Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет *Комп'ютерних наук*
(повна назва)
Кафедра *Системотехніки*
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Дослідження методів кластеризації даних для реалізації рекомендаційної функції CRM-системи мережі кінотеатрів

(тема)

Виконала:

Студентка 2 курсу, групи *СПРм-22-1*

Одинцова В.О.
(прізвище, ініціали)

Спеціальність *122 – Комп'ютерні науки*
(код і повна назва напряму)

Тип програми *освітньо-наукова*
(освітньо-професійна або освітньо-наукова)

Освітня програма *Системне проектуваннч*
(повна назва освітньої програми)

Керівник доцент каф. *Коваленко А.І*
(прізвище, ініціали)

Допускається до захисту
Зав. Кафедри СТ

_____ *Гребеннік І. В.*
(підпис) (прізвище, ініціали)

2024 р.

Я, як студент ХНУРЕ, розумію і підтримую політику закладу із академічної доброчесності. Я не надавала і не одержувала недозволену допомогу під час підготовки кваліфікаційної роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

20.06.2024 Одинцова В.О. 

Атестаційна робота не містить відомостей заборонених до відкритого опублікування.

Атестаційна робота виконана у відповідності до стандартів, що діють в Україні.

Попередній захист проведений «18» грудня 2020 р.

Керівник атестаційної роботи доц. Коваленко А.І.

Харківський національний університет радіоелектроніки

Факультет *Комп'ютерних наук*
(повна назва)
Кафедра *Системотехніки*
(повна назва)
Рівень вищої освіти другий (магістерський)
Спеціальність *122 – Комп'ютерні науки*
(код і повна назва)
Тип програми *освітньо-наукова*
(освітньо-професійна або освітньо-наукова)
Освітня програма *Системне проектування*
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри *СТ проф. Гребеннік І.В.*

(підпис)

" _____ " _____ *2022 р.*

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

Студентці Одинцовій Вікторії Олександрівні
(прізвище, ім'я, по батькові)

1. Тема роботи *Дослідження методів кластеризації даних для реалізації рекомендаційної функції CRM-системи мережі кінотеатрів*

затверджена наказом по університету від « 1 » квітня 2024 р. № 259 Ст

2. Термін подання студентом роботи (проекту) 20 червня 2024 р.

3. **Вихідні дані до роботи:** Провести дослідження методів кластеризації k-середніх, агломеративного та DBSCAN для реалізації рекомендаційної функції у системах перегляду фільмів. Перелік використовуваних програмних засобів: MySQL Server. Клієнтська частина повинна являти собою веб-інтерфейс доступу до БД з використанням технології Django. Операційна система Windows 11, середовище розробки Anaconda, мова програмування Python

4. **Перелік питань, що потрібно опрацювати в роботі**: 4.1 Вступ 4.2 Вибір предметної області 4.3 Аналіз сучасного стану реалізації рекомендаційних функцій в системах перегляду фільмів 4.4 Постановка задачі 4.5 Теоретичні підходи (методи) для визначення рекомендацій в системах перегляду фільмів 4.6 Метод визначення рекомендацій за спільною фільтрацією (Collaborative Filtering) 4.7 Метод кластеризації k-середніх 4.8 Ієрархічні методи кластеризації «згори-вниз» та «знизу-вгору» 4.9 Метод кластеризації DBSCAN 4.10 Порівняння методів кластеризації 4.11 Визначення послідовності дій для дослідження методів кластеризації даних для реалізації рекомендаційної функції мережової системи кінотеатрів 4.12 Опис атрибутів таблиць використаних

для дослідження методів кластеризаццї для реалізації рекомендаційної функції мережової системи кінотеатрів 4.13 Дослідження та підготовка даних до кластеризації 4.14 Дослідження методу кластеризації k-середніх для реалізації рекомендаційної функції мережової системи кінотеатрів 4.15 Дослідження ієрархічного методу кластеризації «згори-вниз» для реалізації рекомендаційної функції мережової системи кінотеатрів 4.16 Дослідження ієрархічного методу кластеризації DBSCAN для реалізації рекомендаційної функції мережової системи кінотеатрів 4.17 Порівняння результатів дослідження методів кластеризації для реалізації рекомендаційної функції мережової системи кінотеатрів 4.18 Реалізація рекомендаційної функції мережової системи кінотеатрів за допомогою методу DBSCAN та порівняння результатів роботи функції за k-середніх, агломеративним та DBSCAN методів 4.19 Висновки 4.20 Перелів джерел посилання

5. **Перелік графічного матеріалу із зазначення креслеників, схем, плакатів, комп'ютерних ілюстрацій:** 1. Варіант класифікації рекомендаційних методів 2. Розподі алгоритмів кластеризації за типами 3. Схематична робота алгоритму «знизу-вгору» 4. Структура послідовності дій дослідження методів кластеризації даних для реалізації рекомендаційної функції мережевої системи кінотеатрів

6. Консультанти розділів роботи (проекту)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Спец. частина	Доц. Коваленко А.І.		05.05.2024

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1.	Отримання завдання на виконання роботи	01.04.2024	Виконано
2.	Аналіз предметної області	02.04.2024-15.04.2024	Виконано
3.	Огляд літератури	16.04.2024-20.04.2024	Виконано
4.	Аналіз методіві та підходів	21.04.2024-01.05.2024	Виконано
5.	Дослідження варіантів реалізації методів	02.05.2024-17.05.2024	Виконано
6.	Формування висновків	18.05.2024-24.05.2024	Виконано
7.	Оформлення пояснювальної записки	25.05.2024-03.06.2024	Виконано
8.	Представлення на рецензування	06.06.2024	Виконано
9.	Підготовка презентації до роботи	07.06.2024-19.06.2024	Виконано
10.	Захист роботи	20.06.2024	Виконано

Дата видачі завдання *22 березня 2024 р.*

Студент _____ *Одинцова В.О.*
(підпис)

Керівник роботи _____ доцент каф. *Коваленко А.І.*
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Кваліфікаційна робота: 72 стор., 23 рис, 1 табл., 1 додаток, 22 джерел. Графічний матеріал атестаційної роботи містить 10 аркушів.

CRM-СИСТЕМА, CONTENT BASED ФІЛЬТРАЦІЯ, COLLABORATIVE ФІЛЬТРАЦІЯ, КЛАСТЕРИЗАЦІЯ, МЕТОД K-СЕРЕДНІХ, МЕТОД DBSCAN, АГЛОМЕРАТИВНИЙ МЕТОД, ANACONDA, MYSQL, PYTHON

Об’єкт дослідження – процес визначення уподобань фільмів клієнтами мережової системи кінотеатрів та подання їх рейтингу за прогнозом в якості рекомендації.

Предмет дослідження – інформаційні технології реалізації рекомендаційної функції CRM-системи мережі кінотеатрів за допомогою методів кластеризації.

Мета роботи – дослідження методів кластеризації даних для реалізації рекомендаційної функції CRM-системи мережи кінотеатрів.

Методи дослідження – системний підхід; рекомендаційні методи, що засновані на описі характеристик товару та даних користувачів (Content-Based); рекомендаційні методи, що засновані на сумісній (колаборативній) фільтрації (Collaborative Filtering), а саме методи кластеризації: метод k-середніх, агломеративний метод та DBSCAN.

У роботі розглянуті особливості практичної реалізації методу кластеризації DBSCAN у комбінації з Content-Based підходом. Для дослідження обрана й розглядається рекомендаційна CRM-система мережі кінотеатрів.

Область застосування – системи перегляду фільмів.

ABSTRACT

Qualification work: 72 p., 23 pic., 1 table, 22 source, , 1 applications. Graphic material attestation work contains 10 poster.

CRM SYSTEM, CONTENT BASED FILTERING, COLLABORATIVE FILTERING, CLUSTERIZATION, K-MEANS METHOD, DBSCAN METHOD, AGGLOMERATE METHOD, ANACONDA, MYSQL, PYTHON

The object of the study is the process of determining movie preferences by customers of the cinema network system and submitting their rating according to the forecast as a recommendation.

The subject of the study is information technologies for the implementation of the recommendation function of the CRM system of the cinema network using clustering methods.

The purpose of the work is to study data clustering methods for the implementation of the recommendation function of the CRM system of the cinema network.

Research methods - systematic approach; recommendation methods based on the description of product characteristics and user data (Content-Based); recommendation methods based on Collaborative Filtering, namely clustering methods: k-means method, agglomerative method and DBSCAN.

The paper examines the features of the practical implementation of the DBSCAN clustering method in combination with the Content-Based approach. The recommendation CRM system of the cinema network is selected and considered for the study.

Field of application – movie viewing systems.

ЗМІСТ

ВСТУП

Необхідність координації діяльності мережі кінотеатрів, розподілених за районами міста, з продажів квитків на фільми за різними сеансами, визначає важливість використання систем управління взаємовідносинами з клієнтами (Customer Relationship Management System). Застосування CRM-систем забезпечує ефективну функцію видачі рекомендаційної інформації у вигляді рекламних пропозицій, що визначаються персональними уподобаннями клієнтів.

Аналіз існуючих рекомендаційних функцій дозволяє зробити висновок, що найбільш перспективними для реалізації рекомендаційної функції CRM-системи мережі кінотеатрів є два методи сумісної (колаборативної) фільтрації (Collaborative Filtering) – метод порівняння користувачів (User-Based) та метод порівняння елементів (Item-Based), у комбінації з фільтрацією на основі вмісту.

За методом порівняння користувачів визначається прогнозна оцінка рекомендованих фільмів, яка розраховується за мірою схожості уподобань інших клієнтів. За методом порівняння елементів також визначається прогнозна оцінка, яка розраховується на основі сумісної схожості оцінок фільмів. Для отримання даних за мірою схожості оцінок клієнтів або фільмів використовується методи кластеризації [1].

Таким чином, реалізація рекомендаційної функції забезпечить шляхи аналізу інформації щодо вподобань на фільми користувачів.

Усе розглянуте вище обумовлює актуальність теми кваліфікаційної роботи «Дослідження методів кластеризації даних для реалізації рекомендаційної функції мережової системи кінотеатрів».

Метою кваліфікаційної роботи є дослідження методів кластеризації даних для реалізації рекомендаційної функції CRM-системи мережи кінотеатрів.

Об’єктом дослідження є процес визначення уподобань фільмів клієнтами мережової CRM-системи кінотеатрів та подання їх рейтингу за прогнозом в якості рекомендації.

Предмет дослідження – інформаційні технології та методи кластеризації, які використовуються для реалізації рекомендаційних функції CRM-системи мережі кінотеатрів.

1 ВИБІР ТА АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ДОСЛІДЖЕНЬ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Вибір предметної області

У сучасному світі, де інформація є дуже цінним ресурсом, здатність швидко і ефективно обробляти великі обсяги даних стає критичною для успіху багатьох організацій. Одним з важливих напрямків, де використовується обробка даних, є системи рекомендацій, які допомагають користувачам вибирати продукти або послуги з великої кількості можливих варіантів.

Цей дипломний проект зосереджений на використанні методів кластеризації для реалізації рекомендаційної функції в CRM-системі мережі кінотеатрів. Кластеризація даних – це процес групування подібних об'єктів в одну групу, або кластер. В контексті рекомендаційних систем, це може означати групування користувачів з подібними інтересами або групування продуктів, які є подібними за деякими характеристиками.

У цьому дослідженні будуть розглянуті три методи кластеризації: K-середні, DBSCAN та агломеративна кластеризація. Ці методи були обрані через їхню популярність та ефективність в різних сценаріях застосування.

1.2 Аналіз сучасного стану реалізації рекомендаційних функцій в системах перегляду фільмів

Рекомендаційні системи стали ключовим компонентом багатьох онлайн-платформ, включаючи платформи для перегляду фільмів, такі як Netflix і Amazon Prime. Ці системи використовують різні алгоритми та методи, включаючи кластеризацію, для надання персоналізованих рекомендацій користувачам.

Netflix використовує різні методи машинного навчання для створення своєї рекомендаційної системи, включаючи кластеризацію. Вони використовують ці методи для аналізу великих обсягів даних про перегляди, оцінки та поведінку користувачів, щоб розробити персоналізовані рекомендації. [2] На рис. 1.1 подано скриншот інтерфейсу рекомендації фільмів від Netflix.

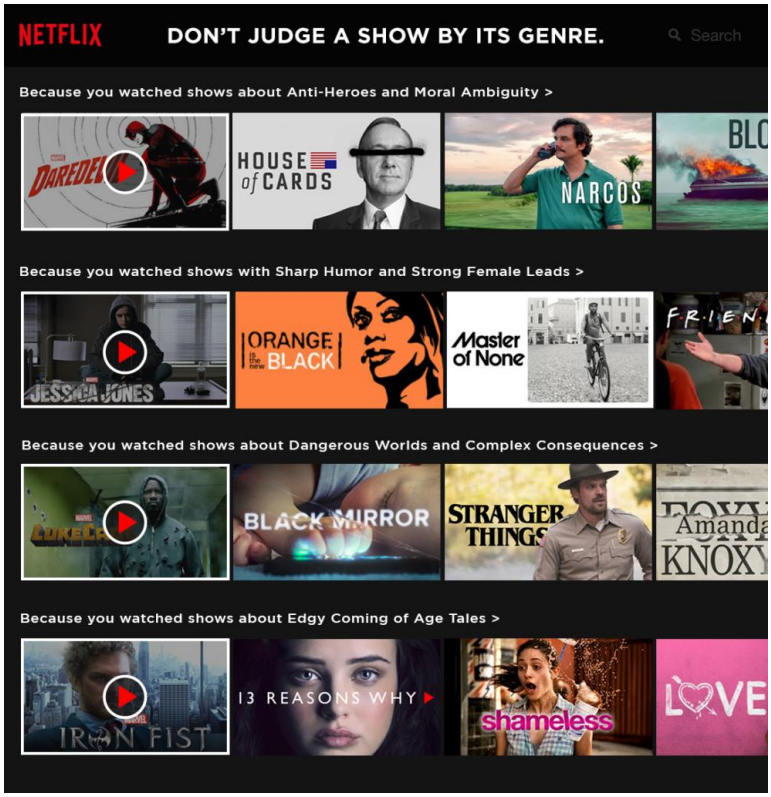

Рисунок 1.1 – Скриншот інтерфейсу рекомендації фільмів від Netflix

Amazon Prime також використовує рекомендаційні системи для підвищення задоволеності користувачів та підтримки високого рівня впровадження. Вони використовують методи, такі як колаборативна фільтрація та кластеризація, для визначення подібності між продуктами та користувачами. [3]

Статистика, яка приведена на рис. 1.1 чітко показує вплив впровадження систем рекомендацій на платформи Netflix та Amazon:

– Netflix: близько 75% того, що переглядають користувачі, надходить із рекомендацій.

– Amazon: 35% доходу Amazon генерується його системою рекомендацій.

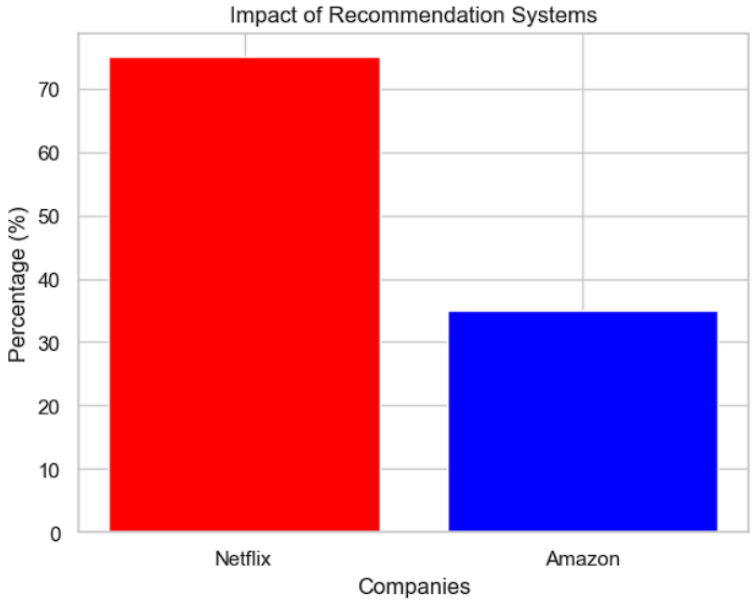

Рисунок 1.2 – Ріст доходу у % у Amazon та Netflix при впровадженні рекомендаційної функції

Обидва сервіси використовують кластеризацію як частину своїх рекомендаційних систем, що підтверджує важливість цього методу. Однак, є багато різних способів використання кластеризації в рекомендаційних системах, і важливо дослідити, які методи найкраще підходять для конкретного застосування, такого як CRM-система мережі кінотеатрів.

1.3 Постановка задачі дослідження методів кластеризації для реалізації рекомендаційної функції CRM-системи мережі кінотеатрів

За проведеним аналізом визначено такі завдання дослідження кваліфікаційної роботи:

– провести дослідження методів кластеризації для реалізації рекомендаційної функції CRM-системи мережі кінотеатрів;

– розробити рекомендаційну функцію CRM-системи мережі кінотеатрів за допомогою обраного методу кластеризації.

Для виконання поставлених завдань потрібно:

– визначити теоретичні підходи (методи) для визначення рекомендацій в системах мережі кінотеатрів;

– проаналізувати такі методи кластеризації: метод кластеризації k-середніх, ієрархічні методи кластеризації «згори-вниз» та «знизу-вгору» та метод кластеризації DBSCAN (Density-based spatial clustering of applications with noise);

– визначити послідовність дій для дослідження методів кластеризації даних для реалізації рекомендаційної функції мережевої системи кінотеатрів;

– провести дослідження метод кластеризації k-середніх, методів кластеризації «згори-вниз» та методу кластеризації DBSCAN (Density-based spatial clustering of applications with noise);

– провести порівняння результатів дослідження та обрати метод для реалізації рекомендаційної функції CRM-системі мережі кінотеатрів;

– реалізувати рекомендаційну функцію мережевої системи кінотеатрів за обраним методом кластеризації для рекомендаційної функції CRM-системі мережі кінотеатрів.

