Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет _____Комп'ютерні науки_____

(повна назва)

Кафедра _____Системотехніки_____

(повна назва)

АТЕСТАЦІЙНА РОБОТА

Пояснювальна записка

_____другий (магістерський)_____

(рівень вищої освіти)

_____ГЮИК 503100-004_____

(позначення документа)

Дослідження рекомендаційних систем для пошуку місць відпочинку в Інтернеті

(тема)

Виконав: студент 2 курсу, групи СПРм-18-1

_____Корнійчук В.О._____

(прізвище, ініціали)

Спеціальності 122 – Комп'ютерні науки____

(код і повна назва спеціальності)

Тип програми освітньо-професійна Освітня програми Системне проектування_____

(повна назва освітньої програми)

Керівник __проф.Міщеряков Ю.В._____

(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ __проф.Гребеннік І.В.

(підпис) (прізвище, ініціали)

2019 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерні науки

Кафедра Системотехніки

Рівень вищої освіти другий магістерський)

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

Тип програми Освітньо-професійна

Освітня програма Системне проектування

ЗАТВЕРДЖУЮ:
Зав. кафедри
_____
(підпис)
«____»_____ 20 ___ р.

ЗАВДАННЯ
НА АТЕСТАЦІЙНУ РОБОТУ

студентові Корнійчук Вадим Олегович
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження рекомендаційних систем для пошуку місць відпочінку в Інтернеті
затверджена наказом по університету від 04 11 2019 р.№
2. Термін подання студентом роботи до екзаменаційної комісії
19 12 2019 р.
3. Вихідні дані до роботи : Функція: Рекомендаційна система, середовище розробки IntelliJ IDEA, язик розробки: JAVA, файловий. Операційна система: Windows 7 та вище, процесор: Pentium4 та вище
4. Зміст пояснювальної записки: Вступ, Аналіз предметної області, Розробка метода підвищення якості рекомендаційної системи, Розробка рекомендаційної системи..
5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів): Алгоритми рекомендаційних систем, форми інтерфейсу, гістограма
6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім’я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Спеціальна частина	Проф.. Міщеряков ,Ю.В.		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання атестаційної роботи00	01.09	Виконав
2	Аналіз завдання, літератури та аналогів з теми атестаційної роботи	15.10 – 23.10	Виконав
3	Вибір засобів для розробки технічних вимог до програми	24.10 – 28.10	Виконав
4	Розробка метода для реалізації ідеї	28.10 – 5.11	Виконав
5	Розробка програми	5.11 – 22.11	Виконав
6	Тестування програми	22.11 – 26.11	Виконав
7	Аналіз роботи програми	26.11- 30.11	Виконав
8	Розробка «Посібника користувача»	30.11 - 5.12	Виконав
9	Оформлення пояснювальної записки та програмної документації	5.12 – 8.12	Виконав
10	Оформлення графічної частини та презентаційних матеріалів комп’ютерного	8.12 – 10.12	Виконав
11	Представлення на рецензування	12.12 – 13.12	Виконав
	Представлення атестаційної роботи в ДЕК	19.12	

Дата видачі завдання 01 09 2019 р.

Студент _____
(підпис)

Керівник роботи _____ __проф. Міщеряков Ю.В
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка до магістерської атестаційної роботи: Арк 75., 18 рис.,23 джерел,табл. 2, 3 додатки

JAVA, WEB-СЕРВІС, РЕКОМЕНДАЦІЙНА СИСТЕМА, КОЛАБОРАТИВНА ФІЛЬТРАЦІЯ, КЛАСТЕРНИЙ АНАЛІЗ

Об’єктом досліджень – рекомендаційні системи.

Предмет досліджень – процес вибору найбільш релевантних користувачу рекомендацій.

Мета досліджень – підвищення релевантності рекомендацій за рахунок врахування соціального образу користувача.

В даній роботі для того щоб підвищити якість рекомендацій було зроблено припущення, що об’єднання декількох рекомендаційних методів з методами кластеризації дасть кращі результати. Спочатку ми реалізуємо метод колаборативної фільтрації, а потім додамо методи кластеризації. Це покращить вихідну рекомендацію для певного користувача. Для рішення проблеми глобального мінімуму, було вирішено використати алгоритм градієнтного спуску з оптимізацією.

Подальшим розвитком даного проекту є інтеграція в соціальні мережі. Вони містять у собі історію користувача протягом тривалого періоду часу, це дозволяє отримати більш об'єктивні дані про користувача і сформувати більш правильніший соціальний образ.

ABSTRACT

Explanatory note to the master's appraisal work: Ark 75., 18 fig., 23 sources, table. 2, 3 applications

JAVA, WEB-SERVICE, RECOMMENDATION SYSTEM, COLLABORATIVE FILTRATION, CLUSTER ANALYSIS

The object of research is the recommendation systems.

The subject of research is the process of selecting the most relevant recommendations for the user.

The purpose of the research is to increase the relevance of the recommendations by taking into account the user's social image.

In order to improve the quality of the recommendations, it has been suggested that combining several recommendation methods with clustering methods will yield better results. First, we implement a collaborative filtering method and then add clustering methods This will improve the original recommendation for a specific user. To solve the problem of the global minimum, it was decided to use an optimized gradient descent algorithm.

A further development of this project is integration into social networks. They contain a history of the user over a long period of time, which allows to obtain more objective data about the user and to form a more correct social image

ПОЗНАЧЕННЯ

Рекомендаційна система - програма, яка намагається передбачити об'єкти, які будуть цікаві користувачеві.

«Холодний старт» - проблема, пов'язана з відсутністю даних про користувачів або об’єктах яка нещодавно з'явилися в системі.

«Сліпі» рекомендації - рекомендації, які носять характер випадкових.

Повнота - метрика характеризує здатність системи давати якомога більше релевантних рекомендацій.

API (application programming interface) - набір готових класів, процедур, функцій, що надаються додатком (бібліотекою, сервісом) для використання в зовнішніх програмних продуктах.

ЗМІСТ

ВСТУП

Відмінною особливістю кінця XX та початку XXI століття є різкий ріст кількості інформації. Так, наприклад, якщо обсяг інформації в інтернеті в 2006 році налічував 161 ексабайт, то до 2020 року він виросте до 44зетабайта, причому дана цифра продовжує зростати приблизно в два рази кожні півтора року. Людині стає все складніше орієнтуватися в такому обсягу інформації, перед ним виникає багато альтернатив. Типові ситуації: вибір відповідного товару в інтернет-магазині, пошук цікавого в стрічці новин, або підбор кінофільмів для вечірнього перегляду, або пошук місця для відпочинку ввечері. Найчастіше, якість знайденої інформації пропорційно витраченому часу, тому не дивно, що в останні десять років особливої популярності набули різні методи машинного навчання, зокрема, рекомендаційні системи. [2]

Дані системи, аналізуючи поведінку користувача і наявну інформацію, намагаються передбачити найбільш релевантні і цікаві об’єкти і, найчастіше, справляються з цією роботою краще самого користувача і значно швидше. Дана область все ще є активно прогресуючою, провідні компанії Силіконової долини постійно покращують свої алгоритми, з метою досягти більш якісних рекомендацій. Особливу роль в прогресі грає електронна комерція, яка активно використовує різні види рекомендаційних систем і, як наслідок фінансує цю галузь науки.

Об’єктом досліджень – рекомендаційні системи.

Предмет досліджень – процес вибору найбільш релевантних користувачу рекомендацій.

Метою даної роботи є розробка рекомендаційної системи, що використовує соціальний образ користувача для поліпшення рекомендацій. Як приклад, система буду рекомендувати місце для відпочинку. Підсумковий результат буде представлений в вигляді web-сервісу.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Історія рекомендаційних систем

Перші рекомендаційні системи з'явилися наприкінці 1990-х. Одним з перших пристроїв був цифровий відеомагнітофон TiVo. Якщо два користувача регулярно переглядали однакові телепередачі, то при появі нової постійно переглядається передачі у одного з цих користувача, пристрій рекомендував її до перегляду для другого користувача. Дані збиралися кожен день шляхом зчитування кліків і відправкою їх на сервер, на якому застосовувався рекомендаційний алгоритм. Вся ця система, звичайно, працювала не ідеально, часто пропонуючи нецікаві передачі, проте простота підходу забезпечила популярність даної ідеї в галузі.[18]

Наступний поштовх в розвитку рекомендаційних систем забезпечила компанія NetflixPrize, оголошений в 2006 році. Перед розробниками була поставлена мета – поліпшити існуючий алгоритм рекомендацій на 10%. Були надані все необхідні набори даних, а метрикою для вимірювання якості пророкувань була обрана середньоквадратична помилка (RMSE). Завдання виявилося непростим і лише через 3 роки команда BellKor'sPragmaticChaos зуміла досягти необхідного результату і забрала приз.

