної подібності та не мають недоліку у вигляді зміщення. Результат такої нормалізації даних зображено на рисунку 4.4 нижче.



user_id	1	2	3	5	7	8	9	10	11	12	14	15	16	17	18	...
name																...
"Bungaku Shoujo" Movie	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0.159846	NaN	NaN	NaN	NaN	...
.hack//Roots	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...
.hack//Sign	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...
.hack//Tasogare no Udewa Densetsu	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...
07-Ghost	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...

Рисунок 4.4 – Центрування даних навколо 0

Оскільки обраним підходом для надання рекомендацій є підхід на основі елементів, для побудови таблиці item-item була використана метрика бібліотеки sclearn – cosine_similarity та попередьо замінені всі значення NaN на 0, оскільки цей метод вимагає, щоб всі значення були реальними числами. Це дозволяє додатково не оброблювати відсутні дані. Після обчислення схожості, була побудована таблиця item-item, яка вказує, наскільки елементи схожі між собою в діапазоні від 0 до 1. Головна діагональ цієї матриці заповнена одиницями, оскільки однакові анімаційні фільми є ідентичними. Частину даної таблиці зображено на рисунку 4.5 нижче.

name	"Bungaku Shoujo" Movie	.hack//Roots	.hack//Sign	.hack//Tasogare no Udewa Densetsu	07-Ghost	...
name						...
"Bungaku Shoujo" Movie	1.000000	0.032753	0.026011	0.036676	0.038012	...
.hack//Roots	0.032753	1.000000	0.289985	0.315053	0.074391	...
.hack//Sign	0.026011	0.289985	1.000000	0.269825	0.058850	...
.hack//Tasogare no Udewa Densetsu	0.036676	0.315053	0.269825	1.000000	0.047875	...
07-Ghost	0.038012	0.074391	0.058850	0.047875	1.000000	...

Рисунок 4.5 – Частина таблиці item-item

Після наведеного вище переліку дій, дані є готовими для застосування алгоритмом рекомендаційної системи.

4.2 Специфікація функцій застосунку

Для знаходження схожих елементів до цільового, створена функція get_similar_anime, яка приймає один параметр – назву анімаційного фільму. Якщо даний анімаційний фільм відсутній у таблиці item-item, то функція повертає None, що в подальшому у зовнішній області видимості дозволить перехопити цю помилку та обробити її. Якщо даний елемент присутній у таблиці, відбувається сортування елементів від найбільш схожого за коефіцієнтом косинусної подібності до найменш, після чого функція повертає список відсортованих елементів та список їх коефіцієнтів подібності. Ця функція є функцією нижчого порядку, використовуватиметься іншими.

Для прогнозу оцінки елементу для обраного користувача, створена функція predict_rating, яка приймає 3 параметри – ідентифікатор користувача, назву анімаційного фільму та кількість сусідів. Кількість сусідів використовується для пришвидшення роботи алгоритму, тобто для зменшення обчилювальної складності. Він визначає максимальну кількість схожих елементів, які будуть використовуватись для прогнозування оцінки. Стандартним значенням якого є 10. Функція працює наступним чином: використовується get_similar_anime для визначення елементів, які найбільш схожі на цільовий, та їх коефіцієнтів, після чого застосовується фільтрація до масиву анімаційних фільмів, щоб залишились тільки ті, які користувач вже оцінив, і на останок користувачу прогнозується оцінка для

обраного елемента на основі зваженої суми оцінок, які він вже дав схожим анімаційним фільмам, де ваги – це коефіцієнти подібності.

Для підбору рекомендацій для обраного користувача, була створена функція get_recommendation, яка приймає два параметри – ідентифікатор користувача та кількість елементів, які необхідно надати у вигляду рекомендаційної вибірки. Дана функція працює наступним чином: створюється порожній масив, який буде використовуватись для зберігання прогнозованих оцінок для кожного елементу, для кожного анімаційного фільму викликається функція predict_rating для прогнозування оцінки, яку цільовий користувач дав би цьому анімаційному фільму, після чого результат predict_rating додається до попередньо створеного масиву, відбувається фільтрація, щоб видалити всі елементи, які користувач вже оцінив, і на останок масив сортується в порядку від найбільшої до меншої оцінки, функція повертає 10 перших елементів, що й є рекомендаційною вибіркою.

Короткий опис специфікації кожної з наведених функцій представлено в таблиці 4.1 нижче.