2 АНАЛІЗ МЕТОДІВ ДЛЯ ВИЗНАЧЕННЯ РЕКОМЕНДАЦІЙ У МЕРЕЖАХ ПЕРЕГЛЯДУ ФІЛЬМІВ

2.1 Теоретичні підходи (методи) для визначення рекомендацій в системах перегляду фільмів

Рекомендаційні системи надають інформацію у вигляді рекламних пропозицій, що визначаються персональними уподобаннями клієнтів. Даний підхід допомагає краще розуміти бажання клієнтів та провести аналіз, на основі якого можна скласти шляхи розвитку системи для покращення її роботи з метою збільшення прибутку від використання системи мережі кінотеатрів користувачами.

У системах електронної комерції мережі кінотеатрів, та у комерційних систем в цілому, налічується багато методів і теоретичних підходів, які використовують для рекомендації фільмів. Шляхи до рекомендацій щодо фільмів можна розділити на дві категорії: персоналізовані та неперсоналізовані.

Неперсоналізовані рекомендації – це рекомендації, які не враховують особистих характеристик, а виводять інформацію користувачам на основі загальних характеристик, наприклад: популярні фільми у прокаті, новинки у прокаті та інше. Такі рекомендації допомагають просунути нові чи старі фільми для користувачів, які переглядають фільми. Неперсоналізовані рекомендації не є найефективнішим шляхом персоналізації, як для користувачів так і для самої мережі кінотеатрів. Вони не надають можливості прогнозувати, які фільми є найбільш доречними для користувачів в залежності від їх вподобань.

Персоналізовані рекомендації – це рекомендації, які надаються користувачам на основі їхніх індивідуальних вподобань, інтересів і попередньої взаємодії з системою, які у свою чергу надають індивідуальний підхід до кожного користувача. Такі рекомендації можуть враховувати багато характеристик для створення

персоналізованого досвіду, наприклад історію покупок фільмів, час сеансів, оцінка фільмів та інші персональні характеристики, такі як стать, вік користувача. [4]

Для створення ефективної персоналізованої рекомендаційної функції для системи мережі кінотеатрів потрібно зібрати дані, які дозволять рекомендаційній системі розуміти і передбачувати вподобання та поведінку користувачів. Необхідні данні можна розділити на дві головні категорії:

– дані користувачів. Це характеристик про користувачів, така як вік, стать, місце проживання та їхня історія покупка білетів, час сеансів, оцінка фільмів та інше.

– дані фільмів та білетів. Це описи об'єктів –назва, опис, жанр фільму, актори, режисери та інформація щодо білетів як сеанс, час та день тижня. А також оцінки та відгуки користувачів про фільми;

Система рекомендацій класифікується за методом отримання даних:

– за пам'яттю (Memory-Based) – використовується інформація, що зберігається в базі даних – це особисті дані клієнтів (здебільшого демографічні) і дані їх історії замовлень білетів;

– за моделлю (Model-Based) – використовується модель, що визначає оцінки фільмів, що надані користувачами.

– гібридний підхід, коли використовується методи отримання даних за пам'яттю (Memory-Based) і за моделлю (Model-Based).

Рекомендаційні методи, можна поділити на чотири види:

– на основі контенту (Content Based);

– на основі знань (Knowledge Based);

– за спільною фільтрацією (Collaborative Filtering);

– гібридні (Hybrid);

На рис. 2.1 представлено варіант класифікації рекомендаційних методів.

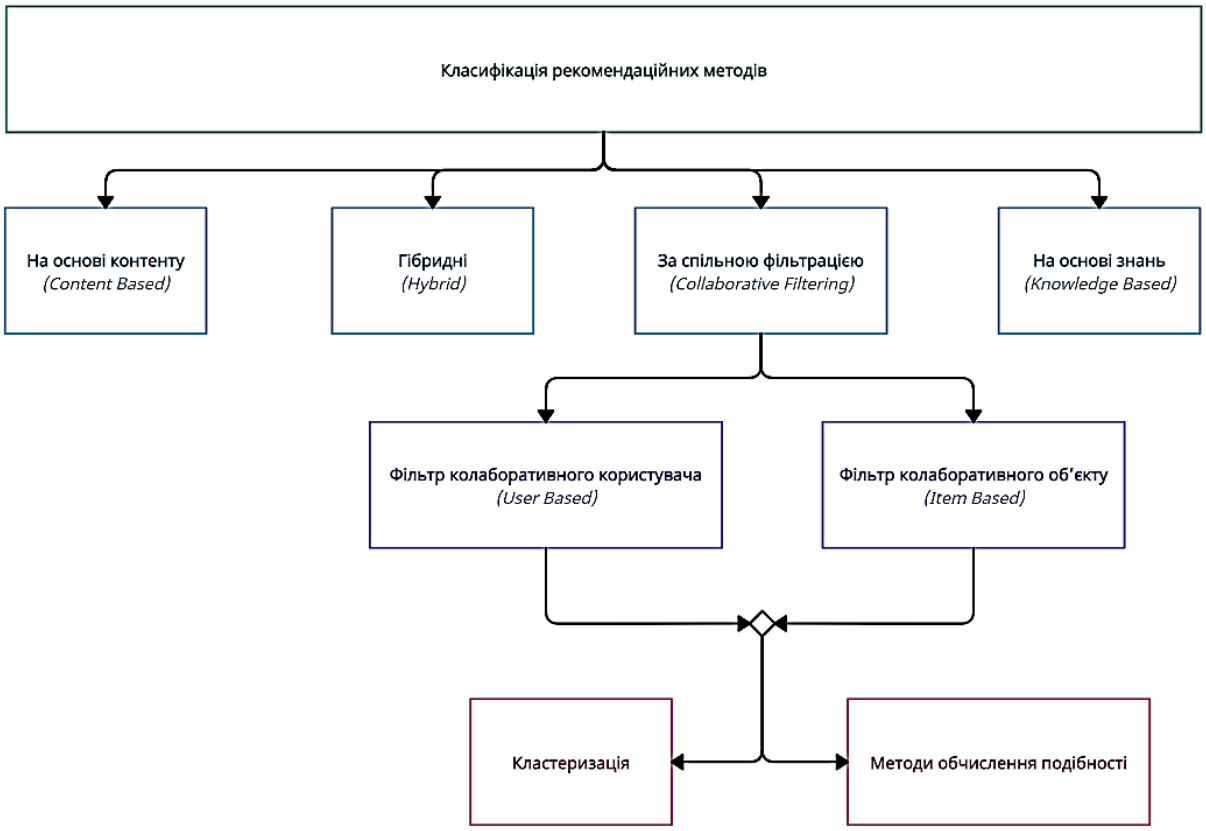

Рисунок 2.1 – Варіант класифікація рекомендаційних методів

Системи рекомендацій на основі вмісту (Content-Based) – це підмножина систем рекомендацій, які підлаштовують рекомендації для користувачів шляхом аналізу внутрішніх характеристик і атрибутів елементів. Ці системи зосереджені на розумінні вмісту елементів і зіставленні його з уподобаннями користувачів. Вивчаючи такі характеристики, як жанр, ключові слова, метадані та інші описові елементи, системи рекомендацій на основі вмісту створюють профілі як для користувачів, так і для елементів. Це дає змогу системі створювати рекомендації, які відповідають уподобанням користувача елементам із подібними рисами вмісту. На відміну від методів спільної фільтрації, які покладаються на історію взаємодії між користувачами, системи на основі вмісту працюють незалежно, що робить їх особливо корисними в сценаріях, коли історія користувачів обмежена або недоступна. Завдяки такому персоналізованому підходу системи рекомендацій на основі вмісту відіграють життєво важливу роль у покращенні взаємодії з

користувачами в різних сферах, від пропонування фільмів і статей до спрямування користувачів у виборі продуктів або місць призначення. Прикладами методів, що засновані на описі характеристик товару та даних користувачів (Content-Based) є методи засновані на аналізі транзакцій (Transaction-Based), метод визначення рейтингу товарів за ключовими словами та методи подання даних про об'єкти. [5]

Система рекомендацій заснована на знаннях (Knowledge-Based), коли вона дає рекомендації не на основі історії оцінок користувача, а на конкретних запитах, зроблених користувачем. Це може запропонувати користувачеві надати низку правил чи вказівок щодо того, як мають виглядати результати, або приклад елемента. Потім система шукає у своїй базі даних елементів і повертає подібні результати. Прикладами методів, що засновані на знаннях (Knowledge-Based) є методи побудови запитів користувачів на основі знань про товари та метод використання жорстких обмежень, призначений для визначення вимог користувача.

Системи рекомендацій на основі спільного фільтрування (Collaborative Filtering) покладаються виключно на минулі взаємодії між користувачами та елементами, щоб пропонувати нові продукти. Особливості кожного окремого предмета не враховуються. У спільній фільтрації історичні дані користувача, який взаємодіє з елементами, записуються та зберігаються. Зазвичай це представлено матрицею, відомою як матриця взаємодії між користувачем і елементом, де рядки представляють користувачів, а стовпці — елементи. Схожі користувачі групуються, і всі їхні взаємодії враховуються під час надання рекомендацій цільовому користувачеві.

Прикладами методів, що засновані на основі спільного фільтрування (Collaborative Filtering) є метод порівняння користувачів (User-Based) та метод порівняння елементів (Item-Based). За методом порівняння користувачів визначається прогнозна оцінка рекомендованих товарів, у даному випадку, — фільмів, яка розраховується за мірою схожості уподобань інших клієнтів. За методом

порівняння елементів також визначається прогнозна оцінка, яка розраховується на основі сумісної схожості оцінок фільмів. [6]

Гібридні методи (Hybrid Filtering) використовують методи на основі спільного фільтрування (Collaborative Filtering), на основі знань (Knowledge-Based) та на основі вмісту (Content-Based) сумісно.

2.2 Метод визначення рекомендацій за спільною фільтрацією (Collaborative Filtering)

Спільна фільтрація (Collaborative Filtering) – це поширений метод персоналізованої системи рекомендацій, яка фільтрує інформацію, наприклад дані про взаємодію інших схожих користувачів. Оскільки він працює шляхом прогнозування оцінок користувачів, його вважають виконанням завдання регресії. Існує два загальні типи спільної фільтрації це метод порівняння користувачів (User-Based) та метод порівняння елементів (Item-Based). Спільна фільтрація між користувачами в основному працює на основі припущення, що користувачі, які дали подібні оцінки певному елементу, ймовірно, матимуть такі ж переваги й для інших елементів. Тому цей метод в основному покладається на пошук подібності між користувачами. Однак у деяких випадках перевага користувача може полягати в тому, щоб абстрагуватися, щоб розбити. Тут стане в нагоді спільне фільтрування по елементам. Тут використовується подібність між елементами замість подібності між користувачами.

Щоб запропонувати нові рекомендації конкретному користувачеві, створюється група схожих користувачів (найближчих сусідів) на основі взаємодії контрольного користувача. Для пропозицій використовуються елементи, які є найпопулярнішими в цій групі, але новими для цільового користувача.

У фільтрації на основі елементів нові рекомендації вибираються на основі попередніх взаємодій цільового користувача. Спочатку розглядаються всі позиції,

які користувач вже вподобав. Потім обчислюються подібні продукти та створюються кластери (найближчі сусіди). Нові елементи з цих кластерів пропонуються користувачеві.

Робочий процес спільної фільтрації виглядає так:

– Процес починається з перетворення даних рейтингу в корисну матрицю, де список користувачів — це рядки, а список елементів — стовпці.

– Наступним кроком є модель спільної фільтрації Neighborhood, у якій ми використовуємо функцію подібності для обчислення подібності між користувачами, а результатом є матриця подібності.

– Береться певна кількість (K) подібних користувачів (також відомих як сусіди), і прогнозування рейтингу буде отримано шляхом виконання регресії даних рейтингу цих сусідів.

– Потім елементи будуть відсортовані на основі найвищого рейтингу, а користувачеві буде рекомендовано найпопулярніші.

За методом порівняння користувачів (User-Based) визначається прогнозна оцінка рекомендованих фільмів, яка розраховується за мірою схожості уподобань інших клієнтів. За методом порівняння елементів (Item-based) також визначається прогнозна оцінка, яка розраховується на основі сумісної схожості оцінок фільмів. Для отримання даних за мірою схожості оцінок клієнтів або фільмів використовуються методи кластеризації.

2.3 Методи кластеризації даних для реалізації рекомендаційної функції мережової системи кінотеатрів

Завдання групування точок даних на основі їх схожості одна з одною називається кластеризацією або кластерним аналізом. Цей метод визначено в розділі «Неконтрольоване навчання», який спрямований на отримання інформації з

немаркованих точок даних, тобто, на відміну від контрольованого навчання, ми не маємо цільової змінної.

Метою кластеризації є формування груп однорідних точок даних із різнорідного набору даних. Він оцінює подібність на основі таких показників, як евклідова відстань, косинусова подібність, манхеттенська відстань тощо, а потім групує точки з найвищим показником подібності. [7]

Існує 2 типи кластеризації, які можна виконати для групування подібних точок даних:

– жорстка кластеризація: у цьому типі кластеризації кожна точка даних повністю чи ні належить до кластеру. Наприклад, припустимо, що є 4 точки даних, і ми повинні згрупувати їх у 2 кластери. Отже, кожна точка даних належатиме до кластера 1 або кластера 2;

– м'яка кластеризація: у цьому типі кластеризації замість призначення кожної точки даних в окремий кластер оцінюється ймовірність того, що ця точка є цим кластером. Наприклад, припустимо, що є 4 точки даних, і ми повинні згрупувати їх у 2 кластери. Отже, ми будемо оцінювати ймовірність того, що точка даних належить до обох кластерів. Ця ймовірність обчислюється для всіх точок даних.

Алгоритми кластеризації в основному використовуються для:

– сегментація ринку – компанії використовують кластеризацію, щоб групувати своїх клієнтів і використовувати цільову рекламу, щоб залучити більше аудиторії;

– аналіз ринкового кошика – власники магазинів аналізують свої продажі та з'ясовують, які товари в основному разом купують клієнти. Наприклад, у США, згідно з дослідженням, памперси та пиво батьки зазвичай купували разом;

– аналіз соціальних мереж – сайти соціальних мереж використовують ваші дані, щоб зрозуміти вашу поведінку в Інтернеті та надати вам цільові рекомендації друзів або рекомендації щодо вмісту;

– медична візуалізація – лікарі використовують кластеризацію, щоб знайти уражені ділянки на діагностичних зображеннях, таких як рентгенівські знімки;

– виявлення аномалій. Для виявлення викидів у потоці набору даних у реальному часі або прогнозування шахрайських транзакцій ми можемо використовувати кластеризацію для їх виявлення;

– спростити роботу з великими наборами даних.

На поверхневому рівні кластеризація допомагає аналізувати неструктуровані дані. Побудова графіків, найкоротша відстань і щільність точок даних – це лише деякі з елементів, які впливають на формування кластера. Кластеризація — це процес визначення того, наскільки пов’язані об’єкти на основі метрики, яка називається мірою подібності. Показники подібності легше знайти в менших наборах ознак. Зі збільшенням кількості ознак стає важче створити показники подібності. Залежно від типу алгоритму кластеризації, який використовується в інтелектуальному аналізі даних, для групування даних із наборів даних використовується кілька методів. На рис. 2.2 зображено схематичний розподіл алгоритмів кластеризації за типами:

– кластеризація на основі центроїда (методи поділу, Partitioning methods);

– кластеризація на основі щільності (методи на основі моделі, Model-based methods);

– кластеризація на основі підключення (ієрархічна кластеризація, Hierarchical clustering);

– кластеризація на основі розподілу.

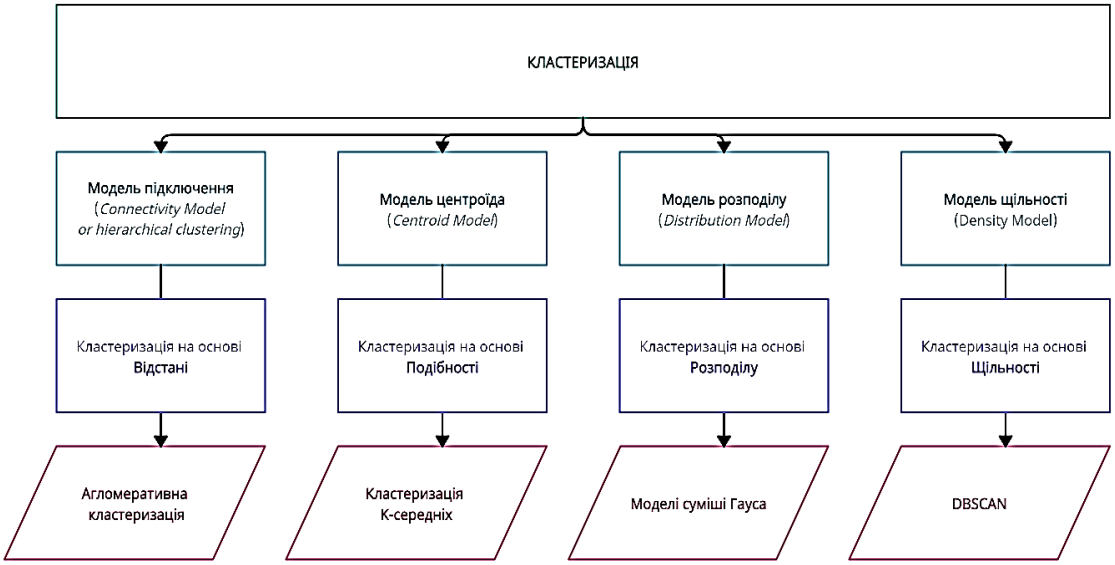

Рисунок 2.2 – Розподіл алгоритмів кластеризації за типами

Методи поділу (Partitioning methods) є найпростішими алгоритмами кластеризації. Вони групують точки даних на основі їх близькості. Як правило, мірою подібності, обраною для цих алгоритмів, є евклідова відстань, відстань Манхеттена або відстань Мінковського. Набори даних розділені на заздалегідь визначену кількість кластерів, і на кожен кластер посилається вектор значень. У порівнянні з векторним значенням змінна вхідних даних не показує різниці та приєднується до кластера. [8]

Основним недоліком цих алгоритмів є вимога, щоб було встановлено кількість кластерів, «k», або інтуїтивно, або науково (за допомогою методу Elbow), перш ніж будь-яка система машинного навчання кластеризації почне розподіляти точки даних. Незважаючи на це, це все ще найпопулярніший вид кластеризації. Кластеризація K-means і K-medoids є деякими прикладами кластеризації цього типу.