В останні роки основною областю застосування рекомендаційних система стала електронна комерція. Найяскравішим представником даній області є Amazon.com –найбільша за оборотом товарів і послуг компанія в світі. Вона використовує кілька видів рекомендаційних систем і одне з застосувань – додаткові пропозиції у вигляді схожих або супутніх товарів. Дана система нав'язлива і приносить користь як користувачам, так і магазину.[3]

1.2 Типи рекомендаційних систем

Існують основні чотири типи рекомендаційних систем:

а) основані на контенті (content-based);

б) колаборативні;

в) основані на знаннях (knowledge-based);

г) гібридні.

Нижче на рисунок 1.1 представлені загальні кроки для формування рекомендацій:

– попередня обробка включає пошук схожості даних, визначення вибірки, розмірності даних;

– аналіз даних – включає в себе класифікацію, кластеризацію даних;

– представлення результатів.

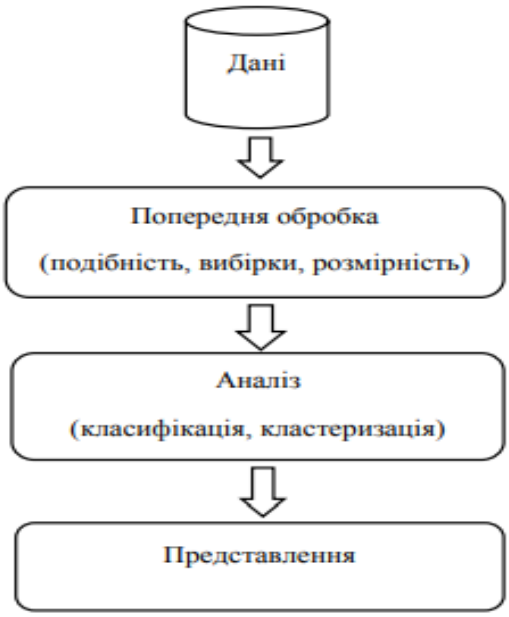

Рисунок 1.1 – Кроки для формування рекомендацій:

Впровадження рекомендаційних систем розширює можливості взаємодії з клієнтами в різних предметних областях. Застосування рекомендаційних систем призводить до збільшення продажу товарів,

розширення продажу більш різноманітних об’єктів, збільшення лояльності користувачів, призводить до кращого розуміння користувацьких потреб та побажань. Розглянемо основні типи рекомендаційних систем.

1.3 Методи контентної фільтрації

Рекомендаційні системи контентної фільтрації будують внутрішні зв’язки між об’єктами, що підлягають рекомендації. Контентна фільтрація (content based) формує рекомендацію на основі об’єктів, оцінених користувачем в минулому. Наприклад, якщо в книжковому магазині користувач придбав книжку для навчання гри на гітарі, то йому автоматично будуть запропоновані інші популярні книжки для самонавчання або література того ж автора.

Нижче на рисунок 1.2 представлена схема основних кроків формування рекомендацій на основі даних користувача системи [5]. Основні кроки в рекомендаційній системі контентної фільтрації: спочатку потрібно проаналізувати контент предметів (об’єктів) і скласти набір його критеріїв (жанри, теги, слова)

Дізнатися, які критерії подобаються користувачу, порівняти ці дані і отримати рекомендації. Критерії об’єднують користувачів і об’єктів в одній системі координат – якщо точка користувача і предмету знаходяться поруч, то об’єкт ймовірно сподобається користувачу.

Даний підхід може використовувати ретроспективну інформацію про перегляд. Наприклад, якщо деякий клієнт закладу харчування замовляє страви виключно без вмісту м’яса, то контентна фільтрація може використовувати цю ретроспективну інформацію для виявлення подібних страв та пропонувати ці страви в якості рекомендованих для даного клієнта.

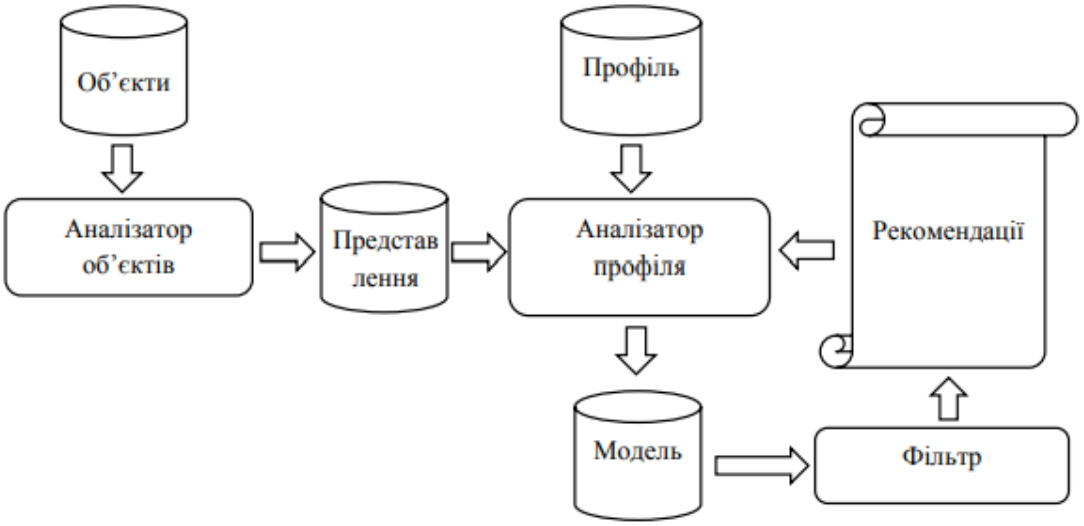

Рисунок 1.2 - Схема формування рекомендації на основі контентної фільтрації

Перевагами даного підходу є [5]:

а) незалежність від інших користувачів, рекомендації формуються на основі конкретного користувача, та не враховується інформація інших клієнтів системи;

б) прозорість;

в) простота добавлення нових об'єктів;

г) для отримання рекомендацій потрібні тільки дані про користувача і об'єкти.

Недоліками підходу контентної фільтрації є:

а) обмежені можливості аналізатора контенту;

б) рекомендації нових областей;

в) складності при нових користувачах.

Часто системи даного підходу використовують до колаборативної фільтрації, коли оцінок користувачів недостатньо для формування рекомендацій на основі колаборативного алгоритму.

1.4 Методи колаборативної фільтрації

Термін «колаборативна фільтрація» вперше використав Девід Голберг (DevidGoldberg) з компанії Xerox PARC в 1992 році в статті «Using collaborative filtering toweavean information ta pestry»

. Він спроектував систему Tapestry, яка дозволяла людям анотувати документ як цікавий чи нецікавий, і використовувала цю інформацію для фільтрації документів, які пропонувались іншим людям. У наш час цей алгоритм колаборативної фільтрації використовується на багатьох сайтах для рекомендації фільмів, інших сайтів, підкастів, статей та навіть анекдотів.

Системи колаборативної фільтрації формують рекомендації на основі інформації про поведінку клієнта в минулому, наприклад про покупки чи оцінки. Даний підхід може бути побудований виключно на основі поведінки даного користувача чи з врахуванням поведінки інших користувачів зі схожими характеристиками, що є більш ефективним.

У випадку, коли колаборативна фільтрація враховує поведінку інших користувачів, вона використовує знання про групу (groupknowledge) для формування рекомендацій на основі подібності користувачів. На рисунку 1.3 представлена схема виконання колаборативної фільтрації [5].

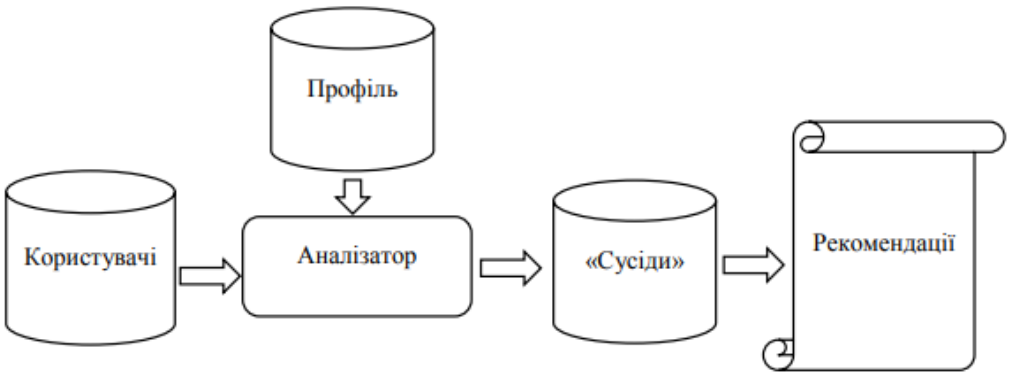

Рисунок 1.3 - Схема виконання колаборативної фільтрації

Спочатку проводиться аналіз інформації конкретного користувача і інших клієнтів системи, з метою виявлення подібності у характеристиках. Результатом роботи аналізатора є так звані «сусіди» - група користувачів з схожими уподобаннями. На основі об’єктів, які сподобались «сусідам», формуються рекомендації для даного клієнта.

Колаборативна фільтрація – найпопулярніший сьогодні вид рекомендаційних систем. Основними перевагами даного підходу для формування рекомендацій є [6]:

- простота алгоритмів;
- простота пояснення;
- стабільність.