Таблиця 4.1 – Специфікація функцій застосунку

Назва функції	Опис
get_similar_anime	Повертає список схожих елементів на цільовий, використовуючи таблицю item-item, що побудована з використанням метрики cosine_similarity (формула 2.1)
predict_rating	Прогнозує оцінку для обраного користувача обраному елементу з використанням формули 2.6
get_recommendation	Повертає список з 10 рекомендованих заданому користувачу елементів

4.3 Керівництво користувача

Для того, щоб розпочати користуватись застосунком, необхідно його запустити. Для цього є декільки шляхів: або натиснути подвійним кліком лівої кнопки мишки по файлу recommender.py (цільовий файл з кодом), або відкрити цей файл у середовищі, що підтримує мову програмування Python і натиснути кнопку “run and debug” в цьому середовищі (наприклад, VSCode). Ці дії відобразять головну форму застосунку, через яку буде відбуватись взаємодія з рекомендаційною системою (рисунок 4.6).

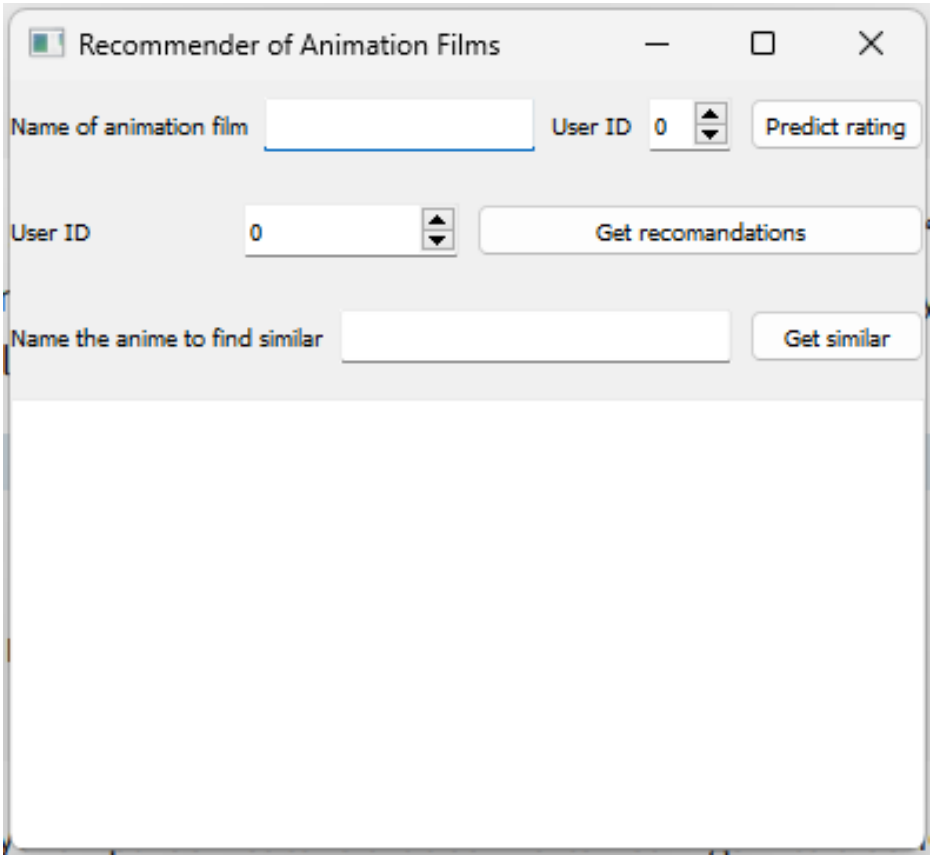


Рисунок 4.6 – Головна форма застосунку

Умовно дану форму можна поділити на три області. Перша область (рисунок 4.7) містить два поля – назва анімаційного фільму та індентифікатор користувача. Перше є текстовим, друге прийматиме тільки числа, починаючи з 0.

Рисунок 4.7 – Перша область форми

Після вводу з клавіатури назви анімаційного фільму та ідентифікатору користувача (або натисканням стрілочок) та натискання кнопки “Predict rating” – буде спрогнозована оцінка для даного елементу, та виведена на екран, що зображено на рисунку 4.8 нижче.