Кластеризація на основі щільності, метод на основі моделі (Model-based methods), знаходить групи на основі щільності точок даних. На відміну від кластеризації на основі центроїда, яка вимагає попереднього визначення кількості кластерів і чутливості до ініціалізації, кластеризація на основі щільності визначає

кількість кластерів автоматично та менш чутлива до початкових позицій. Вони чудово справляються з кластерами різних розмірів і форм, що робить їх ідеальними для наборів даних із кластерами неправильної форми або перекриттям. Ці методи керують як щільними, так і розрідженими областями даних, зосереджуючись на локальній щільності, і можуть розрізняти кластери з різними морфологіями.

Навпаки, групування на основі центроїда, як і k-середні, має проблеми з пошуком кластерів довільної форми. Через попередньо встановлену кількість вимог до кластерів і надзвичайну чутливість до початкового розташування центроїдів результати можуть відрізнятися. Крім того, тенденція підходів, заснованих на центроїді, створювати сферичні або опуклі кластери, обмежує їхню здатність працювати зі складними кластерами або кластерами неправильної форми. Підсумовуючи, кластеризація на основі щільності долає недоліки методів, заснованих на центроїді, шляхом автономного вибору розмірів кластерів, стійкості до ініціалізації та успішного захоплення кластерів різних розмірів і форм. Найпопулярнішим алгоритмом кластеризації на основі щільності є DBSCAN.

Метод об’єднання пов’язаних точок даних в ієрархічні кластери називається ієрархічною кластеризацією. Кожна точка даних спочатку враховується як окремий кластер, який згодом поєднується з кластерами, які є найбільш схожими, щоб утворити один великий кластер, який містить усі точки даних. [9]

Використовуючи кластеризацію на основі розподілу, точки даних генеруються та організовуються відповідно до їхньої схильності потрапляти до того самого розподілу ймовірностей (наприклад, Гауса, біноміального чи іншого) в межах даних. Елементи даних групуються за допомогою ймовірнісного розподілу, який базується на статистичних розподілах. Включено об’єкти даних, які мають більшу ймовірність бути в кластері. Точка даних з меншою ймовірністю буде включена в кластер, чим далі вона знаходиться від центральної точки кластера, яка існує в кожному кластері.

Помітним недоліком підходів, що базуються на щільності та на основі меж, є необхідність апріорного визначення кластерів для деяких алгоритмів і, насамперед,

визначення форми кластера для основної маси алгоритмів. Має бути вибрано принаймні одне налаштування або гіперпараметр, і хоча це має бути просто, неправильне налаштування може мати непередбачені наслідки. Кластеризація на основі розподілу має певну перевагу перед підходами кластеризації на основі близькості та центроїда з точки зору гнучкості, точності та структури кластера. Ключова проблема полягає в тому, що для того, щоб уникнути переобладнання, багато методів кластеризації працюють лише з змодельованими або виготовленими даними, або коли основна частина точок даних, безумовно, належить до попередньо встановленого розподілу. [10] Найпопулярнішим алгоритмом кластеризації на основі розподілу є модель суміші Гауса.

Для виявлення найкращих показників для кластеризації за тим чи іншим методом використовується один із показників «silhouette_score», який використовується для оцінки якості алгоритму кластеризації

«Silhouette_score» або оцінка силуету — це показник, який використовується для оцінки якості алгоритму кластеризації. Це графічна допомога для інтерпретації та перевірки узгодженості в кластерах даних. Оцінка розраховується з використанням середньої відстані між кластерами (a) та середньої відстані до найближчого кластера (b) для кожного зразка.

Оцінка силуету для кожного зразка розраховується наступним чином:

$$\text{silhouette } score = \frac{b-a}{\max(a,b)} \tag{2.1}$$

Нижче приведено пояснення значення кожного терміна:

— «a» є середньою відстанню між зразком і всіма іншими точками в тому самому класі або кластері. Його можна розглядати як міру того, наскільки добре вибірка присвоєна власному кластеру.

— «b» є середньою відстанню між зразком і всіма іншими точками в наступному найближчому кластері. Його можна розглядати як міру того, наскільки добре зразок відділений від сусідніх кластерів.

Оцінка силуету коливається від -1 до +1, де високе значення вказує на те, що об'єкт добре збігається з власним кластером і погано збігається з сусідніми кластерами. Якщо більшість об'єктів мають високе значення, то конфігурація кластеризації вважається відповідною. Якщо багато точок мають низьке або від'ємне значення, тоді конфігурація кластеризації може мати занадто багато або занадто мало кластерів. [11]

Силует можна розрахувати за допомогою будь-якої метрики відстані, наприклад евклідової відстані або відстані Манхеттена.

2.3.1 Метод кластеризації k-середніх

Кластеризація K-середніх — це алгоритм неконтрольованого машинного навчання, який групує набір даних без міток у різні кластери.

Машинне навчання без нагляду — це процес навчання комп'ютера використанню немаркованих, несекретних даних і надання алгоритму можливості працювати з цими даними без нагляду. Без будь-якого попереднього навчання роботі з даними робота машини в цьому випадку полягає в тому, щоб організувати несортовані дані відповідно до паралелей, шаблонів і варіацій. [12]

K означає кластеризацію, призначає точки даних одному з K кластерів залежно від їх відстані від центру кластерів. Він починається з випадкового призначення центроїда кластерів у просторі. Потім кожна точка даних призначається одному з кластерів на основі її відстані від центроїда кластера. Після призначення кожної точки одному з кластерів призначаються нові центроїди кластера. Цей процес виконується ітеративно, доки не буде знайдено хороший кластер. Під час аналізу

припускається, що номер кластера задано заздалегідь, і ми повинні поставити бали в одну з груп.

У деяких випадках K не визначено чітко та треба думати про оптимальну кількість K. K означає, що кластеризація дає найкращі дані, добре розділені. Якщо точки даних перекриваються, ця кластеризація не підходить. K-середніх є швидшим порівняно з іншими методами кластеризації. Це забезпечує міцний зв'язок між точками даних. K означає, що кластер не надає чіткої інформації щодо якості кластерів. Різне початкове призначення центроїда кластера може призвести до різних кластерів. Крім того, алгоритм K Means чутливий до шуму.

Мета кластеризації полягає в тому, щоб розділити генеральну сукупність або набір точок даних на кілька груп, щоб точки даних у кожній групі були більш порівнянними між собою та відрізнялися від точок даних в інших групах. По суті, це групування речей на основі того, наскільки вони схожі та відмінні один від одного.

Нам надається набір даних елементів із певними ознаками та значеннями для цих ознак (як вектор). Завдання полягає в тому, щоб розділити ці предмети на групи. Щоб досягти цього, ми будемо використовувати алгоритм K-means, алгоритм неконтрольованого навчання. «K» у назві алгоритму представляє кількість груп/кластерів, у які ми хочемо класифікувати наші елементи. [13]

Алгоритм класифікує елементи в k груп або кластерів подібності. Щоб обчислити цю подібність, використовуветься евклідова відстань як вимірювання.

Алгоритм роботи наступний:

– Спочатку ми випадковим чином ініціалізуємо k точок, які називаються середніми або кластерними центроїдами.

– Ми класифікуємо кожен елемент відповідно до його найближчого середнього значення та оновлюємо координати середнього значення, які є середніми значеннями елементів, класифікованих у цьому кластері на даний момент.

– Ми повторюємо процес протягом заданої кількості ітерацій, і в кінці ми маємо наші кластери.

«Точки», згадані вище, називаються середніми, оскільки вони є середніми значеннями елементів, класифікованих у них. [14] Для ініціалізації цих засобів у нас є багато варіантів. Інтуїтивно зрозумілим методом є ініціалізація засобів у випадкових елементах у наборі даних. Інший метод полягає в ініціалізації середніх випадковими значеннями між межами набору даних.

Перевага цього методу у тому, що він швидкий та ефективний завдяки тому, що не потребує обчислення всіх попарних відстаней між елементами, на відміну більшості інших методів кластеризації, включаючи ті, що використовуються в процедурах ієрархічного кластерного аналізу. Головними визначеними недоліками методу k-середніх є потрібність заздалегідь задавати кількість кластерів та параметри початкового центру їхнього визначення, що за результатами може надавати різні кінцеві кластери.

2.3.2 Ієрархічні методи кластеризації «згори-вниз» та «знизу-вгору»

Ієрархічна кластеризація — це модель кластеризації на основі зв’язку, яка групує точки даних, розташовані близько одна до одної, на основі міри подібності або відстані. Припущення полягає в тому, що точки даних, які знаходяться близько одна до одної, є більш схожими або пов’язаними, ніж точки даних, які розташовані далі одна від одної.

Дендрограма, деревоподібна фігура, створена шляхом ієрархічної кластеризації, зображує ієрархічні відносини між групами. Окремі точки даних розташовані в нижній частині дендрограми, тоді як найбільші кластери, які включають усі точки даних, розташовані вгорі. Щоб створити різну кількість кластерів, дендрограму можна розрізати на різній висоті. Корінь дерева – це єдиний кластер, який містить у собі всю множину елементів. Листя – це кластери, що складаються лише з одного елемента. [15]

Дендрограмма створюється шляхом ітеративного злиття або поділу кластерів на основі вимірювання подібності або відстані між точками даних. Кластери розділяються або об’єднуються неодноразово, доки всі точки даних не будуть міститися в одному кластері або поки не буде досягнуто заздалегідь визначену кількість кластерів.

Ми можемо подивитися на дендрограму та виміряти висоту, на якій гілки дендрограми утворюють окремі кластери, щоб обчислити ідеальну кількість кластерів. Дендрограмму можна розрізати на цій висоті, щоб визначити кількість кластерів.

Існує два типи ієрархічної кластеризації: агломеративна кластеризація (Agglomerative Clustering) та роздільна кластеризація (Divisive clustering).

Агломеративна кластеризація, також відома як підхід «знизу-вгору». Структура, яка є більш інформативною, ніж неструктурований набір кластерів, отриманий плоскою кластеризацією. Цей алгоритм кластеризації не вимагає попереднього визначення кількості кластерів. Алгоритми «знизу вгору» обробляють усі дані як єдиний кластер на початку, а потім послідовно об’єднують пари кластерів, доки всі кластери не будуть об’єднані в єдиний кластер, який містить усі дані. На рис. 2.2 представлено схематично роботу алгоритму «знизу-вгору». [16]

Роздільна кластеризація, також відома як підхід «зверху-вниз», не вимагає попереднього визначення кількості кластерів. Для низхідної кластеризації потрібен метод для розбиття кластера, який містить цілі дані та продовжується рекурсивним розбиттям кластерів, доки окремі дані не будуть розбиті на одиночні кластери. На рис. 2.3 представлено схематично роботу алгоритму «зверху вниз».

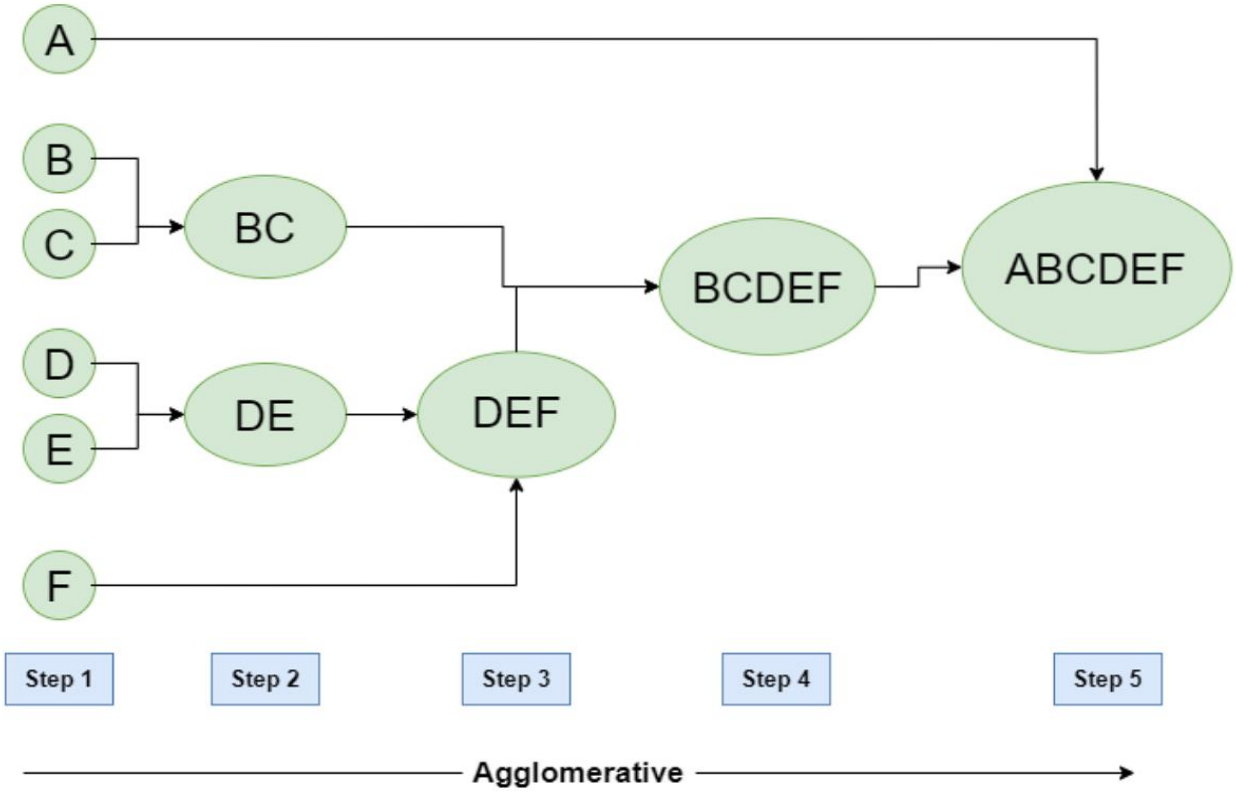

Рисунок 2.3 – Схематична робота алгоритму «знизу-вгору»

Роздільна кластеризація є складнішою порівняно з агломеративною, оскільки у випадку роздільної кластеризації потрібен метод плоскої кластеризації як «підпрограма» для розбиття кожного кластера, доки не отримаємо кожне з даних, що матиме власний єдиний кластер.

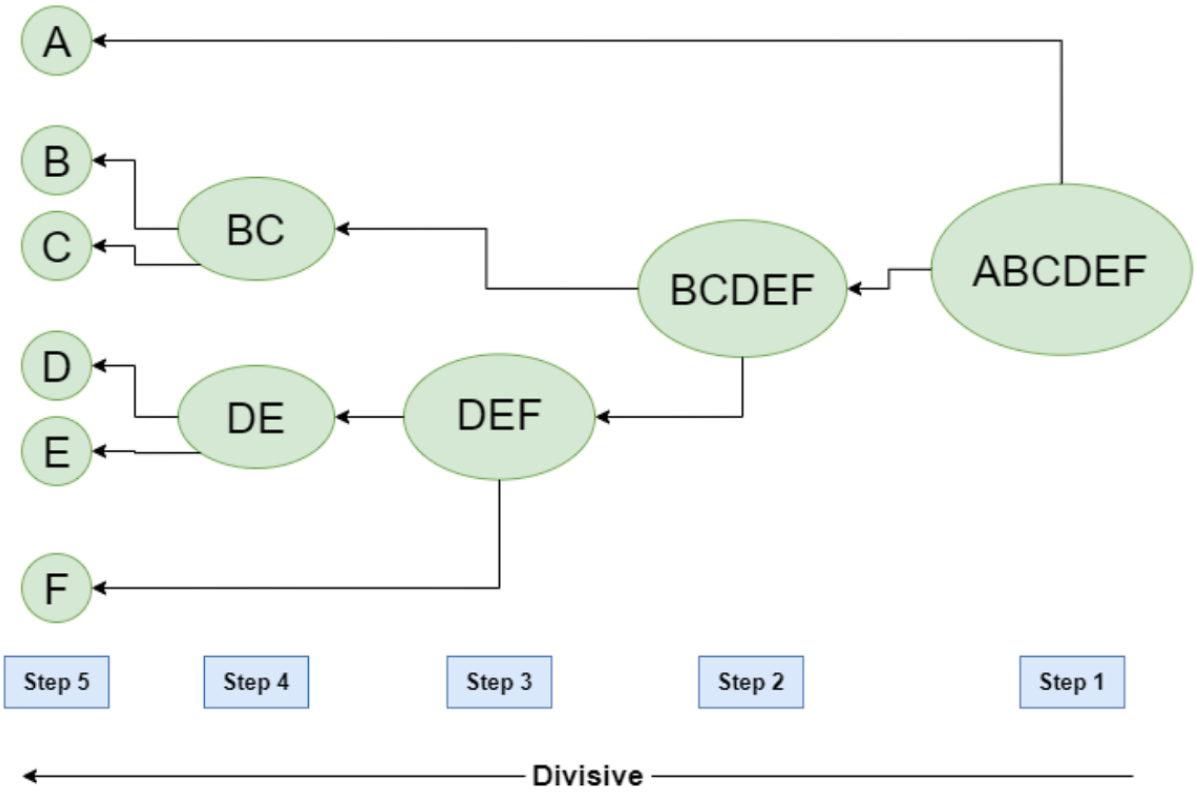

Рисунок 2.4 – Схематична робота алгоритму «зверху-вниз»

Роздільна кластеризація ефективніша, якщо ми не створюємо повну ієрархію аж до окремих листів даних. Часова складність простої агломеративної кластеризації дорівнює $O(n^3)$, оскільки ми повністю скануємо N x N матрицю dist_mat для найменшої відстані в кожній із N-1 ітерацій. Використовуючи структуру даних пріоритетної черги, ми можемо зменшити цю складність до $O(n^{2\log n})$. За допомогою додаткових оптимізацій його можна знизити до $O(n^2)$. У той час як для роздільної кластеризації задана фіксована кількість верхніх рівнів, використовуючи ефективний плоский алгоритм, такий як K-Means, розділові алгоритми є лінійними за кількістю шаблонів і кластерів. [17]

Алгоритм розділення також більш точний. Агломеративна кластеризація приймає рішення, враховуючи локальні шаблони або сусідні точки без початкового врахування глобального розподілу даних. Ці ранні рішення неможливо скасувати.

