НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет/(ННІ) інформаційних технологій

ПОГОДЖЕНО	**ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ**
Декан факультету (Директор ННІ) інформаційних технологій (назва факультету (ННІ))	**Завідувач кафедри** комп'ютерних наук (назва кафедри)
_____ Ігор БОЛБОТ (підпис) (ім'я ПРІЗВИЩЕ)	_____ Белла ГОЛУБ (підпис) (ім'я ПРІЗВИЩЕ)
"___" _____ 2025 р.	"___" _____ 2025 р.

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему «Рекомендаційна інформаційна система фільмів на основі вподобань користувачів»

Спеціальність 122 «Комп'ютерні науки»
(код і найменування)

Освітня програма «Інформаційні управляючі системи і технології»
(назва)

Орієнтація освітньої програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Гарант освітньої програми
кандидат технічних наук, доцент _____ Белла ГОЛУБ
(науковий ступінь та вчене звання) (підпис) (ім'я ПРІЗВИЩЕ)

Керівник магістерської кваліфікаційної роботи
кандидат технічних наук, доцент _____ Віталій СВАТКО
(науковий ступінь та вчене звання) (підпис) (ім'я ПРІЗВИЩЕ)

Виконала _____ Ольга ПОЛЮХОВИЧ
(підпис) (ім'я ПРІЗВИЩЕ здобувача)

КИЇВ – 2025

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет (ННІ) інформаційних технологій

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних наук
доцент, к.т.н. _____ **Белла ГОЛУБ**
(науковий ступінь, вчене звання) (підпис) ((ім'я ПРІЗВИЩЕ))
"____" _____ 2025 року

З А В Д А Н Н Я

ДО ВИКОНАННЯ МАГІСТЕРСЬКОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ ЗДОБУВАЧУ

Полюхович Ользі Вікторівні
(прізвище, ім'я, по батькові)

Спеціальність 122 "Комп'ютерні науки"
(код і найменування)

Освітня програма "Інформаційні управляючі системи та технології"
(назва)

Орієнтація освітньої програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Тема магістерської кваліфікаційної роботи «Рекомендаційна інформаційна система фільмів на основі вподобань користувачів»
затверджена наказом від 01.11.2024р. №1964 «С»

Термін подання завершеної роботи на кафедру _____
(рік, місяць, число)

Вихідні дані до магістерської кваліфікаційної роботи: публічний набір даних про рейтинги фільмів з сайту "MovieLens".

Перелік питань, що підлягають дослідженню:

1.Огляд існуючих методів та алгоритмів роботи рекомендаційних систем, їх порівняльний аналіз.

2.Дослідження можливості застосування OLAP та Data Mining методів для підвищення ефективності роботи рекомендаційних алгоритмів.

3. Розробити прототип рекомендаційної системи фільмів, використовуючи розглянуті методи.

Перелік графічного матеріалу (за потреби) постер; презентація.

Дата видачі завдання "_____"_____20___ р.

Керівник магістерської кваліфікаційної роботи _____ Віталій СВАТКО
(підпис) (ім'я ПРІЗВИЩЕ)

Завдання прийняла до виконання _____ Ольга ПОЛЮХОВИЧ
(підпис) (ім'я ПРІЗВИЩЕ

ЗМІСТ

ВСТУП

Актуальність теми. У сучасному інформаційному суспільстві обсяги даних зростають експоненціально, і користувачі стикаються з проблемою вибору з-поміж величезної кількості доступного контенту. Рекомендаційні системи стали ключовим інструментом для вирішення цієї проблеми, оскільки дозволяють персоналізувати взаємодію користувача з інформаційним середовищем, зменшують інформаційне перевантаження та підвищують рівень задоволення користувачів.

Світові компанії, такі як Netflix, Spotify, YouTube, TikTok, активно застосовують алгоритми рекомендацій, що значно впливає на утримання аудиторії та формування споживчих уподобань. Ефективність цих систем напряму залежить від коректного вибору алгоритмів, їхньої адаптивності та здатності до навчання на основі поведінкових даних.

Дослідження алгоритмів рекомендацій та створення власної моделі є актуальним завданням, оскільки воно поєднує науковий інтерес з практичними потребами бізнесу. Особливу значущість це має у сфері рекомендації фільмів, де якісна персоналізація визначає конкурентоспроможність сервісів.

Мета і завдання дослідження. Основна мета даного дослідження — аналіз сучасних підходів до побудови рекомендаційних систем, виявлення їхніх проблем та обмежень, а також розробка та впровадження гібридної моделі рекомендацій для формування персоналізованих пропозицій фільмів із підвищеною точністю та ефективністю.

Для досягнення поставленої мети необхідно вирішити такі завдання:

- Здійснити огляд існуючих методів та алгоритмів роботи рекомендаційних систем, проаналізувати їх сильні та слабкі сторони, а також провести порівняльний аналіз підходів, що застосовуються у популярних платформах;
- Дослідити можливості використання методів OLAP та Data Mining для підвищення ефективності роботи алгоритмів рекомендацій;

- Формалізувати задачу формування рекомендацій на основі гібридного підходу;
- Розробити прототип рекомендаційної системи фільмів, реалізувавши розглянуті алгоритми та здійснивши їх практичне тестування.

Об’єкт дослідження — алгоритми рекомендацій контенту на основі вподобань користувачів.

Предмет дослідження — рекомендаційна система фільмів на основі вподобань користувачів.

Методи дослідження. Для вирішення поставлених завдань використовувалися методи системного аналізу для оцінки ефективності алгоритмів, об'єктно-орієнтованого проєктування, методи машинного навчання та інтелектуального аналізу даних для створення моделей рекомендацій, експериментальні методи для перевірки працездатності прототипу системи.

Наукова новизна одержаних результатів полягає у формалізації задачі побудови рекомендаційної системи фільмів на основі гібридного підходу до фільтрації даних для вдосконалення процесу формування особистих рекомендацій. Додатковим внеском є дослідження можливостей інтеграції OLAP-технологій для підвищення продуктивності та адаптивності алгоритмів рекомендацій.

Практичне значення роботи полягає у створенні прототипу рекомендаційної системи фільмів, що може бути використаний як основа для розробки повноцінних інформаційних сервісів у сфері кіноіндустрії та медіа. Запропонований підхід може бути адаптований для інших галузей, де необхідна персоналізація інформації.

Структура та обсяг роботи. Пояснювальна записка складається з чотирьох основних розділів.

У першому розділі проводиться аналіз предметної області, огляд рекомендаційних систем та постановка завдання даної роботи, в ході якої визначаються вимоги до розроблюваної системи.

Другий розділ присвячено дослідженню загальноприйнятих алгоритмів створення рекомендацій, визначається їх суть та проблеми, які виникають в ході їх виконання. Проводиться дослідження та практична реалізація зазначених методів програмними засобами на експериментальному наборі даних, аналізуються результати. Досліджуються рекомендаційні системи популярних розважальних платформ.

У третьому розділі описується моделювання системи, наводяться діаграми. Проводиться інтелектуальний аналіз даних робочого датасету, застосування OLAP технологій, досліджується їх застосування в рамках рекомендаційних систем.

Четвертий розділ демонструє готову систему та її архітектуру.

Апробація. Матеріали роботи були опубліковані в тези під назвою «Рекомендаційна інформаційна система фільмів на основі вподобань користувачів» в збірнику наукових праць «ТЕОРЕТИЧНІ ТА ПРИКЛАДНІ АСПЕКТИ РОЗРОБКИ КОМП'ЮТЕРНИХ СИСТЕМ '2025'», 19 квітня 2025 року, НУБіП України, Київ. Режим доступу: http://econference.nubip.edu.ua/index.php/taacsd/2025/paper/view/3626.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1 Поняття та класифікація рекомендаційних систем

Система рекомендацій — є підкласом систем фільтрації інформації, яка прагне передбачити «рейтинг» або «перевагу», яку користувач надасть певному елементу. Вони можуть базуватися на різних критеріях, включаючи минулі покупки, історію пошуку, демографічну інформацію та інші фактори. Системи рекомендацій є дуже корисними, оскільки допомагають користувачам знаходити продукти та послуги, які вони, можливо, не знайшли б самостійно.

Системи рекомендацій навчені розуміти переваги, попередні рішення та характеристики людей і продуктів, використовуючи дані, зібрані про їх взаємодію. До них належать перегляди, кліки, лайки та покупки. Завдяки своїй здатності передбачати інтереси та бажання споживачів на високому персоналізованому рівні, системи рекомендацій є улюбленими серед постачальників контенту та продуктів. Вони можуть привернути увагу споживачів до будь-якого продукту або послуги, що їх цікавить, від книг до відео, курсів з охорони здоров'я та одягу [1].

По суті, будь-яка система рекомендацій працює з двома сутностями – користувачами та об’єктами, де кожен користувач залишає оцінку або задає перевагу певному об’єкту. Такі оцінки збираються двома шляхами: явним та неявним. У першому випадку користувач самостійно виставляє бали за шкалою або обирає позначку з фіксованого діапазону. У другому – інформація отримується опосередковано, наприклад, із журналів кліків чи часу, проведеного на сторінці. Більшість систем поєднують обидва підходи, формуючи матрицю «користувач – об’єкт», відому як utility matrix.

Основна складність полягає в тому, що ця матриця зазвичай є дуже розрідженою: користувачі схильні оцінювати лише невелику частину доступних об’єктів. Тому головне завдання системи рекомендацій – прогнозування відсутніх значень у матриці. При цьому найбільший інтерес становлять високі

оцінки, адже саме вони формують основу для рекомендацій. Якість роботи системи значною мірою залежить від використаного алгоритму та характеристик джерела даних (контекстні, текстові, візуальні тощо).

Рекомендаційні системи зазвичай поділяють на три групи: контентно-орієнтовані, колаборативні та гібридні. Схематичне зображення цього поділу наведено на рис. 1.1 [2].

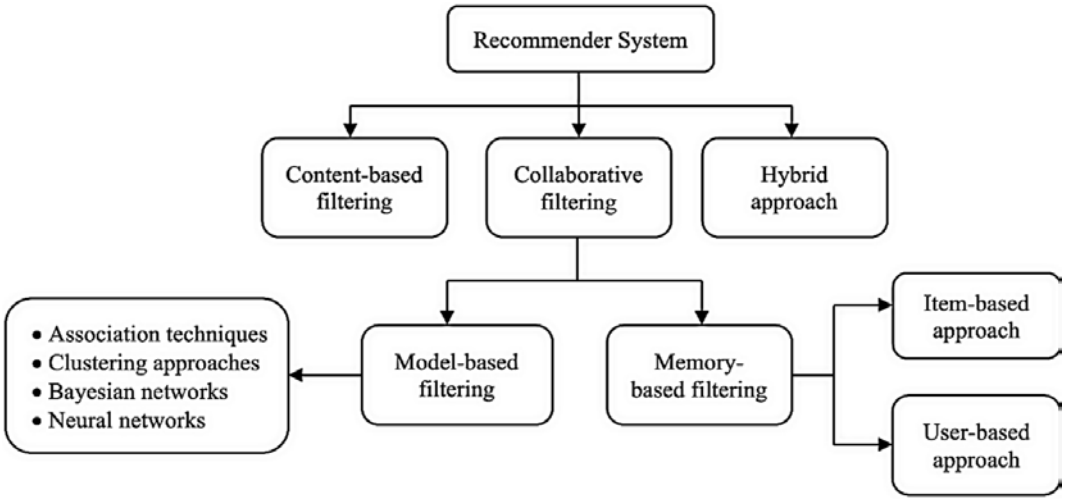


Рис. 1.1 Типи систем рекомендацій

1.2 Етапи процесу формування рекомендацій

1. Збір інформації. На цьому етапі відбувається накопичення даних про користувача з метою формування його профілю або побудови моделі для прогнозування. Використовуються відомості про характеристики користувача, його поведінку та контент, з яким він взаємодіє. Ефективність рекомендаційної системи безпосередньо залежить від точності сформованої моделі користувача: чим більше даних вона враховує, тим якіснішими є рекомендації.

Системи можуть використовувати різні типи вхідних даних:

- явний зворотний зв’язок (фідбек) – коли користувач самостійно оцінює елементи (наприклад, фільми чи товари);

- неявний зворотний зв’язок – коли система робить висновки про інтереси користувача, аналізуючи його поведінку (перегляди, покупки, кліки, час перебування на сторінці тощо);
- гібридний підхід, який поєднує обидва типи, мінімізуючи їхні недоліки [2].

2. Фаза навчання. На цьому етапі система застосовує алгоритми машинного навчання для аналізу зібраного зворотного зв’язку. Завдання – виокремити ключові характеристики користувача та підготувати модель, яка дозволить передбачати його майбутні уподобання.

3. Фаза формування прогнозування, рекомендацій. Завершальний етап полягає у формуванні рекомендацій: система визначає, які об’єкти можуть зацікавити користувача. Це може відбуватися на основі даних, зібраних на етапі збору інформації, або ж за допомогою моделі, побудованої в процесі навчання. Рекомендації можуть ґрунтуватися як на минулих діях користувача, так і на поведінці інших користувачів із подібними інтересами.

1.3 Виклики для рекомендаційних систем

Розглянемо основні проблеми сучасних систем рекомендацій та способи їх подолання.

- Проблема холодного старту виникає, коли система не має достатньо даних для формування рекомендацій для нових користувачів або нових елементів. Можливі рішення: запитати уподобання користувача, попросити оцінити кілька елементів або використовувати демографічні дані для первинних рекомендацій.
- Атаки Шиллінга з’являються, коли зловмисники підробляють оцінки, щоб змінити популярність елементів, знижуючи надійність системи. Вирішення проблеми — швидке виявлення зловмисників та видалення фальшивих профілів і оцінок.

- Проблема синонімічності виникає, коли однакові або схожі товари мають різні назви, що знижує точність рекомендацій. Її можна розв’язати за допомогою демографічного фільтрування, автоматичного розширення термінів або сингулярного розкладу.
- Проблема затримки характерна для колаборативного фільтрування та виникає при додаванні нових елементів, які потрібно перевірити перед рекомендацією. Рішення — застосування контентного фільтрування або кластеризації та обчислення в автономному режимі.
- Розрідженість даних з’являється, коли користувачі оцінюють небагато елементів, що знижує точність рекомендацій. Її пом’якшують за допомогою демографічного фільтрування, сингулярного розкладу та модельних методів.
- Проблема сірих овець виникає, коли відгуки користувача не схожі на жодного сусіда, через що система не може точно прогнозувати вподобання. Вирішення — використання методів на основі контенту.
- Проблема масштабованості з’являється через зростання обсягів даних та користувачів, що ускладнює обробку. Вирішення — зменшення розмірності та кластеризація користувачів для обчислень у невеликих групах [2].

1.4 Постановка завдання

У сучасних умовах інтенсивного зростання обсягів інформації системи рекомендацій відіграють ключову роль у забезпеченні персоналізованого доступу користувачів до релевантних даних. Проте існуючі методи мають низку обмежень, серед яких: проблема холодного старту, розрідженість даних, низька масштабованість, упередженість рекомендацій та уразливість до атак типу «Шиллінг» тощо. Ці фактори знижують ефективність та надійність роботи рекомендаційних систем.

Таким чином, актуальною є задача дослідження сучасних методів побудови систем рекомендацій, аналізу їхніх переваг і недоліків, а також розробки підходу, який забезпечить підвищення точності, масштабованості та стійкості до типових проблем.

Основна мета даного дослідження — аналіз сучасних підходів до побудови рекомендаційних систем, виявлення їхніх проблем та обмежень, а також розробка та впровадження гібридної моделі рекомендацій для формування персоналізованих пропозицій фільмів із підвищеною точністю та ефективністю.

Для досягнення поставленої мети необхідно вирішити такі завдання:

- Здійснити огляд існуючих методів та алгоритмів роботи рекомендаційних систем, проаналізувати їх сильні та слабкі сторони, а також провести порівняльний аналіз підходів, що застосовуються у популярних платформах;
- Дослідити можливості використання методів OLAP та Data Mining для підвищення ефективності роботи алгоритмів рекомендацій;
- Розробити прототип рекомендаційної системи фільмів, реалізувавши розглянуті алгоритми та здійснивши їх практичне тестування.

1.5 Вимоги до системи

Чітко визначені вимоги є важливими ознаками на шляху до успішного проекту. Вимоги до програмного рішення описують конкретні характеристики, які повинен мати продукт, щоб задовольнити потреби зацікавлених сторін.

1.5.1. Функціональні вимоги

Функціональні вимоги – це особливості продукту або функції, які розробники повинні реалізувати, щоб користувачі могли виконувати свої завдання. Функціональні вимоги визначають, що повинен робити продукт, які його особливості та функції, а також описують поведінку системи за певних умов.

Функціональні вимоги розробленої інформаційної системи візуально представлені у вигляді діаграми прецедентів на рис. 1.2.

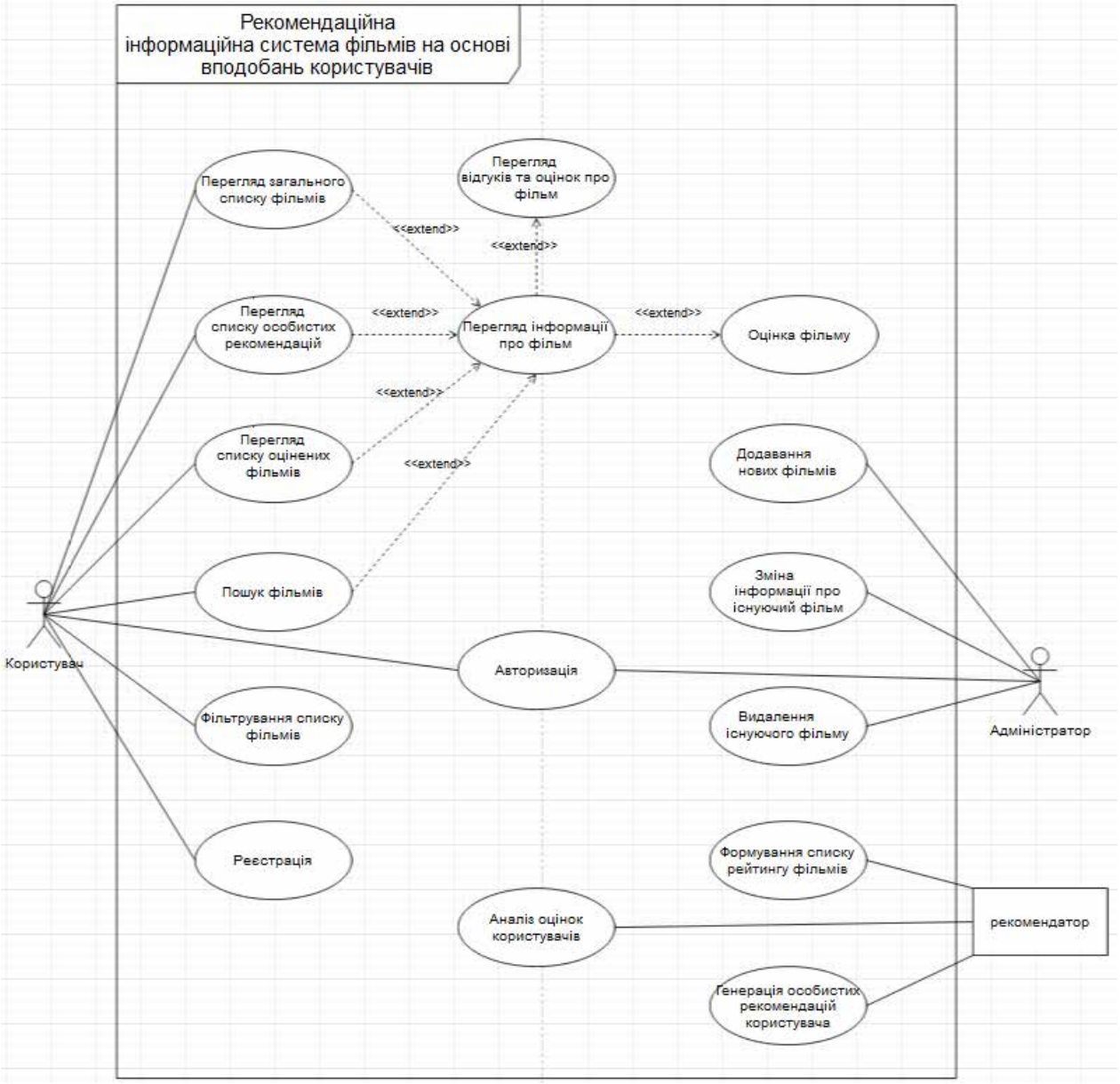


Рис. 1.2 Діаграма прецедентів

Специфікація прецедентів:

Оцінка фільму

Цей юзкейс описує процес оцінки вибраного користувачем фільму

1.1. Передумови

Користувач має намір оцінити вибраний фільм

1.2. Тригер

Користувач натискає кнопку «Залишити відгук»

1.3. Головний потік

1. Користувач натискає кнопку «Залишити відгук».
2. Система завантажує вікно заповнення відгуку.
3. Користувач обирає оцінку для фільму по 5-ти бальній шкалі
4. Якщо користувач хоче залишити текстовий відгук/рецензію, запускається під-потік S1, інакше - текстове поле залишається пустим.
5. Користувач натискає кнопку «Зберегти».
6. Система зберігає відгук і відображає сповіщення про успішність операції.

