Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
Кафедра Програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

другий (магістерський)
(рівень вищої освіти)

Дослідження алгоритмів рекомендацій та їх застосування у сфері відпочинку та розваг

Виконав:
Студент 2 курсу групи ІПЗм-21-1
Стась Б. Л.
(прізвище, ініціали)
Спеціальність 121 — Інженерія програмного забезпечення
Тип програми Освітньо-наукова
Керівник д.т.н проф. Власенко Л. А.
(посада, прізвище, ініціали)

Допускається до захисту
Зав. Кафедри _____ З. В. Дудар

2023 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Програмної інжеренії

Рівень вищої освіти другий (магістерський)

Спеціальність 121 — Інженерія програмного забезпечення

Спеціалізація Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:

Зав. Кафедри _____

«____» _____ 20___ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студенту Стасю Богдану Леонідовичу

(прізвище, ім'я, по батькові)

Тема роботи: «Дослідження алгоритмів рекомендацій та їх застосування у сфері відпочинку та розваг»

затверджена наказом університету від 29 березня 2023 р. № 302 Ст

2. Термін подання студенткою роботи до екзаменаційної комісії « 19 » травня 2023 р.

3. Вихідні дані до роботи *рекомендаційні системи, колаборативна фільтрація, Python, Azure, Databricks*.

4. Перелік питань, що потрібно опрацювати в роботі *мета роботи, аналіз предметної галузі і постановка задачі, дослідження різних методів рекомендацій, зокрема їх недоліки та спосіб реалізації*.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, ілюстрацій (слайдів) *мета завдання, обґрунтування доцільності розроблення, постановка задачі, методи і алгоритми, опис отриманих результатів, демонстраційні матеріали*

6. Консультанти розділів роботи

Найменування розділу	Консультант (посада, прізвище, ім’я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Спецчастина	д.т.н проф. Власенко Л.А.		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз предметної галузі	24.10.2022	виконано
2	Огляд існуючих рекомендаційних систем та алгоритмів реалізації	03.11.2022	виконано
3	Постановка задачі	14.11.2022	виконано
4	Дослідження методів	17.03.2023	виконано
5	Створення оточення для системи, знайомство з Azure Databricks	28.03.2023	виконано
6	Підготовка пояснювальної записки	15.04.2023	виконано
7	Перевірка роботи на антіплагіат	10.05.2023	виконано
8	Нормоконтроль	11.05.2023	виконано
9	Рецензування	13.05.2023	виконано
10	Підготовка презентації та доповіді	13.05.2023	виконано
11	Попередній захист	14.05.2023	виконано
12	Занесення роботи в електронний архів	14.05.2023	виконано
13	Допуск до захисту у зав. кафедри	19.05.2023	виконано

Дата видачі завдання « 24 » січня 202__ р.

Студент _____

(підпис)

Керівник роботи _____ проф, д.т.н. Власенко Л. А.

(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ / ABSTRACT

Квалифікаційна робота магістра містить: 62с., 21 рис., 18 джерел , 4 додатків.

РЕКОМЕНДАЦІЙНІ СИСТЕМИ, ФІЛЬТРАЦІЯ ДАНИХ, КОЛАБОРАТИВНА ФІЛЬТРАЦІЯ, КОНТЕНТНА ФІЛЬТРАЦІЯ.

Об’єктом дослідження є оцінка роботи різних алгоритмів рекомендацій у різних системах для розваг.

Метою роботи є дослідження різних масових систем розваг, а саме їх алгоритмів рекомендацій, таких як контентна чи гібридна фільтрація та їх порівняння.

У результаті виконання роботи було проаналізовано методи для знаходження рекомендацій, також було проведено аналіз для існуючих систем з можливістю рекомендацій.

RECOMMENDATION SYSTEMS, DATA FILTRATION, COLABORATIVE FILTRATION, CONTENT FILTRATION.

The object of the study is to evaluate the performance of various recommendation algorithms in various systems for the search engine.

The purpose of the work is to study various mass entertainment systems, namely their recommendation algorithms, such as content or hybrid filtering, and their comparison.

As a result of the work, methods for finding recommendations were analyzed, as well as an analysis of existing systems with the possibility of recommendations was carried out.

Я, Стась Богдан Леонідович, студент гр. ІПЗм-21-1, здобувач вищої освіти на другому (магістерському) рівні кафедри «Програмна інженерія», заявляю: моя кваліфікаційна робота на тему «Дослідження алгоритмів рекомендацій та їх застосування у сфері відпочинку та розваг», що буде представлена в екзаменаційну комісію для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElAr KhNURE. Всі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлений з діючим положенням «Про протидію академічному плагіату в ХНУРЕ», згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

ЗМІСТ

ВСТУП

Не так багато часу назад, коли люди проживали в невеликих спільнотах. Кожен продавець музичних платівок чи відеокасет знав особисто всіх поточних клієнтів та мав можливість запропонувати персональні рекомендації, які обґрунтовані на знаннях клієнта та його уподобань, щодо музики чи фільмів.

Такий тип відносин між клієнтом та продавцем надавав чудовий сервіс, так як продавець точно знав усі уподобання, можливості платоспроможності та потреби. Завдяки цьому клієнтам надавалися релевантні поради щодо покупки.

На сьогоднішній день, спосіб взаємодії між клієнтом та продавцем змінився, тож майже кожен має доступ до сервісів з надання розваг, таких як музика чи фільми, у мережі. Так, клієнти мають можливість вибору різноманітних видів музики, або фільмів самостійно, але вже не має того особистого зв’язку між клієнтом та продавцем і переваг, що це надавало.

Тому, можна виділити очевидну проблему – нестача знань уподобань користувача призводить до втрати продажів. Для вирішення цієї проблеми зараз активно розроблюються та вдосконалюються алгоритми та сервіси для рекомендацій. Рекомендаційні системи - це набір технік інформаційного фільтрування, які пропонують користувачам потенційно корисні для них товари.

Тож величезна кількість сервісів з надання послуг розваг у мережі мають свій функціонал підбору рекомендацій. Метою роботи є дослідити існуючи сервіси рекомендації та запропонувати власний алгоритм створення рекомендації на основі дворівневої моделі, яка поєднує підходи контентної та колаборативної фільтрації даних [1].

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Актуальність рекомендаційних систем

Рекомендаційні системи активно використовуються всіма відомими компаніями, такими як: Netflix, YouTube, Facebook, Amazon, Spotify, LinkedIn, TripAdvisor, Last.fm, IMDb, Google та багато інших. Також слід підкреслити, що велика кількість медіа компаній на наш час розробляють рекомендаційні системи як складову частину свого сервісу, який вони надають своїм користувачам [2].

Наприклад Netflix, компанія яка надає можливість перегляду різноманітного контенту, нагородила мільйоном доларів команду яка перша змогла покращити швидкодію їхньої рекомендаційної системи.

Також слід підмітити компанію McKinsey, рекомендаційна система якої надає більш 75% контенту, який переглядали клієнти Netflix[1]. Система допомагає користувачам знаходити контент, відповідний до їх уподобань, який вони хочуть переглянути та заощаджую кошти компанії Netflix на витратах для реклами [3].

Якщо брати до уваги компанію Amazon, то за її свідченнями 35% свого доходу вона отримує завдяки рекомендаційним системам. По перше, в системі Amazon показують товари, котрі часто купують з тими товарами, які у них вже є в кошику. Також у них показуються товари, які схожі на ті, що були щойно переглянуті користувачем та товари які є новими версіями до тих, які вони вже купили. Цікаво ще те, що вони вони використовують рекомендаційні системи не лише на сайті, але і в email розсилках, які показують значний приріст у продажах.

Ще цікавим фактом актуальності є щорічна конференція «The Association of Computing Machinery Conference Series on Recommender Systems» на якій різні девелопери діляться своїм досвідом, щодо розробки рекомендаційних систем, також під час цього заходу відбувається змагання «RecSys Challenge»

Поява та зростання інтернет магазинів має значний вплив на поведінку покупців, надаючи їм доступ до значної різноманітності товарів та інформації. Поки ця свобода покупок зробила інтернет комерцію в багато-мільярдну індустрію, вона також ускладнила для покупців вибір товарів, які найбільше відповідають

їхнім потребам. Одним із основних методів вирішення проблеми інформаційного перевантаження є рекомендаційні системи, які забезпечують автоматизоване та персоналізоване пропонування товарів для споживачів [4].

1.2 Аналіз підходів надання рекомендацій

Розрізняють два типи рекомендації контенту або товарів такі як: персоналізовані та не персоналізовані. Не персоналізовані рекомендації – це вибір найпопулярнішого контенту або товарів, чи рекомендація того контенту який потребує бізнес.

Але нас більш цікавить саме персоналізовані рекомендації, які можна поділити на:

– content-based (базовані на контенті);
– collaborative filtering (колаборативну фільтрацію);
– hybrid-based(поєднання content-based та collaborative filtering).

Загальна мета content-based[2] методів – знайти якісь спільні якості у контенту, який отримав гарну оцінку або був переглянутий чи прослуханий(якщо це якийсь трек) користувачем, а потім рекомендувати новий контент зі схожими характеристиками конкретному користувачу.

У цьому методі профіль користувача створюється для надання інформації про типи елементів, які користувач любить, на основі ключових слів, які використовуються для опису елементів. Метод фільтрації на основі вмісту створює профіль для кожного елемента (на основі набору дискретних атрибутів та функцій), який використовується для характеристики елемента в системі. Потім система створює профіль для користувача на основі зваженого вектора характеристик елемента (з предметів, які користувач раніше оцінював або купив, і від елементів, які користувач переглядає). Вага означає важливість кожної функції для користувача [5].

Існує багато можливих способів обчислення цих ваг: баєсів класифікатор, кластерний аналіз, дерева рішень, штучні нейронні мережі. Незалежно від методики обчислення ціль вагового вектора однакова - оцінити ймовірність того, що користувачу сподобається запропонований елемент. Цей підхід може використовувати історію переглядів сторінок, наприклад, які блоги читає користувач та характеристики цих блогів. Якщо користувач часто читає статті про Linux або, ймовірно, залишатиме коментарі в блогах про розробку програмного забезпечення, фільтрація на основі вмісту може використовувати цю історію для ідентифікації та рекомендації аналогічного вмісту (статті на Linux або інших блогах про розробку програмного забезпечення). Цей вміст можна визначити вручну або автоматично вилучати на основі інших методів подібності.

Проте слід зазначити, що рекомендаційні систем, які основані лише на контенті, часто мають проблеми аналізу обмеженого вмісту та надмірної спеціалізації. Аналіз обмеженого вмісту – це випадок коли система має обмежену кількість даних про користувача та контент яким він користується.

Аналіз обмеженого вмісту може виникати через вимогу про конфіденційність приватних даних, що не дозволяє використовувати дані потрібні для аналізу. Також одним з недоліків такого підходу можна зазначити надмірну спеціалізацію, це можна сказати побічний ефект того я к рекомендують content-based системи.

Наприклад, в якомусь додатку для перегляду фільмів, система може рекомендувати фільми однакового жанру, або з тими ж акторами, які найчастіше зустрічаються в фільмах, які переглядає користувач. Але така система не спроможна порекомендувати фільм, який відрізняється від попередньо-переглянутих, але в той же час подобається користувачу.

Ось тут нам на допомогу приходять методи колаборативної фільтрації. Ці методи не залежать від конкретної інформації, вони використовують інформацію основану на оцінках від інших користувачів у системі. Основним принципом є те, що оцінка певного користувача для певного контенту буде схожою з тою, яку поставив інший користувач, якщо вони обидва оцінили інший контент подібними оцінками.