тоді як кластеризація, що розділяє, бере до уваги глобальний розподіл даних під час прийняття рішень щодо розділення на верхньому рівні.

На відміну від методу k-середніх, за ієрархічною кластеризацією отримується однозначний результат, який не залежить від завдання початкового центу і кількості кластерів. Основним визначеним недоліком є отримання надлишкової ієрархії кластерів, яка може бути зайвою в контексті поставленої задачі

2.3.3 Метод кластеризації DBSCAN (Density-based spatial clustering of applications with noise)

Алгоритм DBSCAN базується на інтуїтивно зрозумілому понятті «кластерів» і «шуму». Ключова ідея полягає в тому, що для кожної точки кластера околиці даного радіуса повинні містити принаймні мінімальну кількість точок. [18]

Методи поділу (K-середні, кластеризація PAM) та ієрархічна кластеризація працюють для пошуку кластерів сферичної форми або опуклих кластерів. Іншими словами, вони підходять тільки для компактних і добре розділених скупчень. Крім того, на них також сильно впливає наявність шуму та викидів у даних. Реальні дані можуть містити відхилення, такі як кластери можуть бути довільної форми або ж дані можуть містити шум.

Алгоритм роботи DBSCAN наступний:

– DBSCAN починається з довільної початкової точки даних, яка ще не була відвідана. Усі точки, що знаходяться на відстані ε, розглядаються як точки-сусіди нашої вершини.

– Якщо в цьому околі є достатня кількість точок (більша деякого наперед заданого значення), тоді починається процес кластеризації, а поточна точка даних стає першою точкою в новому кластері. В іншому випадку точка буде позначена як шум (пізніше ця точка ще може стати частиною деякого кластера). Але в обох випадках ця точка позначається як "відвідана".

– Для першої точки в новому кластері усі її точки-сусіди в межах єоколу також стають частиною цього кластера. Ця процедура виконується для всіх точок в околі є та відносить їх до одного кластеру, після чого аналогічний процес відбувається для всіх нових точок, які були щойно додані до кластеру.

– Кроки 2 і 3 повторюються циклічно до тих пір, поки всі точки кластеру не будуть знайдені, тобто всі точки в околицях кластера є вже були відвідані та позначені.

– Після того, як ми закінчимо з поточним кластером, можемо переходити до нової невідвіданої точки, що призведе до знаходження нового кластеру чи шуму. Цей процес повторюється, поки всі точки не будуть позначені як відвідувані. Оскільки в кінці виконання алгоритму всі точки будуть відвідані, то кожна точка буде позначена як належна певному кластеру, або як шум.

Кластеризація за щільністю DBSCAN має значні переваги перед іншими алгоритмами кластеризації тому що безпосереднє сканує базу даних. За цим методом визначаються області концентрації елементів. Вони відокремлюються від розріджених (порожніх) областей і визначаються як кластери. Метод DBSCAN не потребує апріорного завдання кількості кластерів, на відміну від методу k-середніх, і визначається автоматично в ході сканування. [19]

2.4 Порівняння методів кластеризації за визначеними параметрами

У розділі 2.2 класифіковано рекомендаційні системи електронної комерції за підходами – персоналізовані та неперсоналізовані та за способом генерації персоналізованих рекомендацій. [20] Серед персоналізованих підходів рекомендаційні методи класифікуються на системи рекомендацій на основі контенту (Content-Based), методи, що засновані на знаннях (Knowledge-Based), системи рекомендацій із спільною фільтрацією (Collaborative Filtering). [21]

Класифіковано методи кластеризації за моделлю, серед яких виділено модель підключення, модель центроїда, модель розподілу та модель щільності. Розглянуто три методи кластеризації:

– метод k-середніх, який відноситься до моделі центроїда [22];

– агломеративний та роздільний методи, які відносяться до ієрархічна кластеризації або іншими словами до моделі підключення;

– метод DBSCAN, який відноситься до моделі щільності.

Порівняння параметрів методів кластеризації наведено у таблиці 2.1. Визначено, що для створення ефективного рекомендаційного алгоритму потрібно зібрати такі дані: оцінки фільмів користувачами, атрибути фільмів та вимоги користувача.

Таблиця 2.1– Порівняння параметрів методів кластеризації

Характеристики	k-середніх	Ієрархічна	DBSCAN
Часова складність	$O(n^2)$	$O(n^3)$	$O(n \log n)$
Налаштування гіперпараметрів	Потрібно вказувати кількість кластерів k та перенавчати модель для кожного k	Динамічно оновлює значення k без повторного навчання моделі	Динамічно оновлює значення k без повторного навчання моделі
Структура даних	Кластери, утворені в K-Means, мають сферичну або опуклу форму	Кластери, утворені в Ієрархічній кластеризації, мають форму дендрограми	Кластери, сформовані в DBSCAN, можуть мати будь-яку довільну форму.
Варіації	Багато варіацій (наприклад k-медіана, k-медоїд та інше) з різними матрицями відстаней	Два підходи: агломеративний та роздільний	Варіацій немає
Оптимізація	k-середніх++ представляє розумнішу ініціалізацію центроїдів, що робить корвенгенцію швидшою	Роздільна кластеризація (Divisive clustering) зменшує часову складність до $O(n^2)$	DBSCAN можна оптимізувати, вибравши оптимальні значення для епсилон і MinPts, використовуючи різні параметри кластеризації та

			використовуючи інші алгоритми оптимізації, такі як OPTICS.
Викиди (outliners)	Не можуть бути ідентифіковані	Не може обробляти викиди	Може обробляти викиди та шуми
Надійність результату	Результат може відрізнятися при різних прогонах	Одні і ті ж параметри генерують один і той самий результат	Одні і ті ж параметри генерують один і той самий результат
Переваги	Дуже зрозумілий і може масштабуватися для великої кількості наборів даних	Вбудована гнучкість щодо рівня, деталізації за допомогу дендограми	Низька складність, ідентифікує викиди та шуми
Недоліки	Не може оброблювати шуми, викиди та кластери довільного розміру і щільності	Бракує масштабованості для обробки великих наборів даних	Очікується певне падіння щільності, щоб виявити межі кластера
Сценарії використання методів	Рівномірний розмір кластера, плоска геометрія, не надто багато кластерів; для загального призначення	Можливі обмеження підключення, неевклідові відстані та багато кластерів	Нерівномірні розміри кластерів та неплоска геометрія

3 ДОСЛІДЖЕННЯ ВИКОРИСТАННЯ МЕТОДІВ ВИЗНАЧЕННЯ РЕКОМЕНДАЦІЙ У МЕРЕЖІ КІНОТЕАТРІВ

3.1 Визначення послідовності дій для дослідження методів кластеризації даних для реалізації рекомендаційної функції мережевої системи кінотеатрів

Структура послідовності дій дослідження методів кластеризації даних для реалізації рекомендаційної функції мережевої системи кінотеатрів подана на рис. 3.1.

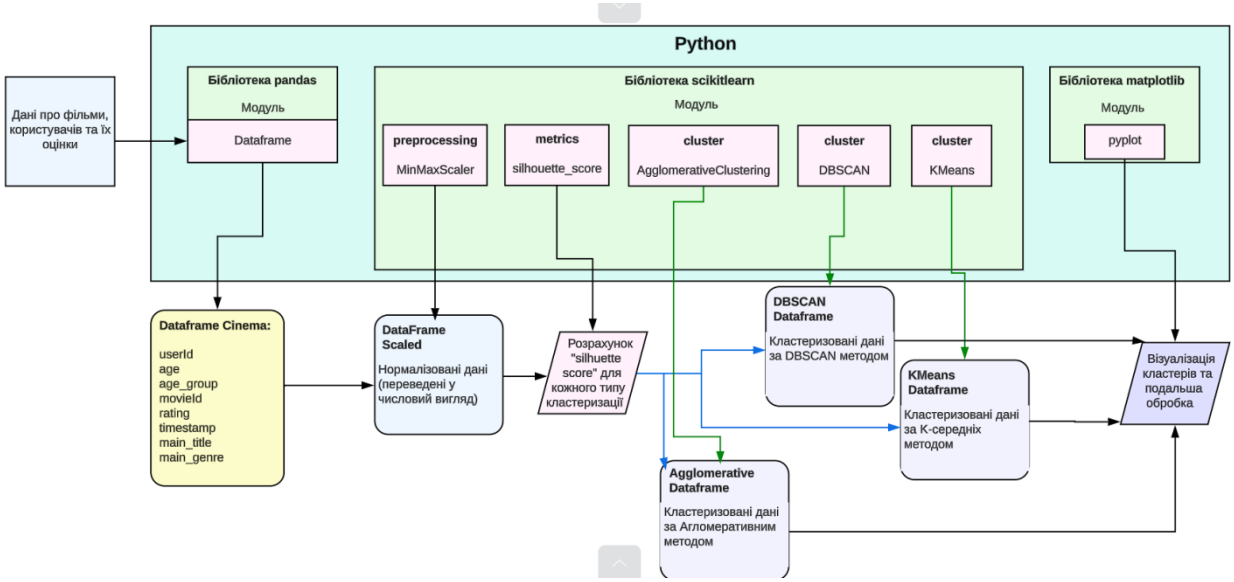

Рисунок 3.1 – Структура послідовності дій дослідження методів кластеризації даних для реалізації рекомендаційної функції мережевої системи кінотеатрів

Для дослідження методів кластеризації даних для реалізації рекомендаційної функції проведені наступні дії:

– проведено дослідження даних мовою програмування Python та середовище програмування Anaconda, у яке вивантажено дані excel файлу з у датаферейм;

– проведено аналіз вивантажених даних та приведено у бажаний вигляд для подальшої нормалізації: додані нові значимі стовпці, модифіковані або видалені стовпці, які вплинуть на кластеризацію;

– проведена нормалізація даних, тобто дані приведені у числовий вигляд і готові для кластеризації;

– разраховано для кожного типу кластеризації найкращу кількість кластерів (K-means, Agglomerative) та найкращі параметри «eps» та «min_samples» (DBSCAN) за допомогою «silhouette_score» показника, який використовується для оцінки якості алгоритму кластеризації;

– проведено кластеризацію даних методами K-means, Agglomerative та DBSCAN за найкращими показниками, який було виявлено на попередньому кроці

– проведено візуалізацію кластерів, обрано та провізуалізовано кластер, який використовується у подальшому аналізі;

– розраховано передбачуваних оцінок фільмів для обраного кластеру та користувача у ньому;

– реалізовано рекомендаційну функцію фільмів, яка видає топ 20 рекомендацій в залежності від жанру та користувача;

– порівняно результати роботи рекомендаційної функції в залежності від вихідних даних кластеризаційних методів.

3.2 Опис та дослідження даних, які будуть використані для кластеризації для реалізації рекомендаційної функції мережевої системи кінотеатрів

3.2.1 Опис атрибутів таблиць використаних для дослідження методів кластеризації для реалізації рекомендаційної функції мережевої системи кінотеатрів

Для дослідження методів кластеризації k-середніх, ієрархічної та DBSCAN було використано датасет MovieLens, який містить понад 60 тисяч рядків з даними щодо фільмів та їх оцінок.

Перш за все, перед застосуванням цим даних у роботі була додана нова таблиця, яка зберігає інформацію щодо користувачів, які переглядали фільми та ставили оцінки.

Таблиця «users» має наступні атрибути:

– ключовий атрибут «userId» – це первинний ключ сутності «users», який є зовнішнім ключем для таблиці «ratings»;

– атрибут «gender» – це стать користувача, яка може бути жіночою(F) або чоловічою (M);

– атрибут «age» – це вік користувача, який було згенеровано з випадковими числами від 7 до 60.

Таблиця «films» зберігає інформацію щодо фільмів та їх жанрів, містить наступні атрибути:

– ключовий атрибут «movieId» – це первинний ключ сутності «films», який є зовнішнім ключем для таблиці «ratings»;

– атрибут «title» – це назва фільму, яка зберігає у собі назву фільму, а також рік випуску фільму;

– атрибут «genres» – це один або декілька жанрів, до яких належить фільм. Можливі жанри це дія, пригоди, мультфільми, дитячий, комедія, кримінал, документальний фільм, драма, фантастика, фільм-нуар, жахи, мюзикл, детективи, романтика, наукова фантастика, трилер, війна, західні фільми або ж жанру зовсім може не бути вказано у фільмі.

Таблиця «ratings» зберігає інформацію щодо оцінок фільмів та містить наступні атрибути:

– ключовий атрибут «movieId» – це зовнішнім ключ для таблиці «ratings»;

– ключовий атрибут «userId» – це зовнішнім ключем для таблиці «ratings»;

– атрибут «rating» – це оцінки фільмів від користувачів, які складаються за 5-бальною шкалою з кроком у півзірки (0,5 зірки - 5,0 зірки);

– атрибут «timestamp» – це мітки часу коли була поставлена оцінка за збережена.

Для роботи з даними була обрана мова Python та використана платформа Anaconda. Дані таблиць excel були послідовно зчитані за допомогою методів ExcelFile() та read_excel() бібліотеки pandas, та записані у окремі датасети:

– «cinema_db» – це основний датасет, у який було зчитано весь excel файл «cinema_film_user_rate»;

– «users_db» – це перший додаткий датасет, у який було зчитано лист excel файлу, який відносився до таблиці «users»;

– «films_db» – це другий додаткий датасет, у який було зчитано лист excel файлу, який відносився до таблиці «films»;

– «ratings_db» – це третій додаткий датасет, у який було зчитано лист excel файлу, який відносився до таблиці «ratings»;

Після цього, для подальшого аналізу даних, був доданий новий стовпчик у датасет «users_db» - «age_group». У листингу 3.1 наведено код перетворення excel файлу у dataset та створення нового стовпцю «age_group».

Лістинг 3.1 – Код для перетворення excel файлу у dataset та створення нового стовпцю 'age_group'

```
cinema_db = pd.ExcelFile('cinema_film_user_rate.xlsx')
users_db = pd.read_excel(cinema_db, 'users')
films_db = pd.read_excel(cinema_db, 'films')
ratings_db = pd.read_excel(cinema_db, 'ratings')
users_db['age_group'] = users_db['age'].apply(get_age_group)
```

Для створення стовпчику «age_group» застосовано функцію «get_age_group», яка повертає значення в залежності від встановлених вікових груп:

– значення «Child» встановлюється для користувачів вік яких менший за 12 років;

– значення «Teen» встановлюється для користувачів вік яких менший за 21 років та більший за 13 років;

– значення «Young» встановлюється для користувачів вік яких менший за 35 років та більший за 22 роки;

– значення «Adult» встановлюється для користувачів вік яких менший за 21 років та більший за 13 років;

– значення «Old age» встановлюється для користувачів яких менший за 21 років та більший за 13 років.

У лістингу 3.2 наведено код реалізації функції «get_age_group».

Лістинг 3.2 – Код реалізації функції «get_age_group»

```
def get_age_group(age):
    if age < 12:
        return 'Child'
    elif age >= 13 and age < 21:
        return 'Teen'
    elif age >= 22 and age < 35:
        return 'Young'
    elif age >= 36 and age < 60:
        return 'Adult'
    else:
        return 'Old age'
```

3.2.2 Дослідження та підготовка даних до кластеризації

Наступним кроком для приведення даних у пригідний вид для кластеризації є об’єднання «users_db», «films_db» та «ratings_db» у один датасет.

Для цього був застосований метод merge() бібліотеки pandas. У параметрах функції вказуються датасети для об’єднання та індекс, який присутній у обох датасетах. Даний метод повністю аналогічний роботі методу join при роботі з SQL базами даних. У лістингу 3.3 наведено код реалізації поєднання 3 датасетів у один.

Лістинг 3.3 – Код реалізації об’єднання датасетів у один

```
ratings_db_title = pd.merge(ratings_db, films_db[['movieId', 'title','genres']], on='movieId' )
users_db_ratings_db_join = pd.merge(users_db[['userId', 'age','age_group']], ratings_db_title, on='userId')
users_db_ratings_db_join
```

У результаті отримано датасет «users_db_ratings_db_join», вигляд якого показано на рис. 3.2.

	userId	age	age_group	movieId	rating	timestamp	title	genres
0	1	16	Teen	296	5.0	1147880044	Pulp Fiction (1994)	Comedy|Crime|Drama|Thriller
1	1	16	Teen	306	3.5	1147868817	Three Colors: Red (Trois couleurs: Rouge) (1994)	Drama
2	1	16	Teen	307	5.0	1147868828	Three Colors: Blue (Trois couleurs: Bleu) (1993)	Drama
3	1	16	Teen	665	5.0	1147878820	Underground (1995)	Comedy|Drama|War
4	1	16	Teen	899	3.5	1147868510	Singin' in the Rain (1952)	Comedy|Musical|Romance
...	...	...	...	...	...	...	...	...
1048570	7045	13	Teen	4447	3.5	1164258032	Legally Blonde (2001)	Comedy|Romance
1048571	7045	13	Teen	4720	4.0	1164257756	Others, The (2001)	Drama|Horror|Mystery|Thriller
1048572	7045	13	Teen	4857	4.0	1164242753	Fiddler on the Roof (1971)	Drama|Musical
1048573	7045	13	Teen	4886	5.0	1168033506	Monsters, Inc. (2001)	Adventure|Animation|Children|Comedy|Fantasy
1048574	7045	13	Teen	4896	4.0	1164122301	Harry Potter and the Sorcerer's Stone (a.k.a. ...	Adventure|Children|Fantasy

1048575 rows × 8 columns

Рисунок 3.2 – Результат об’єднання трьох датасетів у «users_db_ratings_db_join»

На рис. 3.2 чітко видно у якому форматі представлені дані, на даний момент можна побачити, що стовбець «title» має формат у вигляді назви фільму та ріку випуску у дужках, а стовпець «genres» містить декілька жанрів, які розділені знаком «|». Для спрощення роботи з даними, стовпець «title» та стовпець «genres» модифіковано.

Для виділення основного жанру фільму, був використаний метод split(), який прибрав знаки «|» та залишив тільки перший вказаний жанру фільму, який записується у новий стовпець «main_genre», а старий видаляється з датасету.

Для видалення року випуску фільму з назви фільму, був використаний метод extract(), який знаходить дужки та видаляє цю інформацію зі стовпчика «title». Назва

фільму у чистому вигляді записується у новий стовпчик з назвою «main_title», а старий «title» видаляється з датасету.