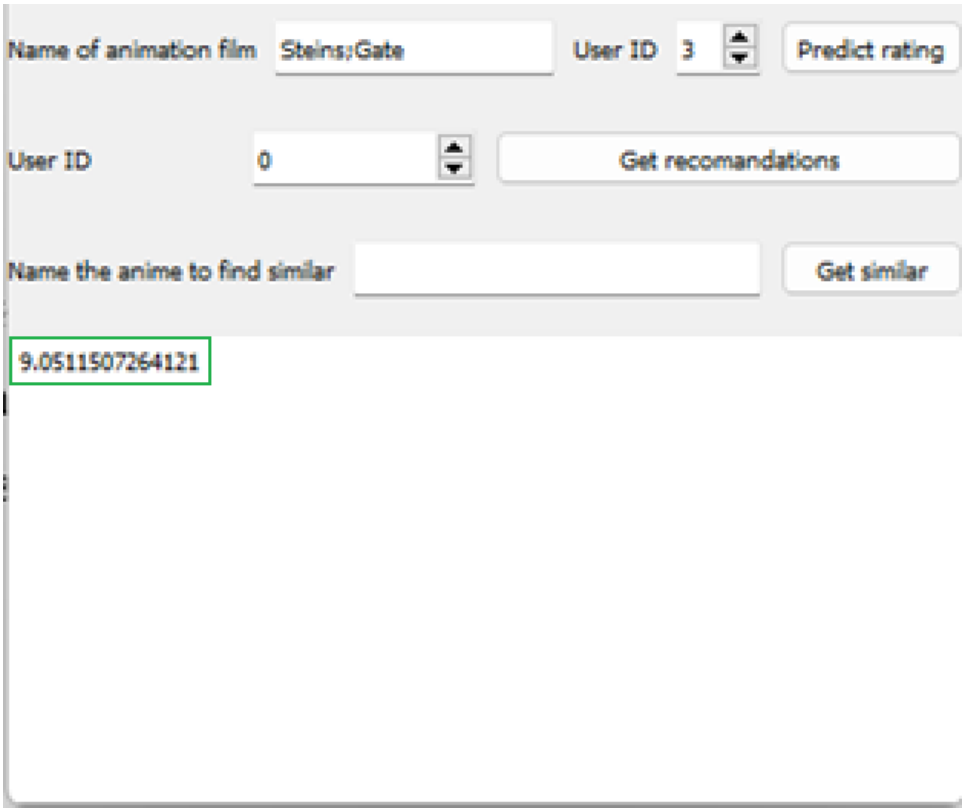


Рисунок 4.8 – Прогнозована оцінка для елементу

Якщо ж введений анімаційний фільм відсутній в системі, дана помилка буде перехоплена та виведено на екран відповідне повідомлення, що зображено на рисунку 4.9 нижче.

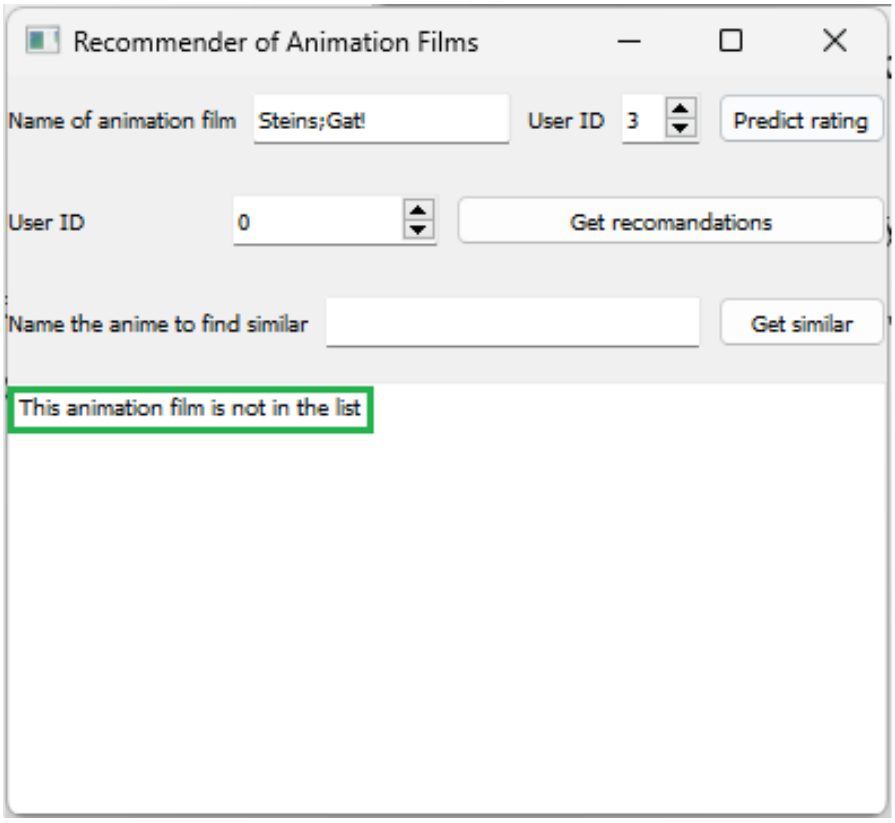


Рисунок 4.9 – Помилка вводу анімаційного фільму

Друга область (рисунок 4.10) містить одне поле для вводу – ідентифікатор користувача. Це поле є числовим.

Рисунок 4.10 – Друга область форми

Слід з використанням клавіатури, чи використанням стрілочок вверх та вниз ввести ідентифікатор користувача і натиснути кнопку “Get recomandations”, щоб отримати вибірку рекомендованих фільмів. Результат продемонстровано на рисунку 4.11 нижче.