1.4. Під-потоки

S1 «Написання відгуку»

1. Користувач заповнює текстове поле відгуку.

1.5. Альтернативні потоки

Відсутні

Пошук фільмів

Цей юзкейс описує процес пошуку фільмів з списку

1.1. Передумови

Користувач має намір знайти фільм за назвою

1.2. Тригер

Користувач натискає на пошуковий рядок у вікні списку фільмів

1.3. Головний потік

1. Користувач натискає на пошуковий рядок та вводить назву фільму
2. Система опрацьовує пошуковий запит
3. Система відображає результат пошуку (E1)

1.4. Під-потоки

Відсутні

1.5. Альтернативні потоки

E1: введено некоректний пошуковий запит, система виводить повідомлення про відсутність результатів та пропонує користувачеві ввести запит заново або завершити прецедент.

Перегляд інформації про фільм

Цей юзкейс описує процес перегляду інформації про вибраний користувачем фільм

1.1. Передумови

Користувач має намір переглянути інформацію про вибраний фільм

1.2. Тригер

Тригером є вибір фільму з списку

1.3. Головний потік

1. Користувач натискає на назву або обкладинку фільму зі списку
2. Система завантажує сторінку з інформацією про фільм

1.4. Під-потоки

Відсутні

1.5. Альтернативні потоки

Відсутні

1.5.2 Нефункціональні вимоги

Продуктивність:

- система повинна забезпечувати час відгуку не більше 5 секунд для більшості операцій;

- система повинна бути здатна обробляти одночасно до 1000 запитів користувачів без помітного зниження продуктивності.

Безпека:

- всі користувачі повинні проходити аутентифікацію з використанням унікального логіну та паролю.

Надійність:

- відновлення після збою: У разі системного збою, система повинна відновлюватися протягом 30 хвилин.

Масштабованість:

- система повинна підтримувати можливість додавання нових користувачів та функцій без суттєвого зниження продуктивності.

Юзабіліті:

- інтерфейс користувача повинен бути інтуїтивно зрозумілим та зручним у використанні для користувачів з різним рівнем технічної підготовки.

1.5.3 Технічні вимоги

Мінімальні апаратні вимоги для розробки та тестування:

1. Процесор: Intel Core i5 або еквівалентний (4 ядра);
2. Оперативна пам'ять: 8 ГБ RAM;
3. Відеокарта: NVIDIA GTX 1050 або еквівалентна;
4. Місце на диску: 10 ГБ вільного простору;
5. Операційна система: Windows 10 або MacOS 10.15.

2 ДОСЛІДЖЕННЯ АЛГОРИТМІВ РЕКОМЕНДАЦІЙ

2.1 Основні підходи до побудови алгоритмів рекомендацій

Системи рекомендацій використовуються для прогнозування оцінки користувача щодо невідомого елемента та рекомендації цього елемента, якщо прогнозований бал є високим. Для досягнення цієї цілі використовують фільтрування на основі співпраці, фільтрування за змістом або гібридні методи, що поєднують дві або більше технік рекомендацій з метою подолання обмежень кожної з них окремо.

2.1.1. Фільтрування за змістом (Content-based filtering)

У контент-орієнтованих рекомендаційних системах кожен елемент описується за допомогою набору характеристик, на основі яких формується його профіль. Наприклад, для книги це може бути автор чи видавництво, для фільму – режисер, актори та інші параметри. Якщо користувач позитивно оцінює певний об’єкт, його характеристики додаються до профілю користувача, що формується з об’єднання профілів усіх схвалених ним елементів [2]. Подальші рекомендації формуються шляхом пошуку нових елементів, які відповідають цим характеристикам (рис. 2.1) [1].

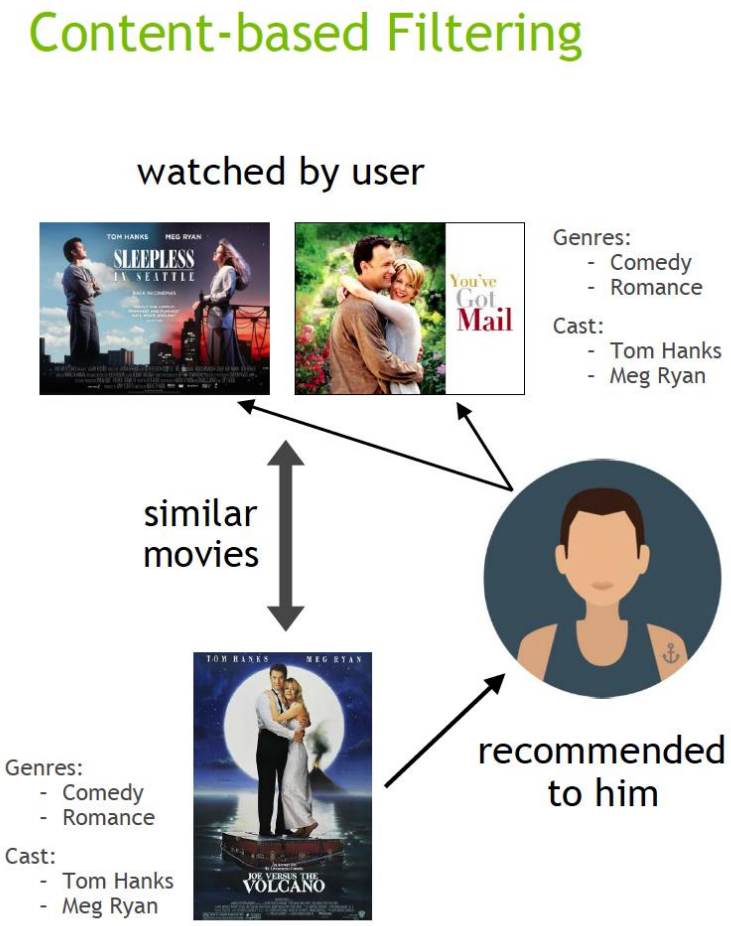


Рис.2.1 Фільтрування за змістом

Основним недоліком такого підходу є потреба у детальному описі кожного елемента, що не завжди можливо. Крім того, система має обмежені можливості розширення інтересів користувача за межі вже наявних уподобань. Водночас метод має низку переваг: він добре адаптується до змін інтересів у часі, працює незалежно від даних інших користувачів (що підвищує рівень приватності) та здатний долати проблему «холодного старту» — за умови наявності достатнього опису навіть новий елемент може бути рекомендований.

Завдяки цим властивостям методи контент-фільтрації широко застосовуються у персоналізованих системах новин, рекомендаціях статей, веб-сторінок та подібних сервісах [2].

2.1.2. Фільтрування на основі співпраці (Collaborative-based filtering)

У колаборативних підходах рекомендації ґрунтуються на вимірюванні схожості між користувачами. Спочатку формується група користувачів X, чиї вподобання найбільш близькі до користувача A; цю групу називають його «сусідством». Нові елементи, які популярні серед користувачів із цього сусідства, пропонуються користувачу A. Ефективність методу залежить від точності визначення сусідства.

На рисунку 2.2 зображено приклад колаборативного фільтрування [1].

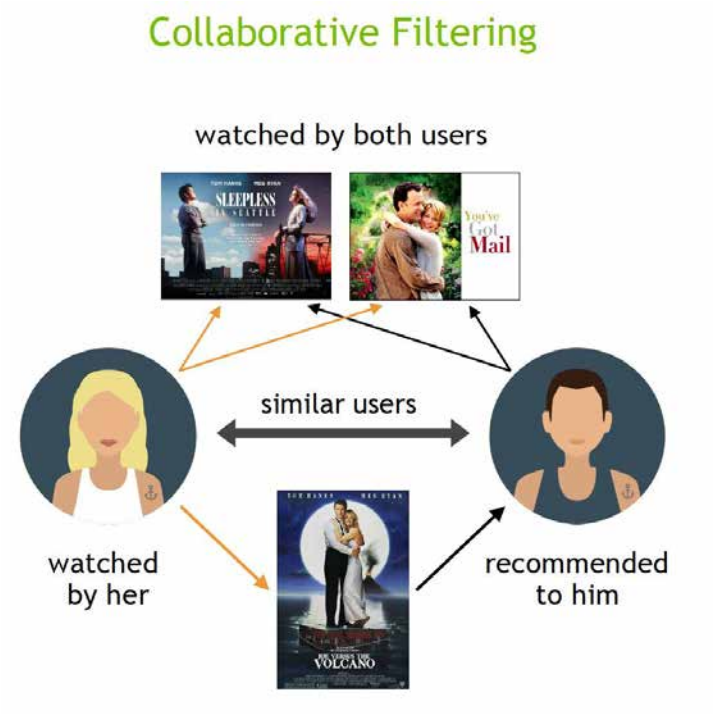


Рис. 2.2 Колаборативне фільтрування

Традиційні алгоритми такого типу мають певні недоліки, зокрема проблему «холодного старту» та ризики для конфіденційності, адже для роботи необхідно обмінюватися користувацькими даними. Водночас важливою перевагою є те, що для формування рекомендацій не потрібно знати характеристики самих елементів, а також можливість розширювати інтереси користувача за рахунок відкриття нових об’єктів.

1.2.3. Гібридне фільтрування (Hybrid filtering)

Гібридні методи поєднують дві або більше технік рекомендацій з метою подолання обмежень кожної з них окремо. Інтеграція може відбуватися різними

способами: об’єднанням результатів кількох алгоритмів, використанням контентного фільтрування всередині колаборативного методу чи, навпаки, застосуванням колаборативного підходу в межах контентного. Така комбінація зазвичай підвищує точність і ефективність рекомендаційних систем у різних сферах. Серед основних стратегій гібридизації виділяють метарівневу, розширення та комбінацію ознак, змішану, каскадну, перемикальну й зважену гібридизацію. Опис цих підходів наведено в таблиці 2.1 [2].

Таблиця 2.1 –Гібридні підходи

Назва	Опис
Метарівень	Попередньо навчена модель використовується як вхідні дані для іншої системи рекомендацій
Комбінація функцій	Функції однієї системи рекомендацій вводяться в іншу.
Розширення функцій	Результат однієї моделі застосовується як вхідні дані для іншої.
Змішана гібридизація	Вихідні дані різних систем рекомендацій змішуються, і комбінований результат подається як рекомендація.
Каскадна гібридизація	Одна система імпровізує вихідні дані іншої.
Перемикання гібридизації	Виберіть одну модель рекомендацій на основі поточних вимог.

Зважена гібридизація	Оцінки різних технік агрегуються для обчислення єдиної рекомендації.

2.3. Реалізація методів та аналіз результатів

В рамках цього дослідження, програмними засобами Python, було реалізовано кожен з вищезазначених методів, аби порівняти отримані результати для визначення найбільш ефективного підходу.

Проєкт базується на припущенні, що користувачі переглянули лише ті фільми, які вони оцінили. Якщо фільм не має оцінки, вважається, що користувач його не бачив — тому його можна рекомендувати. Рекомендаційні системи прогнозують оцінку користувача для невідомого фільму та пропонують його, якщо прогнозований бал високий.

Код з реалізацією цих методів представлено у додатку А. Розглянемо принцип роботи методів:

1) Контентний метод

Принцип роботи даного методу полягає в аналізі характеристик фільмів і порівнянні їх між собою з метою виявлення схожості. Для цього використовується модель TF-IDF (Term Frequency – Inverse Document Frequency), яка перетворює текстові описи фільмів (жанр, сюжет, ключові слова, актори, режисер тощо) у числові вектори. На основі цих векторів обчислюється косинусна міра подібності (Cosine Similarity), що визначає ступінь схожості між двома фільмами. Косинусна міра подібності – це міра схожості між двома векторами. Чим менший кут між ними, тим вища подібність. Якщо розглядати фільми чи користувачів як вектори, де ознаки є їх вимірами, то косинус кута між ними визначає ступінь їх схожості.

Після цього система формує матрицю схожості, у якій кожен фільм порівнюється з іншими. Для користувача, який позитивно оцінив певні фільми, система добирає нові, найбільш подібні за змістом.

2) Колаборативний метод

Колаборативна фільтрація ґрунтується на припущенні, що користувачі з подібними оцінками мають схожі вподобання. Реалізація методу полягає у формуванні матриці “користувач–фільм”, де зберігаються оцінки, виставлені користувачами.

На основі цієї матриці обчислюється схожість між користувачами за допомогою косинусної міри або кореляції. Далі для цільового користувача визначається група найсхожіших користувачів, і на основі їхніх оцінок прогнозується рейтинг фільмів, які цільовий користувач ще не переглядав. Отримані прогнозовані оцінки використовуються для формування списку рекомендацій.

3) Гібридний метод

Гібридний підхід поєднує результати контентної та колаборативної фільтрації для підвищення точності й різноманітності рекомендацій. У межах реалізації спочатку обчислюються окремо контентні та користувацькі показники схожості, після чого їх результати об’єднуються. Остаточний рейтинг формується шляхом усереднення значень косинусної схожості між фільмами та користувачами.

Ефективність запропонованих моделей рекомендаційних систем визначається точністю обчислення косинусної подібності між векторами фільмів та користувачів. Косинусна подібність виступає основним показником схожості між об’єктами або користувачами, що дозволяє системі формувати релевантні рекомендації. Чим вищий показник подібності між профілем користувача та характеристиками фільму, тим більш імовірно, що рекомендація буде корисною. Таким чином, точність розрахунку косинусної подібності безпосередньо впливає на якість і релевантність отриманих результатів, забезпечуючи ефективність роботи системи в цілому.

На рисунку 2.3 зображено графік з значеннями косинусних подібностей кожного з реалізованих методів.

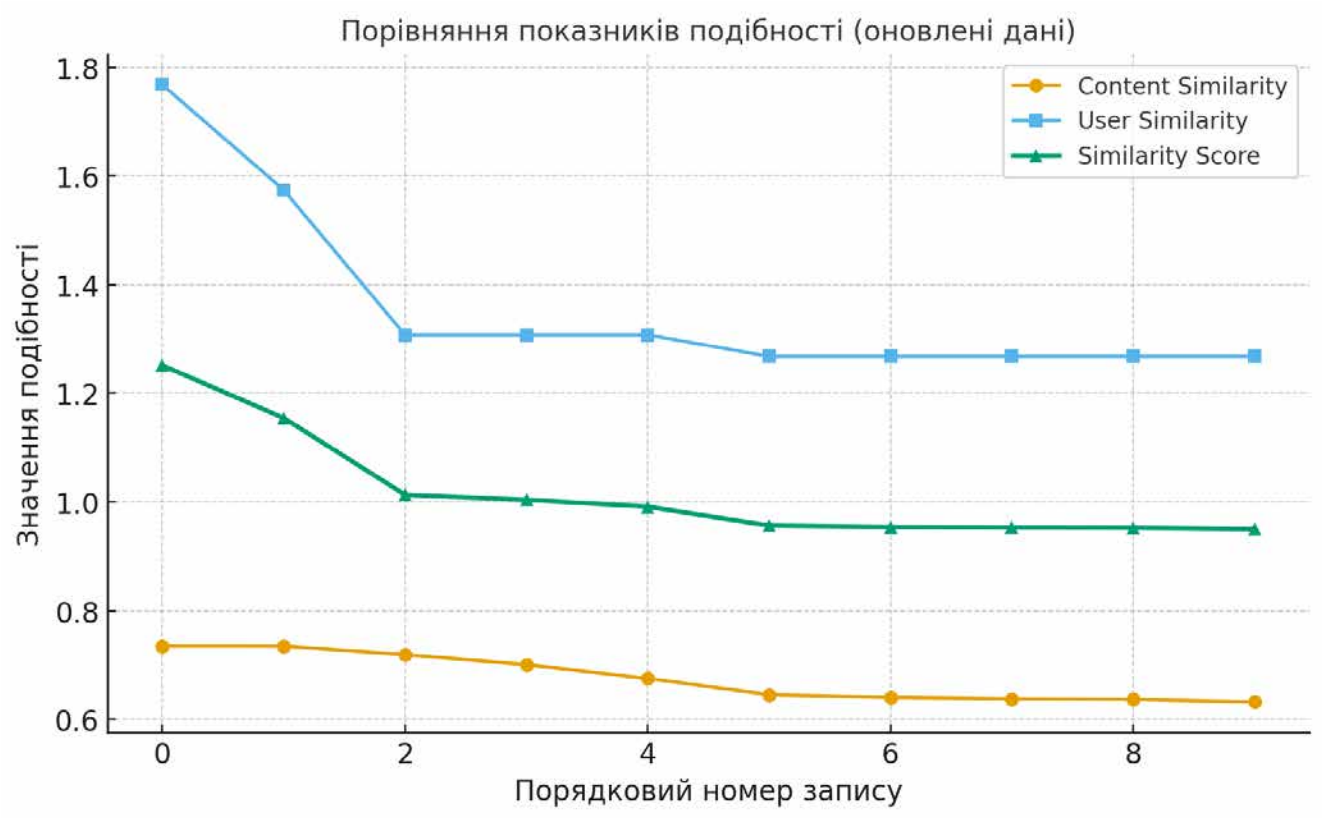


Рис. 2.3 Косинусна подібність

Як можна побачити, найнижчий показник має контентний метод, середній гібридний метод, найвищий — колаборативний.

Гібридний метод показав хоч і не найвищий результат, але має сталий вагомий показник точності отриманих рекомендацій, при цьому запобігаючи проблемам, які виникають з використанням колаборативного та контентного алгоритмів окремо. В зв’язку з чим можна зробити висновок про його загальну підвищену ефективність.

2.4. Аналіз наявних рекомендаційних систем популярних платформ

У сучасному інформаційному суспільстві обсяг цифрового контенту невпинно зростає, що призводить до перевантаження користувачів інформацією та ускладнює процес вибору релевантних матеріалів. У таких умовах забезпечення персоналізованого доступу до контенту стає одним із ключових факторів успішної діяльності компаній у цифровій сфері. Конкуренція за увагу

користувачів змушує провідні платформи активно впроваджувати алгоритми рекомендацій, здатні формувати індивідуалізовані пропозиції та підтримувати довгострокову взаємодію з аудиторією.

Прикладами ефективного використання персоналізованих систем рекомендацій є глобальні платформи TikTok та YouTube. Саме алгоритмічні механізми відбору контенту визначають залученість користувачів, забезпечують їхнє утримання та стимулюють активність у створенні власних матеріалів. Таким чином, рекомендаційні системи виступають не лише інструментом оптимізації користувацького досвіду, але й одним із ключових чинників конкурентоспроможності компаній на сучасному ринку цифрових технологій.