Колаборативна фільтрація[3] змогла подолати певні обмеження, які має content-based. Наприклад, товари, контент яких є відсутнім або важко доступним, можуть бути рекомендовані базуючись на відгуках інших користувачів. Більше того, колаборативні рекомендації базуються на якості товарів які оцінені іншими, ніж покладатись на контент який може бути поганим індикатором якості товару. Також колаборативна фільтрація може рекомендувати різноманітні товари з різним контентом, оскільки інші користувачі проявили інтерес до цих різних товарів.

Колаборативна фільтрація підходить системам, які грунтуються на моделі попередньої поведінки користувачів. Модель може бути побудована виключно з поведінки одного користувача або (більш ефективно) від поведінки інших користувачів, які мають подібні риси. При врахуванні поведінки інших користувачів, колаборативна фільтрація використовує знання про групи, щоб сформувати рекомендації на основі спільних рис. По суті, рекомендації базуються на автоматичній співпраці декількох користувачів і фільтрації на тих, хто виявляє подібні риси або поведінку

Підходи колаборативної фільтрації можуть бути поділені на два загальні класи, такі як: базовані на знаходження сусідів та базовані на навчанні моделі. В підході основою якого є знаходження сусідів, оцінки користувача для контенту чи товару, який збережений в системі, безпосередньо використовуються під час прогнозування оцінок для нового контенту чи товару.

Це може бути здійснено двома способами відомими як user-based та item-based рекомендаціями. User-based системи оцінюють інтерес певного користувача для контенту використовуючи оцінки для цього контенту від інших користувачів, які називають сусідами та мають схожі моделі поведінки оцінювання. Сусіди певного користувача є зазвичай користувачі, оцінки яких найбільше корелюють з оцінками даного користувача. Item-based[4] підхід, з іншого боку, прогнозує оцінки користувача для контенту базуючись на оцінках користувача для схожого контенту. В таких підходах, два контенти є схожими якщо декілька користувачів системи оцінили цей контент подібними оцінками [6].

На відміну від методів на знаходження сусідів, які використовують збережені раніше оцінки користувачів під час прогнозування, методи які базуються на моделях використовують ці оцінки, для того щоб навчити рекомендувати модель. Багато важливої інформації про користувача та контент зберігаються в множені параметрів моделі , котрі були отримані в результаті навчання моделі на тренувальних даних та можуть бути використані для прогнозування майбутніх оцінок.

На завершення, щоб подолати певні обмеження методів базованих на контенті та колаборативної фільтрації, використовують гібридні рекомендаційні методи, які поєднують кращі характеристики обох підходів.

Гібридні рекомендаційні можна поділити на:

– weighted;
– switching;
– mixed;
– feature Combination;
– feature Augmentation;
– cascade;

У системі зважених(Weighted) рекомендацій ми можемо визначити кілька моделей, які можуть добре інтерпретувати набір даних. Система зважених рекомендацій візьме вихідні дані з кожної моделі та об’єднає результат у статичні зважування, вага яких не змінюватиметься в поєднанні та тестовому наборі.

Наприклад, ми можемо об’єднати модель на основі вмісту та модель спільного фільтрування предметів, і кожна з них матиме вагу 50% для остаточного прогнозу [7].

Перевага використання зваженого гібриду полягає в тому, що ми інтегруємо кілька моделей для підтримки набору даних у процесі рекомендацій лінійним способом (див. рис. 1.1).

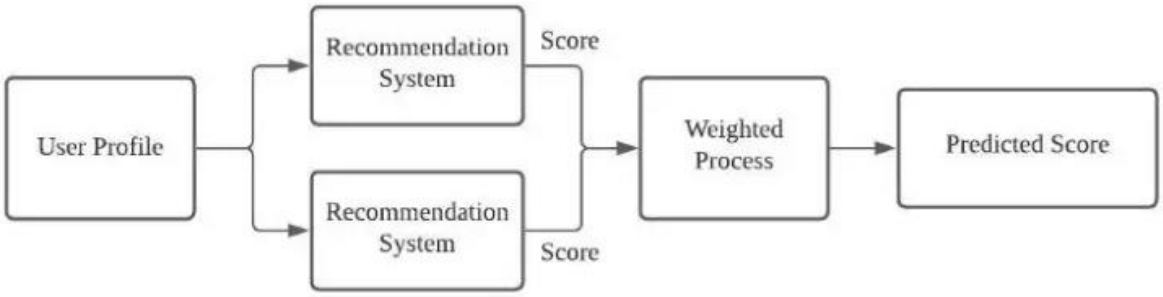

Рисунок 1.1 – Схема weighted рекомендацій

Комутаційний гібридний підхід вибирає єдину систему рекомендацій на основі ситуації. Модель використовується для створення конфіденційного набору даних на рівні елемента, ми повинні встановити критерії вибору рекомендацій на основі профілю користувача або інших функцій [8].

Такий підхід вводить додатковий рівень у рекомендаційну модель, яка вибирає відповідну модель для використання. Система рекомендацій чутлива до сильних і слабких сторін складової моделі рекомендацій (див. рис. 1.2).

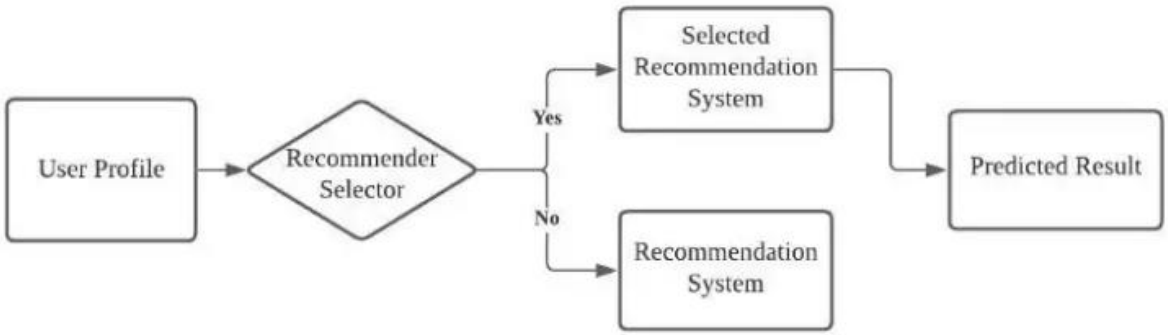

Рисунок 1.2 – Схема Switching рекомендацій

Змішаний гібридний підхід спочатку використовує профіль користувача та функції для створення різних наборів даних-кандидатів. Система рекомендацій відповідно вводить інший набір кандидатів у модель рекомендації та комбінує прогноз для отримання рекомендації результату.

Змішана гібридна система рекомендацій здатна надавати велику кількість рекомендацій одночасно та адаптувати частковий набір даних до відповідної моделі, щоб мати кращу продуктивність (див. рис. 1.3).

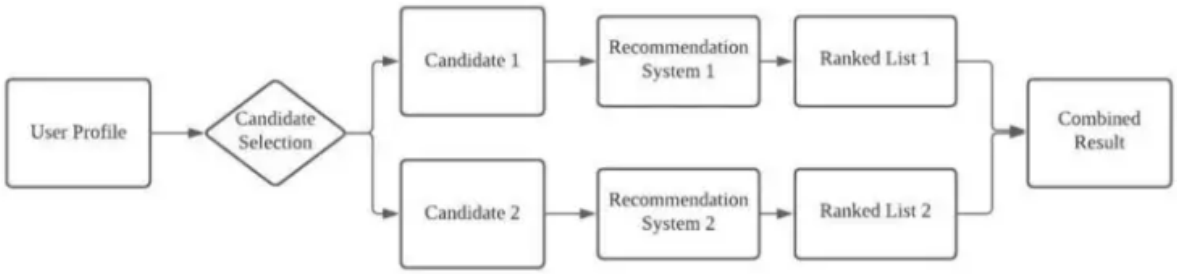

Рисунок 1.3 – Схема Mixed рекомендацій

У гібридній комбінації функцій(Feature Combination) ми додаємо до системи віртуальну модель рекомендацій, яка працює як розробка функцій щодо вихідного набору даних профілю користувача.

Наприклад, ми можемо додати функції спільної моделі рекомендацій до моделі рекомендацій на основі вмісту. Гібридна модель здатна розглядати спільні дані з підсистеми, спираючись виключно на одну модель (див. рис. 1.4).

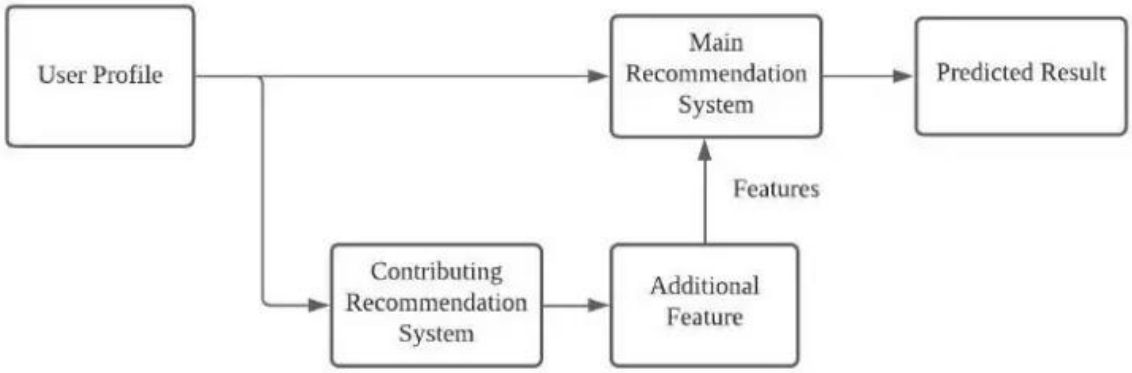

Рисунок 1.4 – Схема Feature Combination рекомендацій

Додаткова модель рекомендацій використовується для створення рейтингу або класифікації профілю користувача/елемента, який далі використовується в

основній системі рекомендацій для отримання остаточного прогнозованого результату.

Гібрид розширення функцій здатний покращити продуктивність основної системи без зміни основної рекомендаційної моделі. Наприклад, використовуючи правило асоціації, ми можемо покращити набір даних профілю користувача. Завдяки розширеному набору даних продуктивність моделі рекомендацій на основі контенту буде покращена (див. рис. 1.5).

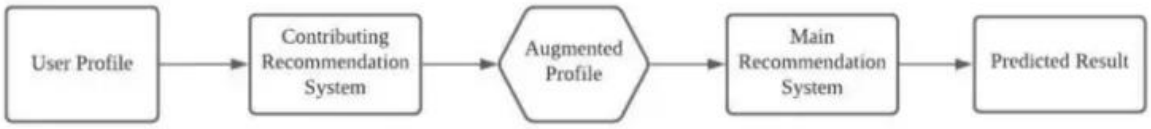

Рисунок 1.5 – Схема Feature Augmentation рекомендацій

Cascade-hybrid визначає сувору ієрархічну структуру системи рекомендацій, так що основна система рекомендацій дає первинний результат, а ми використовуємо вторинну модель для вирішення деяких незначних проблем первинного результату, як-от розірвання нічиєї в оцінці (див. рис. 1.6).