Реалізація цієї логіки приведена у лістингу 3.4.

Лістинг 3.4 – Код реалізації об’єднання датасетів у один

```
users_db_ratings_db_join['main_genre']=users_db_ratings_db_join['genres'].str
.split('|').str[0]
users_db_ratings_db_join.drop(['genres'], axis=1, inplace=True)
users_db_ratings_db_join['title']=users_db_ratings_db_join['title'].astype(st
r)
users_db_ratings_db_join['main_title']=users_db_ratings_db_join['title'].str.
extract(r'^(.*?)(?:\s*\(.*\))?$')
users_db_ratings_db_join.head()
```

Отриманий результат (верхівка датасету) приведено на рис. 3.3.

	userId	age	age_group	movieId	rating	timestamp	main_title	main_genre
0	1	16	Teen	296	5.0	1147880044	Pulp Fiction	Comedy
1	1	16	Teen	306	3.5	1147868817	Three Colors: Red	Drama
2	1	16	Teen	307	5.0	1147868828	Three Colors: Blue	Drama
3	1	16	Teen	665	5.0	1147878820	Underground	Comedy
4	1	16	Teen	899	3.5	1147868510	Singin' in the Rain	Comedy

Рисунок 3.3 – Оновлений вигляд датасету «users_db_ratings_db_join»

Для організації датасету «users_db_ratings_db_join» у вигляді, який би був зручний для подальшої кластеризації, встановлено два індекси «userId» та «main_title», код якого наведено у лістингу 3.5

Лістинг 3.5 – Код реалізації встановлення двух індексів у датасеті

```
users_db_ratings_db_join.set_index(['userId', 'main_title'], inplace=True)
users_db_ratings_db_join.head()
```

При переході на наступний крок спершу за допомогою функції було пораховано кількість рядків та стовпчиків у датасеті «users_db_ratings_db_join». Кількість рядків становить 1 048 575, а стовпців 6.

Для оброблювання такої кількості інформації потребуються значні обчислювані ресурси, яких не має у моєму розпорядженні. Тому для вирішення цієї проблеми, датасет «users_db_ratings_db_join» було зменшено до 49193 рядків, майже у 21 раз. Зменшення датасету було зроблене завдяки зменшенню кількості користувачів за допомогою функції «loc», яких стало рівно 400.

Лістинг 3.6 – Код реалізації зменшення датасету «users_db_ratings_db_join»

```
smaller_dataset = users_db_ratings_db_join.loc[1:400]
smaller_dataset
```

Результатом є зменшений датасет «smaller_dataset», вигляд якого приведено на рис. 3.4. Також на цьому же рис. 3.4 видно встановлені індекси, які були описані у листингу 3.6.

[27]:

userId	main_title	age	age_group	movieId	rating	timestamp	main_genre
1	Pulp Fiction	16	Teen	296	5.0	1147880044	Comedy
	Three Colors: Red	16	Teen	306	3.5	1147868817	Drama
	Three Colors: Blue	16	Teen	307	5.0	1147868828	Drama
	Underground	16	Teen	665	5.0	1147878820	Comedy
	Singin' in the Rain	16	Teen	899	3.5	1147868510	Comedy
...	...	...	...	...	...	...	...
400	Desperately Seeking Susan	12	Old age	2369	3.0	920334341	Comedy
	Jewel of the Nile, The	12	Old age	2405	5.0	920333358	Action
	Romancing the Stone	12	Old age	2406	5.0	920332146	Action
	You've Got Mail	12	Old age	2424	5.0	920331368	Comedy
	She's All That	12	Old age	2485	2.0	920331368	Comedy

49193 rows × 6 columns

Рисунок 3.4 –Датасет «smaller_dataset»

Щоб більше розуміти з якими даними йде робота, на рис. 3.5 наведена візуалізація жанрів, які присутні у датасеті «smaller_dataset». (малюнок зліва) На ньому можна побачити, що більша частина фільмів сфокусована на жанрі комедія, драма та екшн. Також на рис. 3.5. представлено візуалізацію вікових груп, які присутні у датасеті (правий малюнок). Можна побачити, що більш усього у датасеті дорослих та молодих користувачів.

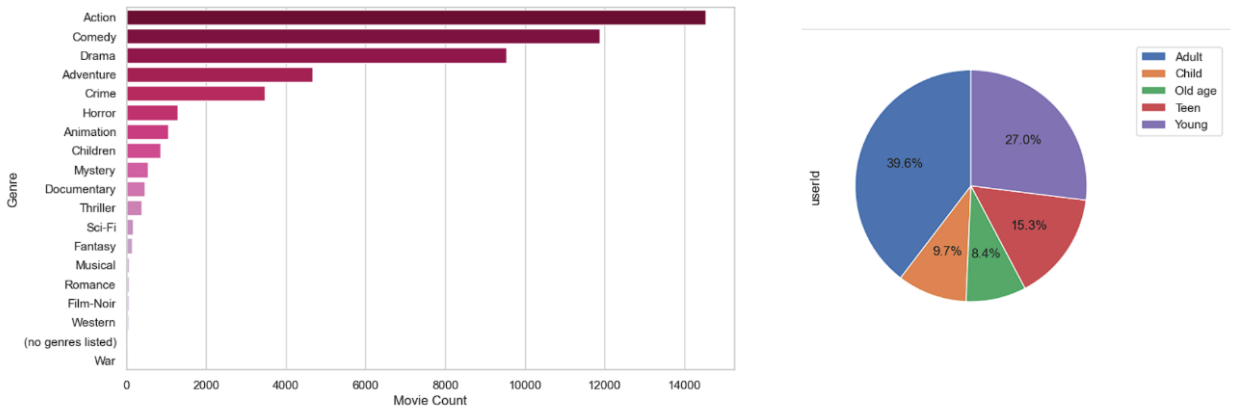

Рисунок 3.5 – Статистика по жанрам та віковим групам у «smaller_dataset» датафреймі

Дані датасету «smaller_dataset» проаналізовані та приведені у задовільний вигляд для їх подальшого розглядання та нормалізування.

Наступним кроком перед нормалізацією даних є приведення деяких стовпців у boolean вигляд для полегшення розуміння комп'ютером даних. Такими сповпцями є «main_title» та «genres». Ми починаємо зі створення нових стовпців для кожної категорії в наших даних. Наприклад, якщо у нас є такі категорії, як «молоді» та «старі» для вікових груп, а також «бойовик» і «комедія» для жанрів, ми створюємо нові стовпці під назвою «молоді», «старі», «бойовик» і «комедія». ".

У цих нових стовпцях ми ставимо «true», якщо фільм належить до цієї категорії, і «false», якщо не належить. Отже, якщо фільм підходить для молоді, ми ставимо «true» у стовпці «молоді» та «false» у стовпці «старі». Якщо це бойовик, ми ставимо «true» у стовпці «екшн» і «false» у стовпці «комедія».

Зробивши це для всіх фільмів, ми об’єднуємо ці нові стовпці і зберігаємо їх у новому датафреймі. Це дає нам новий набір даних, у якому кожен фільм представлено цифрами, а не словами. Нам більше не потрібні вихідні стовпці для вікової групи та жанру, оскільки ми замінили їх новими стовпцями, які мають «true» і «false».

Функція «pd.get_dummies()» використовується для перетворення категоріальних змінних у boolean формат, а «pd.concat()» використовується для об’єднання вихідного набору даних із новими змінними. Нарешті, «drop()» використовується для видалення вихідних категорійних стовпців, залишаючи лише фіктивні змінні в кінцевому кадрі даних.

Код вище описаного процесу представлено у листингу 3.7.

Листинг 3.7 – Код реалізації переведння текстових даних у числовий вигляд

```
dummies = pd.get_dummies(smaller_dataset[['age_group', 'main_genre']], drop_first=True)
df_movies_dum = pd.concat([smaller_dataset, dummies], axis=1)
df_movies_dum.drop(['age_group', 'main_genre'], axis=1, inplace=True)
```

Нормалізація даних це важлива складова, оскільки вона гарантує, що кожна функція однаково впливає на аналіз даних при кластеризації. Без нормалізації, стовпці з більшими діапазонами домінували б в аналізі.

Для нормалізації отриманих даних використаний інструмент «MinMaxScaler» із модуля sklearn.preprocessing, який перетворює функції шляхом масштабування кожної функції до заданого діапазону, як правило, від 0 до 1.

За допомогою функції «fit_transform()» обчислюються мінімальне та максимальне значення для кожного стовпця в наборі даних та дані масштабуються відповідно до цих мінімальних і максимальних значень, перетворюючи всі значення в наборі даних у діапазон від 0 до 1.

Після цих маніпуляцій зміни записуються у новий датафрейм «df_scaled», результат якого проілюстровано на рис. 3.6, а у лістингу 3.8 наведено код виконання цих операцій.

Лістинг 3.8 – Код реалізації нормалізації даних

```
scaler = MinMaxScaler()
df_scaled = scaler.fit_transform(df_movies_dum)
df_scaled = pd.DataFrame(df_scaled, columns=[df_movies_dum.columns])
df_scaled.describe().T
```

	count	mean	std	min	25%	50%	75%	max
age	49193.0	0.494373	0.305025	0.0	0.245283	0.433962	0.754717	1.0
movieId	49193.0	0.087730	0.170068	0.0	0.004978	0.012374	0.033510	1.0
rating	49193.0	0.686173	0.235254	0.0	0.555556	0.777778	0.888889	1.0
timestamp	49193.0	0.477912	0.309786	0.0	0.197854	0.431042	0.796734	1.0
age_group_Child	49193.0	0.097046	0.296024	0.0	0.000000	0.000000	0.000000	1.0
age_group_Old age	49193.0	0.084016	0.277415	0.0	0.000000	0.000000	0.000000	1.0
age_group_Teen	49193.0	0.153091	0.360079	0.0	0.000000	0.000000	0.000000	1.0
age_group_Young	49193.0	0.269713	0.443815	0.0	0.000000	0.000000	1.000000	1.0
main_genre_Action	49193.0	0.295571	0.456303	0.0	0.000000	0.000000	1.000000	1.0
main_genre_Adventure	49193.0	0.094831	0.292984	0.0	0.000000	0.000000	0.000000	1.0
main_genre_Animation	49193.0	0.021304	0.144397	0.0	0.000000	0.000000	0.000000	1.0
main_genre_Children	49193.0	0.017462	0.130986	0.0	0.000000	0.000000	0.000000	1.0
main_genre_Comedy	49193.0	0.241579	0.428045	0.0	0.000000	0.000000	0.000000	1.0

Рисунок 3.6 –Датасет «df_scaled»

Набір даних, який зберігається у датасеті «df_scaled» готовий для кластеризації методами k-середніх, DBSCAN та ієрархічним методом «згори-вниз»

3.3 Дослідження методу кластеризації k-середніх для реалізації рекомендаційної функції мережевої системи кінотеатрів

Перед початком кластеризації даних методом k-середніх для визначення оптимальної кількості кластерів (k) використано показник «silhouette score» або

«оцінка силуету». Обчислення «silhouette score» для різних значень k дає можливість обрати найкращу кількість кластерів. Чим більше значення «silhouette score», тим найкраще кластеризація себе покаже при обраній кількості кластерів.

Для вибору кількості кластерів був обраний проміжок для розрахунку від 5 до 101 кластерів, з кроком 5. Так як є обчислювані обмеження, розглядати більшу кількість кластерів не мало сенсу. За допомогою методу «KMeans» бібліотеки sklearn.cluster у функцію передаються такі параметри:

– «n_clusters=n_clusters» встановлює кількість кластерів, у які алгоритм намагатиметься згрупувати дані. Значення встановлюється поточним значенням n_clusters у циклі;

– «random_state=8» встановлює початкове значення для генератора випадкових чисел, що використовується алгоритмом KMeans. Це робиться для того, щоб результати алгоритму були відтворюваними;

– «max_iter=10000» встановлює максимальну кількість ітерацій для алгоритму KMeans. Це кількість разів, коли алгоритм перемістить центроїди (центри кластерів), щоб мінімізувати відстань між точками та їхніми відповідними центроїдами.

За допомогою методу «fit_predict()» модель KMeans співвідноситься з масштабованими даними з датасету «df_scaled», а потім прогнозує кластер для кожної точки даних. Метод «fit_predict» спочатку обчислює центроїди кластерів шляхом підгонки моделі до даних, а потім призначає кожну точку даних найближчому центроїду. Отримані мітки кластера зберігаються в «model_labels» датафреймі.

Останнім кроком є обчислення «silhouette score» вирішення, при якій кількості кластерів «silhouette score» буде найбільшою. Оцінка силуету є мірою того, наскільки об’єкт схожий на власний кластер порівняно з іншими кластерами. Оцінка розраховується з використанням середньої відстані між кластерами (a) та середньої відстані до найближчого кластера (b) для кожного зразка. Оцінка силуету коливається від -1 до 1, де високе значення вказує на те, що об’єкт добре збігається з

власним кластером і погано збігається з сусідніми кластерами. Середня оцінка силуету для всіх точок даних зберігається в «silhouette_avg» датафреймі.

У лістингу 3.9 наведено код реалізації розрахунку «silhouette score» показника для кожної кількості кластерів.

Лістинг 3.9 – Код реалізації розрахунку «silhouette score»

```
range_n_clusters = range(5, 101, 5)
for n_clusters in range_n_clusters:
        models1      =      KMeans(n_clusters=n_clusters,      random_state=8,
max_iter=10000)
    model_labels = models1.fit_predict(df_scaled)
    silhouette_avg = silhouette_score(df_scaled, model_labels)
    print(
        "For n_clusters =", n_clusters,
        "The average silhouette_score is :", silhouette_avg)
```

Результатом виконання коду є співвідношення кластерів та їх «silhouette score», який представлено на рис. 3.7.

```
For n_clusters = 5 The average silhouette_score is : 0.20820132284781745
For n_clusters = 10 The average silhouette_score is : 0.2996845137240668
For n_clusters = 15 The average silhouette_score is : 0.3576170421894847
For n_clusters = 20 The average silhouette_score is : 0.4038611205246957
For n_clusters = 25 The average silhouette_score is : 0.4570481243251876
For n_clusters = 30 The average silhouette_score is : 0.43670030205723936
For n_clusters = 35 The average silhouette_score is : 0.4143897138649038
For n_clusters = 40 The average silhouette_score is : 0.39431862740499235
For n_clusters = 45 The average silhouette_score is : 0.41160929924233436
For n_clusters = 50 The average silhouette_score is : 0.424534116773073
For n_clusters = 55 The average silhouette_score is : 0.38522673965490656
For n_clusters = 60 The average silhouette_score is : 0.3909788737080982
For n_clusters = 65 The average silhouette_score is : 0.39627736803472624
For n_clusters = 70 The average silhouette_score is : 0.397216154071785
For n_clusters = 75 The average silhouette_score is : 0.39660907968239373
For n_clusters = 80 The average silhouette_score is : 0.3831026635764201
For n_clusters = 85 The average silhouette_score is : 0.3714843061000083
For n_clusters = 90 The average silhouette_score is : 0.37116302145481467
For n_clusters = 95 The average silhouette_score is : 0.37348520954758135
For n_clusters = 100 The average silhouette_score is : 0.3703631712210271
```

Рисунок 3.7 – Значення «silhouette score» для кожної кількості кластерів

На рис. 3.7 видно, що найближче до 1 «silhouette score» при кількості кластерів 25. Далі значення поступово зменшуються і майже однакові. Тому, для кластеризації даних за методом k-середніх було обрано k=25.

Для проведення кластеризації за алгоритмом k-середніх використовується функція KMeans, у передаються такі параметри:

– «n_clusters=25» встановлює кількість кластерів, у які алгоритм намагатиметься згрупувати дані. Значення встановлено на 25, тобто алгоритм спробує згрупувати дані в 25 кластерів.

– «algorithm='lloyd'» встановлює алгоритм для кластеризації KMeans. Алгоритм Ллойда, також відомий як базовий алгоритм KMeans, є типовим алгоритмом, який використовується KMeans scikit-learn.

Після налагодження параметрів для кластеризації функція «fit_predit» підгоняє модель KMeans до масштабованих даних датафрейму «df_scaled». Отримані мітки кластера зберігаються в «cluster_labels» датафреймі.

Останнім кроком є додавання прогнозованих міток кластера до вихідного DataFrame (smaller_dataset). Він створює новий стовпець під назвою «cluster» у DataFrame та призначає йому мітки кластера. Кожен рядок у DataFrame представляє точку даних, а стовпець «cluster» вказує, до якого кластера належить точка даних відповідно до кластеризації KMeans.

У листингу 3.10 представлено реалізацію кластеризації за методом k-середніх.

Лістинг 3.10 – Код реалізації кластеризації даних методом k-середніх

```
kmeans = KMeans(n_clusters=25, algorithm='lloyd')
cluster_labels = kmeans.fit_predict(df_scaled)
smaller_dataset['cluster'] = cluster_labels
```

Для наступного дослідження нам потрібно визначити кластер та користувача, на якому ми хочемо випробувати розрахунок прогнозованої оцінки фільмів.

Для усіх методів кластеризації для початкового дослідження був обраний користувач з номером 284. За результатом сформованих кластерів методом k-середніх був обраний кластер під номером 18.

Часткова візуалізація кластеру 18 приведена на рис. 3.8, так як кластери зберігають великий обсяг фільмів, користувачів та їх оцінок і вмістити усе неможливо на один малюнок.