Розглянемо особливості підходів до рекомендаційних систем на деяких популярних платформах - TikTok, YouTube, Netflix.

1) TikTok

TikTok, міжнародна версія китайського додатку Douyin (2016), з'явився у 2017 році й швидко здобув популярність завдяки унікальній системі рекомендацій. На відміну від традиційних соціальних мереж, де користувачі переважно орієнтуються на підписки, TikTok формує стрічку «Для тебе» (For You) на основі алгоритмів машинного навчання, що аналізують поведінку користувача: перегляди, лайки, час взаємодії з відео. Важливою особливістю є акцент на короткому форматі та здатності відео миттєво привернути увагу, що відповідає швидкому сценарію споживання контенту. Саме рекомендаційна система стала основою стрімкого розвитку TikTok, адже більшість переглядів здійснюється через For You, що робить її критично значущою як для користувачів, так і для авторів [4]. На рис. 2.4 зображена статистика з особистого акаунту, яку надає сам TikTok.

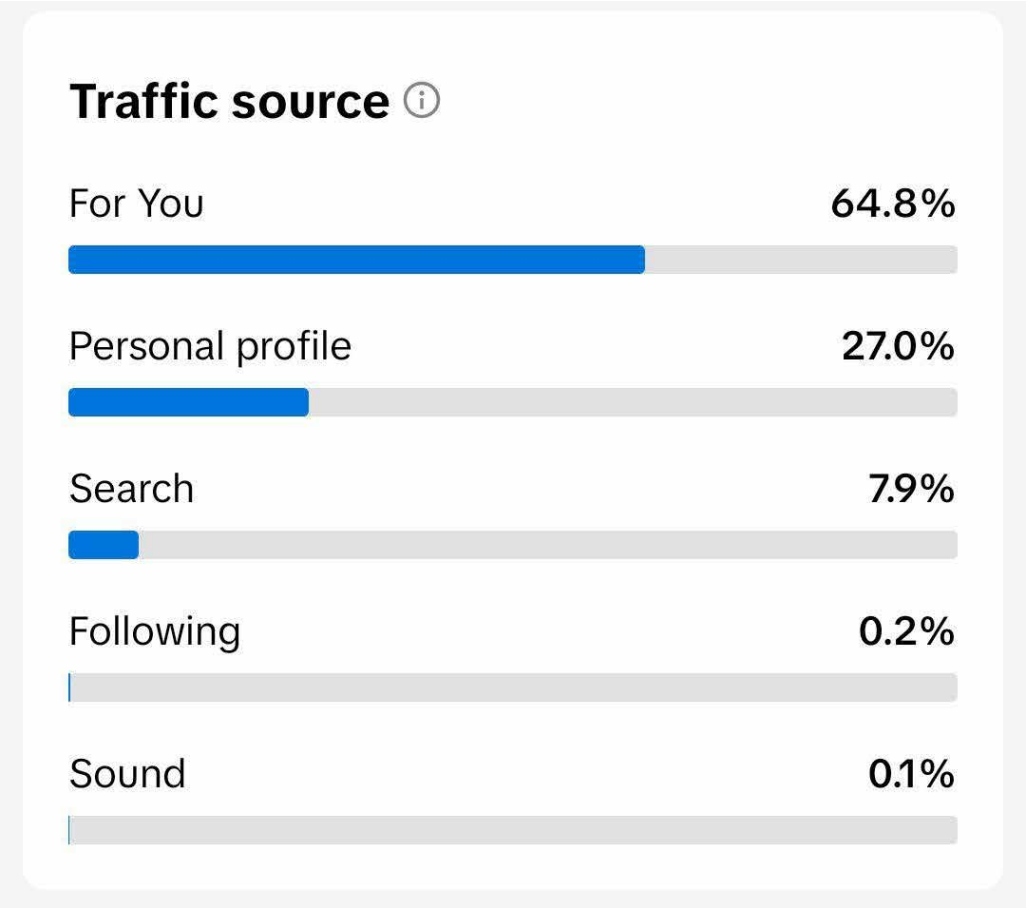


Рис. 2.4 Статистика переглядів TikTok акаунту

За офіційними даними, на формування рекомендацій у TikTok впливають такі чинники:

- взаємодія користувача (лайки, коментарі, поширення, перегляди, пропуски, підписки на акаунти);
- дані про відео (музичний супровід, хештеги, описи, кількість переглядів, країна публікації);
- дані про користувача (мовні налаштування, місцезнаходження, часовий пояс, тип пристрою).

Алгоритм TikTok оцінює ймовірність зацікавленості користувача у відео, орієнтуючись переважно на поведінкові сигнали (наприклад, перегляд до кінця) і мінімізуючи вагу соціальних факторів, як-от кількість підписників автора. Він функціонує як «мережа інтересів», а не як соціальна мережа. Висока чутливість до дій користувача забезпечує динамічне оновлення стрічки, створюючи ефект

«занурення» — за даними внутрішніх звітів TikTok, для формування персоналізованої «фільтрувальної бульбашки» достатньо близько 20 хвилин активності [6].

2) YouTube

Система рекомендацій YouTube є центральним механізмом платформи, що визначає контент на головній сторінці (Home) та у блоці «Up Next». Спершу (2008) вона ґрунтувалася на популярності відео, що обмежувало релевантність і змушувало користувачів звертатися до пошуку чи зовнішніх посилань. Згодом було впроваджено колаборативну фільтрацію, яка враховувала індивідуальні звички перегляду й подібності між користувачами.

Сьогодні система базується на машинному навчанні, яке аналізує понад 80 млрд сигналів на день: кліки, час перегляду, лайки/дизлайки, коментарі, поширення та результати опитувань. Ключовим нововведенням стало врахування часу перегляду (2012), що хоч і зменшило кількість переглядів, проте підвищило точність рекомендацій і рівень задоволеності користувачів.

На рис. 2.5 зображена частина мережі рекомендацій похідних відео, де розмір вузла відображає загальний ступінь, колір вузла відображає категорію відео, вузли позначені назвою відео та кількістю коментарів. Тут ми можемо візуально побачити шляхи, якими може піти користувач – хоча більшість рекомендованих відео є новинами та політичними відео, є декілька, які зачіпають розваги, людей та блоги, спорт та домашніх тварин і тварин [7].

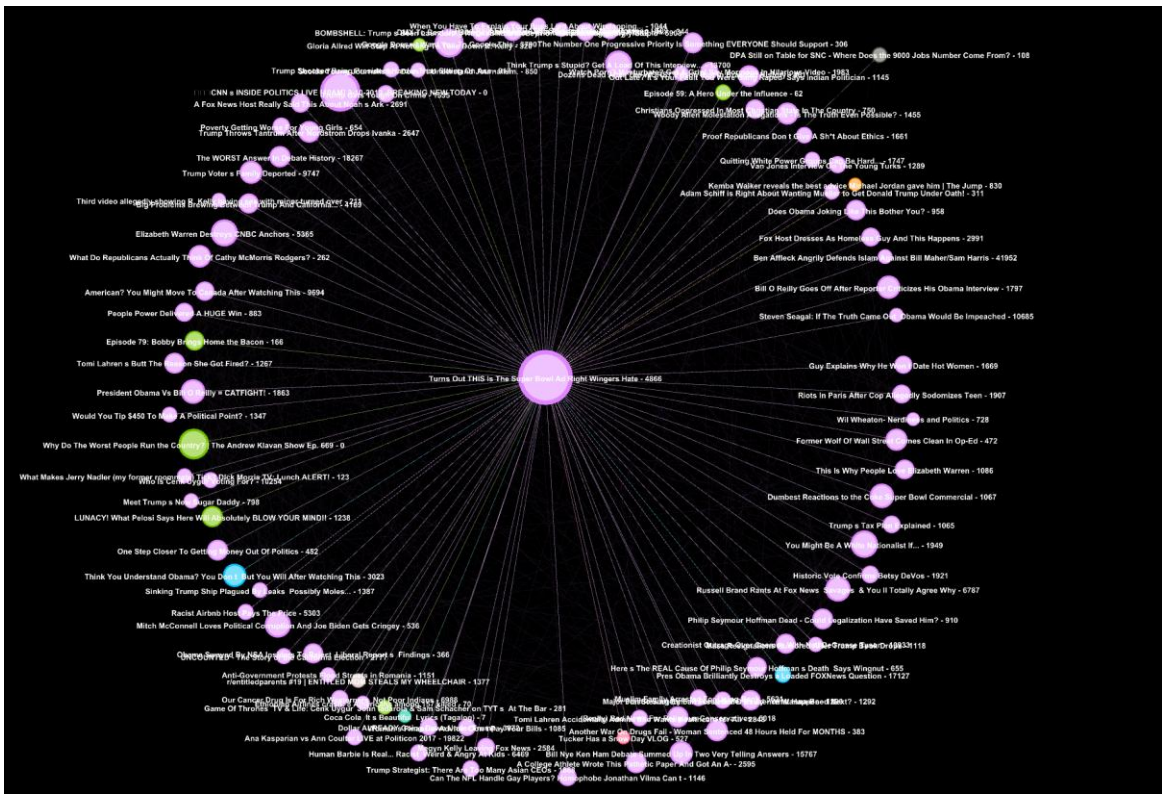


Рис. 2.5 Частина мережі рекомендацій відео

Сучасна система рекомендацій YouTube враховує не лише кількісні сигнали, а й якість та достовірність контенту. Для цього використовуються класифікатори, що відсіюють шкідливі відео, експертні оцінки, а також механізми виявлення межового контенту, схильного до дезінформації. Такі відео знижуються в ранжуванні, а авторитетні джерела — просуваються.

Алгоритм адаптується до індивідуальних звичок: наприклад, якщо користувач часто ділиться випадковим контентом, сигнал «поділитися» для нього поступово знецінюється. У результаті YouTube працює як багаторівнева адаптивна модель, що поєднує колаборативну та контентну фільтрацію, глибинне навчання й оцінку якості. Це демонструє перехід від простих моделей на основі популярності до персоналізованих систем, де враховуються як поведінкові дані, так і етичні обмеження [8].

3) Netflix

Бізнес-модель Netflix базується на підписці, де ключову роль відіграють персоналізовані рекомендації: 80% переглядів здійснюється саме завдяки ним. Алгоритм оцінює ймовірність зацікавленості користувача, враховуючи історію

переглядів, оцінки, «список перегляду», подібність до поведінки інших глядачів та метадані контенту (жанр, актори, рік тощо).

Система також бере до уваги контекст: час доби, мову, пристрій, тривалість перегляду, але не використовує прямі демографічні характеристики. На старті профілю вподобання задаються вибором кількох тайтлів, далі їхня вага зменшується на користь актуальних дій.

Домашня сторінка формується з динамічних «рядів»: алгоритм визначає, які категорії показати, які тайтли додати та в якому порядку їх розташувати. Найрелевантніший контент відображається у верхній частині інтерфейсу.

На рис. 2.6 зображені персоналізовані «ряди» рекомендацій з вказаною причиною для вибору.

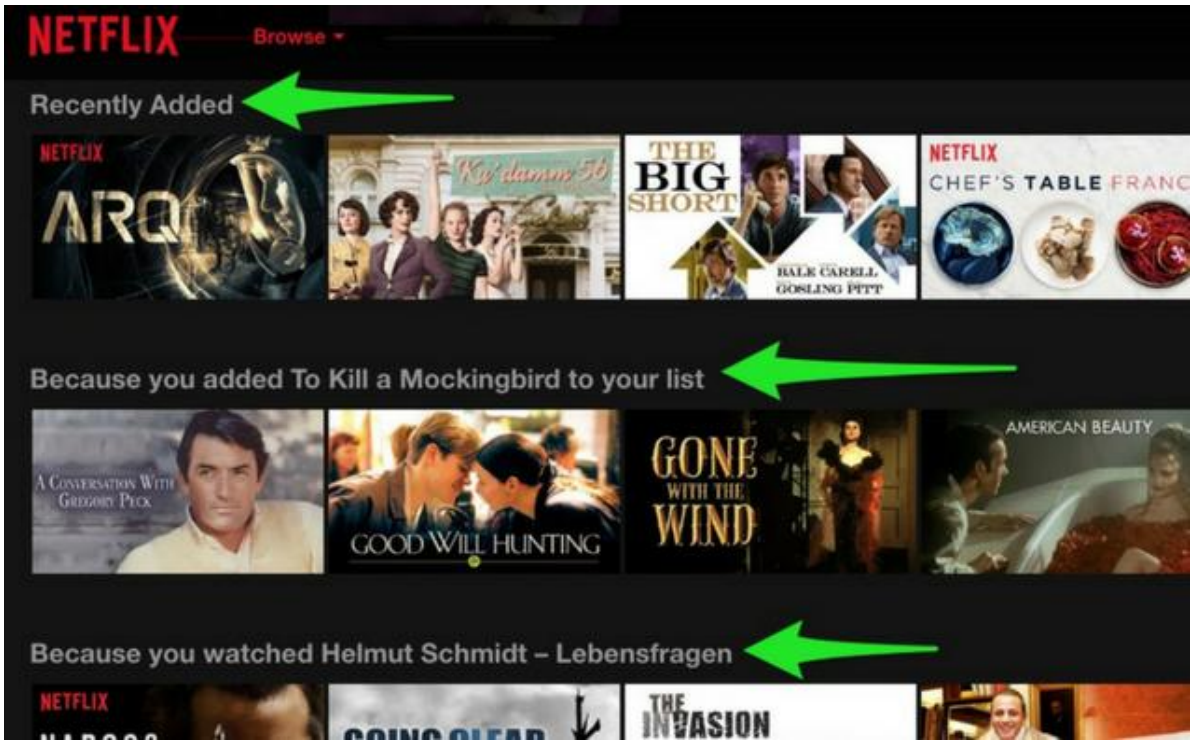


Рис. 2.6 «Ряди» рекомендацій Netflix

Система рекомендацій Netflix постійно вдосконалюється завдяки зворотному зв’язку: запуск, перегляд до кінця, позитивні чи негативні оцінки. Ці сигнали формують замкнений цикл, у якому дані та алгоритми безперервно оновлюють пріоритети.

Якщо користувач звертається до пошуку, результати також персоналізуються — враховується популярність запитів та їхня релевантність для конкретного профілю [9].

У підсумку, алгоритм Netflix являє собою багатофакторну модель, що поєднує колаборативну й контентну фільтрацію, адаптивне ранжування та аналіз поведінкових сигналів. Вона самонавчається й забезпечує високий рівень персоналізації досвіду користувача.

3 МОДЕЛЮВАННЯ СИСТЕМИ

3.1. Діаграма діяльності

На рис 3.1 — 3.3 представлені діаграми діяльності прецедентів системи.