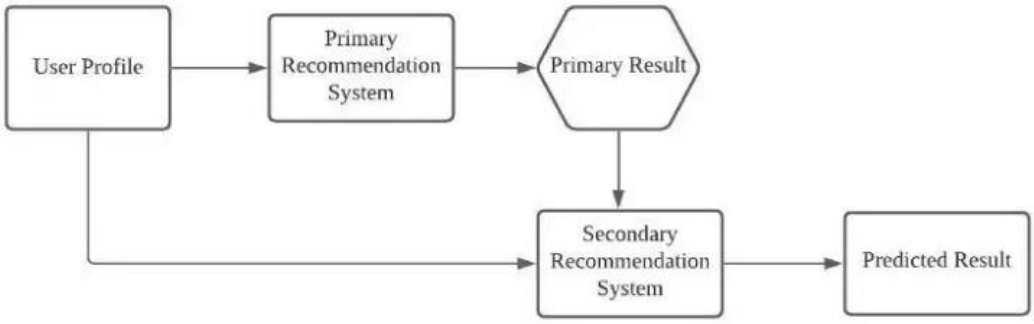

Рисунок 1.6 – Схема Cascade-hybrid рекомендацій

На практиці більшість набору даних є розрідженими, вторинна модель рекомендацій може бути ефективною проти проблеми рівних балів або проблеми з відсутністю даних

1.3 Постановка завдань дослідження

Зробивши підсумки проведеного вище аналізу методів рекомендаційних систем та різноманітних застосунків, котрі активно використовують рекомендаційні системи, можна поставити задачу, котра буде виконана в цій кваліфікаційній роботі.

З наведених вище аргументів можна побачити, що метод колаборативної фільтрації постійно використовується в рекомендаційних системах. Такий вид фільтрації є найбільш популярним підходом та загалом працює краще ніж фільтрація за допомогою вмісту.

Отже, необхідно виконати порівняльний аналіз методів колаборативної фільтрації, описати можливі варіанти реалізацій, виявити сильні та слабкі сторони кожного з підходів та реалізувати один з методів колаборативної фільтрації програмно [9].

2 ДОСЛІДЖЕННЯ ТИПІВ РЕКОМЕНДАЦІЙНИХ СИСТЕМ

2.1 Аналіз методів для реалізації системи рекомендації

Рекомендаційні системи - це програмні інструменти, що допомагають людям знайти продукти, послуги або контент, які вони ймовірно будуть зацікавлені в залежності від їхніх попередніх дій та інформації про них. Рекомендаційні системи використовуються в багатьох різних контекстах, включаючи електронну комерцію, соціальні мережі, медіа, онлайн-ігри та інші [10].

Рекомендаційні системи в цілому класифікуються на три різних типи: рекомендаційні системи на основі вмісту, спільні рекомендаційні системи та гібридні рекомендаційні системи. Діаграмне представлення різних типів рекомендаційних систем наведено на рисунку 2.1.

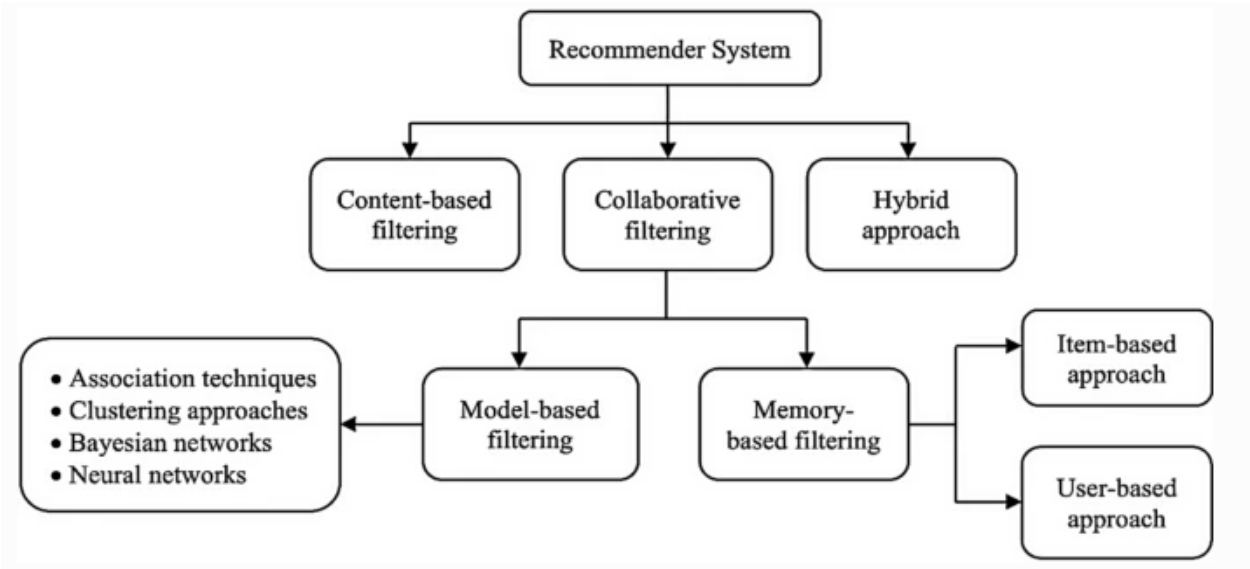

Рисунок 2.1 – Типи систем рекомендацій

За наданою схемою можна зрозуміти, що системи рекомендацій зазвичай використовують фільтрацію на основі співпраці та фільтрацію на основі вмісту (також відому як підхід на основі особистості). Крім того, існують інші системи, такі як системи на основі знань. Підходи до колаборативної фільтрації використовують модель на основі минулої поведінки користувача, яка враховує раніше придбані чи вибрані предмети та числові оцінки цих елементів, а також

рішення, прийняті іншими користувачами. На основі цієї моделі прогнозуються елементи (або оцінки предметів), які можуть зацікавити користувача. Підходи до фільтрації на основі вмісту використовують дискретні характеристики елемента, щоб рекомендувати додаткові елементи зі схожими властивостями, які були попередньо позначені [11].

2.2 Метод фільтрації на основі вмісту

У системах рекомендацій на основі вмісту всі дані зібрані в різні профілі елементів на основі їх опису або характеристик. Наприклад, у випадку книг, характеристиками будуть автор, видавець та інші. У випадку фільмів характеристиками будуть режисер, актори тощо. Коли користувач виставляє позитивну оцінку елементу, то інші елементи, які знаходяться в тому самому профілі, агрегуються разом, щоб створити профіль користувача. Цей профіль користувача поєднує всі профілі елементів, які були оцінені користувачем позитивно. Елементи, що присутні в цьому профілі користувача, потім рекомендуються користувачу, як показано на рисунку нижче (див. рис. 2.2).

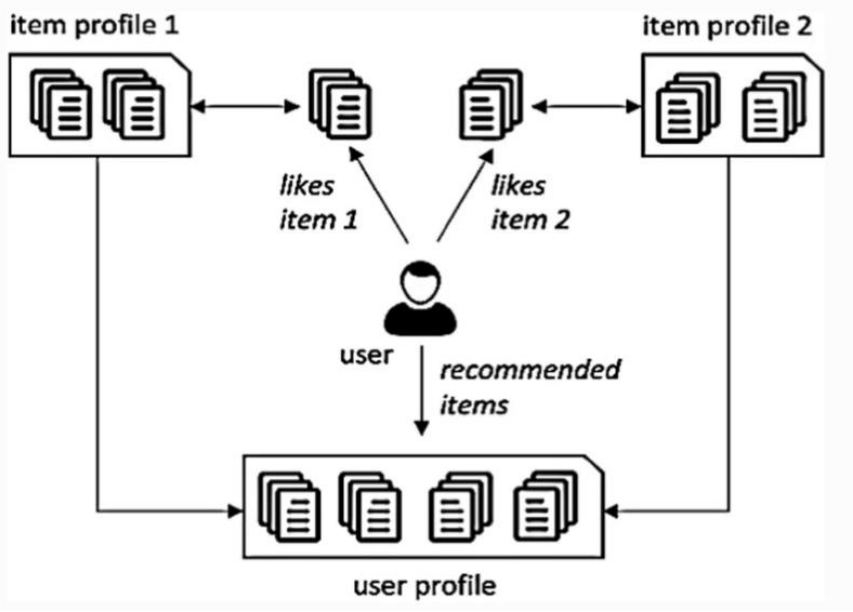

Рисунок 2.2 – Схема роботи фільтрації на основі вмісту

Для виконання абстрагування особливостей елементів у системі, часто використовують алгоритм представлення предметів, зокрема, широко використовується представлення tf-idf, яке також називається представленням векторного простору. За допомогою цього алгоритму, система створює профіль користувача на основі зваженого вектора характеристик елементів. Вагові коефіцієнти вказують на важливість кожної функції для користувача і можуть бути обчислені з індивідуально оцінених векторів вмісту за допомогою різноманітних методів [12].

Прості підходи використовують середні значення вектора оціненого елемента, тоді як складніші методи, такі як байєсівські класифікатори, кластерний аналіз, дерева рішень та штучні нейронні мережі, використовуються для оцінки ймовірності того, що елемент сподобається користувачеві.

Один з недоліків цього підходу полягає в тому, що для точної рекомендації вимагається глибоке знання особливостей предмета. Ці знання можуть не бути доступні для всіх предметів. Крім того, такий підхід має обмежену можливість розширення вибору чи інтересів користувача. Однак, цей підхід має багато переваг. Оскільки вподобання користувачів зазвичай змінюються з часом, цей підхід має швидку здатність динамічно пристосовуватись до змін вподобань користувача. Оскільки один профіль користувача є специфічним лише для цього користувача, цей алгоритм не вимагає детальних профілів інших користувачів, оскільки вони не впливають на процес рекомендації [13].

Це забезпечує безпеку та приватність даних користувача. Якщо нові елементи мають достатній опис, то методи на основі контенту можуть перебороти проблему початкового холодного старту, тобто такий підхід може рекомендувати елемент навіть тоді, коли його раніше не було оцінено жодним користувачем. Підходи на основі контенту більш поширені в системах, таких як персоналізовані рекомендації новин, публікацій, веб-сторінок і т.д.

2.3 Метод колаборативної фільтрації.

Колаборативні підходи використовують міру подібності між користувачами. Ця техніка починається з пошуку групи або колекції користувачів X, чиї вподобання, схильності та неприязні схожі з користувачем A. X називається оточенням A. Потім користувачеві A рекомендуються нові елементи, які сподобалися більшості користувачів з X.

Ефективність алгоритму колаборативної(спільної) фільтрації залежить від того, наскільки точно алгоритм може знайти оточення цільового користувача. Традиційні системи на основі фільтрації за спільною взаємодією стикаються з проблемою старту з холодної початкової точки та проблемами приватності, оскільки потрібно ділитися даними користувачів. Однак підхід на основі спільної фільтрації не потребує жодного знання особливостей елемента для генерації рекомендації. Крім того, цей підхід може допомогти розширити інтереси користувача, відкриваючи нові елементи. Колаборативні підходи знову діляться на два типи: підходи на основі пам'яті та підходи на основі моделі.

Підходи на основі пам'яті колаборативної фільтрації рекомендують нові елементи, враховуючи вподобання їх оточення. Вони безпосередньо використовують матрицю корисності для прогнозування. У цьому підході першим кроком є створення моделі. Модель є функцією, яка приймає матрицю корисності як вхід.

$$\text{Модель} = f\,(\text{матриця корисності}) \tag{2.1}$$

Тоді рекомендації робляться на основі функції, яка приймає модель та профіль користувача як вхідні дані. Тут ми можемо давати рекомендації тільки користувачам, профіль яких належить до матриці корисності. Тому, щоб дати рекомендації новому користувачу, профіль користувача має бути доданий до матриці корисності, і матрицю схожості слід перерахувати, що робить цю техніку обчислювально важкою.