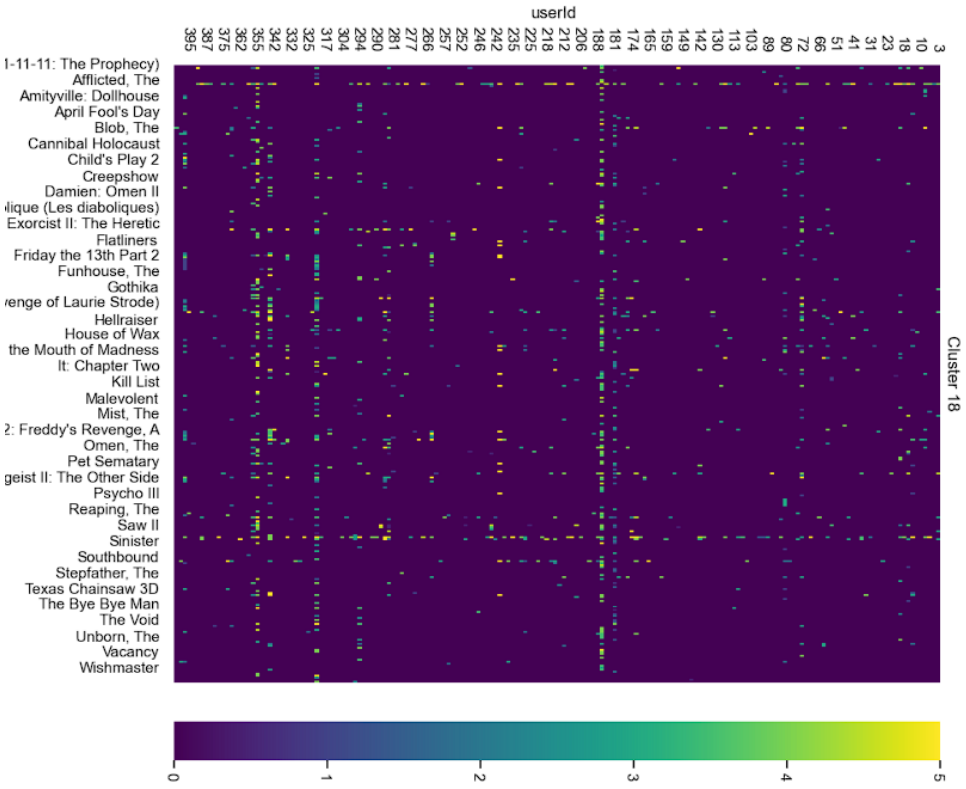

Рисунок 3.7 – Значення «silhouette score» для кожної кількості кластерів

Для того щоб подивитись вміст кластеру 18 створено новий DataFrame «cluster_18», який містить лише рядки з «smaller_dataset», де стовпець «cluster» дорівнює 18. Функція «loc» використовується для доступу до групи рядків і стовпців за мітками або логічним масивом. У цьому випадку він використовується для фільтрації DataFrame на основі умови.

Далі, для того щоб проілюструвати залежність фільмів, користувачів та їх оцінок створено зведену таблицю «cluster_18_ds». Зведена таблиця впорядкована за допомогою «userId» як індексу вздовж рядків, «main_title» як стовпців і «rating» як значень, які збираються. У лістингу 3.11 представлено реалізацію коду вище описаних кроків.

Лістинг 3.11 – Код реалізації зображення кластеру 18 у вигляді датафрейму «cluster_18_ds»

```
cluster_18 = smaller_dataset.loc[smaller_dataset['cluster'] == 18]
cluster_18_ds=cluster_18.pivot_table(index='userId',     columns='main_title',
values='rating')
cluster_18_ds.head(10)
cluster_18_ds.fillna('').head(5)
```

Після результату зведення було отримано, що датафрейм «» має багато «NaN» значень, для того щоб позбутися їх усі такі значення були заміненні на пустий рядок за допомогою функції «fillna()», яка приведена у лістингу 3.11

На рис. 3.8 представлено результат до і після приведення даних датасету «cluster_18_ds» у бажаний вигляд.

Before

main_title	11-11-11 (11-11-11: The Prophecy)	28 Weeks Later	3 Extremes (Three... Extremes) (Saam gaang yi)	30 Days of Night	47 Meters Down	6 Souls (Shelter)	ABCs of Death, The	Absentia	Afflicted, The	Alien	...	Virus	When a Stranger Calls	Wicker Man, The	Wishmaster	Wolf Creek 2	Wrong Turn
userId																	
3	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	4.0	...	NaN	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2.5	...	NaN	NaN	NaN	NaN	NaN	NaN
8	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	4.0	...	NaN	NaN	NaN	NaN	NaN	NaN
9	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	4.0	...	NaN	NaN	NaN	NaN	NaN	NaN
10	NaN	4.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN

After

main_title	11-11-11 (11-11-11: The Prophecy)	28 Weeks Later	3 Extremes (Three... Extremes) (Saam gaang yi)	30 Days of Night	47 Meters Down	6 Souls (Shelter)	ABCs of Death, The	Absentia	Afflicted, The	Alien	...	Virus	When a Stranger Calls	Wicker Man, The	Wishmaster	Wolf Creek 2	Wrong Turn
userId																	
3										4.0	...						
4										2.5	...						
8										4.0	...						
9										4.0	...						
10		4.0									...						

Рисунок 3.8 – Приведення датафрейму «cluster_18_ds» у бажаний вигляд

Для того щоб передбачити оцінку за фільм у користувача використовується функція «mean()», яка рахує середнє значення за кожним фільмом по оцінками, які залишили користувачі. Таким чином можна зробити прогноз яка буде оцінка у цього фільму для іншого користувача, який не оцінював фільми.

Наступним кроком є застосування цієї інформації для певного користувача і зробити передбачення фільмів, які можна було б зарекомендувати користувачу. Код реалізації представлено у лістингу 3.12.

Лістинг 3.12 – Код реалізації передбачення оцінок для користувача

```
user_id = 284
user_284_ratings  = cluster_18_ds.loc[user_id, :]
user_284_unrated_movies =  user_284_ratings[user_284_ratings.isnull()
avg_ratings   =   pd.concat([user_284_unrated_movies,   cluster_18_ds.mean()],
axis=1, join='inner').loc[:,0]
avg_ratings.sort_values(ascending=False)[:20]
```

На рис. 3.9 зображені результати роботи декількох процесів:

– перший рисунок зліва відображає середні знаачення оцінок по кожному фільму;

– другий рисунок посередині відображає перші 20 оцінок від користувача 284, для того щоб розуміти які фільми були оцінені, а які ні;

– третій рисунок справа відображає роботу коду з лістингу 3.12, де були пораховані середні значення оцінок фільмів у кластері та зарекомендовані користувачу 284 починаючи з високооцінених фільмів, які він не оцінював.

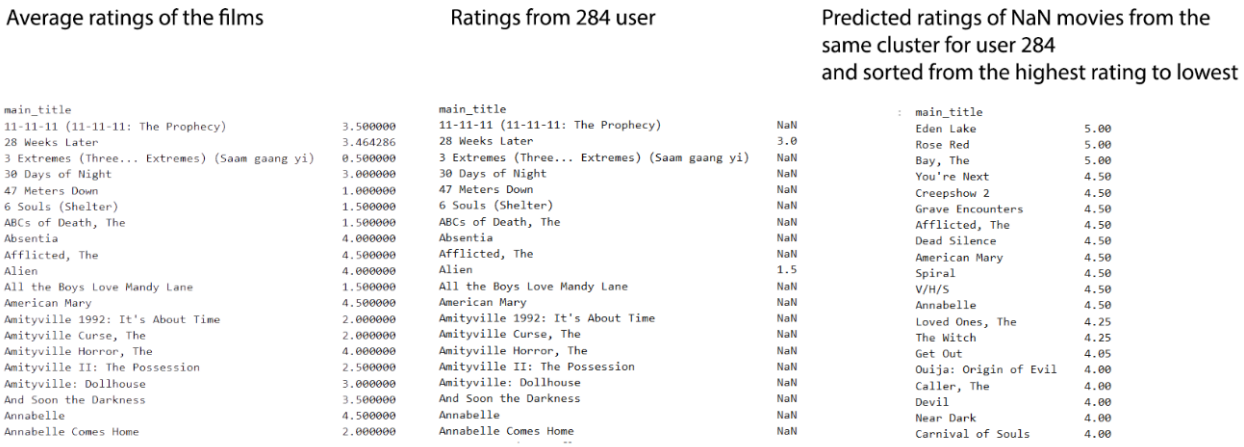

Рисунок 3.9 – Результат прототипу передбачення фільмів для користувача 234

Таким чином, при дослідженні методу кластеризації k-середніх було обрано найкращий параметр для кількості кластерів і дані були згруповані у 25 кластерів. Метод k-середніх показав швидкий час групування даних та їх оброблення при величині датасету 50 000 рядків.

3.4 Дослідження ієрархічного методу кластеризації «згори-вниз» для реалізації рекомендаційної функції мережевої системи кінотеатрів

Перед початком кластеризації даних агломеративним методом для визначення оптимальної кількості кластерів (n) використано показник «silhouette score», який будо попередньо детально описано у пункті 3.2. Обчислення «silhouette score» для різних значень n дає можливість обрати найкращу кількість кластерів.

Для вибору кількості кластерів був обраний проміжок для розрахунку кластерів той самий з 5 до 101, з кроком 5. Агломеративним метод не тягнув обробку повного датасету з інформацією у зв’язку з технічною лімітацією фізичного забезпечення, тому було прийнято рішення роздробити датасет та підрахувати показник оцінки силуету на зменшенному датасеті. Так як є обчислювані обмеження, розглядати більшу кількість кластерів не мало сенсу.

У першу чергу, як було написано вижче, створено меншу підмножину з вихідного набору даних для початкового дослідження. Це зроблено для економії обчислювальних ресурсів. Функція «np.random.choice» використовується для випадкового вибору певної кількості унікальних індексів, визначених «subset_size» параметром (у цьому випадку кількість рядків у новому датасеті буде 20 000). Аргумент «replace=False» гарантує, що вибір виконується без заміни, тобто всі вибрані індекси є унікальними. Потім ці індекси використовуються для вибору відповідних рядків із масштабованого DataFrame «df_scaled», що призводить до випадкової підмножини даних, що зберігається в «df_subset» датасет.

Визначений діапазон потенційних номерів кластерів. Функція діапазону в Python створює послідовність чисел від початкового значення до кінцевого значення, збільшуючи значення кроку. Тут початок — 5, зупинка – 101, крок – 5, тому послідовність буде [5, 10, 15, ..., 100]. Ця послідовність представляє різну кількість кластерів, які будуть перебиратися для агломеративного методу.

«AgglomerativeClustering» функція із модуля «sklearn.cluster» використовується для виконання ієрархічної кластеризації за агломеративним методом або методом «згори-вниз». Екземпляр цього класу створюється з поточною кількістю кластерів (n_clusters), а метод «fit_predict» викликається в «df_subset» датасеті для виконання кластеризації та повернення міток кластера для кожної точки даних у підмножині.

Оцінка силуету є показником того, наскільки спостереження схоже на власний кластер порівняно з іншими кластерами. Функція «silhouette_score» з модуля «sklearn.metrics» обчислює цю оцінку для поточної кластеризації. Вищий бал означає, що точки даних добре згруповані. Якщо оцінка силуету для поточної кількості кластерів краща, ніж попередньо записана найкраща оцінка, код оновлює найкращу оцінку, найкращу кількість кластерів і найкращі мітки кластерів. Це робиться за допомогою простих операцій присвоювання.

У листингу 3.13 наведено код реалізації попередньо описаних кроків для розрахунку «silhouette_score» для агломеративного методу кластеризації.

Листинг 3.13 – Код реалізації підрахунку «silhouette_score» для агломеративного методу

```
subset_size = 20000
subset_indices = np.random.choice(df_scaled.shape[0], subset_size, replace=False)
df_subset = df_scaled.iloc[subset_indices]
range_n_clusters = range(5, 101, 5)
best_n_clusters = None
best_silhouette_score = -1
best_labels = None
for n_clusters in range_n_clusters:
    clusterer = AgglomerativeClustering(n_clusters=n_clusters)
    cluster_labels_subset = clusterer.fit_predict(df_subset)
    silhouette_avg = silhouette_score(df_subset, cluster_labels_subset)
    print(f"For n_clusters = {n_clusters}, the average silhouette_score is : {silhouette_avg}")
    if silhouette_avg > best_silhouette_score:
        best_silhouette_score = silhouette_avg
        best_n_clusters = n_clusters
        best_labels = cluster_labels_subset
```

Результат роботи підрахунку оцінки силуету для зазначеної кількості кластерів продемонстровано на рис. 3.10

```
For n_clusters = 5, the average silhouette_score is : 0.2606521634689581
For n_clusters = 10, the average silhouette_score is : 0.3237206673961871
For n_clusters = 15, the average silhouette_score is : 0.40340850266726336
For n_clusters = 20, the average silhouette_score is : 0.4478326668510906
For n_clusters = 25, the average silhouette_score is : 0.48711364636778026
For n_clusters = 30, the average silhouette_score is : 0.4674491625313077
For n_clusters = 35, the average silhouette_score is : 0.44181007685797213
For n_clusters = 40, the average silhouette_score is : 0.4552172802013143
For n_clusters = 45, the average silhouette_score is : 0.4417247592811158
For n_clusters = 50, the average silhouette_score is : 0.44109456951208686
For n_clusters = 55, the average silhouette_score is : 0.4122151908265053
For n_clusters = 60, the average silhouette_score is : 0.3968300712120645
For n_clusters = 65, the average silhouette_score is : 0.39550360931779094
For n_clusters = 70, the average silhouette_score is : 0.3933611208541341
For n_clusters = 75, the average silhouette_score is : 0.38848853312166054
For n_clusters = 80, the average silhouette_score is : 0.3727905625904602
For n_clusters = 85, the average silhouette_score is : 0.3685372009539762
For n_clusters = 90, the average silhouette_score is : 0.36554423273035846
For n_clusters = 95, the average silhouette_score is : 0.3671164170404751
For n_clusters = 100, the average silhouette_score is : 0.36675742533704353
Best number of clusters: 25
Best silhouette score: 0.48711364636778026
```

Рисунок 3.10 – Визначення «silhouette_score» і оптимального значення кластерів для агломеративного методу

Найкращим параметром для кількості кластерів є 25. Тому для кластеризації за агломеративним методом використовується кількість кластерів 25.

Після того, як у циклі перевірено всю потенційну кількість кластерів, код використовує найкращу кількість кластерів для створення нового екземпляра класу «AgglomerativeClustering». Потім він адаптує цю модель до всього набору даних «df_scaled» і отримує мітки кластера для кожної точки даних у повному наборі даних за допомогою методу «fit_predict».

Нарешті, мітки кластера для кожної точки даних додаються до DataFrame «smaller_dataset» як новий стовпець під назвою «cluster». Цей DataFrame тепер містить додаткову частину інформації для кожної точки даних: кластер, до якого він належить відповідно до найкращої знайденої агломеративної кластеризації.

Вище описані кроки приведені у листингу 3.14.

Листинг 3.14 – Код реалізації алгломеративної кластеризації при найкращьому показнику кластерів 25

```
clusterer_full = AgglomerativeClustering(n_clusters=best_n_clusters)
cluster_labels_full = clusterer_full.fit_predict(df_scaled)

# Calculate silhouette score for the full dataset
silhouette_avg_full = silhouette_score(df_scaled, cluster_labels_full)
print(f"The average silhouette_score for the full dataset is : {silhouette_avg_full}")

smaller_dataset['cluster'] = cluster_labels_full
```

Для наступного кроку нам потрібно визначити кластер 23 та користувача 284, на якому випробувано розрахунок прогнозованої оцінки фільмів.

Часткова візуалізація кластеру 23 приведена на рис. 3.11, так як кластери зберігають великий обсяг фільмів, користувачів та їх оцінок.

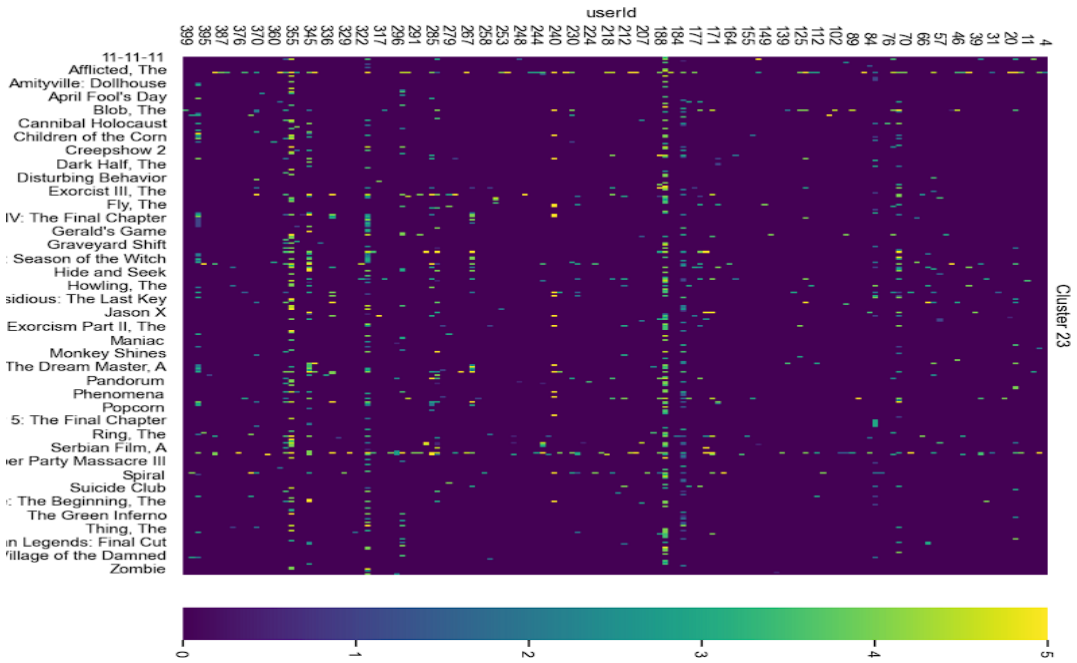

Рисунок 3.11 – Візуалізація кластеру 18, результуючого з агломеративного методу

Наступними кроками є ті ж самі, що були описані у пункті 3.4 для k-середніх кластеризації:

– на обраному кластері 23 і обраному користувачеві 284 показати його неоцінені фільми та оцінені;

– за оціненими фільмами іншими користувачами спрогнозувати оцінку неоціненим фільм користувача 284;

Таким чином, при дослідженні агломеративного методу кластеризації було обрано найкращий параметр кількості кластерів. Дані згруповані у 25 кластерів. Агломеративний метод був не дуже ефективним для великого обсягу даних, але показав схожі результати формування кластерів як і при k-середніх.