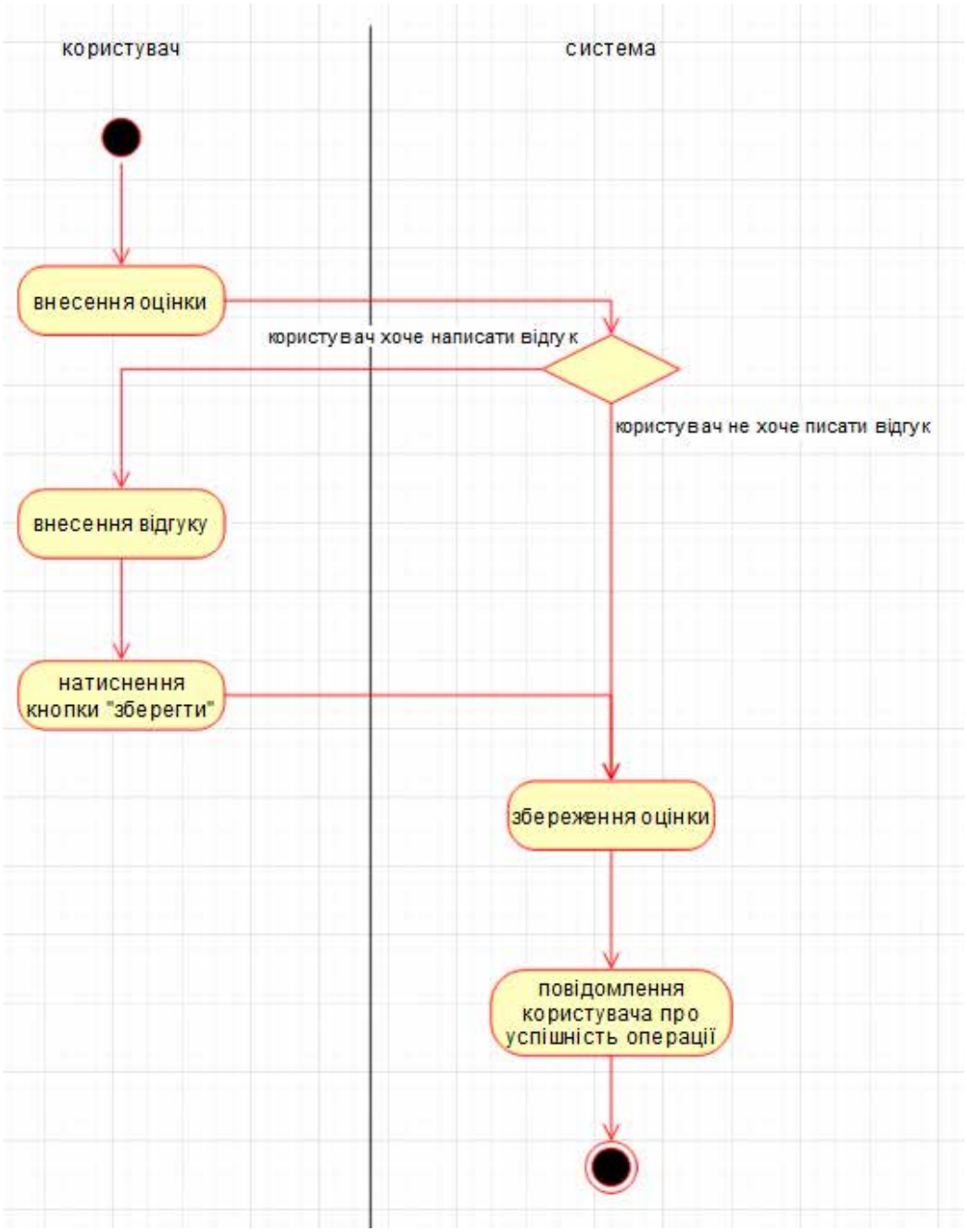


Рис. 3.1 Оцінка фільму

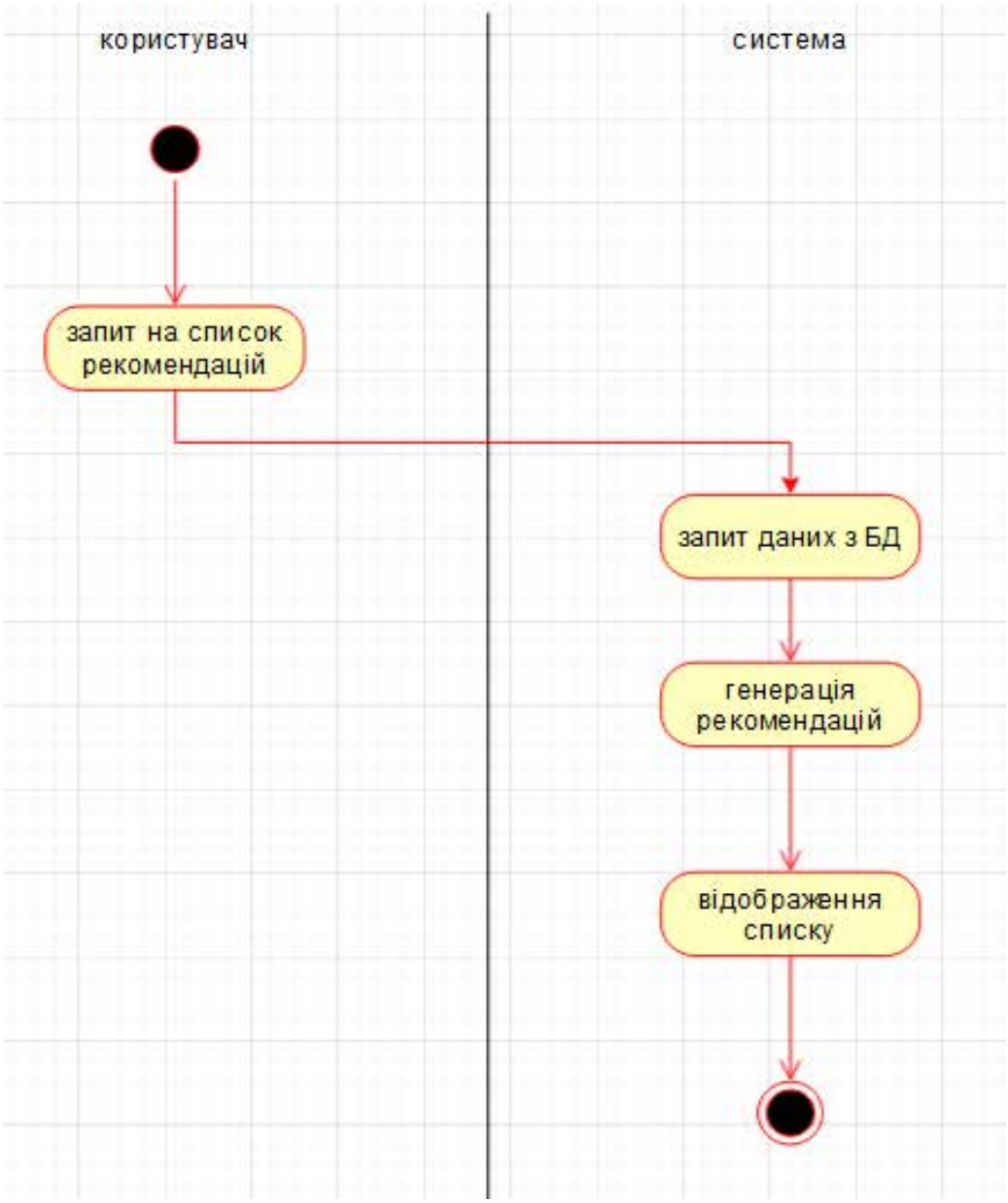


Рис. 3.2 Перегляд списку рекомендацій

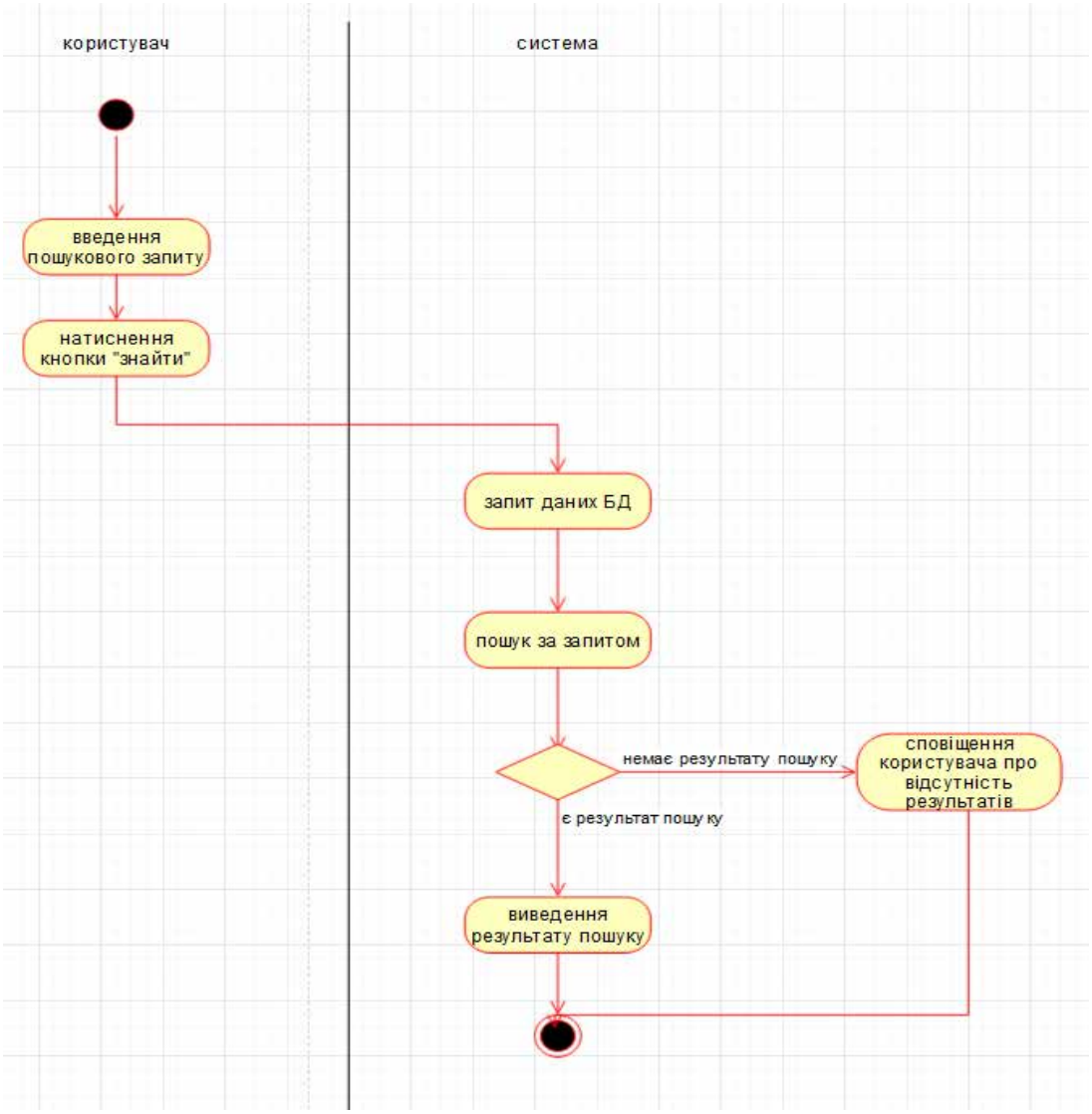


Рис. 3.3 Пошук фільмів

3.2. Діаграма послідовності

На рис 3.4 — 3.5 представлені діаграми послідовності прецедентів системи.

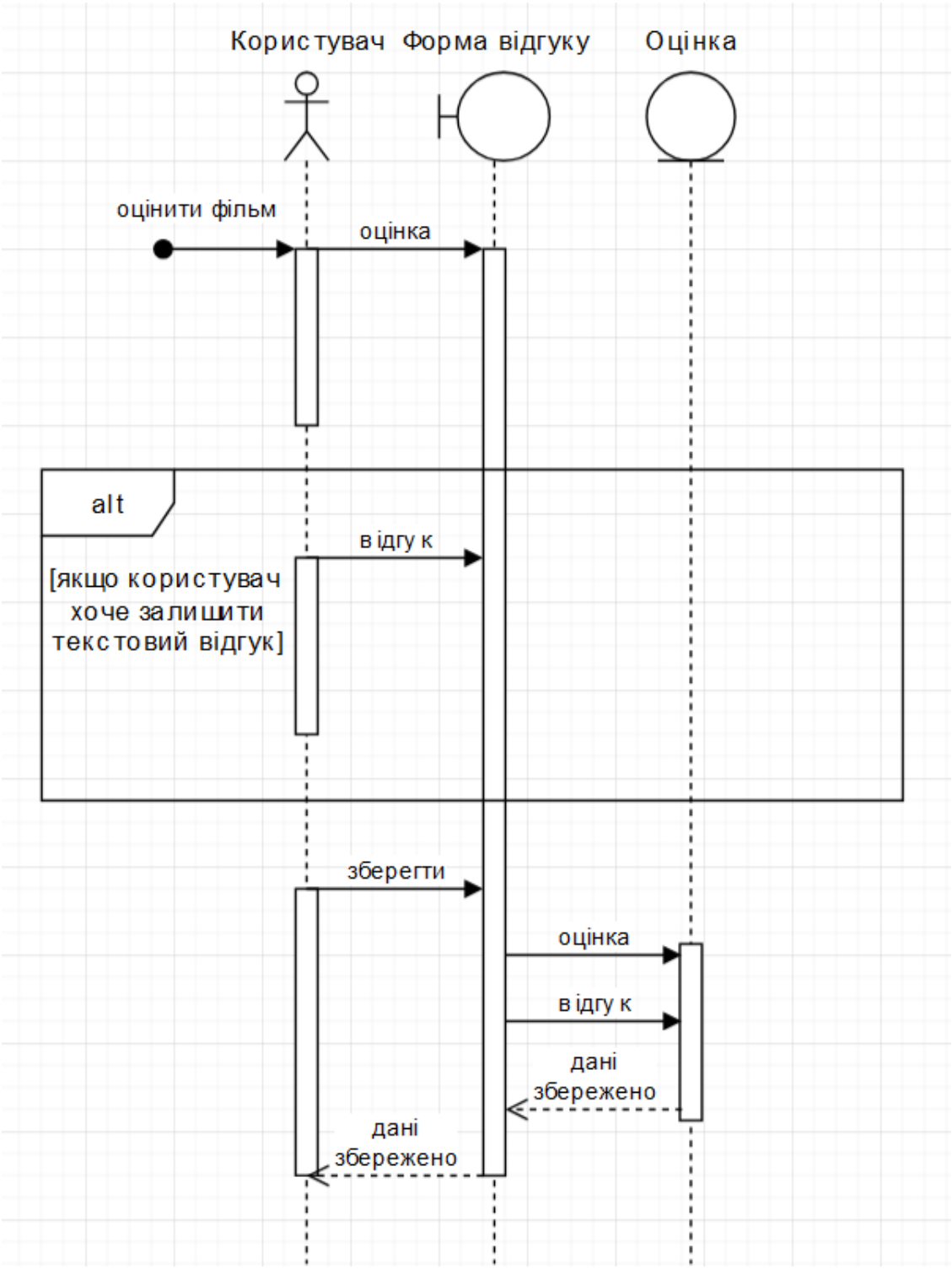


Рис. 3.4 Оцінка фільму

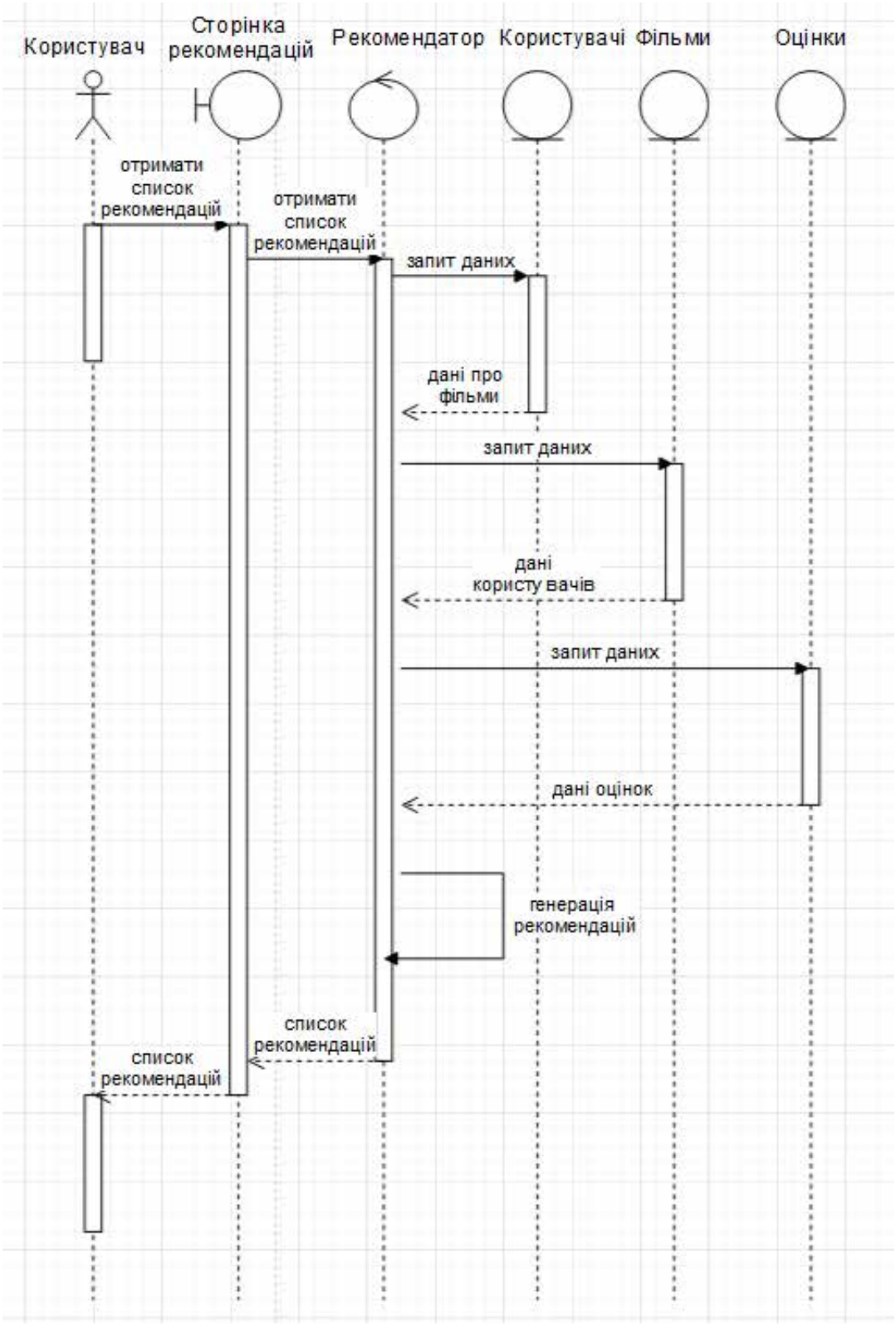


Рис. 3.5 Перегляд списку рекомендацій

3.3. Організаційна структура програмного забезпечення

За допомогою діаграми класів створюється внутрішня структура системи, описується спадкування та відношення класів між собою. Вона відображає

логічне представлення системи, оскільки класи є лише шаблонами, на основі яких потім створюються фізичні об'єкти. Діаграма класів розробленої інформаційної системи зображена на рис. 3.6.

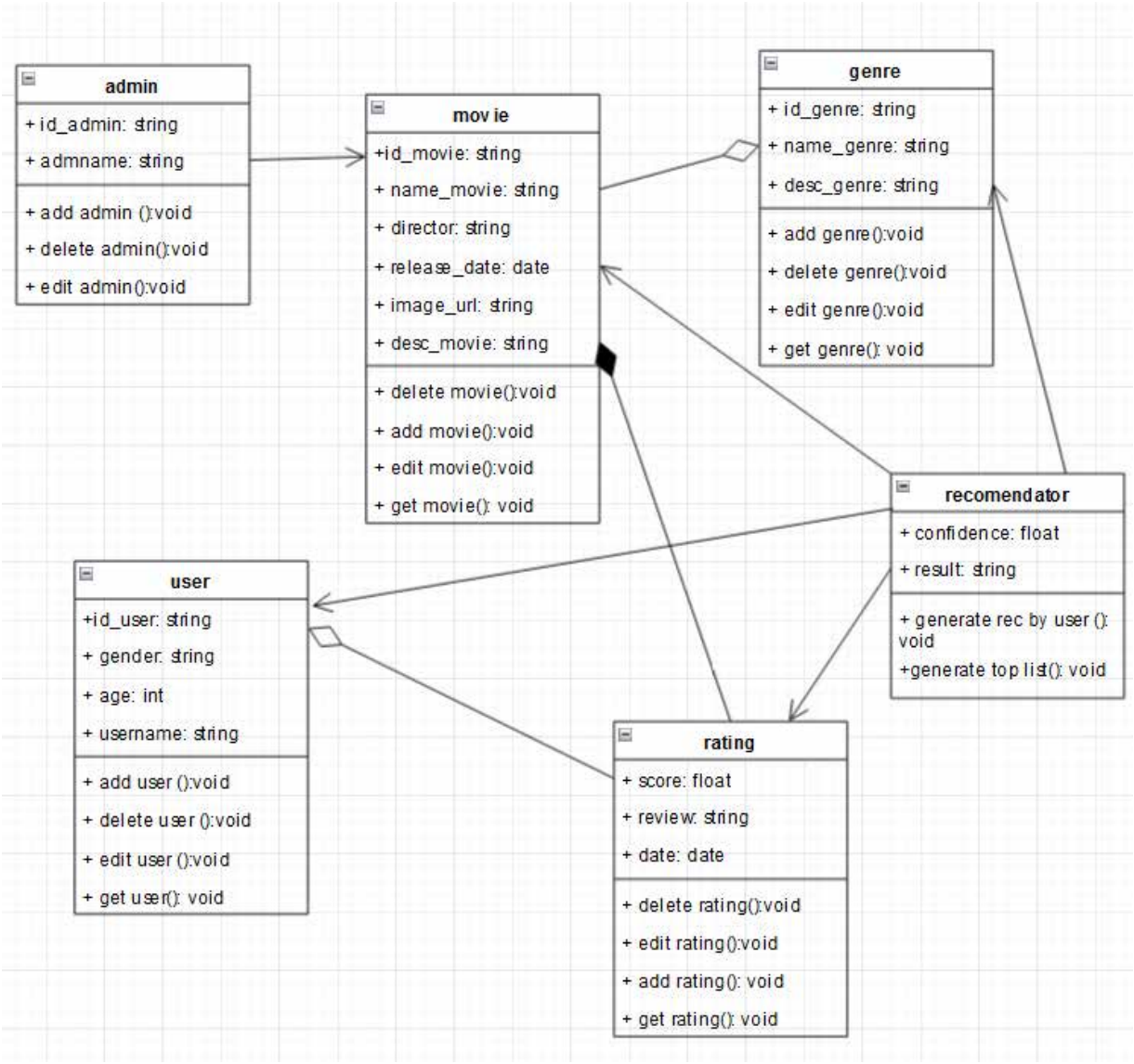


Рис. 3.6 Діаграма класів

Дана діаграма містить:

Admin – клас адміністратора системи, може додавати та змінювати фільми (зв’язок асоціації з сутністю «Movie»), жанри (зв’язок асоціації з сутністю «Genre»).

Movie – клас фільму, має жанр (зв’язок агрегації з сутністю «Genre»).

Genre – клас жанру, відноситься до фільму (зв’язок агрегації з сутністю «Movie»).

Rating – клас оцінки, створюється користувачем (зв’язок агрегації з сутністю «User»), є оцінкою фільму (зв’язок композиції з сутністю «Movie»).

Review – клас відгуку, створюється користувачем (зв’язок агрегації з сутністю «User»), є відгуком фільму (зв’язок композиції з сутністю «Movie»).

User – клас користувача, може залишати відгуки (зв’язок агрегації з сутністю «Review») та оцінки (зв’язок агрегації з сутністю «Rating») до фільмів.

3.4. Застосування OLAP та методів Data Mining для підвищення ефективності рекомендаційних систем

У роботі використано експериментальний підхід: на основі реального датасету фільмів з сайту MovieLens, який було занесено у сховище даних, було проведено аналіз даних із застосуванням методів Data Mining, зокрема кластеризації, методу OneR та асоціативного аналізу. Такий аналіз відкриває значно ширші можливості для виявлення прихованих залежностей та закономірностей у великих масивах інформації.

3.4.1. Створення інформаційної бази

У сучасних інформаційних системах головним викликом є не стільки зберігання великих масивів даних, скільки їх ефективна обробка та використання для прийняття рішень. Це особливо актуально для рекомендаційних алгоритмів, де якість персоналізації напряму залежить від глибини аналізу користувацьких даних. Традиційні підходи до побудови рекомендаційних систем часто обмежуються базовими методами фільтрації, тоді як використання технологій OLAP та Data Mining відкриває значно ширші можливості для виявлення прихованих залежностей та закономірностей у великих масивах інформації.