Рекомендація = f (визначена модель, профіль користувача), де профіль користувача належить до матриці корисності.

Підходи колаборотивної фільтрації, засновані на пам'яті, знову підрозділяються на два типи: фільтрацію на основі користувачів та фільтрацію на основі елементів. У підході на основі користувачів рейтинг користувача нового елемента розраховується шляхом знаходження інших користувачів з сусідства користувача, які раніше оцінювали цей же елемент. Якщо новий елемент отримує позитивні оцінки від сусідів користувача, то цей новий елемент рекомендується користувачеві. На рисунку 2.3 зображено підхід на основі користувачів.

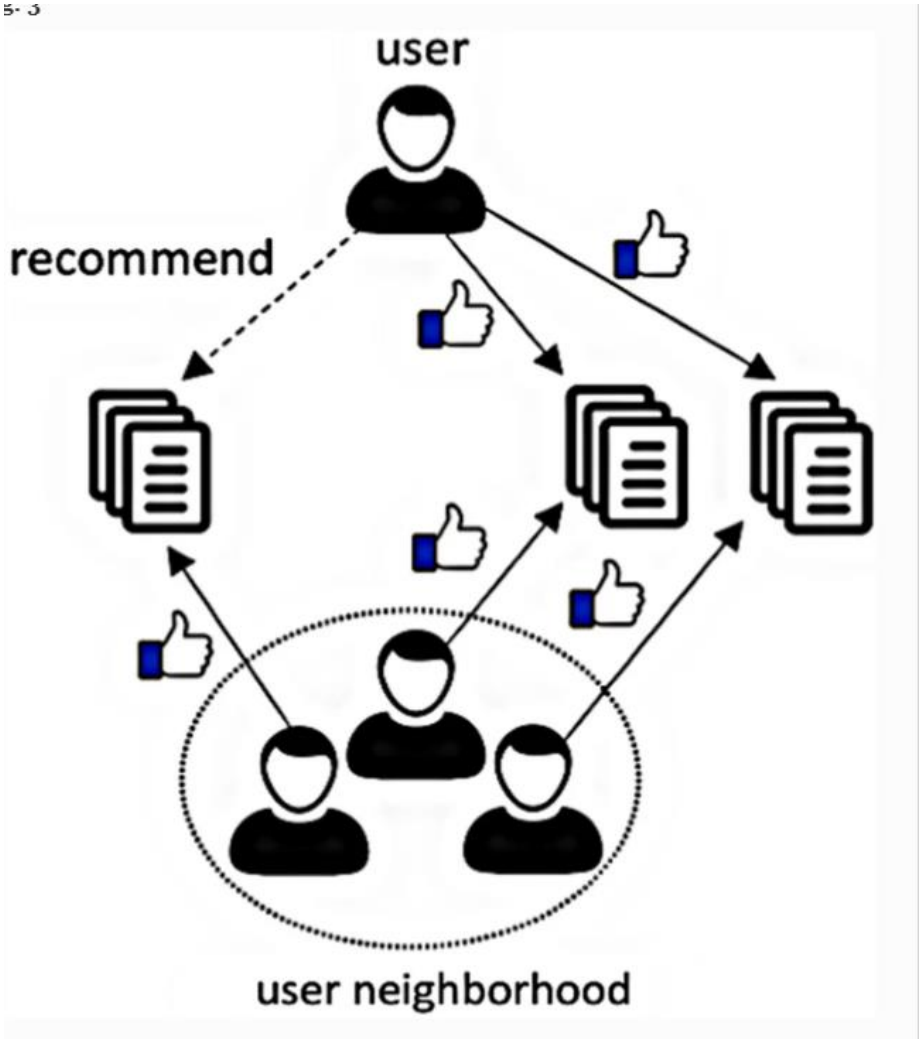

Рисунок 2.3 – Підхід на основі користувачів

У підході, що базується на предметах, будується кластер предметів, що складається з усіх подібних предметів, які користувач раніше оцінював. Потім

рейтинг користувача для нового іншого предмета передбачається шляхом обчислення зваженого середнього усіх рейтингів, що присутні в схожому сусідстві предметів, як показано на рисунку 2.4.

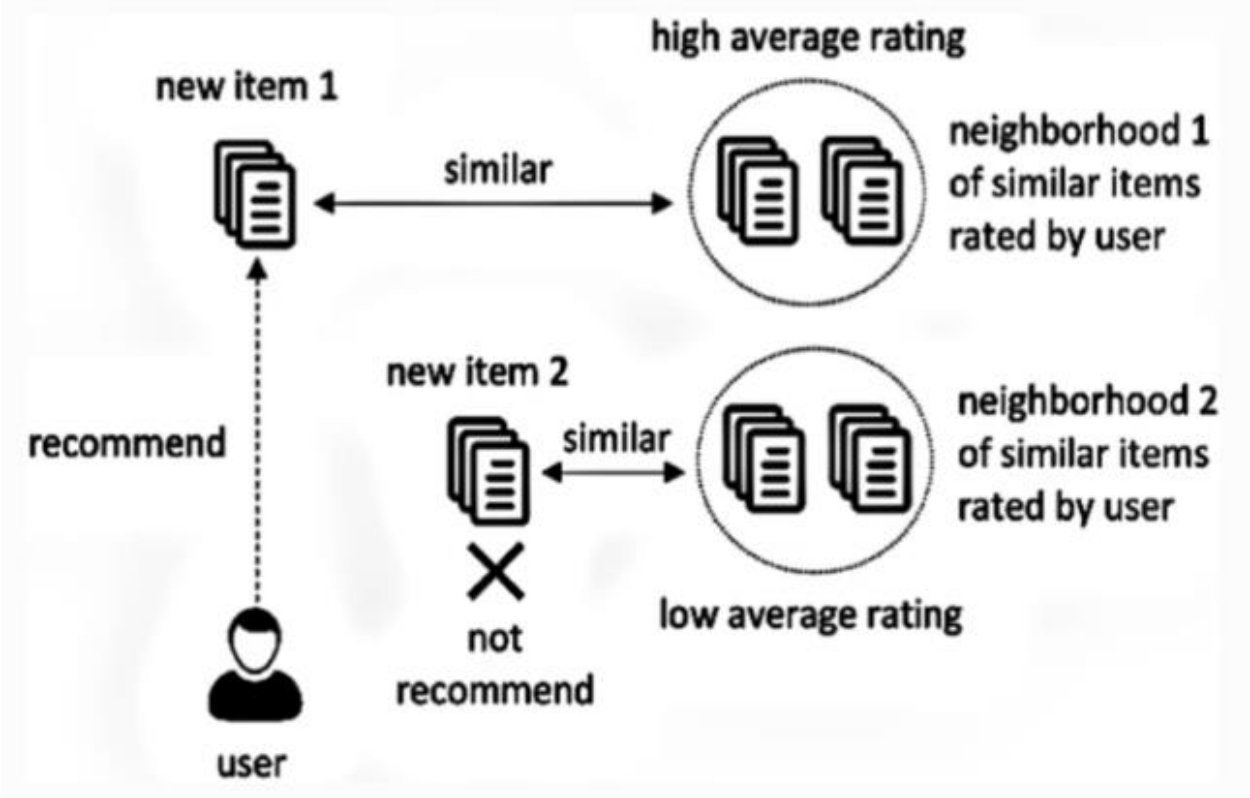

Рисунок 2.4 – Підхід на основі предметів

Системи на основі моделей використовують різні алгоритми з добування даних та машинного навчання для створення моделі, яка передбачає рейтинг користувача для непроглянутих елементів. Під час розрахунку рекомендацій вони не залежать від повного набору даних, а отримують риси з датасету для обчислення моделі. Тому вони називаються "модельно-орієнтованою" технікою. Ці техніки також потребують двох кроків для передбачення - перший крок - побудова моделі, а другий - передбачення рейтингу з використанням функції (f), яка отримує модель, визначену на першому кроці, та профіль користувача в якості вводу [14].

Рекомендація = f (визначена модель, профіль користувача), де профіль користувача ∉ матриці корисності.

Техніки на основі моделей не потребують додавання профілю користувача нового користувача до матриці корисності перед передбаченням. Ми можемо давати рекомендації навіть користувачам, які не присутні в моделі. Системи на основі моделей є більш ефективними для групових рекомендацій. Вони можуть швидко рекомендувати групу елементів, використовуючи навчену модель. Точність цієї техніки в значній мірі залежить від ефективності алгоритму навчання, який використовується для створення моделі. Техніки на основі моделей здатні вирішувати деякі традиційні проблеми систем рекомендацій, такі як розрідженість та масштабованість, використовуючи техніки зменшення розмірності [86] та техніки навчання моделі.

2.4 Метод гібридної фільтрації

Гібридний метод фільтрації - це поєднання двох або більше різних методів рекомендації з метою зниження обмежень, що присутні в окремих методах. Гібридні методи можуть використовувати різні комбінації підходів, таких як колаборативна фільтрація, контент-базована фільтрація та модель-базова фільтрація, для створення більш точних та ефективних рекомендацій.

Наприклад, гібридний метод може використовувати колаборативну фільтрацію для рекомендації товарів, які сподобалися користувачам з подібними смаками, а також контент-базовану фільтрацію для рекомендації товарів на основі характеристик товару та інтересів користувачів.

Гібридні методи можуть бути об'єднані з різними підходами, наприклад, з використанням мета-рівня, збагачення характеристик, комбінації характеристик, змішаної гібридизації, каскадної гібридизації, переключення гібридизації та зваженої гібридизації. Кожен з цих підходів має свої переваги та недоліки, тому гібридні методи можуть бути налаштовані для оптимального використання в конкретному контексті [15].

Гібридні методи можуть бути розділені на декілька типів:

Це підхід до гібридної рекомендації, який поєднує різні алгоритми рекомендації на різних рівнях, зазвичай з метою покращення продуктивності та точності рекомендацій. У цьому підході використовуються кілька різних алгоритмів рекомендації, кожен з яких може бути використаний для рекомендації на окремому рівні системи.

Наприклад, один алгоритм може бути використаний для рекомендації популярних товарів, тоді як інший може бути використаний для рекомендацій нішевих товарів. Мета-рівневий гібрид може бути особливо корисним для рекомендацій великих каталогів, де кількість товарів і користувачів дуже велика, і для яких важко знайти один алгоритм, який працює на всіх рівнях.

Один з підходів до реалізації мета-рівневого гібрида полягає в тому, щоб використовувати декілька алгоритмів рекомендації на різних рівнях системи та об'єднувати їх результати в один вихідний список рекомендацій. Наприклад, можна використовувати і колаборативний, і контентний підходи до рекомендації, а результати кожного з них згрупувати в кінцевий список.

Гібрид з розширенням функцій (feature-augmentation hybrid) поєднує підходи колаборативної та контентної фільтрації, але з урахуванням додаткових атрибутів (features), які відображають характеристики користувачів та об'єктів.

Цей підхід передбачає, що спочатку збирається інформація про користувачів та об'єкти, включаючи їх додаткові характеристики, такі як вік, стать, інтереси, тематика об'єктів тощо. Потім використовуються алгоритми машинного навчання для створення моделі, яка враховує ці додаткові атрибути, які можуть впливати на рекомендації. Наприклад, якщо користувач завжди шукав і купував книжки певного жанру, то цей факт може бути використаний для рекомендації інших книжок цього жанру.