Основним недоліком колаборативної фільтрації є проблема «холодного старту» – відсутність даних про нещодавно з’явлених в системі користувачах, що ускладнює формування рекомендацій.

Колаборативна фільтрація це набір алгоритмів, які використовують інших користувачів, їх ставлення до різного роду предметів і історію конкретного користувача з метою порекомендувати предмет, оцінка якого ще не проведена цим користувачем рисунок 1.4. Основний принцип тут полягає в тому, що користувачі, які раніше однаково оцінювали певні предмети, будуть схожим чином оцінювати майбутні предмети.

Є три типи колаборативної фільтрації:заснований на сусідстві, заснований на моделі і гібридний. Різниця між даними підходами можна показати на наступному прикладі. Нехай вхіднідані представлені у вигляді матриці, що містить оцінки різних товарів, проставлені різними користувачами. Тоді два рядки представлятимуть оцінки двох користувачів по всім товарам, а два стовпчика оцінки для двох товарів, проставлені усіма користувачами. Підхід, заснований на сусідстві,видає рекомендації виходячи з найбільш схожих рядків(User-based підхід) або стовпців (item-based підхід). Всвою чергу, підхід, заснований на моделі, намагається доповнити цю матрицю і передбачити оцінки, які могли б поставити користувачі. Для цього

використовуються алгоритми машинного навчання. Гібридний метод об'єднує в собі вище описані підходи. Він видає більш якісний результат, але даний тип колаборативної фільтрації дорогий і складний в реалізації і застосуванні.[7]

Рисунок 1.4 - Принцип колаборативної фільтрації

1.5 Рекомендаційні системи основані на знаннях (knowledge-based)

Рекомендаційні системи основані на знаннях (knowledge-based) – це системи, які для формування рекомендації використовують отримані знання з предметної області. Дана система формування рекомендацій не враховує оцінки інших користувачів, а лише профіль конкретного користувача і об’єкти. Даний підхід схожий на рекомендаційні системи контентної фільтрації, проте алгоритми основані на знаннях використовують більш глибокий аналіз об’єктів, будуючи зв’язки між об’єктами не по критеріям схожості, а враховуючи взаємозв’язок груп об’єктів.

Рекомендаційні системи, основані на знаннях, враховують ряд додаткових опцій, які відносяться до індивідуальних ознак конкретного

користувача системи. В зв’язку з цим системи можуть враховувати користувацькі побажання, демографічні особливості та ін.. Існують різні підходи для формування рекомендацій на основі знань .

Case-based підхід враховує наявність додаткової сутності: вимоги користувача (user requirements). Задача даного підходу заключається в тому, щоб знайти рекомендовані об’єкти, згідно вимогам .

Demographic-based безпосередньо враховує якості користувача, наприклад, на скільки він забезпечений, де проживає, скільки заробляє і т. д.

Utility-based розраховує відносну користь кожного об’єкту для користувача. Наприклад, якщо користувач придбав фотоапарат, то скоріш за все, йому потрібні будуть об’єктиви для нього [5].

В даних рекомендаційних системах можна створювати багато власних областей знань і ранжувати по ним, досягаючи максимальних результатів. Рекомендації, основані на знаннях, підвищують продажі великих мережевих торгових груп на десятки відсотків. На відміну від методів контентної фільтрації, цей тип рекомендацій наділений високою точністю, пропонуючи користувачу те, що йому дійсно може сподобатись. Ґрунтуючись на додатковій інформації про об’єкти, дані системи не мають «холодного» старту. Проте одним з суттєвих недоліків є необхідність збирання та аналізу великих об’ємів даних.

1.6 Гібридні методи формування рекомендацій

Гібридні підходи поєднують різні рекомендаційні механізми, що підвищує точність надання рекомендацій та збільшує складність рекомендаційних систем.

Найпоширенішим поєднанням є об’єднання результатів колаборативної та контентної фільтрації. Гібридний підхід може бути корисним, якщо застосування колаборативної фільтрації починається зі значної розрідженості даних («холодний» старт). Гібридний підхід дозволяє спочатку зважувати

результати згідно контентної фільтрації, а потім зміщувати ці ваги по направленню до колаборативної фільтрації .

Відомим прикладом гібридної рекомендаційної системи є Netflix (BellKor), яка поєднує в собі 27 алгоритмів рекомендації. Використання гібридних методів формування рекомендацій підвищують складність та також вартість системи. Успішне створення та впровадження гібридного алгоритму вимагає команду досвідчених розробників, які мають ряд професійних навиків, що виділяє розробників рекомендаційних систем в окрему професію.[8]

1.7 Порівняння підходів до побудови рекомендаційних систем

Розглянемо переваги і недоліки колаборативной фільтрації щодо контентної:

а) Немає необхідності виділяти характеристики рекомендованих предметів.

б) Проблема «холодного старту» властива всім рекомендаційним системам. Однак при колаборатівной фільтрації відбувається значно швидше.

в) Зміна інтересів користувача сприймаються значно швидше.

Недоліки:

а) Так звана проблема «білих ворон». Деякі користувачі мають специфічний смак, який не узгоджується з уподобаннями інших користувачів. Рекомендації для таких користувачів мають низьку якість.

б) Інші користувачі можуть давати упереджені і необ’єктивні оцінки деяких предметів, що позначиться на якості рекомендації для інших користувачів.

в) Конфіденційність. Деякі користувачі можуть негативно ставитися до того факту, що їх інтереси, хоч і в неявній формі доступні.

г) Більш складні обчислення. [10]

1.8 Огляд існуючих рекомендаційних систем

1.8.1 Рекомендації Google

Матеріали в рекомендаціях або додаються з джерел, на які користувач підписався самостійно, або підбираються автоматично в залежності від того, які дії він виконував в сервісах Google. При цьому новизна контенту не має значення: якщо є підстави вважати, що користувач чимось зацікавиться, ми це йому запропонуємо.[17]

Крім іншого, картки рекомендацій можуть містити відеоролики, результати спортивних змагань, новини індустрії розваг (наприклад, інформацію про прем'єри фільмів), біржові котирування, відомості про різні заходи (зокрема, списки номінантів будь-якої премії або учасників майбутнього музичного фестивалю) і т . д. Рекомендації Google - невичерпне джерело інформації за інтересами.

1.8.2 Movielens

Movielens є однією з перших рекомендаційних систем, що рекомендують фільми на основі коллаборативной фільтрації. Вона була створена в 1997 році в ході дослідження GroupLensResearch проведеного університетом Меннесота. Для кожного користувача система намагається передбачити оцінки, які він би дав фільму і пропонує фільми з найвищими оцінками. Користувачам постійно пропонується оцінити якомога більше вже переглянутих фільмів, щоб нові рекомендації були більш точними. Проблема «холодного старту» частково вирішується методом індивідуальних переваг. При реєстрації користувачам пропонується оцінити своє ставлення до різного роду жанрів, наприклад висловити своє ставлення до комедій в порівняні з жахами. Дана інформація допомагає підвищити якість «сліпих» рекомендацій.

1.8.3 Amazon

Amazon – одна з найбільших майданчиків інтернет-торгівлі - використовує рекомендації на основі контенту. Коли відвідувач вибирає для покупки будь-якої товар, Amazon на основі цього вихідного товару рекомендує відвідувачеві інші товари, придбані іншими користувачами (за допомогою матриці покупки наступного товару на основі його схожості з попередньою купівлею). Компанія Amazon запатентувала цей підхід під назвою item-to-itemcollaborativefiltering (коллаборативна фільтрація від елемента до елементу).[20]

1.9 Постановка задачі

Основним завданням атестаційної роботи є розробка комбінованого підходу до побудови рекомендаційних систем, що забезпечує найбільш повне використання всіх даних про користувачів та відвідувачів web-сайтів з метою вироблення рекомендацій які найбільш адекватно відображають їх очікування. Практична новизна роботи полягає в ідеї комбінованого використання декількох вже існуючих алгоритмів рекомендацій.

Проблема рекомендаційних систем полягає в тому, що матриця, що відображає оцінки користувачів по відношенню до предметів, найчастіше сильно розряджена. Це сильно ускладнює проблеми «холодного старту», «білих ворон», і така система видає рекомендації не дуже високої якості. Спроба виправити цю ситуацію методами контекстній фільтрації не завжди ефективна. Його користувачі не завжди можуть правильно висловити те, чого вони хочуть, у багатьох випадках це неможливо.

Бажання користувача – знайти гарне місце для відпочинку в вечері, а таке місце характеризується не категоріями, інтер’єром або іншими явними характеристиками. Виділити характеристики за допомогою яких можна відрізнити погане місце відпочинку від хорошого (і така ситуація не тільки с

закладами відпочинку) не представляється можливим, у багатьох випадках вони індивідуальні. Підтвердженням цього може служити рейтинг найпопулярніших закладів, наприклад, з сайту 78.com.ua. У першій десятці представлені різні категорії : ресторан, пивоварні, караоке, кафе, нічні клуби. Деякі з них навіть несумісні.