Структура БД і кубу, в які занесені дані датасету, зображена на рисунках 3.7 — 3.8.

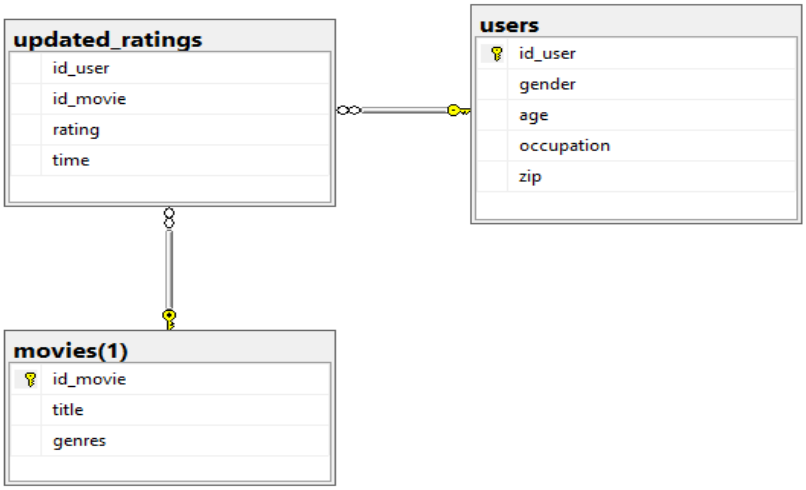


Рис. 3.7 Структура БД

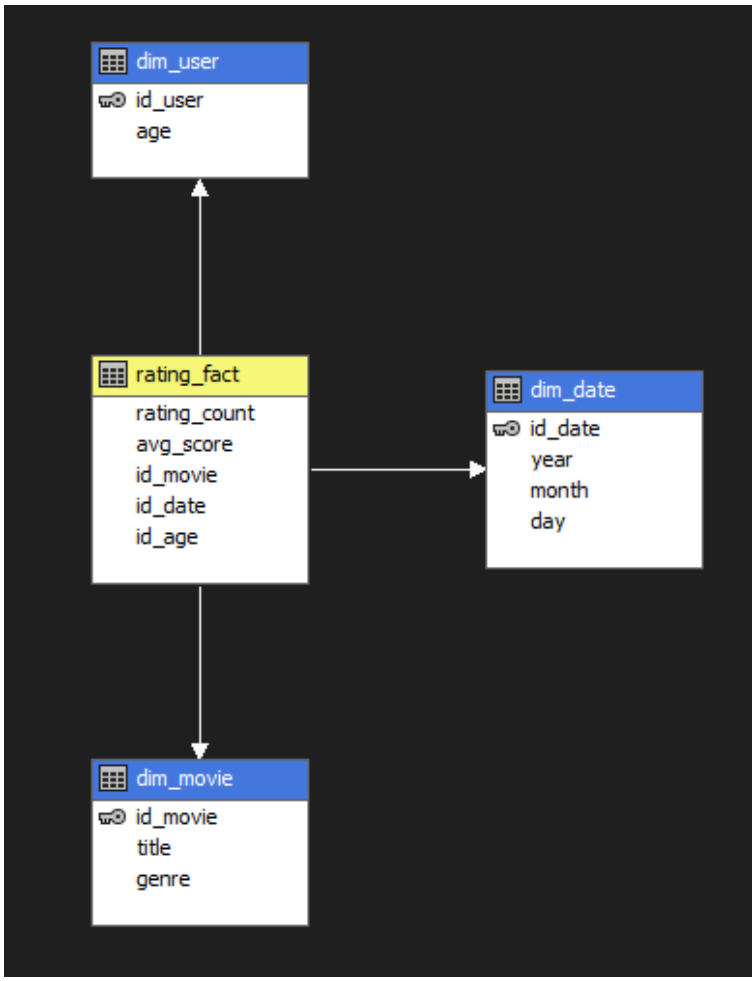


Рис. 3.8 Структура кубу

Фактова таблиця: score_fact: центральна таблиця зберігає агреговані метрики оцінок фільмів. Вона містить ключі, що пов'язують її з вимірювальними таблицями.

- rating_count: Кількість оцінок для фільму.
- avg_score: Середній рейтинг фільму.
- id_movie (FK): Ідентифікатор фільму (посилання на dim_movie).
- id_date (FK): Ідентифікатор дати (посилання на dim_date).
- id_age (FK): Ідентифікатор юзера (посилання на dim_user).

Вимірювальна таблиця: dim_movie (Фільми): містить інформацію про фільми, які аналізуються.

- id_movie (PK): Унікальний ідентифікатор фільму.
- title: Назва фільму.
- genre: Жанр фільму.

Вимірювальна таблиця: dim_date (Дати): зберігає інформацію про дати, що використовуються для аналітики (наприклад, дати оцінок).

- id_date (PK): Унікальний ідентифікатор дати.
- year: Рік.
- month: Місяць.
- day: День.

Вимірювальна таблиця: dim_user (Користувачі): містить інформацію про вік користувачів, які поставили оцінки.

- id_user (PK): Унікальний ідентифікатор вікової категорії користувачів.
- age: Вікова категорія

3.4.2. Інтелектуальний аналіз даних

Дослідимо застосування методів інтелектуального аналізу даних до датасету про фільми. Зокрема, розглянемо алгоритми типу One Rule, Buyers, а також методи кластеризації та пошуку асоціацій. Їх використання дозволяє

структурувати вхідні дані, виокремити групи користувачів зі схожими інтересами, виявити найбільш значущі атрибути контенту та сформувати закономірності спільного перегляду. Такий підхід сприяє підвищенню ефективності рекомендаційних алгоритмів і створює основу для побудови більш гнучких та персоналізованих систем.

Для виконання даного проєкту було створено сховище та базу даних. Джерелом даних є публічний датасет про рейтинги фільмів з сайту MovieLens.

Використаємо можливості Power BI для візуалізації даних, занесених у СД. Звіт який показує загальну кількість поставлених оцінок юзерами різних вікових категорій протягом 2023-2024 року представлений на рисунку 3.9.

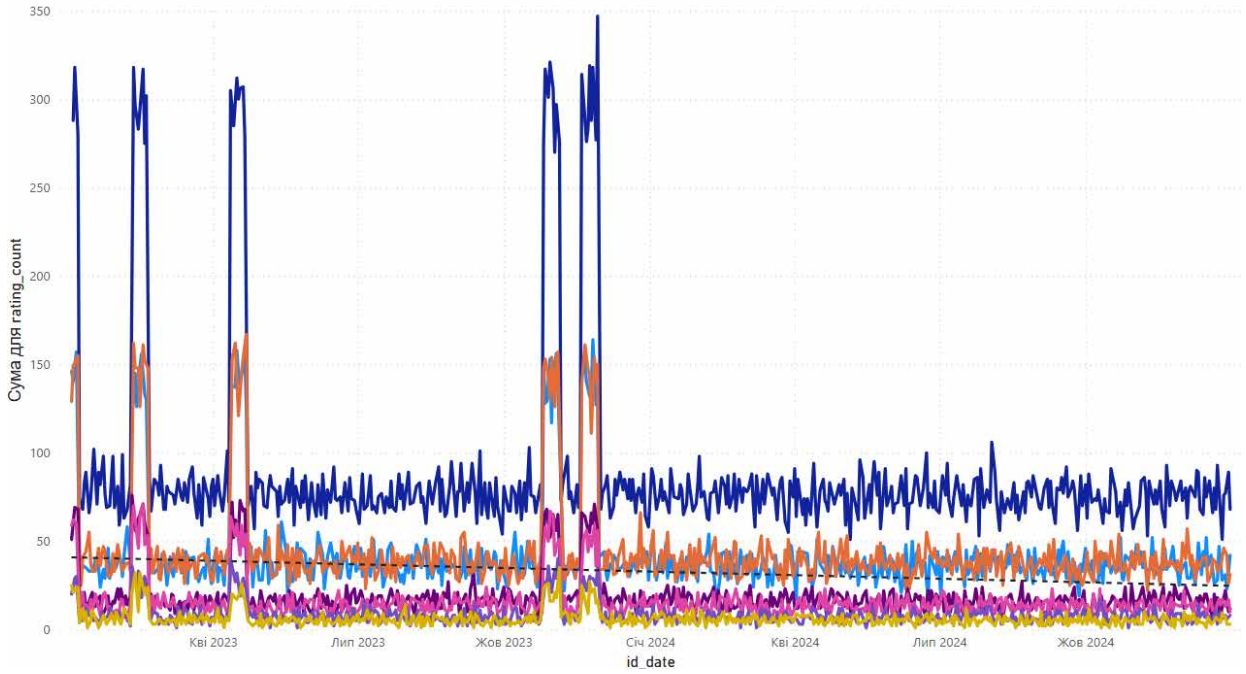


Рис. 3.9 Звіт загальної кількості поставлених оцінок юзерами різних вікових категорій протягом 2023-2024 років

3.4.3. OneR

OneR, скорочення від «One Rule», — це простий, але точний алгоритм класифікації, який генерує одне правило для кожного предиктора в даних, а потім вибирає правило з найменшою загальною помилкою як своє «єдине правило». Щоб створити правило для предиктора, ми створюємо частотну таблицю для кожного предиктора щодо цілі. Було показано, що OneR виробляє

правила лише трохи менш точні, ніж найсучасніші алгоритми класифікації, але виробляє правила, які легко інтерпретувати людям [10].

Для реалізації 1-Rule класифікації, було виділено такі класи та змінні:

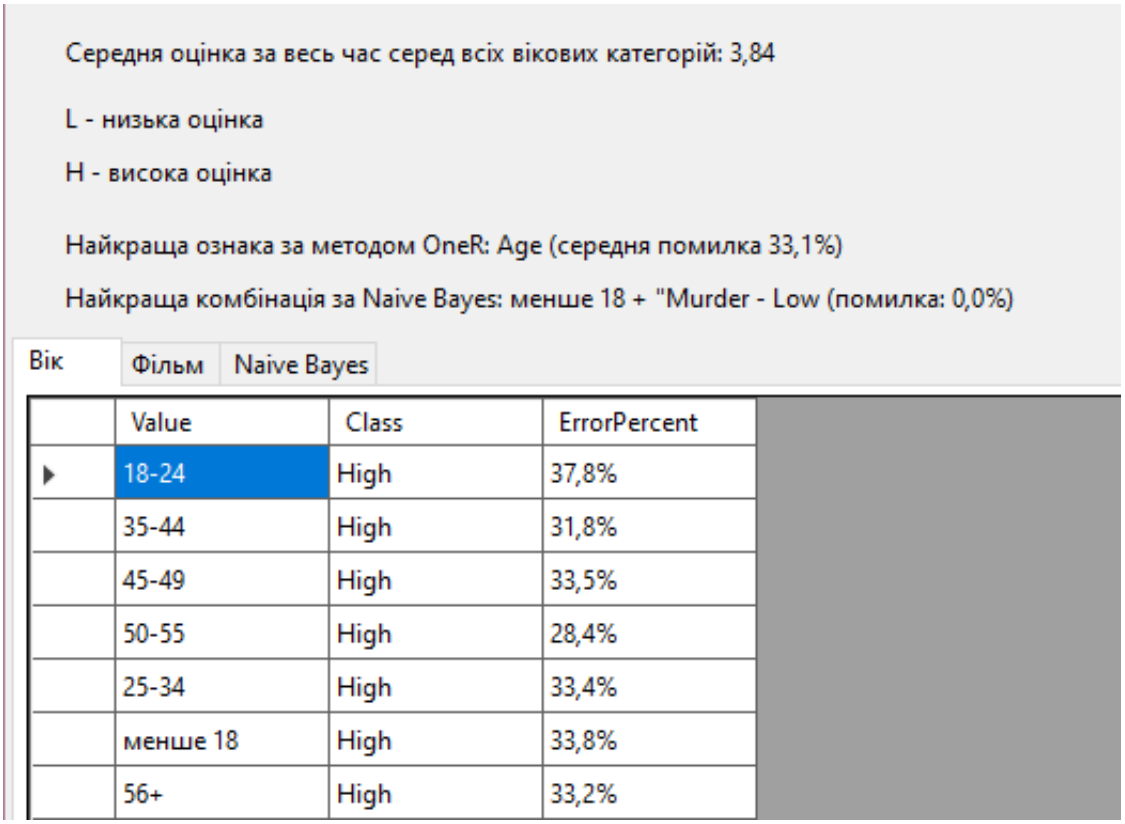
Класи**:**

- Low – низька оцінка, avg_score менша за середнє значення за весь період спостереження;
- High - висока оцінка, avg_score більша за середнє значення за весь період спостереження.

Залежні змінні:

- age користувача;
- title фільму.

Реалізацію алгоритму One Rule було виконано у середовищі Visual Studio за допомогою C#, візуалізацію було зроблено завдяки технології Windows Forms. Скріншоти результатів показані на рисунках 3.10-3.11.



	Value	Class	ErrorPercent
▶	18-24	High	37,8%
	35-44	High	31,8%
	45-49	High	33,5%
	50-55	High	28,4%
	25-34	High	33,4%
	менше 18	High	33,8%
	56+	High	33,2%

Рис. 3.10 Результати 1-Rule

Вік | Фільм | Naive Bayes

	Value	Class	ErrorPercent
▶	Toy Story (1995)	High	21,8%
	Extreme Measu...	Low	25,9%
	Old Yeller (1957)	High	35,8%
	Pollyanna (1960)	Low	44,9%
	"Shaggy Dog	Low	36,2%
	"20	High	40,9%
	"Three Caballeros	Low	37,7%
	Pete's Dragon (...	Low	33,7%
	Bedknobs and ...	High	49,3%
	"Sound of Music	High	32,8%
	Die Hard (1988)	High	20,9%
	Unhook the Sta...	Low	36,4%
	Happy Gilmore ...	High	43,5%
	"Secret Agent	Low	40,0%
	That Thing You ...	High	49,9%
	"Long Kiss Goo...	High	46,5%
	Shadow Conspi...	Low	22,2%

Рис. 3.11 Результати 1-Rule (продовження)

Відповідно до результатів розрахунків правилами з найбільша точність правила Вік:

Якщо вікова категорія користувача 50-55 то висока оцінка (High) з ймовірністю 71.6%.

Найбільша точність правила Назва фільму:

Якщо “Toy Story” то висока оцінка (High) з ймовірністю 78.2%;

Якщо “Die Hard” то висока оцінка (High) з ймовірністю 79.1%;

Якщо “Shadow Conspiracy” то низька оцінка (Low) з ймовірністю 77.8%;

Якщо “Shall We Dance?” то висока оцінка (High) з ймовірністю 84.7%;

Якщо “Sleeper” то висока оцінка (High) з ймовірністю 75%.

3.4.4. Метод наївного Байеса

Наївні класифікатори Байєса – це керовані алгоритми машинного навчання, які використовуються для завдань класифікації на основі теореми Байєса для визначення ймовірностей.

Для реалізації цієї класифікації було вибрано такі ж класи і змінні, як при реалізації 1-Rule, та такі ж інструменти. Блок-схема алгоритму реалізації методу наївного Байеса зображена на рисунку 3.12 [11].

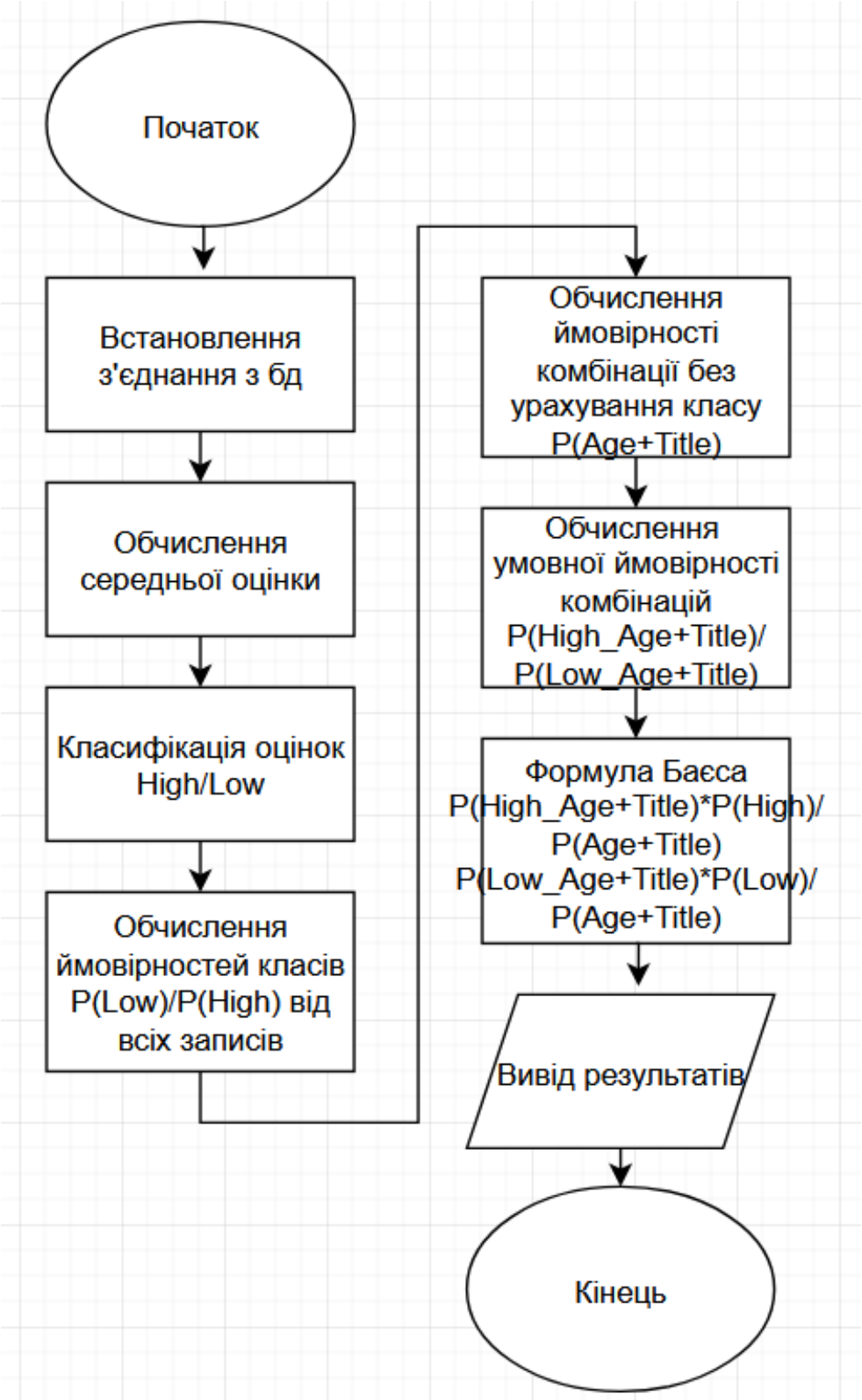


Рис. 3.12 Блок-схема реалізації методу наївного Байеса

Результати проведення класифікації цим методом показано на рисунках 3.13-3.14.

Naive Bayes

	Age	Title	PredictedClass	HighProbability	LowProbability
	50-55	"20	High	63,1%	36,9%
	18-24	"20	High	56,3%	43,7%
	35-44	"20	High	60,8%	39,2%
	56+	"20	High	66,0%	34,0%
	25-34	"20	High	57,5%	42,5%
	45-49	"20	High	55,6%	44,4%
	менше 18	"20	Low	45,5%	54,5%

Рис. 3.13 Результати наївного Байеса

м Naive Bayes

	Age	Title	PredictedClass	HighProbability	LowProbability
	25-34	Bedknobs and ...	High	60,2%	39,8%
	35-44	Bedknobs and ...	Low	41,1%	58,9%
	18-24	Bedknobs and ...	Low	47,8%	52,2%
	менше 18	Bedknobs and ...	Low	45,5%	54,5%
	56+	Bedknobs and ...	Low	37,5%	62,5%
	50-55	Bedknobs and ...	High	55,6%	44,4%
	45-49	Bedknobs and ...	Low	47,1%	52,9%

Рис. 3.14 Результати наївного Байеса (продовження)

По результатам аналізу найймовірнішими правилами є:

Якщо вік 56+ і місяць “20”, то висока оцінка (High) з ймовірністю 66%;

Якщо вік 25-34 і “Bedknobs and broomsticks”, то висока оцінка (High) з ймовірністю 60.2%.