Один з варіантів застосування гібриду з розширенням функцій - це використання додаткових атрибутів для змішування результатів, отриманих від колаборативної та контентної фільтрації. Наприклад, враховуючи вік користувача та об'єкти, які йому подобаються, можна встановити більш вагому роль

колаборативної фільтрації для молодших користувачів та контентної фільтрації для старших.

Гібрид з розширенням функцій дозволяє отримати більш точні та персоналізовані рекомендації, враховуючи додаткові атрибути користувачів та об'єктів. Однак вимагає більшої кількості даних та більш складних алгоритмів машинного навчання для створення моделі.

Гібрид з комбінацією функцій (англ. feature-combination hybrid) - це підхід, в якому використовуються кілька алгоритмів, які обчислюють різні ознаки для предметів та користувачів. Наприклад, можна використовувати як колаборативний, так і контент-базовий підходи, але кожен з них буде використовуватися для виконання різних обчислень. Ознаки, які були обчислені кожним алгоритмом, об'єднуються, щоб створити остаточну рекомендацію.

Наприклад, якщо користувач зазвичай купує продукти з певної категорії веб-сайту, то контент-базовий алгоритм може бути використаний для знаходження подібних продуктів. З іншого боку, якщо користувач має високі рейтинги для фільмів з певного жанру, то колаборативний алгоритм може бути використаний для знаходження подібних фільмів. Ознаки, знайдені обома алгоритмами, об'єднуються для створення остаточної рекомендації.

Цей підхід дозволяє комбінувати переваги різних алгоритмів і знижує їхні недоліки, забезпечуючи більш точні та персоналізовані рекомендації для користувачів.

Метод змішаного гібриду включає в себе поєднання різних методів рекомендацій в одній системі. Цей метод може використовувати різні комбінації методів, включаючи як колаборативну, так і контент-базову фільтрацію, а також залучати врахування додаткової інформації, такої як демографічні дані або контекстуальні дані.

Наприклад, можна поєднати колаборативну та контент-базову фільтрацію, додавши до колаборативної фільтрації інформацію про зміст товарів. Також можна включити фактори додаткової інформації, наприклад, рекомендувати товари в залежності від місця знаходження користувача або пори доби.

Змішаний гібрид дозволяє отримувати більш точні рекомендації, оскільки враховує більш широкий спектр даних про користувачів і товари. Однак, враховуючи більшу кількість факторів, цей метод може бути складним для реалізації та вимагати значних обчислювальних ресурсів.

Каскадний гібридний метод поєднує кілька різних алгоритмів у послідовних етапах рекомендації. На кожному етапі, система використовує один алгоритм для виділення набору ітемів, які вважає потенційно привабливими для користувача.

Наприклад, на першому етапі система може використовувати контентно-базовий алгоритм, щоб знайти ітеми, які відповідають інтересам користувача. На другому етапі, коли вже відібрано деякі пропозиції, система може використовувати колаборативний алгоритм, щоб підібрати ітеми, які співпадають зі смаком користувача на основі даних з його рейтингів і відгуків.

Цей підхід забезпечує більш точні рекомендації, оскільки він поєднує в собі переваги різних методів і зменшує їхні недоліки. Однак, він може бути більш складним в реалізації, оскільки потрібно правильно настроїти кожен етап і забезпечити, щоб рекомендації на одному етапі не перекривалися з рекомендаціями на наступному етапі.

При підході перемикаючого гібрида, система може використовувати різні підходи до рекомендацій в залежності від обставин. Наприклад, якщо користувач використовує систему вперше, то відображені йому рекомендації можуть бути засновані на контенті, такі як різноманітні категорії товарів. Якщо ж користувач уже здійснив деякі покупки, то система може переключитися на колаборативний підхід і рекомендувати товари, що сподобалися іншим користувачам з подібними інтересами. Варто зазначити, що перехід між різними підходами зазвичай залежить від того, наскільки повна та досить інформація про користувача та його дії в системі [16].

Такий гібридний підхід може забезпечити якісні рекомендації для різних користувачів та у різних ситуаціях. Однак, також існує ризик того, що занадто часті переходи між різними підходами можуть призвести до зниження точності та якості рекомендацій. Тому важливо збалансувати перехід між різними підходами,

використовуючи, наприклад, певний поріг певної кількості інформації про користувача, щоб перейти до колаборативного підходу.

Зважений гібрид (Weighted Hybrid) - це гібридний метод рекомендацій, який поєднує кілька алгоритмів рекомендацій в один шляхом зважування їх внеску. У цьому підході кожен алгоритм має своє вагове значення, яке визначає його вплив на кінцевий результат.

Для визначення вагових коефіцієнтів можна використовувати різні підходи, такі як експертна оцінка, кількісні метрики або машинне навчання. Наприклад, експерти можуть оцінювати кожен алгоритм на шкалі від 1 до 10, і його ваговий коефіцієнт буде відповідати цій оцінці. У кількісних метриках вагові коефіцієнти можуть визначатися за допомогою крос-валідації, аналізу кореляцій або інших методів оцінки ефективності алгоритмів.

Зважений гібрид може бути використаний для покращення точності рекомендацій в порівнянні з одиночними алгоритмами рекомендацій. Наприклад, якщо один алгоритм рекомендацій добре вирішує проблему холодного старту, а інший - працює краще з персоналізованими рекомендаціями, зважений гібрид може поєднувати ці два алгоритми і отримувати більш якісні рекомендації. Однак, визначення правильних вагових коефіцієнтів може бути складним завданням і вимагати експертної оцінки або налагодження методу машинного навчання.

3 ПРОБЛЕМИ РЕКОМЕНДАЦІЙНИХ СИСТЕМ

3.1 Проблема холодного старту

Проблема старту з холодного підходу виникає, коли рекомендаційна система не може зробити висновок з існуючих даних, які є недостатніми. "Холодний старт" відноситься до стану, коли система не може продукувати ефективні рекомендації для "холодних" (або нових) користувачів, які не оцінили жоден елемент або оцінили дуже малу кількість елементів. Це, як правило, виникає, коли новий користувач увійшов до системи або нові елементи (або продукти) були вставлені до бази даних. Наразі є деякі рішення для цієї проблеми такі як:

– просити нових користувачів явно зазначити свої переваги стосовно елементів;

– просити нового користувача оцінити деякі елементи на початку;

– збирати демографічну інформацію (або метадані) від користувача та рекомендувати елементи відповідно.

Розглянемо проблему атаки на шилінг.

3.2 Проблема атаки на шилінг

Ця проблема виникає, коли зловживаючий користувач підробляє свою ідентичність і входить в систему, щоб дати фальшиві оцінки елементам. Така ситуація виникає, коли зловживаючий користувач хоче збільшити або зменшити популярність деяких елементів, спричинивши упередження до обраних цільових елементів. Атаки зловживанням значно знижують надійність системи. Один з варіантів вирішення цієї проблеми полягає у швидкому виявленні зловживаючих користувачів та видаленні фальшивих оцінок і фальшивих профілів користувачів з системи.

3.3 Проблема синонімії

Ця проблема виникає, коли схожі або пов'язані предмети мають різні записи або назви, або коли той самий предмет представлений двома або більше назвами в системі [78]. Наприклад, дитячий одяг та дитячі тканини. Багато систем рекомендацій не вміють розрізняти ці відмінності, що знижує їхню точність рекомендацій. Для полегшення цієї проблеми використовуються різні методи, такі як демографічна фільтрація, автоматичне розширення термінів та розклад Singular Value Decomposition.

3.4 Проблема затримки

Проблема затримки виникає у випадках колаборативної фільтрації і трапляється, коли в базу даних часто додаються нові елементи. Ця проблема характеризується невдачею системи рекомендацій при рекомендації нових елементів. Це стається тому, що нові елементи мають бути пройдені перед тим, як їх можна рекомендувати у середовищі колаборативної фільтрації. Використання фільтрації на основі контенту може вирішити цю проблему, але це може призвести до перенасичення та зниження часу обчислень та продуктивності системи. Для покращення продуктивності, обчислення можуть бути виконані в офлайн середовищі та використовувати техніки, засновані на кластеризації.

3.5 Проблема розрідженості

Розрідженість даних є поширеною проблемою у аналізі великого масштабу даних, яка виникає, коли в наборі даних відсутні деякі очікувані значення. У випадку систем рекомендацій ця ситуація виникає, коли активні користувачі оцінюють дуже мало предметів. Це знижує точність рекомендацій. Для полегшення

цієї проблеми можна використовувати кілька технік, таких як демографічна фільтрація, сингулярний розклад значень та використання модельних колаборативних технік.

3.6 Проблема масштабованості.

Рекомендаційні системи, особливо ті, які використовують техніки колаборативної фільтрації, потребують великої кількості тренувальних даних, що призводить до проблем з масштабованістю. Проблема масштабованості виникає, коли кількість даних, які використовуються в рекомендаційній системі, швидко збільшується. У цей епоху великих даних все більше і більше елементів та користувачів швидко додаються до системи, і ця проблема стає загальною в рекомендаційних системах. Для вирішення проблеми масштабованості використовують два поширені підходи: зменшення розмірності та використання технік на основі кластеризації для пошуку користувачів в невеликих кластерах замість повної бази даних [17].

4 ОПИС ВИКОРИСТАНИХ ТЕХНОЛОГІЙ ТА АЛГОРИТМІВ

4.1 Опис використаної мови програмування.

У ході виконання практичної частини було обрано мову програмування Python. Python - це інтерпретована мова програмування високого рівня зі строгою динамічною типізацією. Це означає, що вам не потрібно компілювати код перед його виконанням, і типи змінних перевіряються під час виконання програми. Python підтримує об'єктно-орієнтоване, процедурне та функціональне програмування, і має багатий стандартний бібліотеку.

Python має простий та зрозумілий синтаксис, який зробив його дуже популярним в академічних та індустріальних середовищах. Це також дозволяє швидше створювати програми порівняно з іншими мовами програмування, тому що менше коду потрібно для виконання тих самих завдань.

Python підтримує різні парадигми програмування, такі як структурне, об'єктно-орієнтоване та функціональне програмування, що робить його гнучким та легко розширюваним.

Python використовується для розробки веб-додатків, наукових обчислень, машинного навчання, розробки ігор та багатьох інших завдань. Він має велику і активну спільноту розробників, яка постійно розширює бібліотеки та модулі, доступні для використання.

У загальному, Python є потужною мовою програмування, яка має багато переваг, таких як простий та зрозумілий синтаксис, широку функціональність та активну спільноту розробників.

Так як Python широко використовується у напрямку «Data Sience», та має багато вбудованих методів для обробки великих данних, він ідеально підходить для реалізації колаборативної фільтрації.

4.2 Опис середовища розробки.