Дана ситуація наштовхує на пошук інших способів допомогти рекомендаційним системам робити якісніші рекомендації. Одним з рішень є облік характеристик користувачів. Сукупність цих характеристик можна виділити в соціальний образ. Ними можуть бути: вік, стать, освіта, професія, захоплення, творчі інтереси і інші. Знайти пряму або зворотну кореляцію між даними характеристиками користувача і його ставленням до рекомендованих предметів дуже складно, проте є інше рішення.

Ідея полягає в тому, щоб розділити користувачів на різні групи з урахуванням їх соціального образу, і при рекомендації будь-якого предмета надавати перевагу тим предметам, які найбільш популярні в соціальній групі даного користувача.

2 РОЗРОБКА МЕТОДА ПІДВИЩЕННЯ ЯКОСТІ РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ

Існує два шляхи вирішення даного завдання, і обидва є проблемами машинного навчання. Перший шлях це класифікація. Заздалегідь виділяються соціальні групи, і кожен користувач записується в ту, до якої він найближче. Другий шлях це кластеризація. Спеціальний алгоритм виділяє логічну структуру серед безлічі соціальних образів користувачів і ділить їх на групи.

Другий варіант видається більш переважним, особливо на ранньому етапі розробки проекту, тому що неможливо передбачити які користувачі і в якому співвідношенні будуть користуватися даною рекомендаційної системою.

В даній роботі для того щоб підвищити якість рекомендацій було зроблено припущення, що об’єднання декількох рекомендаційних методів з методами кластеризації дасть кращі результати. Спочатку ми реалізуємо метод колаборативної фільтрації, а потім додамо методи кластеризації. В теорії це повинно покращити вихідну рекомендацію для певного користувача.

Метод кластеризації будується на тому як користувач відповість на питання задані йому при реєстрації. Проблемою цього алгоритму може бути відсутність гарантії досягнення глобального мінімуму. Для рішення цієї проблеми ми додамо метод градієнтного спуску, який повинен знайти глобальний мінімум.

2.1 Набір даних для побудови рекомендаційної системи

Збір оцінок нових користувачів проводиться явним способом за допомогою web-сайту, на якому надана можливість шукати місце для відпочинку за назвою або з використанням фільтрів. База даних всіх закладів

відпочинку була взята с сайту 78.com.ua. Для тестування та в якості початкового набору оцінок. Дані представлені вфайлах формату .csv, для обробки яких в рамках проекту була написана невелика бібліотека. Два основні файли: restaurants.csv та links.csv. Перший має структуру userId, restaurantsId, rating, timestamp.UserId – це анонімний ідентифікатор, а restaurantsId – це ідентифікатор реального ресторану або кафе, посилання на який можна знайти в файлі links.csv. Посилання представлено в вигляді набора ідентифікаторів цього закладу.

2.2 Огляд алгоритму колаборативної фільтрації

Виходячи з представленого набору даних, найбільш якісні результати дасть підхід колаборативної фільтрації, заснований на моделі. Уявимо вхідний набір даних у вигляді матриці Y. Рядки цієї матриці відповідають місцям відпочинку, а стовпці користувачам. Оцінки варіюються від 1 до 5, якщо користувач не оцінював фільм, то відповідно буде стояти 0.[12]

Приклад матриці Y:

користувачі

місця відпочинку

$$\begin{vmatrix} 5\ 0\ 3 \ldots 0\ 3 \\ 0\ 4\ 0 \ldots 2\ 4 \\ 0\ 0\ 0 \ldots 0\ 0 \\ .. .. .. \ldots .. .. \\ 5\ 0\ 3 \ldots 0\ 0 \end{vmatrix}$$

Нехай деякий набір векторів$x^{(1)}$,$x^{(2)}$, …, $x^{(\text{ кількість закладів})}$ $x^{i} \in R^{k}$ описує k характеристик кожного закладу. Другий набір векторів $\theta^{(1)}$,$\theta^{(2)}$, …,$\theta^{(\text{кількість користувачів})}$ $\theta^{j} \in R^{k}$ описує ставлення користувача до кожної характеристики. Таким чином, вектори $\theta^{(j)}$ та $x^{(i)}$ мають однакову розмірність, а параметр $\theta_n^{(j)}$ відображає відношення користувача j до характеристики $x_n^{(i)}$ закладу i. В такому випадку оцінка даного закладу цим користувачем розраховується як

$(\theta^{(j)})^T x^i$. Надалі кількість користувачів позначається як n_u, а кількість закладів n_m.

Представимо, що набір векторів $\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(n_u)}$ відомий. В такому випадку можливо знайти значення вектора $x^{(j)}$, мінімізувавши наступний функціонал.

$$J = \frac{1}{2}\sum_{j:r(i,j)} ((\theta^{(j)})^T x^i - y^{(i,j)})^2 + \frac{\lambda}{2}\sum_{l=1}^{n}(x_l^i)^2 \quad (2.1)$$

Де r(i,j) =1,якщо j користувач оцінив і заклад або 0, якщо не оцінив.

$\lambda /2 \sum_{l=1}^{n}(x_l^i)^2$ – регуляризація.

Дана функція рахує квадратичну помилку того, на скільки різняться оцінки, отримані за допомогою параметрів θ та x, порівняно зі справжніми. Таким чином, мінімізувавши функціонал J, ми отримаємо деякі характеристики x^i для і заклада.

Але на потрібні характеристики для всіх закладів, в такому випадку функціонал буде виглядати наступним чином:

$$J(x^{(1)}, \ldots, x^{(nm)}) =$$
$$= \frac{1}{2}\sum_{i=1}^{n_m}\sum_{j:r(i,j)=1}((\theta^{(j)})^T x^i - y^{(i,j)})^2 + \frac{\lambda}{2}\sum_{i=1}^{n_m}\sum_{l=1}^{n}(x_l^i)^2 \quad (2.2).$$

Розглянемо протилежну ситуацію. Нехай нам відомі характеристики закладів $x^{(1)}, x^{(2)}, \ldots, x^{(n_m)}$. Тоді значення векторів $\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(n_u)}$ можливо обчислити, мінімізувавши наступний функціонал:

$$J(\theta^{(1)}, \ldots, \theta^{(nu)}) =$$
$$= \frac{1}{2}\sum_{j=1}^{n_m}\sum_{i:r(i,j)=1}((\theta^{(j)})^T x^i - y^{(i,j)})^2 + \frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{l=1}^{n}(\theta_l^j)^2 \quad (2.3)$$

Але обидва набора векторів невідомі. Існує два способи рішення даної проблеми. Перший полягає в тому, щоб форматувати один з наборів маленькими випадковими значеннями та обчислити другий набір векторів. Далі, на основі другого набору векторів, знайти перший набір і так далі. Уявити це можна так:

$$\theta \to x \to \theta \to x \to \theta \to x \dots$$

Алгоритм, який реалізує дану ідею, називається «The Alternating Least Squares» (алгоритм чергуються найменших квадратів, скорочено ALS).

Другий спосіб більш простий. Можливо обчислити обидва набориθ таxодночасно. Якщо придивитися до функціоналу формул (2.2) та (3.3), можливо виявити, що за винятком регуляризації вони обчислюють однакову суму. У першому фунціоналі обчислюється сума помилок для тих користувачів, хто оцінив даний заклад. У другому випадку для кожного користувача обчислюється сума квадратичних відхилень для тих закладів, які оцінені даними користувачем. Таким чином, в обох випадках береться сума відхилень для всіх пар(користувач, заклад) значення яких в таблиці r дорівнює один, тобто даний користувач оцінив цей заклад. Виходить наступний функціонал (який називають функцією втрат або функція вартості), який необхідно мінімізувати:λ

$$J(x^{(1)}, \dots, x^{(n_m)}, \theta^{(1)}, \dots, \theta^{(n_u)} = \tag{2.4}$$
$$= \frac{1}{2}\sum_{(i,j):r(i,j)=1}\left(\left(\theta^{(j)}\right)^T x^i - y^{(i,j)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{l=1}^{n}(\theta_l^j)^2 + \frac{\lambda}{2}\sum_{i=1}^{n_m}\sum_{l=1}^{n}(x_l^i)^2$$

Вектора θ і x ініціалізіруются випадковим маленькими значеннями. Після обчислення всіх характеристик виходить наступна матриця пророкувань:

$$\begin{bmatrix} (\theta^{(1)})^{T}x^{(1)} & (\theta^{(2)})^{T}x^{(1)} & \cdots & (\theta^{(n_u)})^{T}x^{(1)} \\ (\theta^{(1)})^{T}x^{(2)} & (\theta^{(2)})^{T}x^{(2)} & \cdots & (\theta^{(n_u)})^{T}x^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ (\theta^{(1)})^{T}x^{(n_m)} & (\theta^{(2)})^{T}x^{(n_m)} & \cdots & (\theta^{(n_u)})^{T}x^{(n_m)} \end{bmatrix} \quad (2.5)$$

В цій матриці містяться передбачення оцінок всіх користувачів для всіх закладів відпочинку. Наприклад, передбачення оцінки користувача j для закладу і розраховується як $(\theta^{(j)})^{T}x^{(i)}$.