Для порівняння, реалізацію цього методу також було виконано за допомогою мови Python з використанням вбудованих функцій бібліотеки sklearn. Результати зображено на рисунку 3.15. Порівнявши бачимо, що обидва шляхи дали майже ідентичні результати.

title	age	avg_score	class
20	18-24	3.5422535211267605	Low
20	25-34	3.701492537313433	High
20	35-44	3.6891666666666665	High
20	45-49	3.738095238095238	High
20	50-55	3.8461538461538463	High
20	56+	3.84	High
20	менше 18	3.272727272727273	Low
bedknobs and broomsticks (1971)	18-24	3.1985074626865675	Low
bedknobs and broomsticks (1971)	25-34	3.6610619469026546	High
bedknobs and broomsticks (1971)	35-44	3.3767123287671232	Low
bedknobs and broomsticks (1971)	45-49	3.5588235294117645	Low
bedknobs and broomsticks (1971)	50-55	3.5555555555555554	Low
bedknobs and broomsticks (1971)	56+	3.5	Low
bedknobs and broomsticks (1971)	менше 18	3.5454545454545454	Low

Рис. 3.15 Результати наївного Байеса в Python

3.4.5. Пошук асоціативних правил

Інтелектуальний аналіз правил асоціації знаходить цікаві асоціації та зв’язки між великими наборами елементів даних. Це правило показує, як часто набір елементів зустрічається в транзакції. Типовим прикладом є аналіз ринку. Аналіз ринку є одним із ключових методів, який використовується великими партнерами, щоб показати асоціації між товарами. Він дозволяє роздрібним торговцям визначати зв’язки між товарами, які люди часто купують разом. Маючи набір транзакцій, ми можемо знайти правила, які передбачуватимуть появу елемента на основі появи інших елементів у транзакції [12].

Реалізацію пошуку асоціативних правил виконаємо за допомогою мови Python. У Python є кілька бібліотек, які реалізують алгоритми для пошуку асоціативних правил, зокрема, mlxtend - найпопулярніша бібліотека для частих наборів та асоціативних правил, що включає реалізацію алгоритмів Apriori і FP-growth).

Алгоритм Apriori є одним з найпоширеніших методів пошуку асоціативних правил і працює на основі ідеї "підтримки" для пошуку частих наборів елементів. Основні кроки алгоритму Apriori:

Крок 1: Генерація кандидатних елементів (set) з одиничних елементів (одиноких товарів або фільмів).

Крок 2: Обчислення підтримки для кожного кандидата.

Крок 3: Продовження генерації більш складних елементів, якщо їх підтримка перевищує мінімальний поріг.

Крок 4: Створення правил на основі знайдених частих наборів [13].

Блок схему реалізації пошуку асоціативних правил зображено на рисунку 3.16.

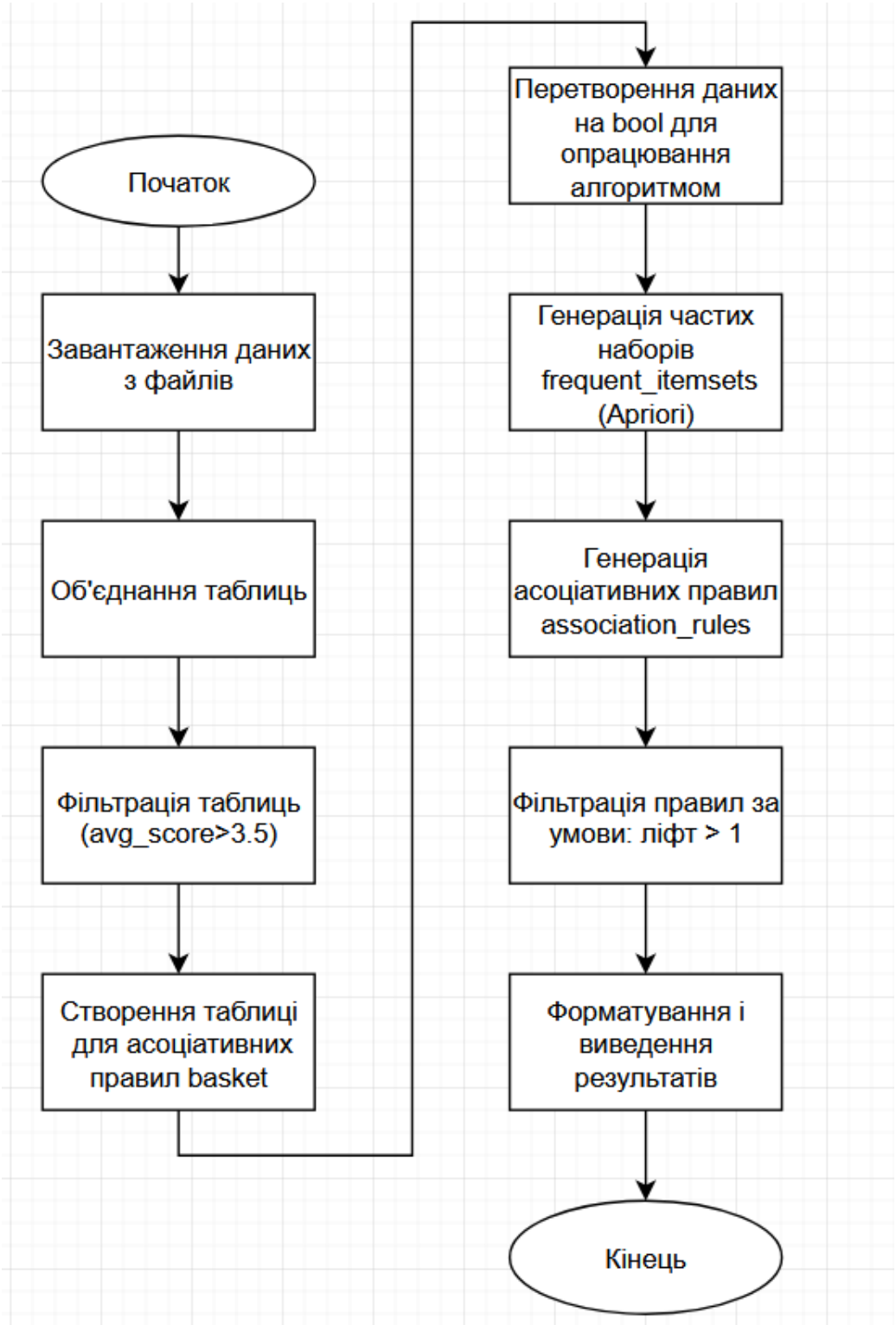


Рис. 3.16 Пошук асоціативних правил

Результати пошуку асоціативних правил представлено на рис. 3.17.

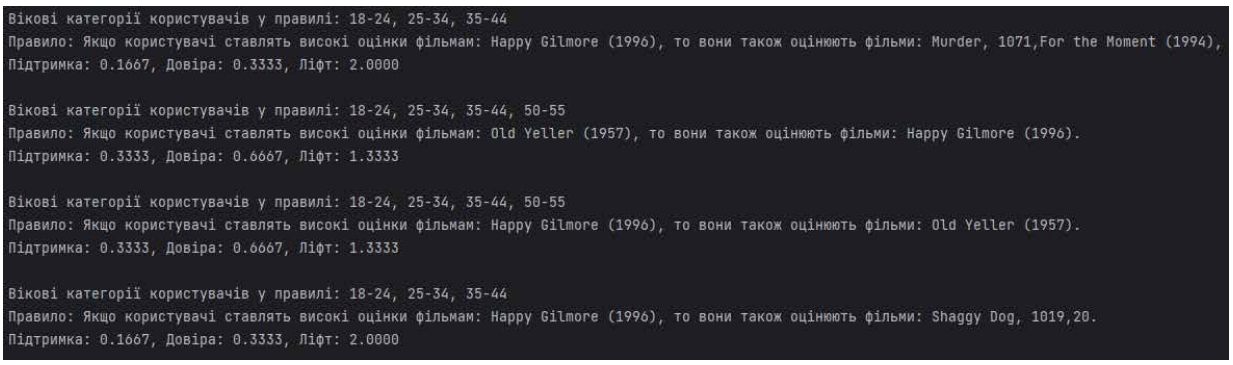


Рис. 3.17 Пошук асоціативних правил

Проаналізуємо правило:

Вікові категорії користувачів у правилі: 18-24, 25-34, 35-44, 50-55

Правило: Якщо користувачі ставлять високі оцінки фільмам: Old Yeller (1957), то вони також оцінюють фільми: Happy Gilmore (1996).

Підтримка: 0.3333, Довіра: 0.6667, Ліфт: 1.3333

Вікові категорії користувачів: це правило має сенс для користувачів з віковими категоріями 18-24, 25-34, 35-44, 50-55. Це може свідчити про те, що ці фільми популярні серед цих вікових груп, або вони мають подібні характеристики, які приваблюють цих користувачів.

Підтримка (Support): 0.3333. Підтримка показує, яку частку всіх транзакцій складають користувачі, які одночасно оцінили і фільм Old Yeller (1957), і Happy Gilmore (1996). Значення 0.3333 означає, що 33.33% всіх записів (користувачів) містять обидва ці фільми. Це досить велика підтримка і свідчить про значний зв'язок між оцінками цих двох фільмів серед користувачів.

Довіра (Confidence): 0.6667. Довіра вказує на ймовірність того, що якщо користувач оцінив фільм Old Yeller (1957), то він також оцінить Happy Gilmore (1996). Значення 0.6667 означає, що 66.67% користувачів, які оцінили Old Yeller (1957), також оцінили Happy Gilmore (1996). Це високий показник довіри, що свідчить про сильний зв'язок між цими фільмами серед користувачів.

Ліфт (Lift): 1.3333. Ліфт порівнює фактичну ймовірність появи двох подій (фільмів) з теоретичною ймовірністю їх спільного появи, якщо вони були б незалежними. Ліфт більше ніж 1 означає, що існує позитивний зв'язок між двома подіями. Значення 1.3333 вказує на те, що ймовірність того, що користувач

оцінить Happy Gilmore (1996) після того, як оцінив Old Yeller (1957), на 33.33% більша, ніж якщо ці події були б незалежними.

3.4.6. Кластерний аналіз

Кластерний аналіз, також відомий як кластеризація, — це техніка інтелектуального аналізу даних, яка групує схожі точки даних у кластери. Мета полягає в тому, щоб точки даних у кластері були більш схожі одна на одну, ніж на точки в інших кластерах. Ця техніка широко використовується в дослідницькому аналізі даних для виявлення прихованих закономірностей.

Реалізацію кластеризації виконаємо за допомогою мови Python, використовуючи функції бібліотеки sklearn для виконання кластеризації методом K-середніх та методом Ліктя [14]. На рисунку 3.18 зображений результат реалізації методу Ліктя для визначення оптимальної кількості кластерів.

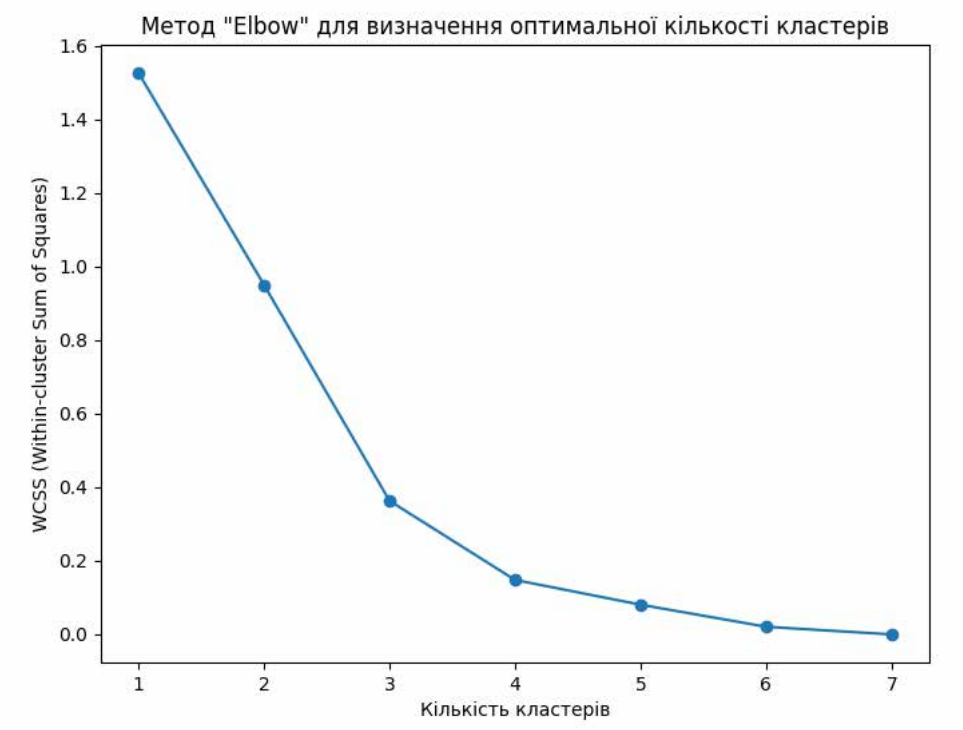


Рис. 3.18 Метод Ліктя

Як бачимо, оптимальною кількістю кластерів буде 3. На рисунку 3.19 представлена діаграма розкиду кластерів.

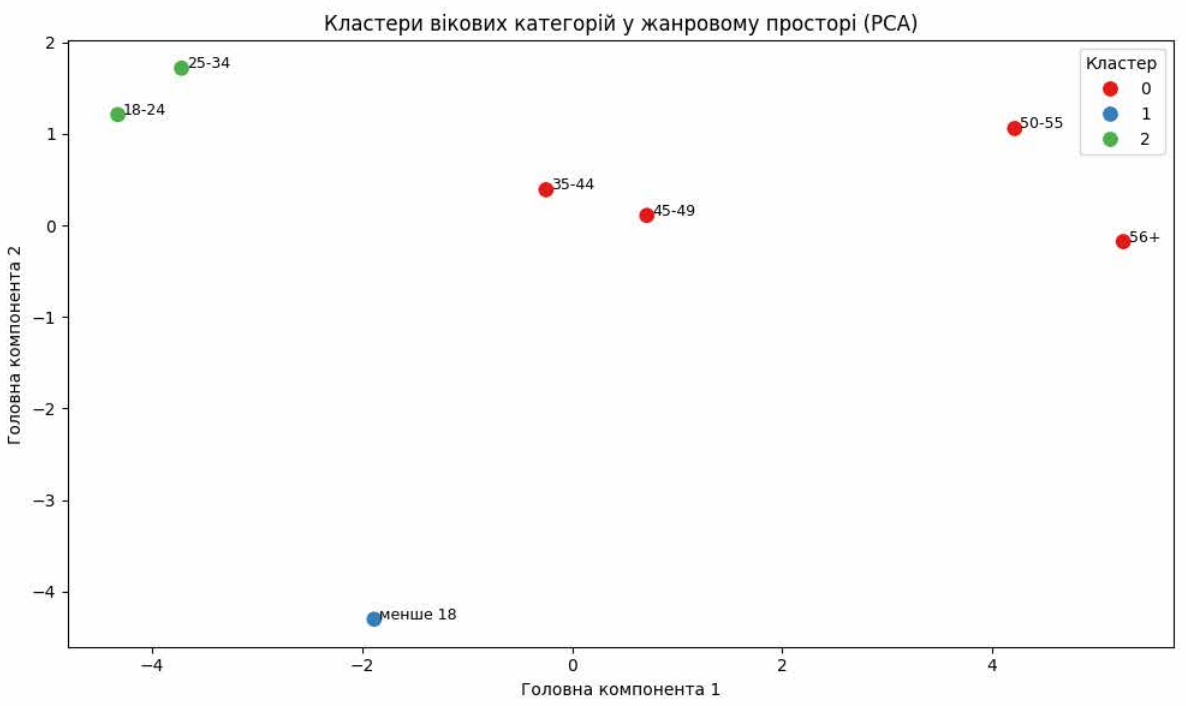


Рис. 3.19 Результати кластеризації

Кластери представляють групи вікових категорій з схожими вподобаннями (відповідно до поставлених оцінок) в жанрах фільмів. На рисунку 3.20 зображена діаграма, що показує відмінності середніх оцінок по окремих жанрам серед кластерів.

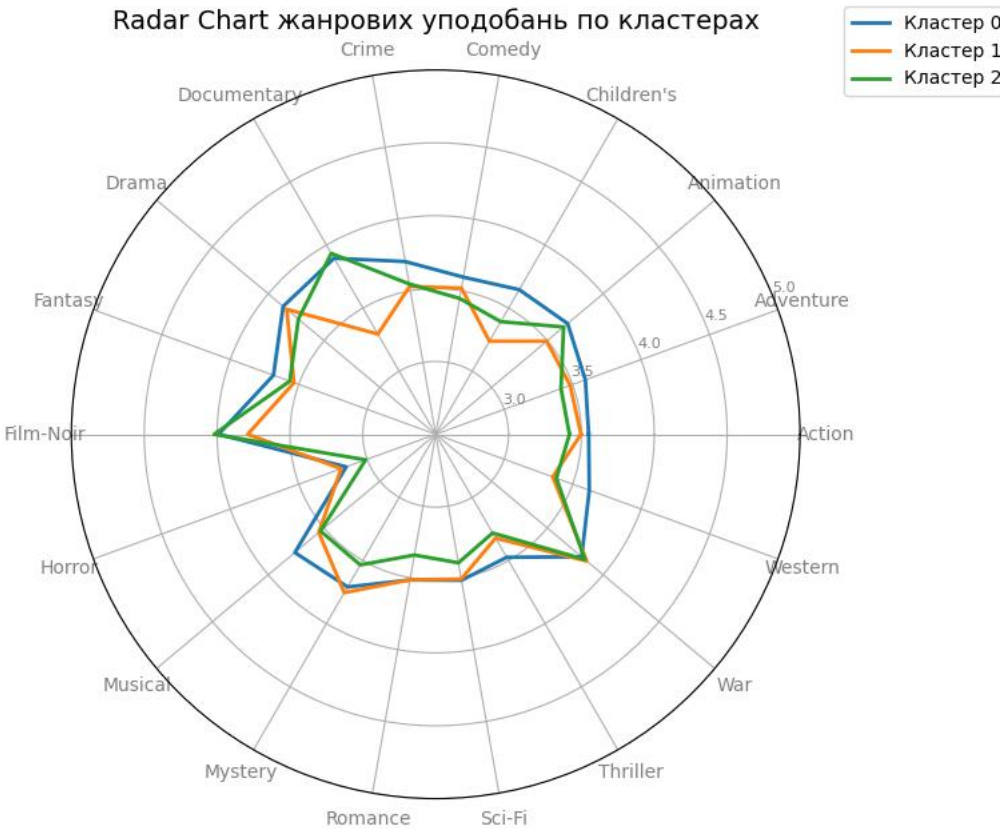


Рис. 3.20 Відмінності кластерів

Проаналізувавши, можемо скласти такий опис знайдених кластерів:

Кластер 0 (Старше покоління): Високі оцінки на Пригоди, Мюзикл, Комедія, Драма, Вестерн, Війна, Нуар.

Кластер 1 (Діти): Комедія, Містика, Наукова фантастика, Анімація, Екшн, Хоррор. Менше подобається: Вестерн, Романтика, Нуар, Документальні фільми.