Середовищем розробки було обрано Azure Databricks. Azure Databricks - це хмарна платформа для аналізу даних, яка дозволяє ефективно працювати з великими обсягами даних та виконувати аналіз даних у режимі реального часу. Вона забезпечує доступ до розширеної функціональності Apache Spark та інтегрується з різними інструментами Azure, такими як Azure Storage, Azure Data Lake та інші. Ось деякі з переваг Azure Databricks:

– швидкість та продуктивність: Azure Databricks забезпечує швидку обробку даних завдяки використанню Apache Spark, який підтримує паралельну обробку даних на кластерах з високими потужностями. Це дозволяє ефективно працювати з великими обсягами даних та виконувати аналіз даних у режимі реального часу;

– масштабованість: Azure Databricks легко масштабується в залежності від потреб користувачів. Вона підтримує автоматичне масштабування для вирішення збільшення обсягів даних та навантаження на систему;

– гнучкість: Azure Databricks підтримує різні мови програмування, такі як Python, R та Scala, що дозволяє розробникам використовувати мову програмування за власним вибором та зручно працювати з даними;

– інтеграція з Azure: Azure Databricks інтегрується з різними інструментами Azure, такими як Azure Storage, Azure Data Lake та інші. Це дозволяє легко і ефективно зберігати та оброблювати дані на хмарній платформі Microsoft Azure;

– захист даних: Azure Databricks забезпечує високий рівень безпеки для даних, що зберігаються та оброблюються на платформі;

– спільна робота: Azure Databricks дозволяє розробникам працювати в команді над проектами та спільно редагувати код. Це забезпечує більшу ефективність та швидкість розробки;

– візуалізація даних: Azure Databricks підтримує різні інструменти візуалізації даних, такі як Matplotlib та Plotly, що дозволяє розробникам візуалізувати дані та спростити процес аналізу даних;

– надійність: Azure Databricks забезпечує високий рівень надійності та стійкості, завдяки розподіленому архітектурному підходу та підтримці мультирегіональної реплікації даних;

– відкритість: Azure Databricks є відкритою платформою, що дозволяє розробникам використовувати власні бібліотеки та інструменти для аналізу даних;

– економія коштів: Azure Databricks дозволяє економити кошти завдяки оптимізованому використанню ресурсів та ефективній обробці даних. Вона також підтримує підписки на різні плани, що дозволяє розробникам вибрати план, який відповідає їх потребам та бюджету;

Всі ці позитивні фактори, роблять Azure Databricks ідеальним середовищем для розробки колаборативної фільтрації.

4.3 Опис методу нормалізації оцінок

Для розробки колаборативної фільтрації буде використано дата сет з певним набором оцінок користувачів, але так як всі користувачі різні за своїми вподобаннями та метриками оцінювання того чи іншого объекту нам треба перейти до більш чіткого діапазону, в нашому випадку це [-1,1]. Ще це потрібно для здатності використати методи подібності, які будуть наведені нижче.

У цьому нам допоможе Мін-Макс шкалювання (min-max scaling). Мін-Макс шкалювання (min-max scaling) - це метод нормалізації даних, що використовується для перетворення значень ознак таким чином, щоб вони були в межах заданого діапазону (наприклад, [0, 1]). Цей метод працює шляхом перетворення значень ознак на новий діапазон значень за допомогою формули:

$$X\_norm = (X - X\_min) / (X\_max - X\_min) \quad (4.1)$$

де X - оригінальне значення ознаки;

X_min - мінімальне значення ознаки;

X_max - максимальне значення ознаки.

Таким чином, після шкалювання всі значення ознак будуть в межах [0, 1]. Цей метод допомагає унормалізувати значення ознак, що мають різний масштаб, і зробити їх порівнянними. Min-max шкалювання може бути корисним при побудові моделей машинного навчання, оскільки багато алгоритмів працюють краще, коли дані нормалізовані.

4.4 Методи подібності

Методи подібності - це підходи до визначення ступеня схожості між об'єктами у великих наборах даних. В контексті машинного навчання та аналізу даних, методи подібності використовуються для розрахунку відстані або коефіцієнта подібності між об'єктами, що можуть бути відображені у вигляді числових значень.

До одних з найпопулярніших методів подібності можна віднести такі як, косинусна подібність, кореляційна подібність(Кореляція Пірсона), евклідова відстань.

4.4.1 Косинусна подібність

Косинусна подібність - це метод обчислення схожості між двома векторами, який використовується для порівняння текстових даних, рекомендацій, зображень, тощо.

Косинусна подібність вимірює кут між векторами, представляючи їх як точки в n-вимірному просторі, і повертає косинус цього кута. Цей косинус буде значенням подібності між векторами: чим ближче значення до 1, тим більш вони схожі, а чим ближче до 0, тим менш вони схожі.

Формула для обчислення косинусної подібності між двома векторами a і b може бути записана так:

$$\cos(\vec{a}, \vec{b}) = \frac{\vec{a}.\vec{b}}{\|\vec{a}\|^2 * \|\vec{b}\|^2} \quad (4.2)$$

де a * b позначає добуток скалярний добуток векторів a і b;

||a|| та ||b|| позначають їхні норми (довжини) відповідно.

В Python цю формулу можна реалізувати таким чином:

```
from numpy import dot
from numpy.linalg import norm
def cosine_similarity(a, b):
    return dot(a, b) / (norm(a) * norm(b))
```

де dot використовується для обчислення скалярного добутку;

norm - для обчислення норми вектора.

4.4.2 Кореляція Пірсона

Кореляція Пірсона - це міра лінійної залежності між двома змінними. Ця міра вказує на те, наскільки сильно дві змінні корелюють між собою, тобто наскільки сильно вони пов'язані в лінійний спосіб. Кореляція Пірсона відноситься до діапазону мір кореляції, де значення знаходяться в інтервалі від -1 до 1, де значення +1 вказує на дуже сильну позитивну кореляцію, -1 - на дуже сильну негативну кореляцію, а 0 - на відсутність кореляції.

Формула кореляції Пірсона має наступний вигляд:

$$s(u,v) = \frac{\sum_{i \in I_{uv}} (r_{ui} - \bar{r}_u)(r_{ui} - \bar{r}_u)}{\sqrt{\sum_{i \in I_{uv}} (r_{ui} - \bar{r}_u)^2 \sum_{i \in \mathcal{L}_{uv}} (r_{vi} - \bar{r}_u)^2}} \quad (4.3)$$

Дана формула відображає подібність між користувачами u та v, де s(u, v) показує ступінь їх подібності. Набір елементів, що були оцінені користувачами u та v, позначається як I_{uv}. r_{ui} та r_{vi} відповідають рейтингам, що користувачі u та v віддали елементу i, а $\overline{r}_u$ та $\overline{r}_v$ - середні оцінки користувачів u та v відповідно для всіх елементів, які вони оцінили.

4.4.3 Евклідова відстань

Евклідова відстань - це міра відстані між двома точками в n-вимірному просторі. Ця міра обчислюється за допомогою теореми Піфагора. Якщо ми розглядаємо дві точки у двовимірному просторі, то їх евклідова відстань обчислюється як довжина прямої, яка з'єднує ці точки.

Формула для обчислення евклідової відстані:

$$s(u,v) = \sqrt{\frac{\sum_{i \in u_v}(r_{ui} - r_{vi})^2}{|I_{uv}|}} \tag{4.4}$$

У формулі $s(u,v)$, яка використовується для визначення схожості між користувачами u та v, враховуються всі елементи, що зазвичай оцінюються.

5 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

5.1 Підготовка програмного оточення.

Для того щоб реалізувати колаборативну фільтрацію першою справою було створено ресурс групу, куди ми можемо додати всі необхідні компоненти. Створення ресурс групи вказано на рисунку 5.1.

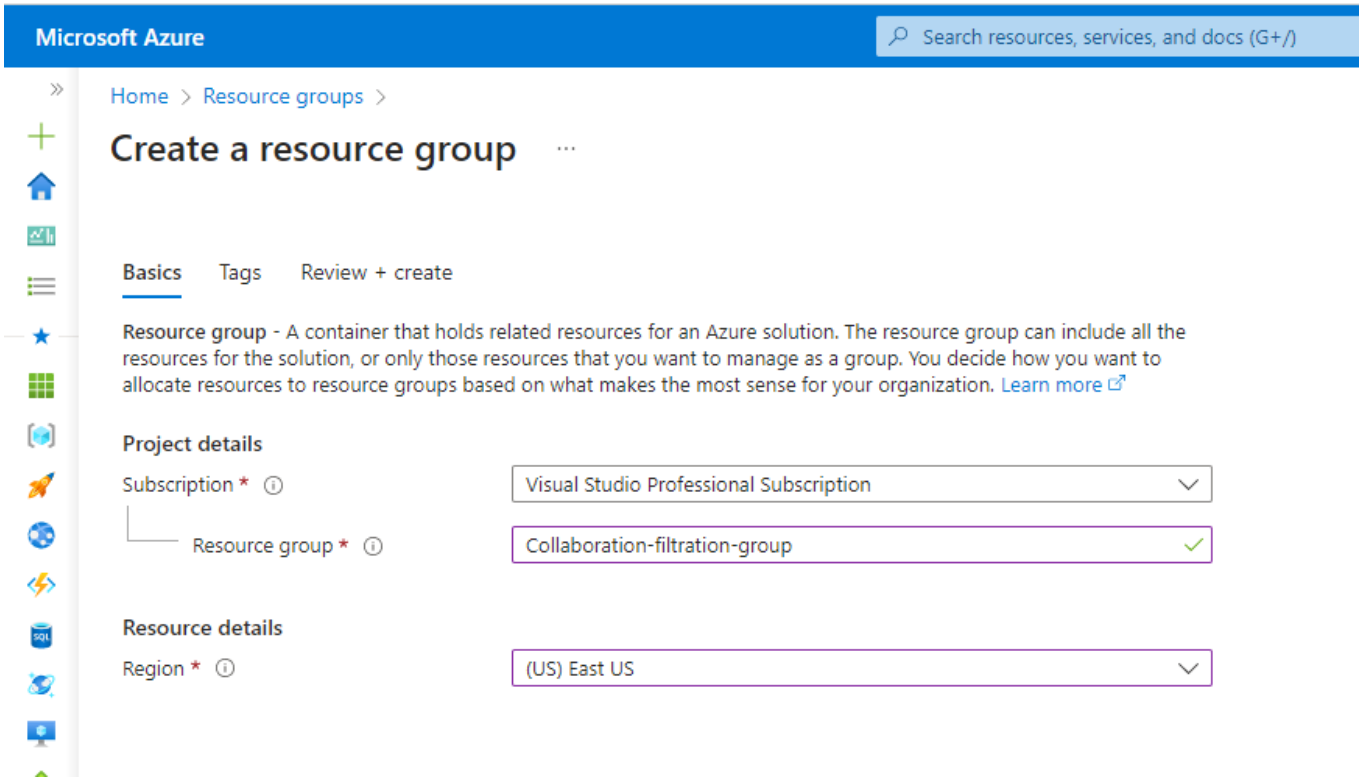

Рисунок 5.1 – Створення ресурсної групи

Група ресурсів була створенна з такими параметрами:

– підписка: «Visual Studio Professional Subscription»;

– назва групи ресурсів: «Collaboration-filtration-group»;

– регіон: «(US) East US».

Після створення ресурсної групи необхідно створити Azure Databricks сервіс, та налаштувати його. Для початку перейдемо в Marketplace та знайдемо там відповідний сервіс (див. рис. 5.2).