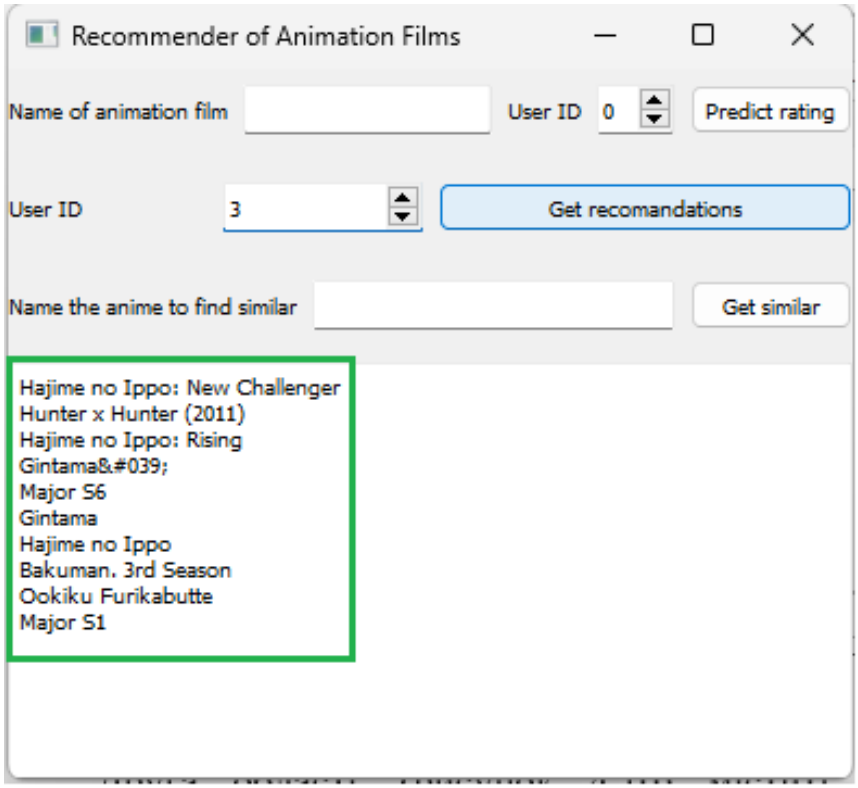


Рисунок 4.11 – Результат підбору рекомендацій

Третя область (рисунок 4.12) містить також одне поле для вводу – назва анімаційного фільму в тектовому форматі.

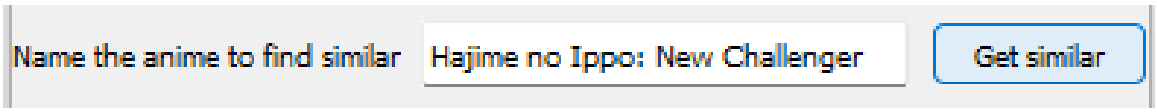


Рисунок 4.12 – Третя область форми

Слід ввести назву елементу з клавіатури та натиснути на кнопку “Get similar”, щоб отримати вибірку зі схожими анімаційнийними фільмами на той, що був введений. Результат відображено на рисунку 4.13 нижче.