Запишемо матрицю (2.5) в векторному вигляді. Для цього вводимо матриці Мта U:

$$M = \begin{bmatrix} \cdots & (x^{(1)})^{T} & \cdots \\ \cdots & (x^{(2)})^{T} & \cdots \\ \cdots & \vdots & \cdots \\ & (x^{(n_m)})^{2} & \end{bmatrix} \quad (2.6)$$

$$U = \begin{bmatrix} \cdots & (\theta^{(1)})^{T} & \cdots \\ \cdots & (\theta^{(2)})^{T} & \cdots \\ \cdots & \vdots & \cdots \\ & (\theta^{(n_u)})^{T} & \end{bmatrix} \quad (2.7)$$

Матриця (2.6) має розмірність n_m×k та зберігає характеристики всіх закладів відпочинку. Матриця (2.7) має розмірність n_u×k та зберігає параметри, описують відношення даного користувача до характеристик.

Матрицю передбачень можна представити таким чином:

$$Y = M(U)^{T} \quad (2.8)$$

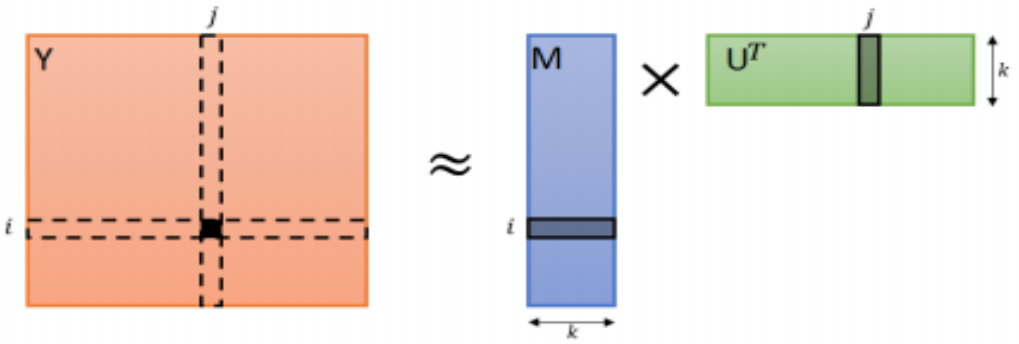

Рисунок 2.1 - Інтуїтивно-зрозуміле уявлення

Вище описаний метод має назву «Low rank matrix factorization» (факторизація матриці низького рангу).[13]Ми апроксимуємо матрицю Y, представляючи її як добуток двох матриць. Однак слід врахувати той факт, що апроксимація неточна і в цьому перевага алгоритму. Метою є отримати передбачення для тих закладів, які користувач ще не оцінив, тобто замінити нулі в початковій матриці прогнозами. Досягти цього допомагає те, що мінімізіруємий функціонал враховує помилки тільки тих ділянок матриці, для яких проставлені оцінки.

2.3 Оптимізація градієнтного спуску

Раніше було описано дві стратегії обчислення наборів векторів$\theta^{(1)}$, $\theta^{(2)}$, … , $\theta^{(nu)}$ и $x^{(1)}$, $x^{(2)}$, … , $x^{(nm)}$. Перший – алгоритмALS, другий – мінімізація функції втрат (2.4). Скористаємося другою стратегією, як більш простий в реалізації. Одним з найпопулярніших алгоритмів мінімізації є градієнтний спуск. Ідея полягає в тому, щоб оновлювати параметри функції втрат в протилежному напрямку градієнта даної функції. Іншими словами, якщо функція при поточних параметрах зростає, тобто градієнт позитивний, то мінімум знаходиться зліва, і ми віднімаємо позитивний градієнт. якщо градієнт негативний, то мінімум знаходиться праворуч і негативний градієнт віднімається.[14]

$$\omega = \omega - \lambda \times \nabla\omega J(\omega) \quad (2.9)$$

Де ω - параметри,

J(ω) – функція втрат,

Λ - коефіцієнт швидкості навчання (learning rate).

З формули (3.9) видно, що параметри оновлюються на величину кратну градієнту. Коефіцієнт швидкості навчання необхідний для регуляції стабільності алгоритму і швидкості його сходження. При занадто великому значенні коефіцієнта, алгоритм може розходитися, а надто маленькому, сходиться дуже повільно.

Градієнт для набору $x^{(1)}$, $x^{(2)}$, … , $x^{(nm)}$, при функції втрат (2.4) розраховується наступним чином:

$$\frac{dY}{dx_k^{(i)}} = \sum_{j:r(i,j)=1}((\theta^{(j)})^T x^i - y^{(i,j)})\theta_k^{(j)} + \lambda x_k^{(i)} \quad (2.10)$$

Де λ - коефіцієнт регуляризації

Для набору$\theta^{(1)}$, $\theta^{(2)}$, … , $\theta^{(nu)}$:

$$\frac{dY}{d\theta_k^{(j)}} = \sum_{i:r(i,j)=1}\left(\left(\theta^{(j)}\right)^T x^i - y^{(i,j)}\right)x_k^{(i)} + \lambda\theta_k^{(j)}s \quad (2.11)$$

Де λ - коефіцієнт регуляризації.

Алгоритм градієнтного спуску можна описати в три кроки:

а) Задаються початкові значення для параметрів (в випадку колаборативної фільтрації це маленькі випадкові значення) і задається точність розрахунку ε;

б) Розраховуються оновлені значення параметрів за формулою (2.9);

в) Перевіряється умова зупинки. Наприклад, якщо функція втрат змінилася менше, ніж на величину ε, алгоритм припиняє роботу, в іншому випадку переходить в пункт б.

Даний тип алгоритму носить назву batch gradient descent. Однак він погано працює на великих даних, тому що кожне оновлення вимагає обчислень для всього набору даних. Є альтернативний алгоритм, який називається stochastic gradient descent (алгоритм стохастичного градієнта, скорочено sgd). Оновлення параметрів відбувається для кожної пари вхідних даних. Таким чином, замість формули (2.9) застосовується формула (2.12).

$$\omega = \omega - \lambda \times \nabla_{\omega} J\left(\omega, x^{(i)} ; y^{(i)}\right) \quad (2.12)$$

Де ω - параметри,

$(x^{(i)} ; y^{(i)})$ - приклад з вхідного набору даних.

Градієнтний спуск має один значний мінус - він сходиться в локальний мінімум. Наглядно дана проблема демонструється на рисунку 2.2.

Якщо початкові значення параметрів задаються вточці A, то градієнтний спуск благополучно сходиться в глобальний мінімум C. Якщо в точці B, то алгоритм сходиться в перший же локальний мінімум, в даному випадку D. Функція втрат (2.4) має безліч локальних мінімумів, і потрапляння в один з таких мінімумів може помітно позначитися на якості пророкувань. Можна намагатися запускати алгоритм кілька разів, намагаючись ініціювати параметри в такому місці, де градієнтний спуск зможе уникнути локальних мінімумів. Однак через специфіку функції втрат на це піде величезна кількість спроб і ніколи не можна бути впевненим у тому, що алгоритм зійшовся саме в глобальний мінімум.

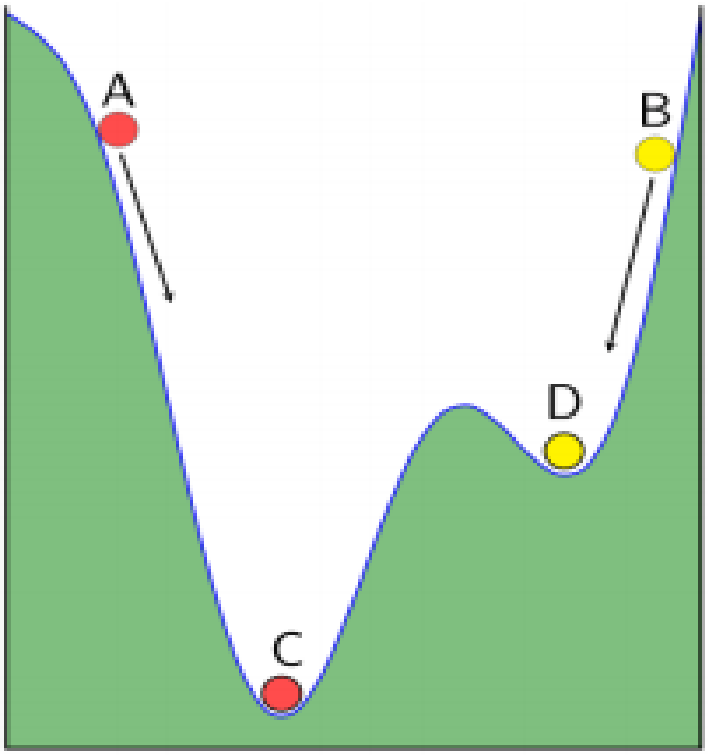

Рисунок 2.2 - Проблема локального мінімуму

Однак існують алгоритми, які оптимізують класичний градієнтний спуск і допомагають уникнути таких проблем, як локальні мінімуми. Одним з найбільш простих і ефективних є оптимізація, яка носить назву momentum (імпульс). Ідея продемонстрована на рисунок 2.3

Для ясності скористаємося фізичної інтерпретацією. Уявімо, що кулька котиться по поверхні, представленої на рисунку 2.3, і починає свій рух в точці B. Даний схил крутий, куля починає збільшувати свій імпульс і, при досягненні локального мінімуму в точці D, долає дану перешкоду. Цьому сприяє два фактори: крутий спуск, що допомагає накопичити імпульс (за рахунок збільшення швидкості), досить пологий підйом. В випадку, якщо куля рухається з точки A, він не проскочить глобальний мінімум C, тому що відрізок CE крутий. У градієнтному спуску роль імпульсу виконує градієнт.