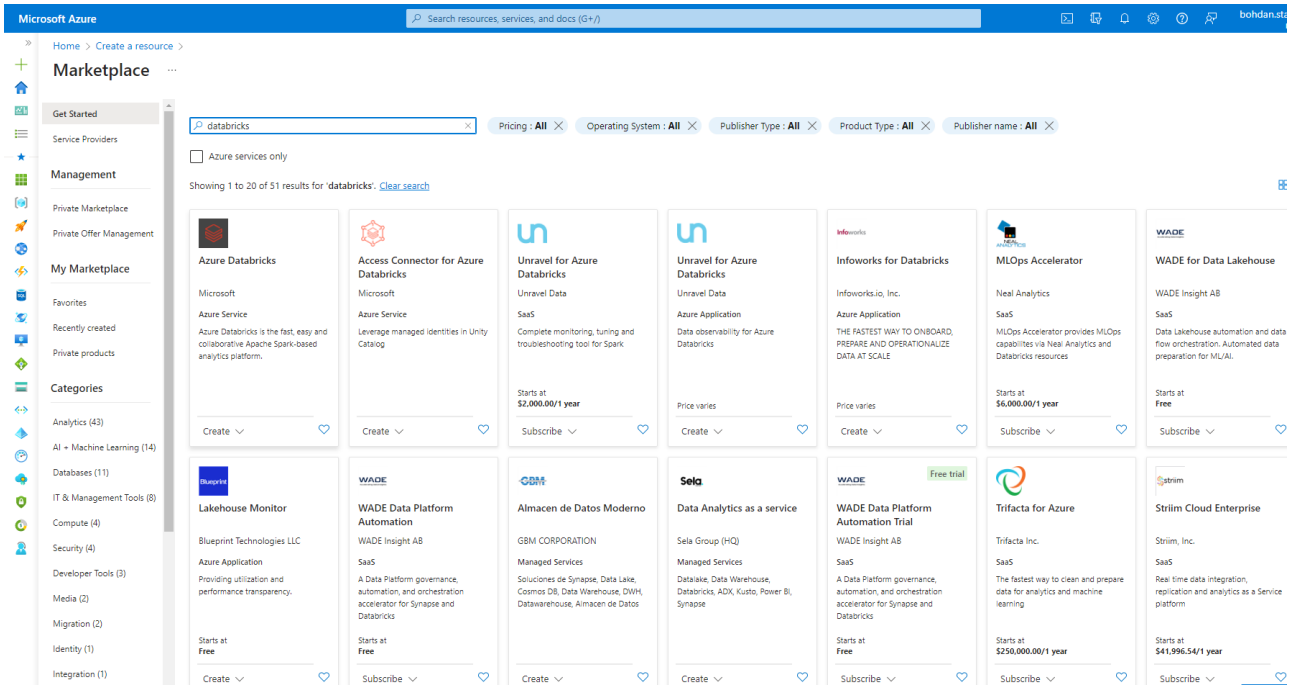

Рисунок 5.2 – Azure Databricks Service in Marketplace

Отже перейдемо до створення самого сервісу. Для цього було заповнено відповідні поля:

– підписка: «Visual Studio Professional Subscription»;

– назва групи ресурсів: «Collaboration-filtration-group»;

– назва робочого середовища: «Collaboration-filtration-workspace»;

– регіон: «(US) East US»;

– ціновий рівень: «Trial (Premium – 14-days Free DBUs)».

Налаштування сітки та шифрування було обрано за замовчуванням, так як ці налаштування не впливають на подальшу реалізацію (див. рис. 5.3).

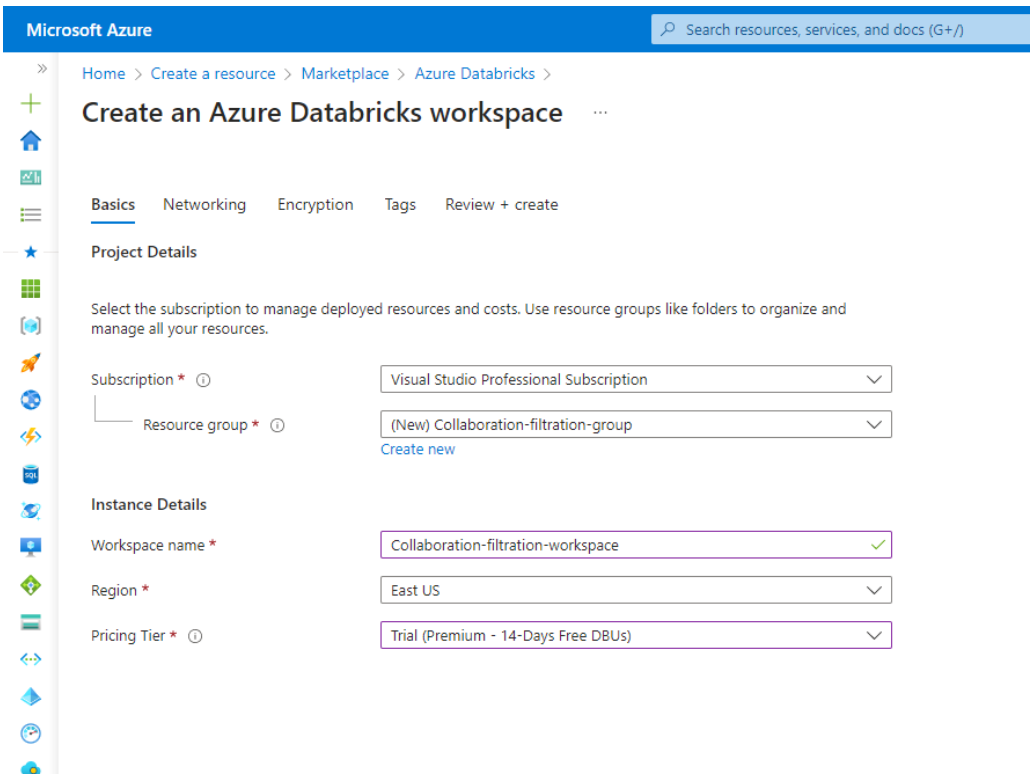

Рисунок 5.3 – Створення Azure Databricks Service

Далі вже в самому Azure Databricks Service, створимо кластер в якому подальше буде виконуватися робота. (див. рис 5.4).

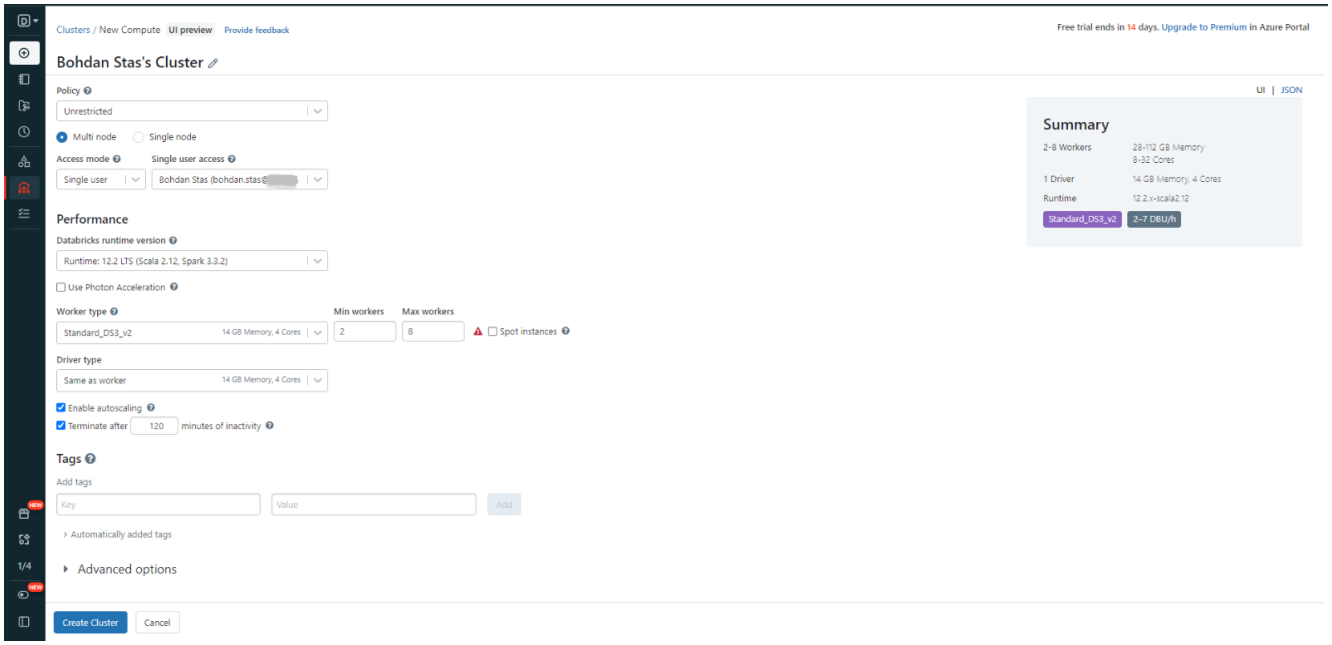

Рисунок 5.3 – Створення кластеру

Як можна побачити на рисунку кластер був створений з такими параметрами:

– правила: Unrestricted;
– режим доступу: Single user;
– однокористувацький доступ: Bohdan Stas;
– версія середовища виконання databricks: 12.2 LTS;
– тип системи: Standard_DS3_v2 14 GB memory, 4 Cores;
– тип драйверу: Same as worker;
– мінімум інстансів: 2;
– максимум інстансів 8;
– включити автомасштабування: Yes;
– вимкнути після 120 хв.

І нарешті після створення кластеру ми можемо створити ноутбук де буде виконуватися реалізація алгоритму для знаходження рекомендацій на основі колаборативної фільтрації.

5.2 Створення тестових даних для реалізації системи

Для перевірки функціональності системи було створено тестовий дата сет, який описує оцінки юзерів для різних фільмів, в даному випадку це бойовики та драми. Сам дата сет зберігається в csv файлі. Приклад тестового дата сету можно побачити на рисунку 5.4.

	A	B	C	D	E	F	G
1		action1	action2	action3	drama1	drama2	drama3
2	user 1	4	5	3		2	1
3	user 2	5	3	3	2	2	
4	user 3	1			4	5	4
5	user 4		2	1	4		3
6	user 5	4	4	4	4	4	4

Рисунок 5.4 – Тестовий дата сет

Як можна побачити в тестовому дата сеті не всі юзери поставили оцінку через те, що фільми не були переглянуті. Далі завантажимо дата сет до Azure Databricks. Для цього у вкладці «Data» оберемо секцію «Upload data» та завантажимо тестовий датасет .

5.3 Реалізація колаборативної фільтрації

Спочатку за допомогою бібліотеки pandas зчитаемо датасет з файлового сховища та вставимо неоціненим фільмам 0-вий рейтинг (див. рис. 5.5.).

```
1  import pandas as pd
2  from scipy import sparse
3  from sklearn.metrics.pairwise import cosine_similarity
4  ratings=pd.read_csv("/dbfs/FileStore/tables/test.csv",index_col=0)
5  ratings = ratings.fillna(0)
6  ratings
7
```

	action1	action2	action3	drama1	drama2	drama3
user 1	4.0	5.0	3.0	0.0	2.0	1.0
user 2	5.0	3.0	3.0	2.0	2.0	0.0
user 3	1.0	0.0	0.0	4.0	5.0	4.0
user 4	0.0	2.0	1.0	4.0	0.0	3.0
user 5	4.0	4.0	4.0	4.0	4.0	4.0

Рисунок 5.5 – Відображення тестового датасету

Наступним кроком буде нормалізація оцінок за допомогою min-max scaling методу. Нормалізація оцінок потрібна, щоб перевести усі оцінки до одного діапазону, та після використати один з методів подібності.

	action1	action2	action3	drama1	drama2	drama3
user 1	0.24	0.44	0.20	-0.7	-0.12	-0.35
user 2	0.44	0.04	0.20	-0.2	-0.12	-0.60
user 3	-0.36	-0.56	-0.55	0.3	0.48	0.40
user 4	-0.56	-0.16	-0.30	0.3	-0.52	0.15
user 5	0.24	0.24	0.45	0.3	0.28	0.40

Рисунок 5.6 – Результат після нормалізації оцінок

Далі використовуючі метод подібності косинусів отримаємо матрицю подібності фільму відповідно інших фільмів.

	action1	action2	action3	drama1	drama2	drama3
action1	1.000000	0.707412	0.856188	-0.593067	0.153806	-0.609404
action2	0.707412	1.000000	0.885881	-0.668424	-0.293359	-0.472200
action3	0.856188	0.885881	1.000000	-0.408248	-0.046829	-0.368514
drama1	-0.593067	-0.668424	-0.408248	1.000000	0.258093	0.800095
drama2	0.153806	-0.293359	-0.046829	0.258093	1.000000	0.480065
drama3	-0.609404	-0.472200	-0.368514	0.800095	0.480065	1.000000

Рисунок 5.7 – Матриця подібності

Як можна побачити на рисунку 5.7 у результаті виконання косинус подібності було відтворено матрицю подібності кожного фільму відносно інших фільмів. Значення яке ближче до одного вказує на те, що фільм є схожим та його можна використати для рекомендації, а значення які ближче до -1 навпаки вказують на різність.