3.5 Дослідження методу кластеризації DBSCAN для реалізації рекомендаційної функції мережевої системи кінотеатрів

Для початку аналізу алгоритму DBSCAN (просторова кластеризація додатків на основі щільності з шумом) ініціалізувалося два масиви, «eps_array» і «min_samples_array». Ці масиви представляють діапазон значень для двох ключових параметрів алгоритму DBSCAN:

– «eps» або епсилон – це максимальна відстань між двома зразками, щоб вони розглядалися як такі, що знаходяться в одному сусідстві. Функція «np.arange» використовується для створення масиву значень від 0,2 до 1,0 з кроком 0,1.

– «min_samples» – це кількість зразків у околиці для точки, яка розглядається як основна точка. Це включає саму точку. Функція «np.arange» використовується для створення масиву значень від 55 до 75 з кроком 5.

Перевіряючи кожну комбінацію значень «eps» і «min_samples», створюється екземпляр класу «DBSCAN», передаючи поточні значення цих параметрів як аргументи.

```
For eps = 0.2 For min_samples = 55 Count clusters = 33 The average silhouette_score is : 0.3802596301843489
For eps = 0.2 For min_samples = 60 Count clusters = 35 The average silhouette_score is : 0.3492913102832032
For eps = 0.2 For min_samples = 65 Count clusters = 36 The average silhouette_score is : 0.33351634828377924
For eps = 0.2 For min_samples = 70 Count clusters = 33 The average silhouette_score is : 0.3297818825698729 5
For eps = 0.30000000000000004 For min_samples = 55 Count clusters = 44 The average silhouette_score is : 0.4924327355655548
For eps = 0.30000000000000004 For min_samples = 60 Count clusters = 44 The average silhouette_score is : 0.4885134717283791
For eps = 0.30000000000000004 For min_samples = 65 Count clusters = 41 The average silhouette_score is : 0.4829282241960272
For eps = 0.30000000000000004 For min_samples = 70 Count clusters = 39 The average silhouette_score is : 0.4802080648558711
For eps = 0.4000000000000001 For min_samples = 55 Count clusters = 53 The average silhouette_score is : 0.5192232493814959
For eps = 0.4000000000000001 For min_samples = 60 Count clusters = 50 The average silhouette_score is : 0.5132061469498796
For eps = 0.4000000000000001 For min_samples = 65 Count clusters = 49 The average silhouette_score is : 0.5101048901456959
For eps = 0.4000000000000001 For min_samples = 70 Count clusters = 46 The average silhouette_score is : 0.5069682515028209
For eps = 0.5000000000000001 For min_samples = 55 Count clusters = 48 The average silhouette_score is : 0.5360606939949374
For eps = 0.5000000000000001 For min_samples = 60 Count clusters = 48 The average silhouette_score is : 0.5342023468948183
For eps = 0.5000000000000001 For min_samples = 65 Count clusters = 48 The average silhouette_score is : 0.5320562582516263
For eps = 0.5000000000000001 For min_samples = 70 Count clusters = 46 The average silhouette_score is : 0.5292838620925733
For eps = 0.6000000000000001 For min_samples = 55 Count clusters = 48 The average silhouette_score is : 0.5374113783662713
For eps = 0.6000000000000001 For min_samples = 60 Count clusters = 48 The average silhouette_score is : 0.5372565507826287
For eps = 0.6000000000000001 For min_samples = 65 Count clusters = 48 The average silhouette_score is : 0.537128469696689
For eps = 0.6000000000000001 For min_samples = 70 Count clusters = 47 The average silhouette_score is : 0.535878493986402
For eps = 0.7000000000000002 For min_samples = 55 Count clusters = 48 The average silhouette_score is : 0.5381442123958731
For eps = 0.7000000000000002 For min_samples = 60 Count clusters = 48 The average silhouette_score is : 0.5381442123958731
For eps = 0.7000000000000002 For min_samples = 65 Count clusters = 48 The average silhouette_score is : 0.5381442123958731
For eps = 0.7000000000000002 For min_samples = 70 Count clusters = 47 The average silhouette_score is : 0.5367512483186523
For eps = 0.8000000000000003 For min_samples = 55 Count clusters = 50 The average silhouette_score is : 0.5394095338164043
For eps = 0.8000000000000003 For min_samples = 60 Count clusters = 49 The average silhouette_score is : 0.5388543073752868
For eps = 0.8000000000000003 For min_samples = 65 Count clusters = 48 The average silhouette_score is : 0.5382976463717557
For eps = 0.8000000000000003 For min_samples = 70 Count clusters = 48 The average silhouette_score is : 0.5382826210066777
For eps = 0.9000000000000001 For min_samples = 55 Count clusters = 51 The average silhouette_score is : 0.5401703763142139
For eps = 0.9000000000000001 For min_samples = 60 Count clusters = 49 The average silhouette_score is : 0.5388543073752868
For eps = 0.9000000000000001 For min_samples = 65 Count clusters = 49 The average silhouette_score is : 0.5388543073752868
For eps = 0.9000000000000001 For min_samples = 70 Count clusters = 48 The average silhouette_score is : 0.5382976463717557
```

Рисунок 3.12 – Візуалізація кластеру 18, результуючого з агломеративного методу

Параметри «eps» та «min_samples», які призводять до найкращої кластеризації, виміряної за допомогою оцінки силуету, є 0.9 для «eps» та «min_samples» 55. На рис. 3.12 показано результат підрахунку «silhouette score» для значень епсілон та кількості зразків, а листинг коду наведено у листингу 3.15.

Листинг 3.15 – Код реалізації підрахунку «silhouette_score» для методу DBSCAN

```
eps_array = np.arange(0.2, 1.0, 0.1)
min_samples_array = np.arange(55, 75, 5)
for eps in eps_array:
  for min_samples in min_samples_array:
    dbscan_clusterer = DBSCAN(eps=eps,
min_samples=min_samples).fit(df_scaled)
    dbscan_cluster_labels = dbscan_clusterer.labels_
    if len(set(dbscan_cluster_labels)) == 1:
        continue
    silhouette_avg = silhouette_score(df_scaled, dbscan_cluster_labels)
    print("For eps =", eps,
          "For min_samples =", min_samples,
          "Count clusters =", len(set(dbscan_cluster_labels)),
          "The average silhouette_score is :", silhouette_avg)
```

Найкращі показники адаптуються у модель до масштабованих даних «df_scaled» за допомогою методу «fit» і отримуються мітки кластера, призначені алгоритмом DBSCAN за допомогою атрибута «labels_». Код реалізації цих кроків наведено у листингу 3.16

Листинг 3.16 – Код реалізації DBSCAN кластеризації при найкращих параметрах

```
dbscan_clusterer = DBSCAN(eps=0.9, min_samples=55).fit(df_scaled)
dbscan_cluster_labels = dbscan_clusterer.labels_
smaller_dataset['cluster'] = dbscan_cluster_labels
```

Для наступного кроку нам потрібно визначити кластер 37 та користувача 284, на якому випробувано розрахунок прогнозованої оцінки фільмів, де візуалізація кластеру 37 приведена на рис. 3.13.

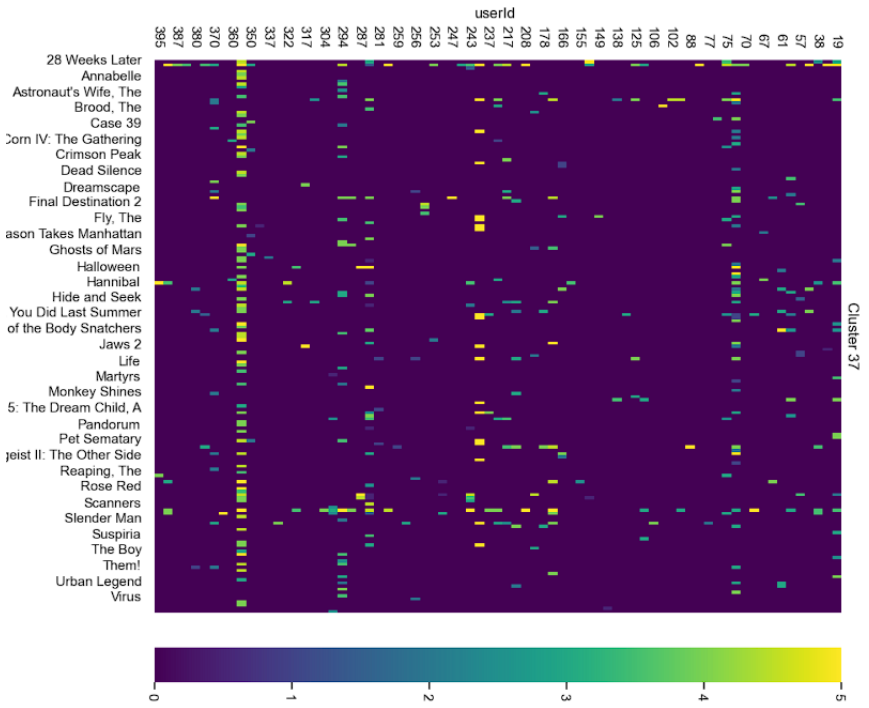

Рисунок 3.13 – Візуалізація кластеру 37 методу DBSCAN

Наступними кроками є ті ж самі, що були описані у пункті 3.5 для агломеративної кластеризації:

– на обраному кластері 37 і обраному користувачеві 284 показати його неоцінені фільми та оцінені;

– за оціненими фільмами іншими користувачами спрогнозувати оцінку неоціненим фільм користувача 284;

Таким чином, при дослідженні методу DBSCAN кластеризації було обрано найкращі параметри «eps» та «min_samples» для отримання найкращих результатів кластеризації. Дані згруповані у 51 кластер. Метод DBSCAN був ефективним та показав інші, більш чіткі результати ніж попередні методи.

3.6 Порівняння результатів дослідження методів кластеризації для реалізації рекомендаційної функції мережевої системи кінотеатрів

Методи K-середніх та агломеративна найкраще працюють із 25 кластерами. Це означає, що відповідно до оцінки силуету, розподіл даних на 25 груп призвів до найбільш узгоджених і чітких кластерів.

K-середніх та агломеративна кластеризація є методами поділу, тобто вони ділять дані на підмножини або кластери, що не перекриваються. Основна відмінність між ними полягає в тому, як вони формують ці кластери: K- середніх базується на центроїді та намагається мінімізувати дисперсію всередині кожного кластера, тоді як агломеративна кластеризація є ієрархічним методом, який будує ієрархію кластерів, а потім розбиває цю ієрархію на певному рівні.

Той факт, що обидва методи показали найкращі результати з 25 кластерами, свідчить про те, що це хороший розподіл даних, принаймні відповідно до використаного критерію оцінки. Однак, агломеративна кластеризація потребувала дроблення даних перед використанням, так як він потребує великих обчислювальних витрат. Тому, агломеративна кластеризація хоч і дає схожі результати, але дуже непрактична при даних, які будуть динамічно збільшуватися.

Метод DBSCAN показав найкращі результати зі значенням eps 0,9 і мінімальним розміром вибірки 55, що призвело до 51 кластера. Це більша кількість кластерів, ніж для K-середніх і агломеративної кластеризації, що свідчить про те, що DBSCAN знаходить більш гранульовану структуру в даних.

Порівнюючи метод DBSCAN та K-середніх можна сказати, що найкращий результат дає метод DBSCAN за «silhouette score», хоч і дає більшу кількість кластерів. У той час коли для агломеративного методу та K-середніх оцінка силуету майже не змінювалась при збільшенні кількості кластів.

Також DBSCAN знайшов більшу кількість кластерів, що може означати, що в даних є багато невеликих щільних областей, які не були ідентифіковані як окремі кластери за допомогою K-середніх і агломеративної кластеризації.

Метод DBSCAN через наступні переваги, які були виявлені у ході дослідженні:

– виявляє щільні області, які не виявляються K-середніх і агломеративним методами;

– має вищий показник «silhouette» ніж K-середніх і агломеративної, у яких збільшення кластерів не дає кращого показника оцінки силуету;

– метод не потребує «разщіплення» даних на декілька кусочків перед кластеризацією, а потім збирання їх у один датасет.

Отже, для реалізації рекомендаційної функції прийнято рішення використовувати метод DBSCAN.

3.7 Реалізації рекомендаційної функції мережевої системи кінотеатрів за допомогою методу DBSCAN та порівняння результатів роботи функції, реалізованих за методами k-середніх, агломеративного та DBSCAN

За результатами дослідження методів кластеризації створена функція «recommend_movies(user_id, main_genre)», яка призначена для рекомендації фільмів користувачеві на основі членства користувача в кластері та визначеного жанру. На рис. 3.14 зображено етапи роботи рекомендаційної функції фільмів.

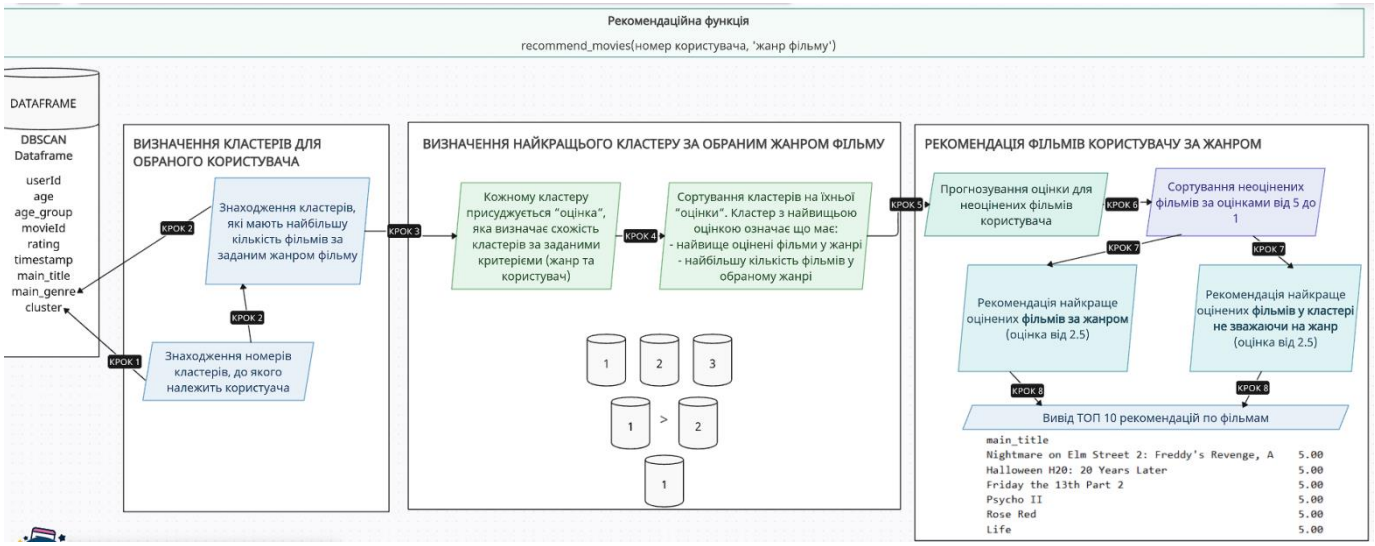

Рисунок 3.14 – Рекомендаційна функція «recommend_movies (user_id, main_genre)»

Першим кроком функції «recommend_movies (user_id, main_genre)» є визначення кластерів, до яких належить вказаний користувач, тобто «user_id». Це робиться шляхом встановлення в DataFrame «smaller_dataset» лише тих рядків, де «userId» атрибут відповідає наданому «user_id», а потім отримує унікальні значення кластерів зі стовпчику «cluster» для цих рядків.

Далі для кожного кластера, до якого належить користувач, функція обчислює оцінку на основі кількості фільмів у цьому кластері та зазначеного жанру, а також їх середнього рейтингу. Ця оцінка є зваженою сумою кількості жанрів і середнього рейтингу з рівними вагами 0,5. Потім функція сортує кластери в порядку спадання їх балів.

Потім функція перебирає відсортовані кластери та для кожного кластера рекомендує фільми, які користувач ще не оцінив. Спершу рекомендуються фільми зазначеного жанру. Обчислюється середній рейтинг для кожного фільму (для всіх користувачів у кластері), а потім сортуються ці фільми в порядку спадання їхнього середнього рейтингу. Він рекомендує 20 найкращих фільмів із середнім рейтингом понад 2,5. Якщо фільмів із зазначеного жанру не знайдено, процес повторюється для всіх фільмів, незалежно від їхнього жанру. Якщо фільмів не знайдено, середній поріг рейтингу знижується, щоб рекомендувати 20 найкращих фільмів. Нарешті, якщо жоден фільм не може бути рекомендований навіть після зниження середнього порогу оцінки, функція повертає порожню серію pandas.