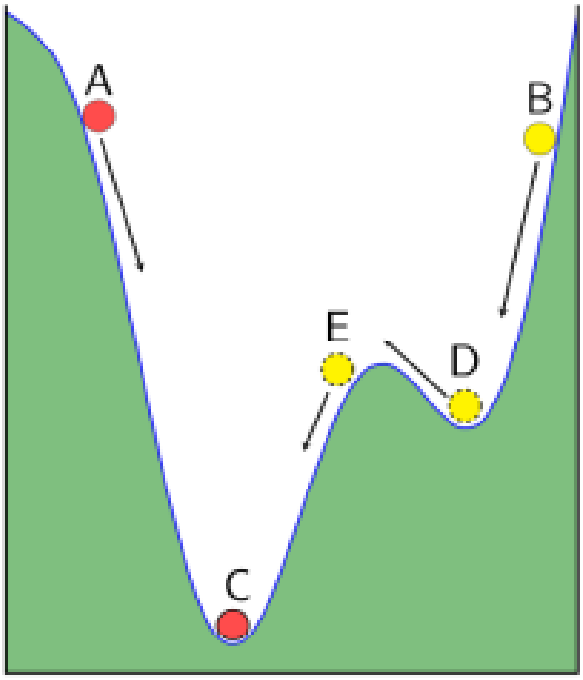

Рисунок 2.3 Градієнтний спуск з імпульсом

Для реалізації даного типу оптимізації необхідно формулу (2.9) замінити на формули (2.13) та (2.14):

$$v_t = 0.9 \times v_{t-1} - \lambda \times \nabla_\omega J(\omega) \quad (2.13)$$

$$\omega = \omega - v_t \quad (2.14)$$

Імпульс для кожного параметра накопичується в векторі v.

2.4 Огляд методів кластеризації

Кластеризація - це завдання угруповання об'єктів таким чином, щоб схожі об'єкти знаходилися в одній групі (кластері). Дане завдання вирішується в таких важливих областях як розпізнавання образів, аналіз зображень, пошук інформації, біоінформатика та інші. існує кілька основних підходів, розглянемо їх:[11]

а) Ієрархічна кластеризація - Даний метод кластеризації заснований на побудові ієрархії кластерів. Існує дві стратегії, які реалізують даних підхід: агломеративний та дівізівний. Перша виділяє кожен об'єкт як окремий кластер і потім починає об'єднувати схожі пари кластерів, поступово зменшуючи їх загальну кількість. Друга відштовхується від ідеї, що всі об'єкти спочатку складаються в одному кластері, який рекурсивно ділиться на підкластери .

б) Кластиризація на основі центроїд - в даному методі завдання ставиться у наступному вигляді: вводяться центри кластерів, і об’єкт присвоюється кластеру з найближчим центром. Далі стоїть завдання знайти такі центри, які мінімізували б квадратні відстані від кластера. Дане завдання є NP складної і тому загальний підхід полягає в пошуку наближених рішень. Найбільш популярні алгоритми дають рішення в локальному мінімумі, проте їх можна оптимізувати.

в) Кластеризація на основі щільності - в даному випадку кластери визначаються як області з більш високою щільністю, ніж інші дані. Об'єкти в розряджених областях вважаються шумовими і прикордонними точками.

Існують і інші підходи кластерного аналізу, однак вони незастосовні на наборах даних, представлених в даній роботі.

2.5 Огляд алгоритму кластеризації

В даній роботі розробляється рекомендаційна система яка використовує кластеризацію користувачів для поліпшення якості рекомендацій. Характеристики користувача представлені масивом чисел від 1 до 10. Найбільш популярним алгоритмом кластеризації даних подібного типу є алгоритмk-means. Саме він використовується в даній роботі.

Даний метод розбиває безліч об’єктів на задане число кластерів. Перший крок алгоритму полягає в ініціалізації центрів кластерів (Центроїд)μ_1, μ_2, …, $\mu_K \in \mathbb{R}^n$. Даний запис визначає, що кількість кластерів

дорівнює K, а розмірність кожного об’єкта n. Кожен елемент безлічі відноситься до кластеру з найближчим центром. Наступні два кроки повторюються, поки алгоритм не зійдеться

На другому кроці заповнюється вектор c. Це можливо записати наступним шляхом:

$$\text{fori} = 1\,\text{tom}$$
$$c^{(i)} = \{p: ||x^{(i)} - \mu_p||^2 \leq ||x^{(i)} - \mu_j||^2 \forall j, 1 \leq j \leq K \qquad (2.15)$$

Де $x^{(i)}$– об’єкт з безлічі вхідних даних

c – вектор, який зберігає відповідність між об’єктом $x^{(i)}$ та кластером $c^{(i)}$

m – кількість об’єктів безлічі вхідних даних

З формули (2.15) випливає, що кожен об'єкт присвоюється до кластеру з найближчим центром. На третьому кроці оновлюється положення центроїд.

$$\text{fork} = 1\,\text{toK}$$
$$\mu_k = \frac{1}{S_k}\sum_{x^{(j)}:c^{(j)}=k} x^{(j)} \qquad (2.16)$$

З формули (2.16) випливає, що кожному центу кластера k присвоюється значення центру мас безлічі об’єктів, що належать даному кластеру. Алгоритм завершується,якщо після поточної ітерації не відбулося зміни положення центрів кластерів. Однак алгоритм k-means має ряд проблем:

а) Досягнення глобального мінімуму не гарантується;

б) Різні початкові положення центрів кластерів можуть призводити до різних розбиттів;

в) Число кластерів необхідно знати заздалегідь.

Розглянемо способи вирішення даних проблем. Для початку введемо функцію втрат, яка буде оцінювати якість розбиття. Вона наведена формулою (2.17).

$$J\left(c^{(1)}, \ldots, c^{(m)}, \mu 1, \ldots, \mu_K\right) = \frac{1}{m}\sum_{i=1}^{m} ||x^i - \mu_{c^{(i)}}||^2 \qquad (2.17)$$

Мету алгоритму можна уявити як мінімізацію даного функціоналу, тобто суми квадратичних відстаней від центрів кластерів до об'єктів відповідних цим кластерам. Розглянемо першу проблему – відсутність гарантії досягнення глобального мінімуму. Проблема проілюстрована на рисунках 2.4, 2.5, 2.6.