Останнім кроком буде реалізація методу котрий виведе найбільш схожі фільми за вподобанням певного користувача. Метод бере, кожен фільм та його оцінку від конкретного користувача та вираховує коефіцієнт подібності по кожному фільму в матриці подібності. Для перевірки створимо тестового користувача якому більше подобаються бойовики (див. рис. 5.8).

```
test_user = [("action1",5),("drama2",1),("drama3",1)]
```

Рисунок 5.8 – Тестовий юзер

Після розрахунку коефіцієнту подібності для тестового юзера ми отримали матрицю з коефіцієнтами де рядки – це ті фільми, що користувач подивився та надав оцінку, а колонки це всі фільми в датасеті (див. рис. 5.9).

	action1	action3	action2	drama2	drama1	drama3
0	2.500000	2.140469	1.768529	0.384514	-1.482668	-1.523510
1	-0.230709	0.070244	0.440039	-1.500000	-0.387139	-0.720098
2	0.914106	0.552771	0.708300	-0.720098	-1.200142	-1.500000

Рисунок 5.9 – Матриця коефіцієнтів для тестового юзера

І на останок, щоб зробити рекомендацію просто просумуємо коефіцієнт по кожному фільму та виведемо рекомендацію. Результат можно побачити на рисунку 5.10.

```
action1    3.183397
action2    2.916869
action3    2.763484
drama2    -1.835583
drama1    -3.069949
drama3    -3.743608
```

Рисунок 5.10 – Рекомендації для тестового юзера

Як можно бачити в результаті для тестового юзера ми спочатку отримаємо всі фильми з жанром «бойовик», а вже потім драма, так як користувач оцінював усі бойовики з високою оцінкою, а драми з низькою.

5.4 Перевірка реалізації на реальних даних

Для того щоб перевірити реалізацію алгоритму, використаємо датасет для рейтингу фільмив з серевісу kaggle (див. рис. 5.11).

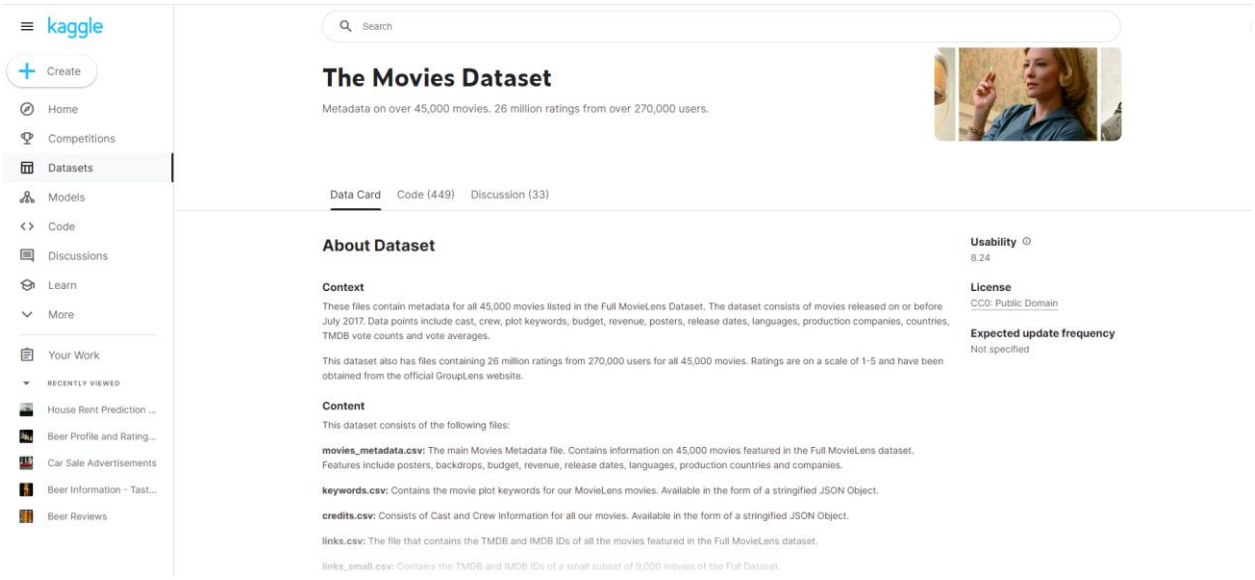

Рисунок 5.11 – Датасет з сервісу kaggle

Далі завантажимо датасет з інформацією про фільми та рейтинг від користувачів. Ці файли містять метадані для всіх 45 000 фільмів, перелічених у повному наборі даних MovieLens. Набір даних складається з фільмів, що були випущені у липні 2017 року або раніше. До даних входять актори, знімальна група, ключові моменти сюжету, бюджет, дохід, постери, дати випуску, мови, кінопродукційні компанії, країни, кількість голосів TMDB та середні голоси.

Також цей набір даних містить файли з 26 мільйонами рейтингів від 270 000 користувачів для всіх 45 000 фільмів. Рейтинги мають шкалу від 1 до 5 і були отримані з офіційного веб-сайту GroupLens.

Для перевірки роботи створимо користувача з відповідним рейтингом фільмів:

– Amazing Spider-Man, The (2012) - рейтинг 5;

– Mission: Impossible III (2006) - рейтинг 4;

– 2 Fast 2 Furious (Fast and the Furious 2, The) (2003) – рейтинг 4;

– "Toy Story 3 (2010)" – рейтинг 2.

Після виконання алгоритму пошуку рекомендації система надала такий список (топ 10 рекомендацій):

– Over the Hedge (2006) – коефіцієнт: 2.229721

– Crank (2006) – коефіцієнт: 2.176259

– Mission: Impossible - Ghost Protocol (2011) – коефіцієнт: 2.159666
– Hancock (2008) – коефіцієнт: 2.156098
– The Amazing Spider-Man 2 (2014) – коефіцієнт: 2.153677
– Hellboy (2004) – коефіцієнт: 2.137518
– Snakes on a Plane (2006) – коефіцієнт: 2.137396
– Jumper (2008) – коефіцієнт: 2.129716
– Chronicles of Riddick, The (2004) – коефіцієнт: 2.121689
– Tron: Legacy (2010) – коефіцієнт: 2.111843

Як можна побачити рекомендації дійсно мають сенс, так як всі фільми дуже схожі за жанром та контентом з тих, що було найвище оцінено користувачем.

ВИСНОВОК

На сьогоднішній день, величезна кількість топових компаній не бачить своє подальше існування без рекомендаційної системи, так як це допомагає налагодити більш гарний контакт з клієнтами, котрі використовують їх сервісу для пошуку розваг, покупок тощо.

Було розглянуто безліч методів, які використовуються для пошуку правильних рекомендацій користувачу. Всі вони різні та мають певні позитивні та негативні якості.

Було проаналізовано актуальність задачі формування рекомендацій товарів, виявлено проблеми та можливі напрямки досліджень в цій галузі. Було розглянуто особливості рекомендаційних систем, їхні переваги для впровадження в бізнесі. Також було проаналізовано існуючі підходи до формування рекомендацій. На підставі розглянутих підходів можна виділити 3 основних напрямки:

– рекомендаційні системи на основі вмісту;

– рекомендаційні системи на основі колаборативної фільтрації;

– гібридні системи, які поєднують 2 попередні підходи. Було сформовано постановку завдання.

Було реалізовано підхід колаборативної фільтрації в сервісі Azure Databricks, та зроблений аналіз на реальних даних рейтингів фільмів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Netflix – Вікіпедія / URL: https://uk.wikipedia.org/wiki/Netflix (дата звернення: 06.04.2023);

2. Zhao Y., Zhang B. Research on Recommendation Algorithms Based on Collaborative Filtering. Journal of Physics: Conference Series. 2019. Т. 1237. С. 022094. URL: https://doi.org/10.1088/1742-6596/1237/2/022094 (дата звернення: 13.04.2023).

3. Chalyi, S., Leshchynskyi, V., Leshchynska, I. V. DETAILING EXPLANATIONS IN THE RECOMMENDER SYSTEM BASED ON MATCHING TEMPORAL KNOWLEDGE Eastern-European Journal of Enterprise Technologies, 2020, 4(2-106), стр. 6–13

4. Chalyi, S., Leshchynskyi, V., Leshchynska, I. Method of forming recommendations using temporal constraints in a situation of cyclic cold start of the recommender system Chalyi, S., Leshchynskyi, V., Leshchynska, I. EUREKA, Physics and Engineering, 2019, 2019(4), с. 34-40

5. Rafter R. Evaluation and Conversation in Collaborative Filtering. College of Engineering Mathematical and Physical Sciences. 2010. С. 15.

6. Resnick P., Iacovou N., Suchak M. GroupLens: an open architecture for collaborative filtering of netnews. Masloriy Kikabidze – College of Engineering Mathematical and Physical Sciences. 1994. С. 219.

7. A. Rutkas, Vlasenko, L.A. Optimal control of undamped Sobolev-type retarded systems Mathematical Notes, vol. 102 (3-4), P. 297-309

8. Recommendation Systems – How Companies are Making Money URL: https://sigmoidal.io/recommender-systems-recommendation-engine/ (дата звернення: 17.03.2023) .

9. The ACM Conference Series on Recommender Systems / URL: https://recsys.acm.org/ (дата звернення: 19.03.2023) .

10. Montaner, M., López, B., de la Rosa, J.L.: A taxonomy of recommender agents on the internet. Artificial Intelligence Review 19(4), 285–330 (2003) (дата звернення 29.04.2023) .

11. Query Dependent Ranking Using K-Nearest Neighbor / URL: https://www.andrewoarnold.com/fp025-geng.pdf (дата звернення: 29.04.2023) .

12. Artificial Intelligence in Retail – 10 Present and Future Use Cases / URL: https://emerj.com/ai-sector-overviews/artificial-intelligence-retail/ (дата звернення: 01.05.2023) .

13. O'Connor M. PolyLens: A Recommender System for Groups of Users / URL: http://files.grouplens.org/papers/poly-camera-final.pdf (дата звернення: 01.05.2023) .

14. Large-Scale Machine Learning with Stochastic Gradient Descent / URL: https://leon.bottou.org/publications/pdf/compstat-2010.pdf (дата звернення: 01.05.2023) .

15. Collaborative Filtering for Implicit Feedback Datasets / URL: http://yifanhu.net/PUB/cf.pdf (Last accessed: 01.05.2023) .

16. Understanding TF-IDF for Machine Learning. / URL: https://www.capitalone.com/tech/machine-learning/understanding-tf-idf (дата звернення: 02.05.2023)

17. Azure DataBricks service / URL: https://learn.microsoft.com/en-us/azure/databricks/introduction/ (дата звернення: 25.04.2023).

18. Movie Dataset Kaggle / URL: https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset (дата звернення: 03.05.2023).