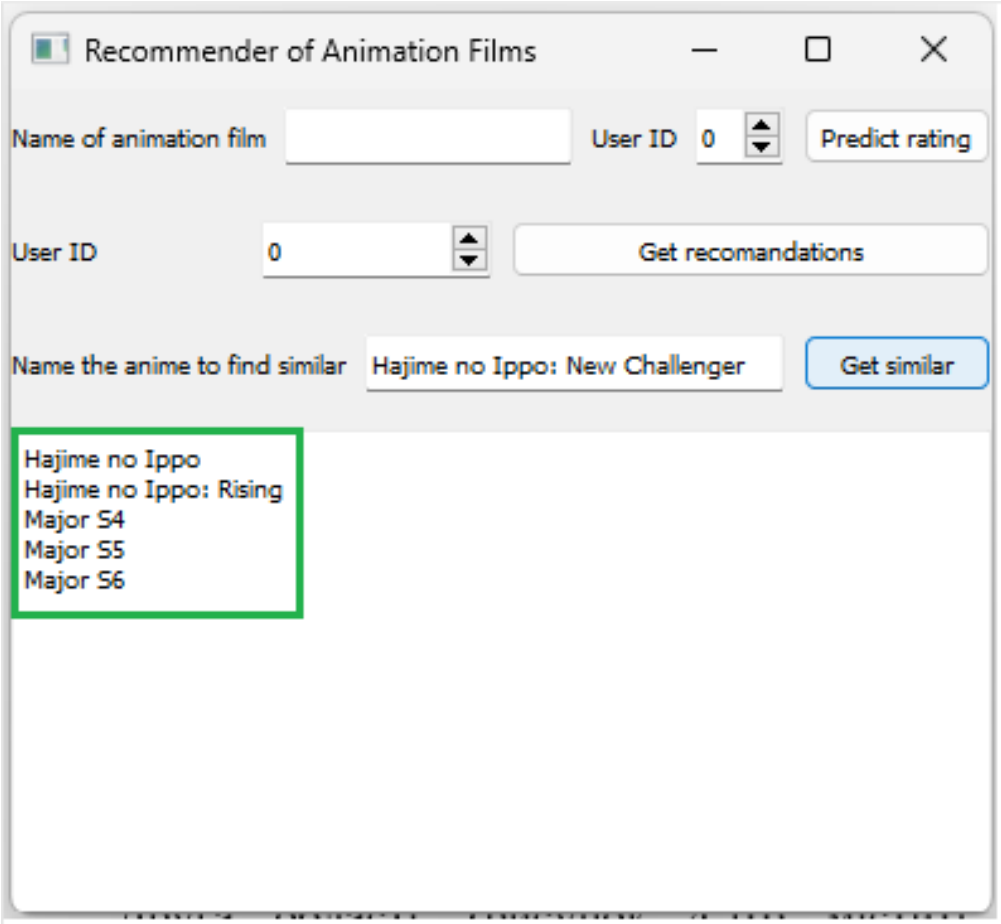


Рисунок 4.13 – Результат знаходження схожих елементів

У випадку, якщо введеного елементу немає в системі, буде перехоплена помилка та виведено відповідне повідомлення на екран подібно до того, як це було продемонстровано на рисунку 4.9 вище у даному розділі.

4.4 Випробовування розробленого програмного продукту

З метою дослідження якості програмного продукту, необхідно провести ряд випробувань для перевірки відповідності елементів застосунку їх властивостям. Під час даних випробовувань проводилась перевірка розробленого функціоналу, а результати записані до таблиць 4.2 – 4.5 нижче.

Таблиця 4.2 – Перевірка функціоналу прогнозування оцінки

Мета випробовування	Перевірити функціонал прогнозування оцінки для визначеного користувача визначеному анімаційному фільму
Початковий стан моделі	Відкрита основна форма застосунку
Випробовування	Перевірка функції прогнозування оцінки. Ввести ідентифікатор користувача та назву анімаційного фільму. Натиснути кнопку “Predict rating”
Очікуваний результат	Прогнозована оцінка повинна бути виведена у пусте поле нижче у форматі з плавуючою точкою
Стан моделі після випробовування	Прогнозована оцінка виведена у пусте поле нижче у форматі з плавуючою точкою

Отож, з таблиці 4.2 видно, що стан моделі після випробовування відповідає очікуваному результату. Це означає що функціонал прогнозування оцінки елементу працює правильно та без помилок.

Таблиця 4.3 – Перевірка функціоналу підбору рекомендацій

Мета випробовування	Перевірити функціонал формування рекомендаційної вибірки для визначеного користувача
Початковий стан моделі	Відкрита основна форма застосунку

Продовження до таблиці 4.3

Випробовування	Перевірка функції рекомендацій анімаційний фільмів. Ввести ідентифікатор користувача. Натиснути кнопку “Get reccomandations”
Очікуваний результат	Список рекомендацій (анімаційних фільмів) повинен бути виведений у пусте поле нижче у формі.
Стан моделі після випробовування	Список рекомендацій виведений у пусте поле нижче у формі

Отож, з таблиці 4.3 видно, що стан моделі після випробовування відповідає очікуваному результату. Це означає що функціонал підбору рекомендацій працює правильно та без помилок.

Таблиця 4.4 – Перевірка функціоналу знаходження схожих елементів

Мета випробовування	Перевірити функціонал знаходження схожих анімаційних фільмів
Початковий стан моделі	Відкрита основа форма застсосунку
Випробовування	Перевірка функції знаходження схожих елементів. Ввести назву анімаційного фільму. Натиснути кнопку “Get similar”
Очікуваний результат	Список схожих анімаційних фільмів на цільовий повинен бути виведений у пусте поле нижче у формі
Стан моделі після випробовування	Список схожих анімаційних фільмів на цільовий виведений у пусте поле нижче у формі

Отож, з таблиці 4.4 видно, що стан моделі після випробовування відповідає очікуваному результату. Це означає що функціонал знаходження схожих елементів працює правильно та без помилок.

Таблиця 4.5 – Перевірка функціоналу при негативній валідації даних

Мета випробовування	Перевірити роботу системи при невалідних вхідних даних
Початковий стан моделі	Відкрита основна форма застосунку
Випробовування	Перевірка системи, коли введеного елементу чи користувача не існує. Ввести id користувача, значення якого є 0. Ввести назву елементу, назва якого є “Stan1d;Err”. Натиснути на будь-яку кнопку у формі.
Очікуваний результат	Система повинна перехопити помилку, а повідомлення цієї помилки вивести у пусте поле нижче у формі.
Стан моделі після випробовування	Система перехопила помилку, а повідомлення цієї помилки виведено у пусте поле нижче у формі.