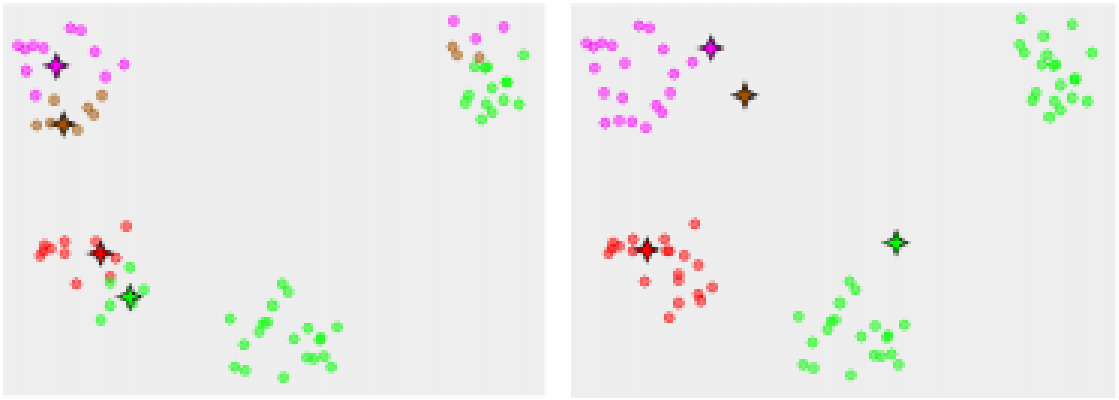

Рисунок 2.4 – Початкове положення і перша ітерація алгоритму k-means

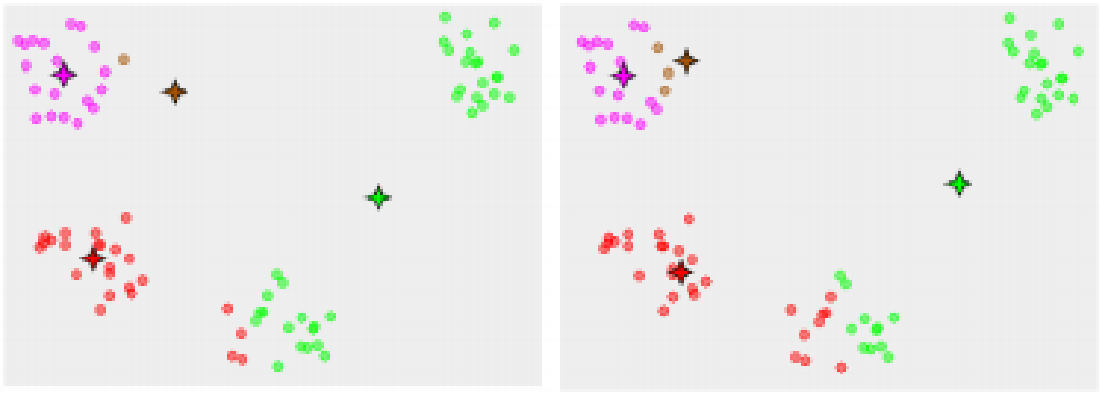

Рисунок 2.5 – Ітерації 2,3 алгоритму k-means

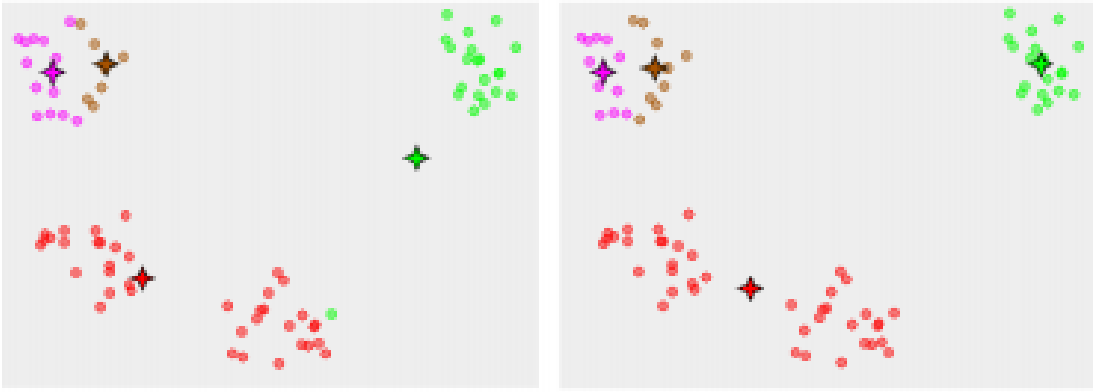

Рисунок 2.6 – Ітерації 4,5 алгоритму k-means

На рисунках продемонстровано роботу алгоритму k-meansпри невдалій ініціалізації центрів кластерів. Кружечками позначені точки даних, а зірки – це центроїди. Алгоритм зійшовся в локальний мінімум за 5 ітерацій, його результат суперечить очевидної кластерної структурі даних. Вирішити дану проблему допомагає множинна ініціалізація. Алгорит мініціалізується певною кількістю різних центроїдів, і вибирається той результат роботи алгоритму, при якому функція втрат приймає найменше значення.

Третя проблема – кількість кластерів необхідно знати заздалегідь. Рішенням служить метод ліктя (elbowfork-meansclustering). Він пропонує розробнику самостійно вибрати кількість кластерів, оцінюючи зменшення функції втрат, при збільшенні їх кількості. Для цього будуються графіки як на рисунку 2.7.

Графік представляється як рука і вибирається місце з найбільшим вигином, звідси випливає і назва методу. Місце сильному вигину говорить про те, що зменшення функції при переході від поточної точки до наступної значно менше, ніж при переході від попередньої і поточної. Таким чином, подальше збільшення кількості кластерів не сильно покращує функцію втрат, і поточне розбиття безлічі можна вважати оптимальним. [16]

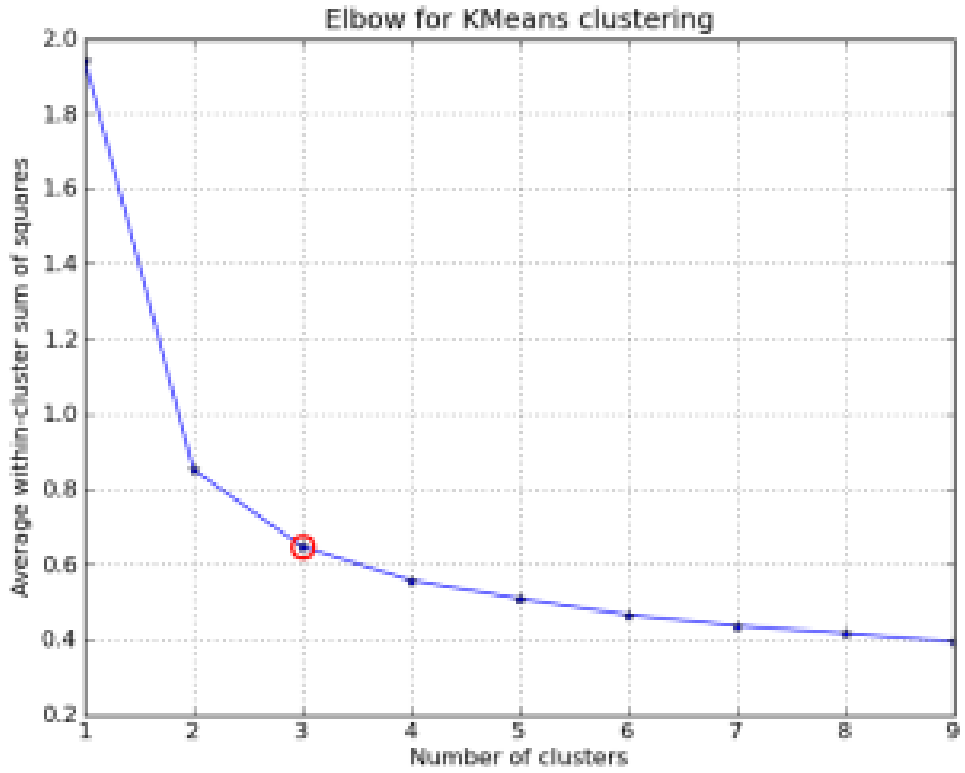

Рисунок 2.7 – Метод ліктя для визначення кількості кластерів

2.6 Вимірювання якості рекомендацій

Після того як зроблені рекомендації необхідно оцінити їх якість. Існує велика кількість метрик, що дозволяють оцінити параметри якості рекомендаційної системи, розглянемо основні. [21]

2.6.1 Точність і повнота

Даний тип метрик відноситься до метрик ранжирування. Для використання даної метрики необхідно перевести бальну систему в бінарну, тобто поміняти оцінки на передбачення «рекомендувати» чи «не рекомендувати». Введемо дві безлічі: Ri P. Безліч R включає в себе всі рекомендовані об’єкти, а безліч Роб’єкти, які дійсно сподобалися користувачеві.

Таким чином оцінити точність рекомендації можна за формулою:

$$\text{Точність} = \frac{|R \cap P|}{|R|} \tag{2.18}$$

А повноту рекомендації формулою:

$$\text{Повнота} = \frac{|R \cap P|}{|P|} \tag{2.19}$$

Найчастіше ці метрики об’єднують в одну величину, звану F-мірою.

$$F = \frac{2 \times \text{точність} \times \text{повнота}}{\text{точність} + \text{повнота}} \tag{2.20}$$

2.6.2 RMSE

Однією з самих популярних метрик, які вимірюють якість рекомендацій, вважається міра RMSE(Root Mean Square Error). Вона обчислюється за формулою(2.15).

$$RMSE = \sqrt{\frac{1}{T}\sum_{(i,j \in T}(y_{ij}^{i} - y_{ij})^2} \tag{2.21}$$

Де T– загальна кількість тестових оцінок.

(i,j) – пара (заклад, користувач) з тестового набору.

y^i– оцінка передбачення.

y – реальна оцінка користувача.

Дана метрика не є ідеальною і має ряд недоліків, наприклад користувачі з більш широким розкидом оцінок будуть впливати більше. Також можна мати ідеальну метрику RMSE, але дуже погане ранжування. Однак дана метрика є найпопулярнішою в даній сфері, тому будемо використовувати її.

2.6.3 Евклідова метрика

Евклідова метрика (евклідова відстань) - метрика в евклідовому просторі - відстань між двома точками евклідового простору, що обчислюється за теоремою Піфагора

Для точок $p=(p_1,\ldots,p_n)$ і $q=(q_1,\ldots,q_n)$ евклідова відстань визначається наступним чином:

$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2} = \\ = \sqrt{\sum_{k-1}^{n}(p_k - q_k)^2} \quad (2.22)$$

2.7 Висновки до розділу

В даному розділі було сформульовано математичну постановку задачі та представлено модифікований метод кластеризації даних

Обрано та обґрунтовано метрику для роботи

3 РОЗРОБКА РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ

Розробляється рекомендаційна система представлена клієнт-серверним додатком. Архітектура реалізована за допомогою шаблон проектування MVC (Model-View-Controller, «Модель-Представлення-Контролцер». Для зберігання даних використовується база даних PostgreSql. Докладний архітектурний рівень продемонстрований на рисунку 4.1.