У листингу 3.17 подано код реалізації рекомендаційної функції за мтеодом DBSCAN

Листинг 3.17 – Код реалізації рекомендаційної функції за мтеодом DBSCAN

```
def recommend_movies(user_id, main_genre):
    df_reset = smaller_dataset.reset_index()
    user_clusters          =          df_reset[df_reset['userId']          ==
user_id]['cluster'].unique()
    print(f'User {user_id} belongs to clusters: {user_clusters}')
    sorted_clusters = []
    for cluster in user_clusters:
```

```
        cluster_df = df_reset[df_reset['cluster'] == cluster]
        genre_df                                                    =
cluster_df[cluster_df['main_genre'].str.contains(main_genre)]
        genre_count = genre_df.shape[0]
        average_rating = genre_df['rating'].mean()
        score = genre_count * 0.5 + average_rating * 0.5
        sorted_clusters.append((cluster, score))
    sorted_clusters = [c for c in sorted_clusters if not np.isnan(c[1])]
    sorted_clusters.sort(key=lambda x: x[1], reverse=True)
    print(f'Sorted clusters: {sorted_clusters}')
    for cluster, score in sorted_clusters:
        print(f'Processing cluster {cluster} with score {score}')
        cluster_df = df_reset[df_reset['cluster'] == cluster]
        genre_df                                                    =
cluster_df[cluster_df['main_genre'].str.contains(main_genre)]
        genre_df                  =                 genre_df.groupby(['userId',
'main_title']).rating.mean().reset_index()
        genre_pivot  =  genre_df.pivot(index='userId',  columns='main_title',
values='rating')
        if user_id in genre_pivot.index:
            user_ratings = genre_pivot.loc[user_id]
            unrated_movies                                          =
user_ratings[user_ratings.isnull()].index.tolist()
            if unrated_movies:
                mean_ratings = genre_pivot[unrated_movies].mean()
                high_rated_movies      =      mean_ratings[mean_ratings      >
2.5].sort_values(ascending=False)
                if not high_rated_movies.empty:
                    return high_rated_movies[:20]
        cluster_df               =               cluster_df.groupby(['userId',
'main_title']).rating.mean().reset_index()
        cluster_pivot            =            cluster_df.pivot(index='userId',
columns='main_title', values='rating')
        if user_id in cluster_pivot.index:
            user_ratings = cluster_pivot.loc[user_id]
            unrated_movies                                          =
user_ratings[user_ratings.isnull()].index.tolist()
            if unrated_movies:
                mean_ratings                                        =
cluster_pivot[unrated_movies].mean().sort_values(ascending=False)
                return mean_ratings[:20]
    for cluster, score in sorted_clusters:
        print(f'Processing cluster {cluster} with score {score}')
        cluster_df = df_reset[df_reset['cluster'] == cluster]
        cluster_df               =               cluster_df.groupby(['userId',
'main_title']).rating.mean().reset_index()
        cluster_pivot            =            cluster_df.pivot(index='userId',
columns='main_title', values='rating')
```

```
        if user_id in cluster_pivot.index:
            user_ratings = cluster_pivot.loc[user_id]
            unrated_movies                                                    =
user_ratings[user_ratings.isnull()].index.tolist()
            if unrated_movies:
                mean_ratings                                                  =
cluster_pivot[unrated_movies].mean().sort_values(ascending=False)
                high_rated_movies = mean_ratings[mean_ratings > 2.5]
                if not high_rated_movies.empty:
                    return high_rated_movies[:20]
    for cluster, score in sorted_clusters:
        print(f'Processing cluster {cluster} with score {score}')
        cluster_df = df_reset[df_reset['cluster'] == cluster]
        cluster_df              =              cluster_df.groupby(['userId',
'main_title']).rating.mean().reset_index()
        cluster_pivot           =           cluster_df.pivot(index='userId',
columns='main_title', values='rating')
        if user_id in cluster_pivot.index:
            user_ratings = cluster_pivot.loc[user_id]
            unrated_movies                                                    =
user_ratings[user_ratings.isnull()].index.tolist()
            if unrated_movies:
                mean_ratings                                                  =
cluster_pivot[unrated_movies].mean().sort_values(ascending=False)
                return mean_ratings[:20]
    return pd.Series()
```

Таким чином, ця функція використовує комбінацію спільної фільтрації (враховуючи членство користувача в кластері) і фільтрації на основі вмісту (враховуючи жанр фільму), щоб рекомендувати фільми користувачеві. Він визначає пріоритетність фільмів із жанру, якому подобається користувач, і з кластерів, до яких належить користувач і які мають велику кількість фільмів із цього жанру та високий середній рейтинг.

При різних видах кластеризації були різні рекомендації, яка приведені на рис. 3.15

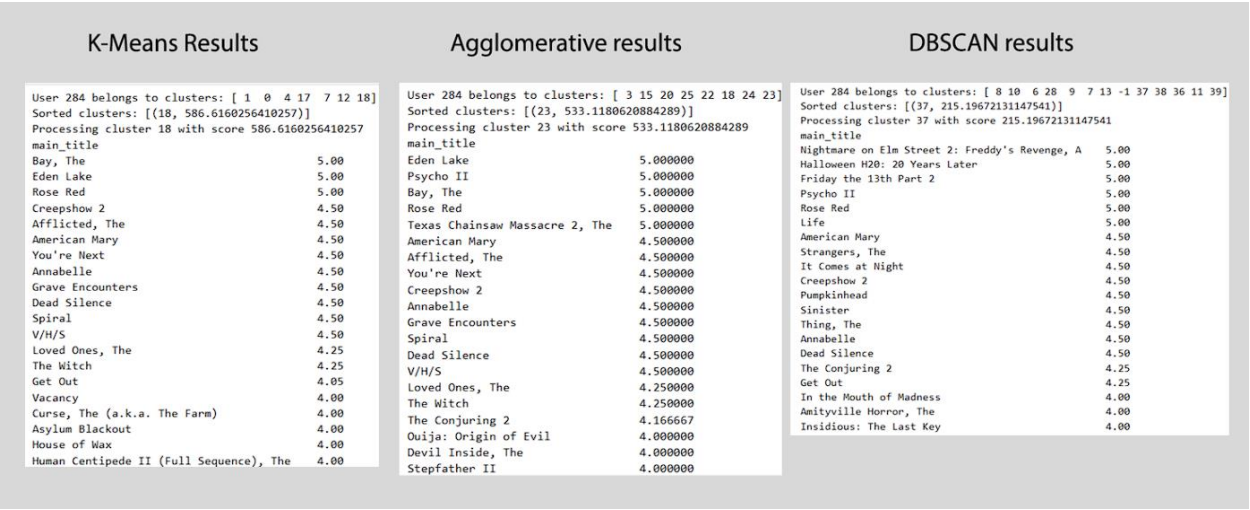

Рисунок 3.15 – Результати рекомендаційної функції «recommend_movies (user_id, main_genre)» за різними методами

За результатами методу DBSCAN рекомендаційна функція була реалізована у вигляді сторінок веб-додатку. Для того щоб отримати персональну рекомендацію за фільмами, перш за все, наприклад, треба бути на сторінці неоціненого фільму. Сторінка для оцінювання фільму приведена на рис. 3.16, сторінка має інформацію про фільм та сеанси на фільм. Назва фільму, рік випуску, опис фільму, жанр фільму та картинка фільму є інформацією про фільм. Під інформацією про фільм є інформація про відкриті за закриті сеанси на фільм. Під закритими сеансами розуміється те, що усі квитки на сеанс вже заброньовані та вже немає можливості купувати квитки на цей сеанс. Закриті сеанси відображені сірим, а відкриті відображені фіолетовим.

Під картинкою фільму є 5 зірочок, якщо фільм не був попередньо оцінений користувачем, то зірочкі сірі та відображають середню оцінку фільму за іншими користувачами, які вже дивились та оцінювали цей фільм. Коли зірочки сірі, система відображає повідомлення під ними та просить користувача оцінити фільм, щоб у системи була можливість зробити рекомендацію.

Для того, щоб отримати рекомендацію треба обрати кількість зірочок під картинкою фільму і після того, як користувач натискає на зірочки, вони становляться жовтими та показують повідомлення, що цей фільм був оцінений з певною оцінкою.

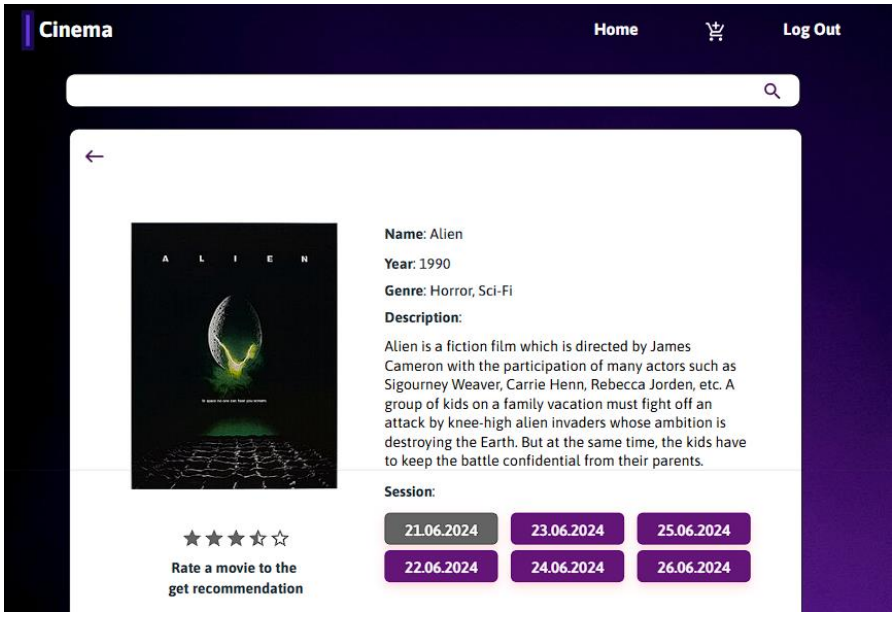

Рисунок 3.16 – Сторінка для оцінювання фільму

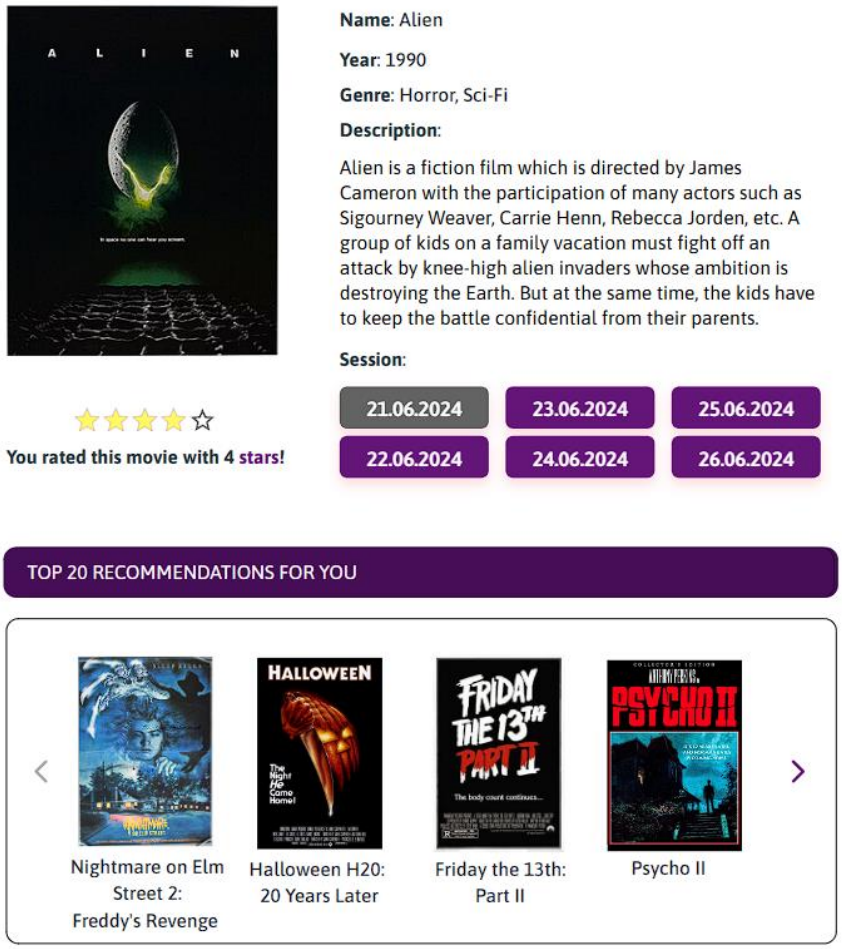

Рисунок 3.17 – Рекомендація ТОП 20 фільмів

На рис. 3.17 приведений скриншот сторінки результату того, що користувач оцінив фільм і система видає йому персоналізовану рекомендацію фільмів, де першими фільмами є найбільш оцінені фільми у кластері, до якого належить користувач, у обраному жанрі (у даному прикладі як на скриншоті, жанр є хоррор). Система відображає перші 4 фільми, а далі пропонує користувачу стрілочками побачити більшу кількість рекомендацій. Рекомендаційна функція реалізована так, що система рекомендує до 20 фільмів.

ВИСНОВКИ

За проведеним дослідженням реалізації рекомендаційної функції за допомогою методів кластеризації у системах перегляду фільмів виконані наступні задачі:

– проведений аналіз сучасного стану рекомендаційних функцій в системах перегляду фільмів та обрана предметна область CRM-мережі кінотеатрів;

– визначено теоретичні підходи (методи) для визначення рекомендацій в системах мережі кінотеатрів та проведений аналіз теоретичних відомостей про методи кластеризації: k-середніх, ієрархічні методи кластеризації «згори-вниз» та «знизу-вгору» та DBSCAN;

– визначено послідовність дій для дослідження методів кластеризації даних для реалізації рекомендаційної функції мережевої системи кінотеатрів;

– визначено вхідні дані для аналізу: дані користувачів (вік, стать, вікова група), оцінка фільмів та інформація про фільми (назва фільму, жанр фільму);

– проведено дослідження методу k-середніх, агломеративного та DBSCAN для реалізації рекомендаційної функції у CRM-системі мережі кінотеатрів;

– проведено порівняння результатів дослідження методів кластеризації для реалізації рекомендаційної функції у CRM-системі мережі кінотеатрів;

– проведена реалізація рекомендаційної функції мережевої системи кінотеатрів за допомогою спільної фільтрації (методом DBSCAN) і фільтрації на основі вмісту з елементами для веб-додатку.

За розглянутими методами кластеризації було прийнято не реалізовувати методи кластеризації k-середніх та агломеративний.

Реалізовано комбінацію спільної фільтрації (враховуючи членство користувача в кластері за результатами кластеризації методом DBSCAN) і фільтрації на основі вмісту (враховуючи жанр фільм) для рекомендаційної функції фільмів користувачеві у CRM-системі мережі кінотеатрів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Одинцова В.О. Дослідження методів кластеризації даних для реалізації рекомендаційної функції CRM-системи мережі кінотеатрів. 28-й Міжнародний молодіжний форум «Радіоелектроніка та молодь у XXI столітті». Том 6: секція 4: Харків – 2024 – С.587.

2. Sameer Chhabra. Netflix says 80 percent of watched content is based on algorithmic recommendations [Електронний ресурс]. – Режим доступу: https://mobilesyrup.com/2017/08/22/80-percent-netflix-shows-discovered-recommendation/.

3. Decoding Amazon's Recommendation System. URL: https://www.argoid.ai/blog/decoding-amazons-recommendation-system.

4. D. Jannach, M. Zanker, A. Felfernig, G. Friedrich. Recommender Systems. An Introduction. Cambridge University Press 32 Avenue of the Americas, New York, NY 10013-2473. 2011. URL: https://assets.cambridge.org/97805214/93369/frontmatter/9780521493369_frontmatter.pdf.

5. Pasquale Lops, Marco de Gemmis, Giovanni Semeraro. Content-based Recommender Systems: State of the Art and Trends. Recommender Systems Handbook. Springer New York, 2020. P. 73–105. ISBN eBook: 978-0-387-85820-3. DOI: https://doi.org/10.1007/978-0-387-85820-3.

6. A Step by Step approach to Solve DBSCAN Algorithms by tuning its hyper parameters [Електронний ресурс] – Режим доступу до ресурсу https://medium.com/@mohantysandip/a-step-by-step-approach-to-solve-dbscan-algorithms-by-tuning-its-hyper-parameters-93e693a91289.

7. CRM система, її компоненти, класифікація [Електронний ресурс] – Режим доступу до ресурсу http://websekretar.chizh.ua/?p=548.

8. Francesco Ricci, Lior Rokach, Bracha Shapira Recommender Systems Handbook Second Edition Springer Science+Business Media New York 2011, 2015. 77

9. X. Su, T.M. Khoshgoftaar; A Survey of Collaborative Filtering Techniques; Advances in Artificial Intelligence, 2009.

10. P. Melville, V. Sindhwani; Recommender systems; Encyclopedia of Machine Learning, 2010.

11. Greg Linden, Brent Smith, Jeremy York. Amazon.com Recommendations: Item-to-Item Collaborative Filtering. IEEE Computer Society, 2003.

12. Jannach, M. Zanker, A. Felfernig, G. Friedrich; Recommender Systems. An Introduction; Cambridge University Press 32 Avenue of the Americas, New York, NY 10013-2473, USA, 2011; 352 p.

13. Robin Burke. Knowledge-based recommender systems. Encyclopedia of Library and Information Science. Department of Information and Computer Science University, URL: https://www.cs.odu.edu/~mukka/cs795sum09dm/Lecturenotes/Day6/ burke-elis00.pdf .

14. Noor Ifada, Triyani Fatchur Rahman, Mochammad Kautsar Sophan. Comparing Collaborative Filtering and Hybrid based Approaches for Movie Recommendation. 2020 6th Information Technology International Seminar URL: https://www.researchgate.net/publication/348673145_Comparing_Collaborative_Filtering_and_Hybrid_based_Approaches_for_Movie_Recommendation.

15. Bei-Bei CUI. Design and Implementation of Movie Recommendation System Based on Knn Collaborative Filtering Algorithm. ITM Web of Conference, DOI: 10.1051/ 71204008, 2017.

16. Noor Ifada, Syafrurrizal Naridho, Mochammad Kautsar Sophan. Multi-criteria based Item Recommendation Methods. Journal of Science and Technology URL: https://doi.org/10.21107/rekayasa.v12i2.5913

17. Phongsavanh Phorasim, Lasheng Yu. Movies recommendation system using collaborative filtering and k-means. International Journal of Advanced Computer Research,

Vol 7(29), 2017. ISSN (Print): 2249-7277 ISSN (Online): 2277-7970 URL: http://dx.doi.org/10.19101/ IJACR.2017.729004 . 78

18. Rahul Pradhan, Ashish Chandra Swami, Akash Saxena, Vikram Rajpoot. A Study on Movie Recommendations using Collaborative Filtering. Rahul Pradhan et al 2021 IOP Conf. Ser.: Mater. Sci. Eng. 1119 012018 URL: https://iopscience.iop.org/article/10.1088/1757-899X/1119/1/012018 .

19. Noor Ifadaa, Irvan Syachrudina, Mochammad Kautsar Sophana, Sri Wahyunib. Enhancing the Performance of Library Book Recommendation System by Employing the Probabilistic-Keyword Model on a Collaborative Filtering Approach. 4th International Conference on Computer Science and Computational Intelligence 2019 (ICCSCI)

20. Chong Chen, Liya Zhang, Huan Qiao, Shihong Wang, Yuchu Liu, Xiaohong Qiu. Book Recommendation Based on Book-Loan Logs. International Conference on Asian Digital Libraries, 2012. URL: https://www.researchgate.net/ publication/293099285_Book_Recommendation_Based_on_Book-Loan_Logs

21. Токмаков,Г. П. Бази даних та знань. Проектування баз даних за технологією «клієнт-сервер» та розробка клієнтських додатків: Навчальний посібник/Г.П. Токмаков.- Ульяновськ; УлГТУ, 2005. – 143 с.

22. K-means Clustering – Introduction [Електронне джерело] – Режим доступу до ресурсу: https://www.geeksforgeeks.org/k-means-clustering-introduction/?ref=lbp