Отож, з таблиці 4.5 видно, що стан моделі після випробовування відповідає очікуваному результату. Це означає що система правильно оброблює помилку та не виходить з ладу при негативній валідації даних.

4.5 Тестування рекомендаційної системи

Для оцінки якості розробленої рекомендаційної системи анімаційних фільмів на базі алгоримту колаборативної фільтраці на основі елементів необхідно використати спеціальні метрики [15 - 18].

4.5.1 Покриття(Coverage)

Покриття є метрикою, яка вимірює відсоток елементів, які рекомендаційна система здатна запропонувати. Це показує різноманітність рекомендацій та здатність системи рекомендувати менш популярні елементи.

Для виміру порикриття, необхідно виконати наступні кроки:

– використовуючи рекомендаційну систему, необхідно згенерувати рекомендації для випадкової підмножини користувачів;

– об’єднати всі рекомендації та видалити повторення, щоб отримати унікальний набір елементів;

– розрахувати покриття, ділячи кількість унікальних рекомендованих елементів на загальну кількість елементів (усі анімаційні фільми, доступні для рекомендації);

– помножити отриманий результат на 100, щоб перетворити дріб у відсоток.

Формула 4.1 для розрахунку покриття наведена нижче.

$$cov = \frac{uniq}{general} 100\,, \tag{4.1}$$

де *uniq* – кількість унікальних елементів, *general* – загальна кількість елементів, а *cov* – покриття.

Вирішено зробити рекомендації для 15 користувачів і розширити список рекомендованих елементів для кожного до 25. Після отримання загального списку з 375 рекомендованих елементів, в результаті видалення повторень залишилось

335. Обрахунок відсотку покриття розробленої системи показано у формулі 4.2 нижче.

$$cov = 100 * \left(\frac{335}{1534}\right) = \ 21{,}83833116036506\ \%. \qquad (4.2)$$

Отриманим відсотком покриття $\approx 22\%$, що свідчить про гарний результат, обгрунтування якого зазначено в таблиці 4.6 нижче.

Таблиця 4.6 – Обгрунтування отриманого відсотку покриття

Різноманітність	Система демонструє здатність рекомендувати велику кількість унікальних елементів.
Релевантність	При цьому, система зберігає високий рівень релевантності рекомендацій
Ефективне використання ресурсів	Система не намагається рекомендувати все підряд, замість цього вона зосереджена на наданні найбільш релевантних рекомендацій, що оптимально використовує ресурси

Слід враховувати також, що такий відсоток покриття саме у нашому випадку можна рахувати гарним результатом, адже колекція елементів нараховує 1534, що є великим набором даних.

4.5.2 Середньоквадратична помилка(RMSE)

Середньостатистичне відхилення, чи помилка – це статистична метрика, яка вимірює середню величину помилки системи прогнозування. Це відстань, у середньому, між тим, що система прогнозує, та реальним значенням. У контексті рекомендаційних систем використовується для вимірювання точності прогнозованих рейтингів порівняно з фактичними рейтингами, які користувачі вже дали елементам. Чим менше значення RMSE, тим точніше система рекомендацій.

Середньоквадратична помилка обчислюється за формулою 4.3, що наведена нижче.

$$RMSE = \sqrt{\frac{1}{N} * \sum(predicted - actual)^2}, \quad (4.3)$$

де N – кількість оцінок в наборі для тестування; *predicted* – прогнозована оцінка; *actual* – фактична оцінка.

За тестовий набір було взято користувачів з id 1,2 та 5. А елементами, яким вони вже надали оцінки Hokuto no Ken та Hunter x Hunter (2011). Для кожного користувача була спрогнозована оцінка, результат чого занесено у таблицю 4.7 і наведено нижче.

Таблиця 4.7 – Прогнозовані та фактичні рейтинги елементам

Користувач	Назва анімаційного фільму	Фактичний рейтинг	Прогнозований рейтинг
1	Hokuto no Ken	9	9.345
1	Hunter x Hunter (2011)	8	7.654
2	Hokuto no Ken	9	9.788
2	Hunter x Hunter (2011)	6	5.004

Продовження до таблиці 4.7

Користувач	Назва анімаційного фільму	Фактичний рейтинг	Прогнозований рейтинг
5	Hokuto no Ken	7	7.303
5	Hunter x Hunter (2011)	4	3.433

На основі отриманих даних, проведено розрахунок RMSE, який показано у формулі 4.4 нижче.

$$RMSE = \sqrt{\frac{1}{6} * ((9.345-9)^2 + (7.654-8)^2 + (9.788-9)^2 + (5.004-6)^2 + (7.303-7)^2 + (3.433-4)^2)} =$$

$$= \sqrt{\frac{0.119025 + 0.119716 + 0.620944 + 0.992016 + 0.091809 + 0.321489}{6}} =$$

$$= \sqrt{\frac{2.264999}{6}} \approx 0.6144. \qquad (4.4)$$

У контексті шкали оцінки від 1 до 10 (в нашому випадку), середньквадратичне відхилення(RMSE) 0.6144 є відносно низьким, що свідчить про високу точність розробленої рекомендаційної системи. Це означає, що прогнози в середньому відхиляються на 0.6144 від фактичних рейтингів, тобто ми досить близькі до реальних оцінок користувачів.