Кластер 2 (Дорослі люди): Документальні фільми, Нуар, Війна. Менше подобається: Хоррор, Містика, Наукова фантастика, Вестерн, Дитячі фільми, Мюзикл.

3.4.7. KPI

Ключові показники ефективності (KPI) - це управлінський інструмент або засіб, за допомогою якого можна відстежувати діяльність або процес, контролювати його (якщо він відхиляється, то можна розпізнати причину і виправити її) і забезпечити досягнення бажаних результатів.

Одним із способів досягнення хороших показників в оцінці роботи

співробітників є використання методу ключових показників ефективності. KPI порівнюють те, що було створено, з тим, що було визначено. Успішне впровадження буде залежати від реалізації хорошої стратегії обслуговування відповідно до того, що було визначено [15].

Створені КПІ зображені на рис. 3.21.

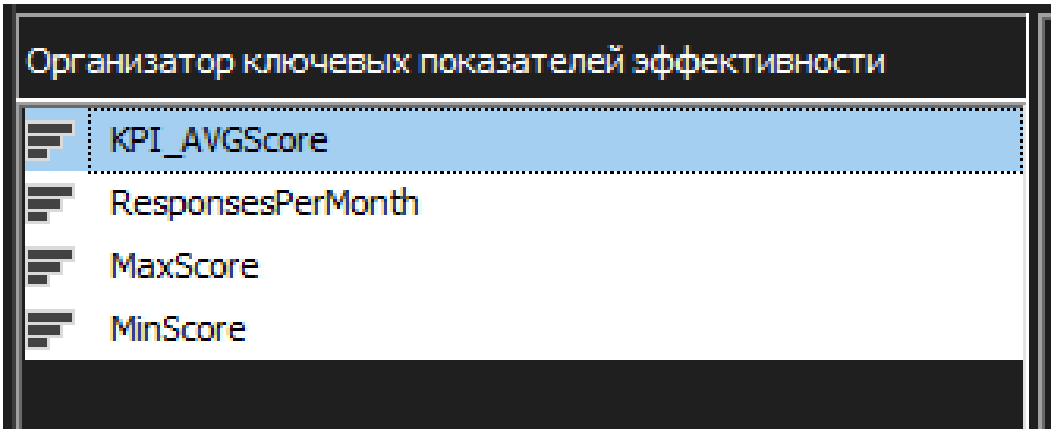


Рис. 3.21 Створені КПІ

KPI_AVGScore – середня оцінка за весь час, за ціль береться максимальна можлива – 10.

ResponsesPerMonth - Кількість оцінок за місяць. Порівнюється з результатами за попередній місяць.

MaxScore - максимальна оцінка за певний проміжок часу (місяць), за ціль береться максимальна можлива – 10.

MinScore - мінімальна оцінка за певний проміжок часу (місяць), за ціль береться 6.

Огляд ResponsesPerMonth:

Вираз значення - Сумує кількість оцінок за місяць

SUM(([Dim Date].[Id Date].&[6] : [Dim Date].[Id Date].&[10] , [Measures].[Rating Count]))

Вираз цілі – те ж саме, що й значення, але вже за минулий місяць:

SUM(([Dim Date].[Id Date].&[1] : [Dim Date].[Id Date].&[5] , [Measures].[Rating Count]))

Вираз стану:

CASE

```
   WHEN KPIValue("ResponsesPerMonth") < KPIGoal("ResponsesPerMonth")
THEN -1
   WHEN KPIValue("ResponsesPerMonth") > KPIGoal("ResponsesPerMonth")
THEN 1
   ELSE 0
END
```

Якщо результат більше 1, тобто кількість оцінок більша, ніж за попередній, - стрілочка тяжіє до зеленого. Якщо навпаки – до червоного.

Результат зображений на рис. 3.22:

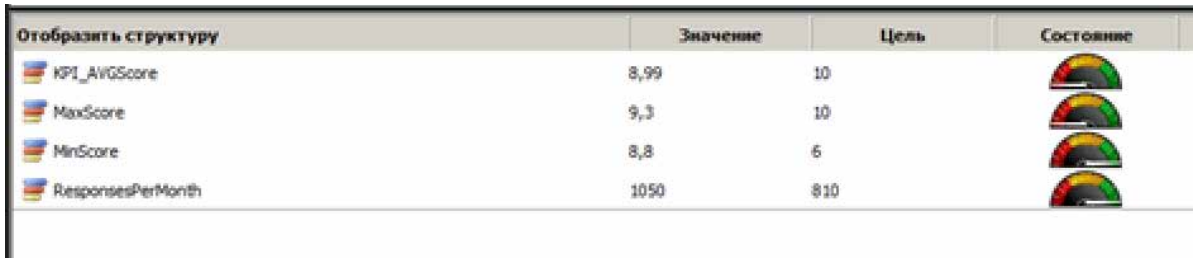


Рис. 3.22 Результат КПІ

Як показує результат, КПІ середньої оцінки 8.9 – нижче умовного значення 10.

КПІ максимальної оцінки 9.3 – нижче значення цілі 10.

КПІ мінімального значення 8.8 – вище цілі 6.

КПІ кількості оцінок за місяць 1050 – вище цілі 810.

3.4.8. Побудова звітів та прогнозування

Середовище Power BI надає можливість створення звітів даних з прогнозуванням за вказаними критеріями. Використаємо це для аналізу активності користувачів.

Як можна побачити по звіту на рис. 3.23, кількість оцінок, тобто активних користувачів має «сплески» у кількості. Дослідивши період їх виникнення, можна побачити, що вони випадають на святкові періоди/канікули. На прогнозі можна побачити передбачення таких сплесків після зимових свят та у весною, у період Великодня.

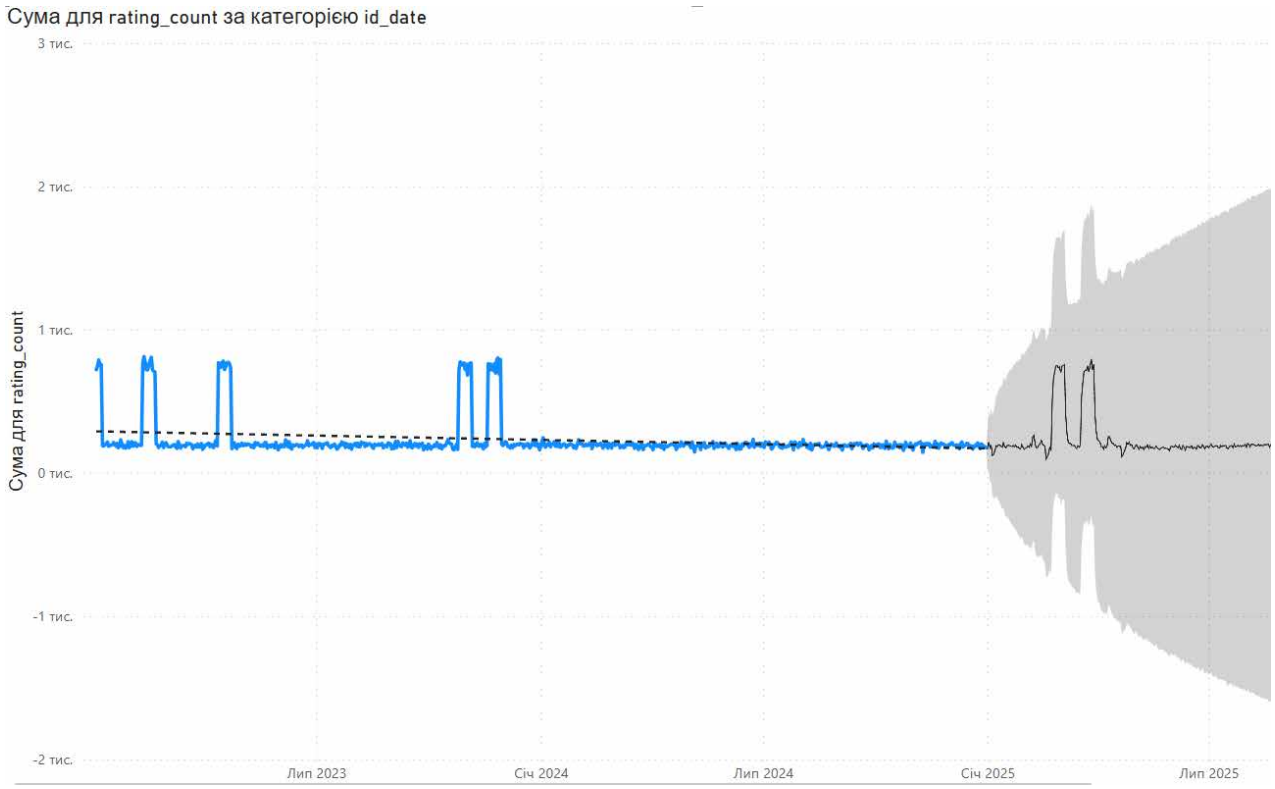


Рис. 3.23 Звіт з прогнозом в Power BI

Таку ж функцію має і Excel. Звіт за такими ж даними показано на рис.3.24. Як бачимо, прогноз Excel вийшов дещо іншим, проте довірчі границі такі самі, як і в Power BI.

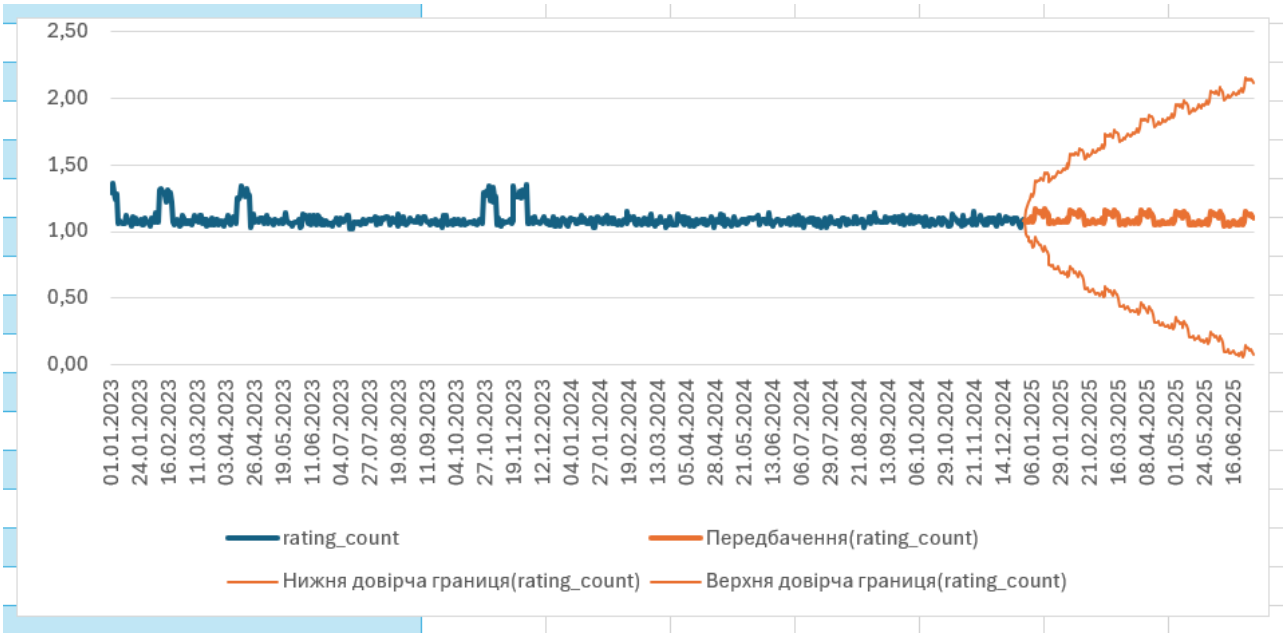


Рис. 3.24 Звіт з прогнозом в Excel

4 РОЗРОБКА СИСТЕМИ

4.1. Вибір інструментарію для розробки системи

Система розроблена із застосуванням наступного стеку веб-технологій:

- HTML використовується для структурування веб-сторінок та розміщення основних елементів інтерфейсу [16].
- CSS забезпечує стилізоване та привабливе оформлення, що робить інтерфейс зрозумілим та зручним для користувачів [17].
- JavaScript застосовується для реалізації інтерактивних функцій, таких як динамічне відображення рекомендацій, обробка подій користувача та взаємодія з сервером без перезавантаження сторінки [18].
- Для спрощення розробки адаптивного дизайну та забезпечення коректного відображення веб-додатку на різних пристроях використано Bootstrap, що надає готові компоненти та сіткову систему [19].
- Серверна частина системи реалізована на Django, який забезпечує обробку запитів користувачів, управління базою даних тощо [20].
- Базою даних для веб-системи рекомендацій фільмів використано SQLite, що забезпечує зручне зберігання та швидкий доступ до інформації про користувачів, фільми та їхні оцінки, а також ефективну роботу серверної частини на Django [21].

4.2. Демонстрація інтерфейсу системи

Каталог фільмів на головній сторінці зображений на рис. 4.1.Тут користувач може ознайомитись з всіма доступними фільмами, або виконати пошук за запитом.

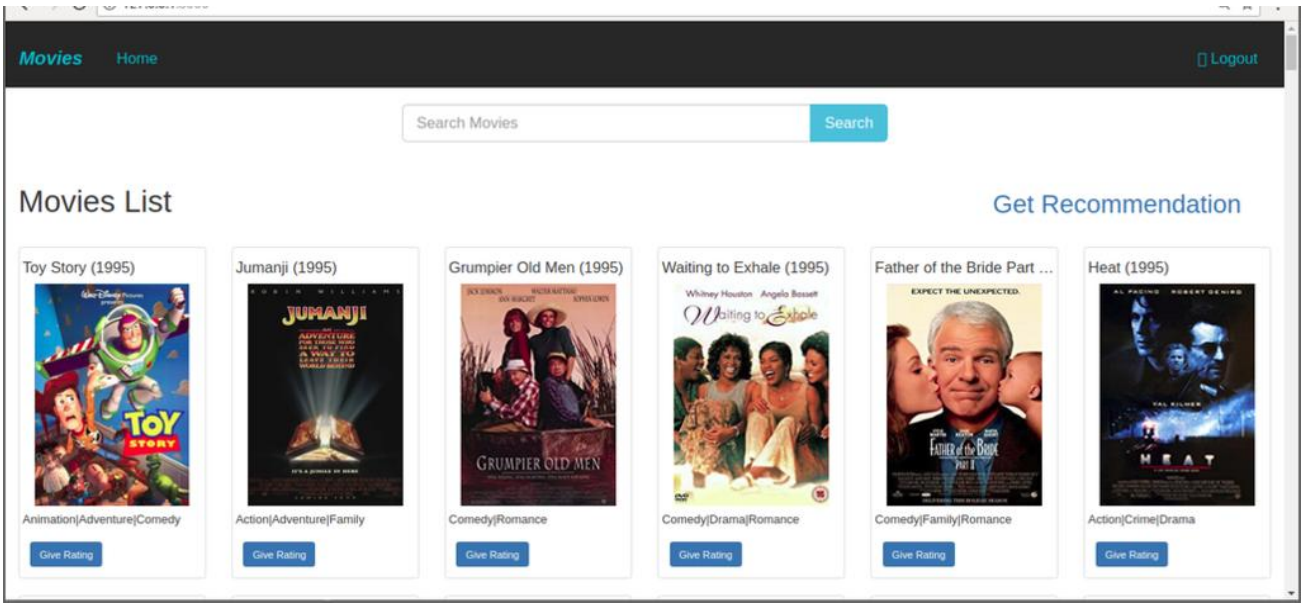


Рис. 4.1 Каталог фільмів

На рис. 4.2 зображена секція персональних рекомендацій користувача.Користувач отримує фільми, відібрані для нього, опираючись на його активність. Доступно лише зареєстрованим та авторизованим користувачам.

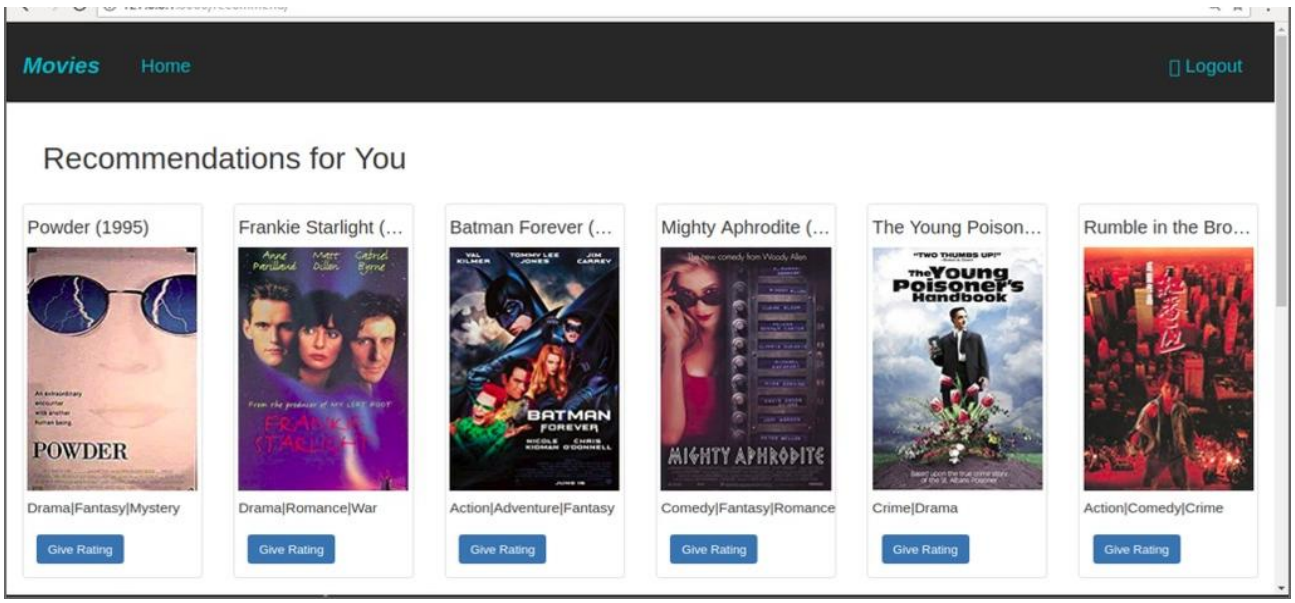


Рис. 4.2 Персональні рекомендації фільмів

На рис. 4.3 зображено вікно, в якому користувач може залишити відгук фільму.