Таким чином було вимірено розподіл помилок прогнозування, що є важливим фактором при оцінці якості роботи будь-якої рекомендаційної системи.

4.5.3 Середня абсолютна помилка(MAE)

Подібно до RMSE, MAE - це метрика якою можна виміряти точність системи, тут також обчислюється середнє відхилення прогнозованих значень від фактичних,

але не квадратично, а асболютно, з використанням модулю. Дана метрика була залучена для додаткової перевірки точності рекомендаційної системи. Формула 4.5 показує, яким чином можна обчислити середню абсолютну помилку, і зображена нижче.

$$MAE = \frac{1}{N} * \sum| \, actual - predicted \, | \, , \tag{4.5}$$

де N – кількість оцінок в наборі для тестування; *predicted* – прогнозована оцінка; *actual* – фактична оцінка.

Для обрахунку середньої абсолютної похибки візьмемо дані з минулого експерименту з таблиці 4.7. Обчислення MSE зображено у формулі 4.6 нижче.

$$\mathrm{M}AE = \frac{1}{6} * (0.345 + 0.346 + 0.788 + 0.996 \; + \; 0.303 \; + \; 0.567) = \\ = 0.5575. \tag{4.6}$$

У контексті рекомендаційних систем, при шкалі оцінок від 1 до 10 (в нашому випадку), значення середньої абсолютної похибки 0.5575 є відносно невеликим відхиленням. Так, як і у випадку з RMSE, це свідчить про досить високу точність прогнозів, за MAE метрикою прогнозована оцінка в середньому відхиляється від фактичної на 0.5575, що є гарним результатом для рекомендаційної системи відео-елементів на базі алгоритму колаборативної фільтрації.

Слід зазначити, що дані результати задовольняють наші вимоги на високому рівні, але якщо б таке значення було, наприклад, в області медицини чи фінансів, то воно рахувалось занадто високим [19].

Результати оцінки якості розробленої рекомендаційної системи занесено до таблиці 4.8, що наведена нижче.

Таблиця 4.8 – Результати оцінки якості системи

Покриття	RMSE	MAE
22%	0.6144	0.5575

Висновок до розділу 4

У даному розділі було продемонстровано яким чином проходить нормалізація набору даних, та його попередня обробка для його подальшого використання алгоритмом формування рекомендацій. Це включало скорочення кількості елементів, злиття двох структур даних в одну, заміна та конвертування значень оцінок, формування таблиць item-item та item-user.

Також було наведено інструкцію, яка для більшої зручності була проілюстрована. За допомогою неї корисутвач знатиме як запустити застосунок, та користуватись ним в подльшому для роботи з рекомендаційною системою. Застосунок успішно пройшов ряд випробувань на відповідність функціоналу елементам, стан моделі був очікуваним при нестандартних ситуаціях. На останок було проведено оцінку якості рекомендаційної системами. Метриками було обрано MAE, RMSE та відсоток покриття. Згідно результатів оцінки, можна стверджувати, що розроблена система є якісною.

ВИСНОВКИ

У пояснювальній записці було оглянуто існуючі сервіси, що використовують рекомендаційні системи, проаналізовано наявні типи рекомендаційних систем, проведено порівняльний аналіз підходів надання рекомендацій, обґрунтовано вибір типу системи рекомендацій, стратегії та підходу, сфери впровадження та виду елементів. Досліджено способи обрахунку коефіцієнтів схожості, методу прогнозування оцінки, їх особливості та математичний зміст. Спроектовано та описано реалізацію системи. Проведено ряд випробувань та тестувань.

У першому розділі було розглянуто сервіси, що використовують рекомендаційні системи, різних сфер. Описано особливості предметної області, проведено порівняльний аналіз стратегій до створення систем та підходів.

У другому розділі проаналізовано та наведено математичні основи способів знаходження схожих елементів в рекомендаційних системах колаборативної фільтрації на основі елементів та методу прогнозування оцінок для неї.

У третьому розділі було обґрунтовано вибір інформаційного забепечення, засобів для розробки системи та візуалізовано поведінку й структуру розроблюваної системи за допомогою UML та DFD діаграм, поставлено змістовну та математичну задачу, обґрунтовано метод вирішення. Також було спроєктовано архітектуру застосунку. Принцип роботи кожної з діаграм також був описаним у розділі.