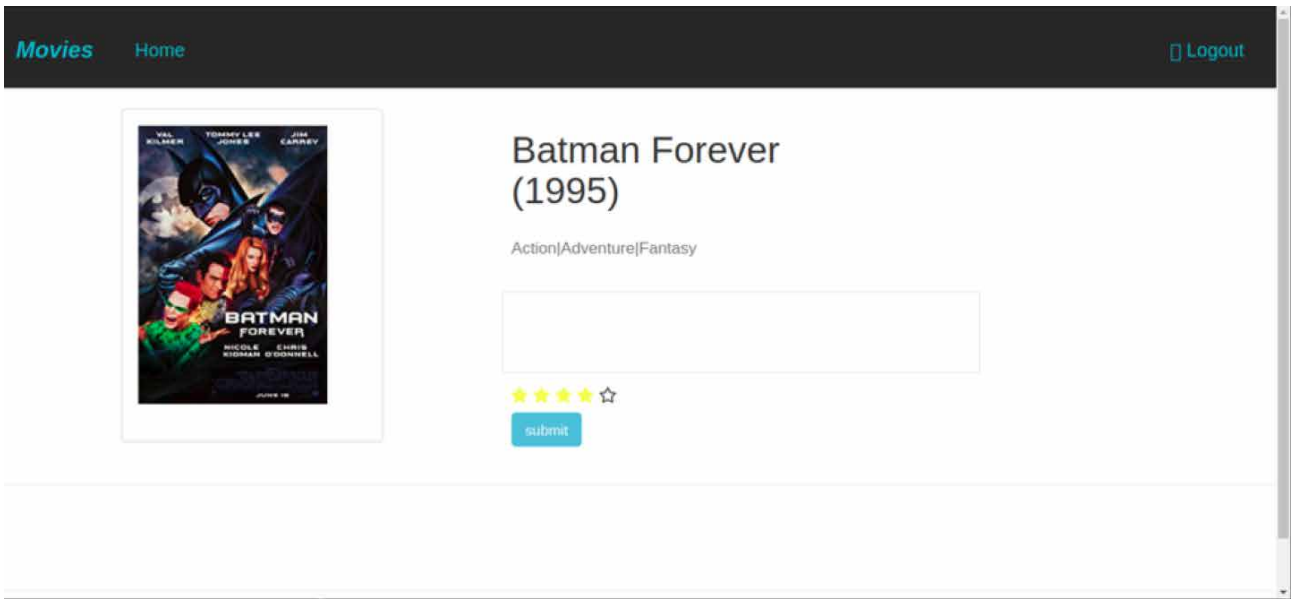


Рис. 4.3 Вікно відгуку фільму

4.3. Архітектура системи

Для кращого розуміння процесу впровадження розробленої інформаційної системи створюється діаграма розгортання. Діаграма розгортання в UML – це візуальне представлення конфігурації вузлів обробки та компонентів, які на них виконуються. Вона належить до категорії структурних діаграм і використовується для моделювання фізичних аспектів об'єктно-орієнтованої системи. Основне призначення діаграм розгортання – моделювання статичного представлення розгортання системи, тобто топології апаратного забезпечення. Це робить їх цінним інструментом для планування та документування інфраструктури ІС. Діаграми розгортання показують структуру системи часу виконання, відображають апаратне забезпечення, яке буде використовуватися для реалізації системи, та зв'язки між різними елементами апаратного забезпечення, моделюють фізичні апаратні елементи та шляхи зв'язку між ними, а також корисні для документування розгортання програмних компонентів або вузлів [22]. На рис. 4.4 представлена діаграма розгортання розробленої інформаційної системи.

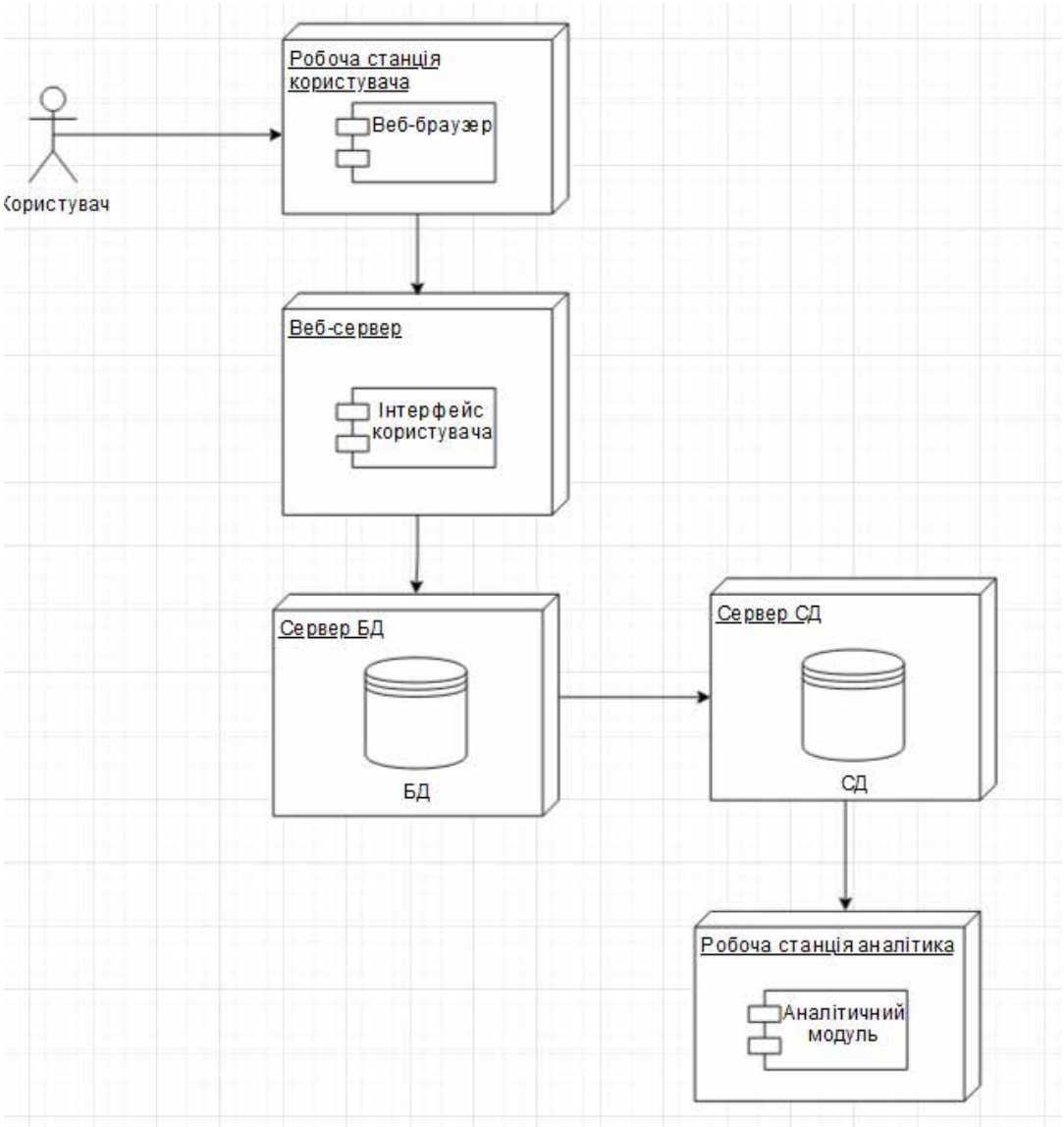


Рис. 4.4 Діаграма розгортання

ВИСНОВКИ

В рамках даного проекту було виконано огляд існуючих методів та алгоритмів роботи рекомендаційних систем, а саме - колаборативної фільтрації, фільтрації за вмістом та гібридної фільтрації. Розглянуто особливості їх застосування та описані основні принципи роботи даних алгоритмів.

Здійснено порівняльний аналіз різних методів та алгоритмів роботи рекомендаційних систем та основні проблеми, які можуть виникнути у результаті застосування певного алгоритму.

Здійснено програмну реалізацію алгоритмів, проведено порівняння показників косинусної подібності, в результаті якого доведено ефективність застосування гібридного методу.

Проведено аналіз даних, використовуючи методи OLAP та Data Mining. Такий аналіз відкриває значно ширші можливості для виявлення прихованих залежностей та закономірностей у великих масивах інформації. Таку інформацію можна використати для формування рекомендацій окремих груп користувачів аби, наприклад, вирішити проблему «холодного старту» нових користувачів у системі.

Розроблено прототип рекомендаційної системи фільмів, на основі гібридного підходу. Змодельована і описана архітектура програмного та технічного забезпечення системи, засоби які використовувались під час розробки, продемонстровано інтерфейс користувача розробленої системи.

У висновку, розроблювана рекомендаційна система покликана покращити користувацький досвід отримання персоналізованої підбірки контенту, за допомогою гібридного підходу генерації рекомендацій. Реалізація інформаційної системи передбачає можливість подальшого розширення функціоналу та удосконалення.

В результаті виконання цієї магістерської кваліфікаційної роботи, були успішно виконані всі задачі, поставлені на етапі її ініціалізації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Recommendation System. *Nvidia*. URL: https://www.nvidia.com/en-us/glossary/recommendation-system/.

2. Dutta M., Deepjyoti R. A systematic review and research perspective on recommender systems. *Springer Open*. URL: https://journalofbigdata.springeropen.com/articles/10.1186/s40537-022-00592-5.

3. Casalegno F. Recommender Systems — A Complete Guide to Machine Learning Models. *Medium*. URL: https://medium.com/data-science/recommender-systems-a-complete-guide-to-machine-learning-models-96d3f94ea748.

4. Pönkä, H. AI-based recommendation algorithms for social media services. *FaktaBaari*. URL: https://faktabaari.fi/edu/ai-based-recommendation-algorithms-for-social-media-services/.

5. Pönkä, H. (29.4.2024). Uusi selvitys TikTokin riskeistä ja raportti TikTokin Kiina-yhteyksistä, https://harto.wordpress.com/2024/04/29/uusi-selvitys-tiktokin-riskeista-ja-raportti-tiktokin-kiina-yhteyksista/

6. NPR. (11.10.2024). TikTok executives know about app's effect on teens, lawsuit documents allege, https://www.npr.org/2024/10/11/g-s1-27676/tiktok-redacted-documents-in-teen-safety-lawsuit-revealed

7. Korsunska A. Network Analysis on Youtube: Visualizing Trends in Discourse and Recommendation Algorithms. *Temple*. 26.03.2019. URL: https://sites.temple.edu/tudsc/2019/03/26/network-analysis-on-youtube/.

8. Goodrow C. On YouTube's recommendation system. *Youtube Official Blog*. 15.09.2021. URL: https://blog.youtube/inside-youtube/on-youtubes-recommendation-system/.

9. How Netflix's Recommendations System Works. *Netflix Help Centre*. URL: https://help.netflix.com/en/node/100639.

10. OneR. *Data Mining Map*. URL: http://www.saedsayad.com/oner.htm.

11. What are Naïve Bayes classifiers?. *IBM*. URL: https://www.ibm.com/think/topics/naive-bayes.

12. Sandaruwan H. Fundamentals of Associate Rule mining. *Medium*. URL: https://medium.com/image-processing-with-python/fundaments-of-associate-rule-mining-468801ec0a29.

13. Moez A. Association Rule Mining in Python. *Datacamp*. URL: https://www.datacamp.com/tutorial/association-rule-mining-python.

14. Sadrach P. How to Form Clusters in Python: Data Clustering Methods. *Builtin*. URL: https://builtin.com/data-science/data-clustering-python.

15. Key performance indicator (KPI) guide — examples and types.. *Adobe*. URL: https://business.adobe.com/blog/basics/kpi.

16. HTML. *Wikipedia*. URL: https://uk.wikipedia.org/wiki/HTML.

17. CSS. *Wikipedia*. URL: https://uk.wikipedia.org/wiki/CSS.

18. JavaScript. *Wikipedia*. URL: https://uk.wikipedia.org/wiki/JavaScript.

19. Bootstrap. *Getbootstrap*. URL: https://getbootstrap.com/docs/5.3/getting-started/introduction/.

20. Django. *Djangoproject*. URL: https://www.djangoproject.com/start/overview/.

21. sqlite. *Sqlite*. URL: https://sqlite.org/.

22. What is Deployment Diagram? [Електронний ресурс] – Режим доступу до ресурсу: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-deployment-diagram/.

ДОДАТКИ

ДОДАТОК А

РОЗРОБЛЕНИЙ КОД ДЛЯ РЕАЛІЗАЦІЇ МЕТОДІВ РЕКОМЕНДАЦІЙ

Сторінок – 3

```
import pandas as pd
import numpy as np
from scipy import sparse
import pickle
from sklearn.metrics.pairwise import cosine_similarity

df_content = pd.read_csv('../data/clean_content.csv')
df_content.head()

df_user = pd.read_csv('../data/ratings_title.csv')
df_user.head()

df_user.rename(columns={'userId':'user_id', 'movieId':'movie_id'}, inplace=True)

#Content-Based Filtering
content_similarity = pickle.load(open('../data/movie_similarity_matrix.pkl','rb'))
df_content_sim = pd.DataFrame(content_similarity, index=df_content['title'].values,
          columns=df_content['title'].values)

df_current_user = df_user[df_user['user_id'] == 569]
df_current_user

def get_content_similar_movies(user):

   #Current/target user
   df_current_user = df_user[df_user['user_id'] == user]

   #Movies watched by the current/target user
   user_watched_movies = df_current_user['title'].values

   #User's mean rating
   user_mean_rating = df_current_user['rating'].mean()

   #Filter the list of movies by like/dislike based on user's rating
   user_movies = []
   for movie in user_watched_movies:
      if df_current_user[df_current_user['title'] == movie]['rating'].values >=
user_mean_rating:
         user_movies.append(movie)

   #Create an empty dataframe to store movie recommendations for each movie seen
by the user
   similar_movies = pd.DataFrame()
   #Loop through each movie seen by the user
```

```
        for movie in user_movies:
        #Add similarity score for each movie with user_movie
        #Remove movies that the user has already seen
        similar_movies = similar_movies.append(df_content_sim[movie].drop(user_watched_movies))
    #Add the similarity score of each movie and select the movies with high scores
    content_rec = pd.DataFrame(similar_movies.sum()).reset_index().rename(columns={'index': 'title',
                    0: 'content_similarity'})
    return pd.merge(df_content[['title', 'genres']], content_rec, how='inner').sort_values(by='content_similarity', ascending=False)

content_based_scores = get_content_similar_movies(569)
#Collaborative Filtering
user_item = df_user.pivot_table(values = 'rating', index = 'user_id', columns= 'title')
norm_user_item = user_item.subtract(user_item.mean(axis=1), axis = 'rows')
user_similarity = cosine_similarity(sparse.csr_matrix(norm_user_item.fillna(0)))
df_user_sim = pd.DataFrame(user_similarity, index=user_item.index, columns=user_item.index)
def get_user_similar_movies(user, similarity_threshold):

    #Extract similar users and their similarity score with the target user
    similar_users = df_user_sim[df_user_sim[user] > similarity_threshold][user].sort_values(ascending=False)[1:]

    #Extract movies watched by the target user and their score with the target user
    target_user_movies = norm_user_item[norm_user_item == user].dropna(axis =1, how= 'all')

    #Extract movies watched by similar users and their score with the similar users
    similar_user_movies = norm_user_item[norm_user_item.index.isin(similar_users.index)].dropna(axis=1, how = 'all')

    #Keep the movies watched by similar users but not by the target user:
    for column in target_user_movies.columns:
        if column in similar_user_movies.columns:
            similar_user_movies.drop(column, axis=1, inplace=True)

    #Weighted average
    movie_score = {}
    #Loop through the movies seen by similar users
    for movie in similar_user_movies.columns:
```

```
#Extract the rating for each movie
        movie_rating = similar_user_movies[movie]
        #Variable to calculate numerator of the weighted average
        #This must be calculated for each movie
        numerator = 0
        #Variable to calculate the denominator of the weighted average
        denominator = 0
        #Loop through the similar users for that movie
        for user in similar_users.index:
            #If the similar user has seen the movie
            if pd.notnull(movie_rating[user]):
                #Weighted score is the product of user similarity score and movie rating by
the similar user
                weighted_score = similar_users[user] * movie_rating[user]
                numerator += weighted_score
                denominator += similar_users[user]
        movie_score[movie] = numerator / denominator
    #Save the movie and the similarity score in a dataframe
    movie_score = pd.DataFrame(movie_score.items(), columns=['title',
'user_similarity'])
    user_rec = pd.merge(df_content[['title','genres','year']], movie_score[['title',
'user_similarity']], how='inner')
    return user_rec.sort_values(by=['user_similarity', 'year'], ascending=False)

user_based_scores = get_user_similar_movies(569, .1)
#Hybrid
def hybrid_recommender(user):
        content_user_scores = pd.merge(get_content_similar_movies(user),
get_user_similar_movies(user, 0.1))
    content_user_scores['similarity_score'] = (content_user_scores['content_similarity']
+ content_user_scores['user_similarity']) / 2
    top_scores = content_user_scores.sort_values(by=['similarity_score', 'year'],
ascending=False)[:10]
    recommendations = pd.merge(df_content[['title','genres','imdb_rating',
'tmdb_rating']], top_scores[['title','similarity_score']], on='title')
    recommendations.rename(columns={'title':'Movie Title', 'imdb_rating': 'IMDb
Rating', 'tmdb_rating':'TMDB rating', 'similarity_score':'Similarity Score'},
inplace=True)
    return recommendations.sort_values(by='Similarity Score', ascending=False)

df_current_user
hybrid_recommender(569)
```

Календарний план

№ з/п	Назва етапів виконання бакалаврської кваліфікаційної роботи	Строк виконання етапів бакалаврської кваліфікаційної роботи	Примітка
1	Видача завдання	01.11.2024	
2	Аналіз предметної області	02.11.2024-28.11.24	
3	Моделювання предметної області	01.12.2024-26.12.24	
4	Постановка завдання	13.01.2025	
5	Проєктування системи	17.01.2025-03.03.2025	
6	Розробка системи	06.03.2025-30.06.2025	
7	Аналіз результатів	02.07.2025-14.09.2025	
8	Оформлення записки	20.09.2025-09.11.2025	
9	Оформлення постеру	17.10.2025-20.10.2025	
10	Підготовка тез	21.10.2025-23.10.2025	
11	Постерна сесія	28.10.2025-29.10.2025	
12	Перевірка на плагіат	14.11.2025	
13	Попередній захист	24.11.2025-28.11.2025	
14	Захист	05.12.2025-13.12.2025	

РЕФЕРАТ

Основна частина пояснювальної записки займає сторінок 6, із них 56 сторінок основного тексту. Додатки мають обсяг в 3 сторінки. Робота містить 37 рисунків і 1 таблицю. Для виконання дослідження та підготовки матеріалів використано 22 джерела інформації.

Об’єкт дослідження — алгоритми рекомендацій контенту на основі вподобань користувачів.

Предмет дослідження — рекомендаційна система фільмів на основі вподобань користувачів.

Методи дослідження. Для вирішення поставлених завдань використовувалися методи системного аналізу для оцінки ефективності алгоритмів, об'єктно-орієнтованого проєктування, методи машинного навчання та інтелектуального аналізу даних для створення моделей рекомендацій, експериментальні методи для перевірки працездатності прототипу системи.

Мета і завдання дослідження. Основна мета даного дослідження — аналіз сучасних підходів до побудови рекомендаційних систем, виявлення їхніх проблем та обмежень, а також розробка та впровадження гібридної моделі рекомендацій для формування персоналізованих пропозицій фільмів із підвищеною точністю та ефективністю.

Наукова новизна одержаних результатів полягає у формалізації задачі побудови рекомендаційної системи фільмів на основі гібридного підходу до фільтрації даних для вдосконалення процесу формування особистих рекомендацій. Додатковим внеском є дослідження можливостей інтеграції OLAP-технологій для підвищення продуктивності та адаптивності алгоритмів рекомендацій.

Практичне значення роботи полягає у створенні прототипу рекомендаційної системи фільмів, що може бути використаний як основа для розробки повноцінних інформаційних сервісів у сфері кіноіндустрії та медіа.

ABSTRACT

The main part of the explanatory note occupies 66 pages, of which 56 pages are the main text. The appendices are 3 pages long. The work contains 37 figures and 1 table. Twenty-two sources of information were used to conduct the research and prepare the materials.

The object of the study is content recommendation algorithms based on user preferences.

The subject of the study is a movie recommendation system based on user preferences.

Research methods. To solve the tasks set, methods of system analysis were used to evaluate the effectiveness of algorithms, object-oriented design, machine learning and intelligent data analysis methods to create recommendation models, and experimental methods to test the prototype system's performance.

The purpose and objectives of the study. The main purpose of this study is to analyze current approaches to building recommendation systems, identify their problems and limitations, and develop and implement a hybrid recommendation model for generating personalized movie suggestions with increased accuracy and efficiency.

The scientific novelty of the results obtained lies in the formalization of the task of building a movie recommendation system based on a hybrid approach to data filtering to improve the process of forming personal recommendations. An additional contribution is the study of the possibilities of integrating OLAP technologies to improve the performance and adaptability of recommendation algorithms.

The practical significance of the work lies in the creation of a prototype movie recommendation system that can be used as a basis for developing full-fledged information services in the film industry and media. The proposed approach can be adapted for other industries where personalization of information is required.