В останньому розділі описано модуль нормалізації та попередньої обробки даних, специфікацію функцій застосунку, наведено інструкцію користувача. Було проведено ряд випробувань програмного застосунку для перевірки правильності роботи функціоналу та реакції системи на нестандартні ситуації. Проведено тестування розробленої рекомендаційної системи для оцінки її якості.

Система, розроблена в даній дипломній роботі, може в подальшому бути використана як готовий компонент, для створення повноцінного веб-сервісу потокового відтворення відео. Для цього необхідно розробити презентаційний рівень застосунку – користувацький інтерфейс. Також, може бути розроблена

власна база даних, яка буде зберігати не тільки явні дані, як оцінки чи відгуки, а й неявні, як історія перегляду чи взаємодія з елементом.

До переваг розроблюваної системи можна віднести її просту архітектуру, що в подальшому дозволить легко модернізувати та розширити рекомендаційну систему, високу точність надання рекомендацій та відповідний відсоток покриття, що свідчить про високу якість.

Розробивши рекомендаційну систему анімаційних фільмів на базі алгоритму колаборативної фільтрації та провівши її оцінку, можна стверджувати, що мета дипломної роботи – полегшення процесу пошуку контенту користувачам зі схожими вподобаннями, досягнута в повній мірі.

В подальшому, для покращення системи, можна розглянути моделі, побудовані з використанням нейронних мереж та злиття різних типів рекомендаційних систем, наприклад базовану на вмісті та колаборативної фільтрації.

ПЕРЕЛІК ПОСИЛАНЬ

1. Ekstrand, M. D., Riedl, J., & Konstan, J. A. (2015). Collaborative filtering recommender systems. Foundations and Trends® in Human-Computer Interaction, 4(2), P. 81-173.
2. Ricci, F., Rokach, L., & Shapira, B. Introduction to recommender systems handbook // Recommender systems handbook. Springer, 2011. P. 1-35.
3. Herlocker, J. L., Konstan, J. A., & Riedl, J. Explaining collaborative filtering recommendations // Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work. 2000. P. 241-250.
4. Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. Item-based collaborative filtering recommendation algorithms // Proceedings of the 10th International Conference on World Wide Web. 2001. P. 285-295.
5. Desrosiers, C., & Karypis, G. A comprehensive survey of neighborhood-based recommendation methods // Recommender systems handbook. Springer, 2011. P. 107-144.
6. Zhang, J., & Wang, S. A survey on collaborative filtering-based recommender systems // Advanced Data Mining and Applications. Springer, 2013. P. 1-17.
7. Ricci, F., & Werthner, H. Context-aware recommender systems // Recommender systems handbook. Springer, 2010. P. 217-253.
8. Ekstrand, M. D., Riedl, J., & Konstan, J. A. (2015). Collaborative filtering recommender systems. Foundations and Trends® in Human-Computer Interaction, 4(2), P. 81-173.
9. Ricci, F., Rokach, L., & Shapira, B. (2015). Recommender systems handbook (2nd ed.). Springer.
10. Ponomarenko, I., Afinogenov, E., & Afinogenova, E. (2021). Context-aware recommendation system for movie selection in video-on-demand platform. In Proceedings of the 2021 IEEE Conference of Young Researchers in Electrical and Electronic Engineering (EIConRus) P. 2151-2155.

11. Linden, G., Smith, B., & York, J. (2003). Amazon.com recommendations: Item-to-item collaborative filtering. IEEE Internet Computing, 7(1), P. 76-80.
12. McFee, B., Bertin-Mahieux, T., Ellis, D. P., & Lanckriet, G. R. (2012). The million song dataset challenge. In Proceedings of the 21st International Conference on World Wide Web P. 909-916.
13. Kostikov, A., Fisun, M., & Gasanov, E. (2019). Collaborative filtering with weighted NMF for movie recommendation on the Megogo video platform. In Proceedings of the 4th International Conference on Digital Transformation and Global Society P. 301-311.
14. Herlocker, J. L., Konstan, J. A., & Riedl, J. (2004). Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems (TOIS), 22(1), P. 5-53.
15. Pyforms [Електронний ресурс] – Режим доступу до ресурсу: https://pyforms.readthedocs.io
16. UML [Електронний ресурс] – Режим доступу до ресурсу: https://evergreens.com.ua/ua/articles/uml-diagrams.html
17. Scikit-learn [Електронний ресурс] – Режим доступу до ресурсу: https://scikit-learn.org
18. DFD [Електронний ресурс] – Режим доступу до ресурсу: https://www.techtarget.com/data-flow-diagram-DFD
19. Covington, P., Adams, J., & Sargin, E. (2016). Deep neural networks for YouTube recommendations. In Proceedings of the 10th ACM Conference on Recommender Systems P. 191-199.