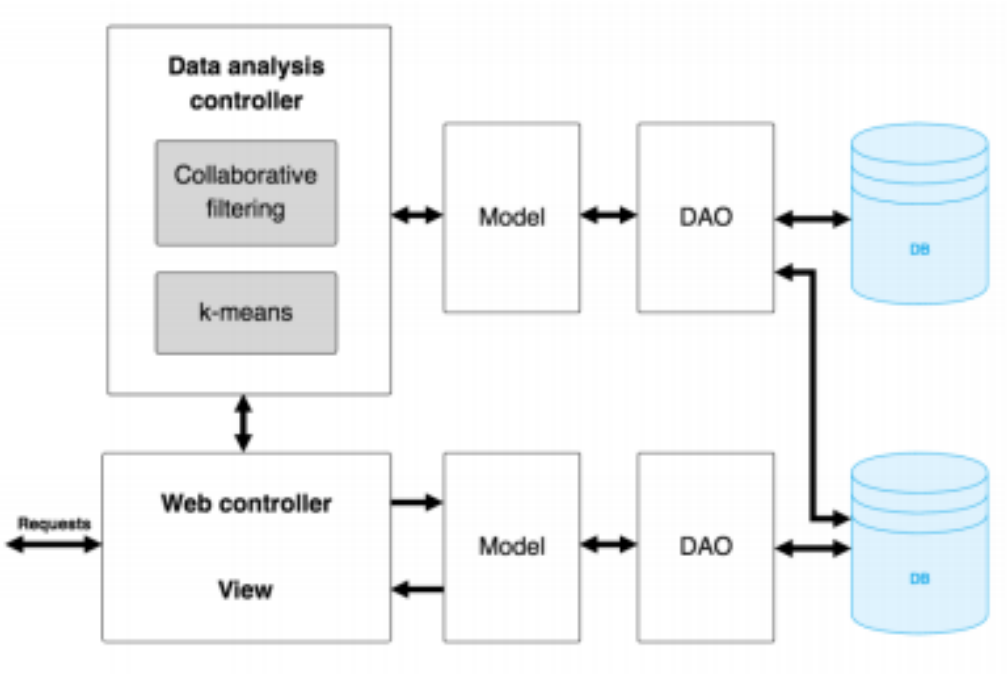

Рисунок 3.1 Докладний архітектурний рівень

3.1 Опис серверної частини

Серверна частина цього додатка реалізована на мові високого рівня java і складається з двох логічних частин. Перша (webcontroller) займається обробкою запитів клієнта і роботою з базою даних, друга (data analysis controller) створює рекомендації, реалізуючи алгоритми, описані в теоретичної частини

3.1.1 Webcontroller

За організацію роботи сервлетів відповідає вільний контейнер сер влетів Jetty. його головним перевагою перед конкурентами є швидкість обробки запитів при невеликої та середньої завантаженні. Взаємодія з клієнтом відбувається згідно архітектурному стилю REST. Управління даними відбувається по протоколу HTML за допомогою методів: GET (отримати), POST (додати, змінити, видалити), PUT (додати, замінити), DELETE (видалити). Найчастіше використовуються перші два методи. Для реалізації архітектури REST використовується фреймворкJersey.

Обробкою запитів займаються такі контролери: LoginResource, LogonResource, MyPageResource, SearchResource, FavoritesResource, GetRecResource. Для роботи з базою даних використовуються сервіси, які взаємодіють з контролерами за допомогою спеціального механізму, який називається Dependencyinjection (впровадження залежності). Підключення до бази даних здійснюється за допомогою класу Database. При першому підключенні створюється connection pool. Це спеціальний кеш, який зберігає з'єднання з базою даних для повторного їх використання. Сервіс, який бажає виконати читання або запис в базу даних, повинен викликати статичний метод getConnction (), який поверне готове до роботи підключення.

3.1.2 Data analysis controller

Дана частина реалізує всі описані алгоритми і результатом роботи є id рекомендованого закладу. Взаємодія з Web controller відбувається за допомогою класу Recommender. Data analysis controller складається з двох модулів: модуль обробки даних і модуль побудови рекомендацій.

Перший перетворює вхідні дані в зручний для роботи вид. Інформація зберігається в файлах формату csv - це текстовий формат, призначений для зберігання табличних даних. Розроблено спеціальні методи, які дозволяють

читати даний файл в двовимірний масив і записувати двовимірний масив в csv файл. Таким чином, вся робота з даними представляється як робота з масивом. Ще одне завдання, що виконується даними модулем, це парсинг вхідних даних, які представлені у вигляді JSON об'єкта.

У другому модулі реалізований алгоритм факторизації матриці і k-means. У класі GradientDescent реалізовані алгоритми градієнтного спуску і градієнтного спуску з оптимізацією momentum. Ці методи приймають об'єкти реалізують інтерфейс CostAndGrad. Даний інтерфейс містить два абстрактних методу: getJ() – розрахунок функції втрат і getGrad () - розрахунок градієнта. Інтерфейс реалізований класами ColFilCostAndFunc, який обчислює функцію втрат (2.4) і градієнти (2.10) і (2.11), а також класом KMeansCost, обчислює функцію втрат (2.15).

Робота з двовимірними масивами проводиться за рахунок використання бібліотеки лінійної алгебри jblas. Вона володіє не найширшим функціоналом, однак є найшвидшою бібліотекою для java.

Також в даному модулі реалізований механізм оцінки якості за допомогою метрики RMSA. Здається розмір вибірки і відсоток даних, на яких будуть навчатися параметри.

3.2 Опис клієнтської частини

Клієнтська частина реалізована у вигляді web-сайту, що складається з шести сторінок, кожну обробляє відповідний контролер. Статичні дані зберігаються на стороні сервера в форматі jsp, після GET запиту користувача відбувається формування html сторінки. Динамічні дані завантажуються за допомогою асинхронних javascript запитів, виконуваних на стороні сервера. Вони передаються в текстовому форматі JSON. Новий користувач проходить реєстрацію, яка включає в себе введення електронної пошти, пароля, імені, короткого опису себе, року народження, статі та оцінка інтересів по десятибальній шкалі для створення соціального образу.

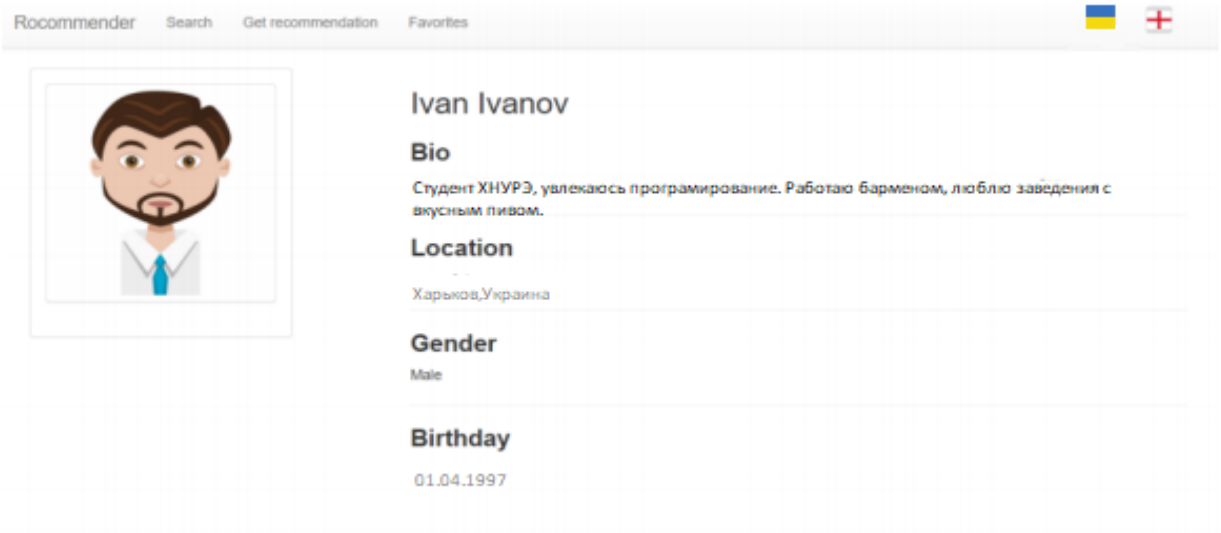

Рисунок 3.2 Профіль користувача

На web-сайті доступна навігація по трьомрозділах: search, getrecommendation і favorites. В першому розділі можна знайти будь-який заклад відпочинку,представлений базою даних. Користувач вводить часткове або повну назву закладу, і після натискання кнопки search виконується ajax запит за допомогою бібліотеки javascript - jQuery. Результатом запиту є JSON відповідь, що містить масив об'єктів, кожен з яких містить інформацію про це місце відпочинку.

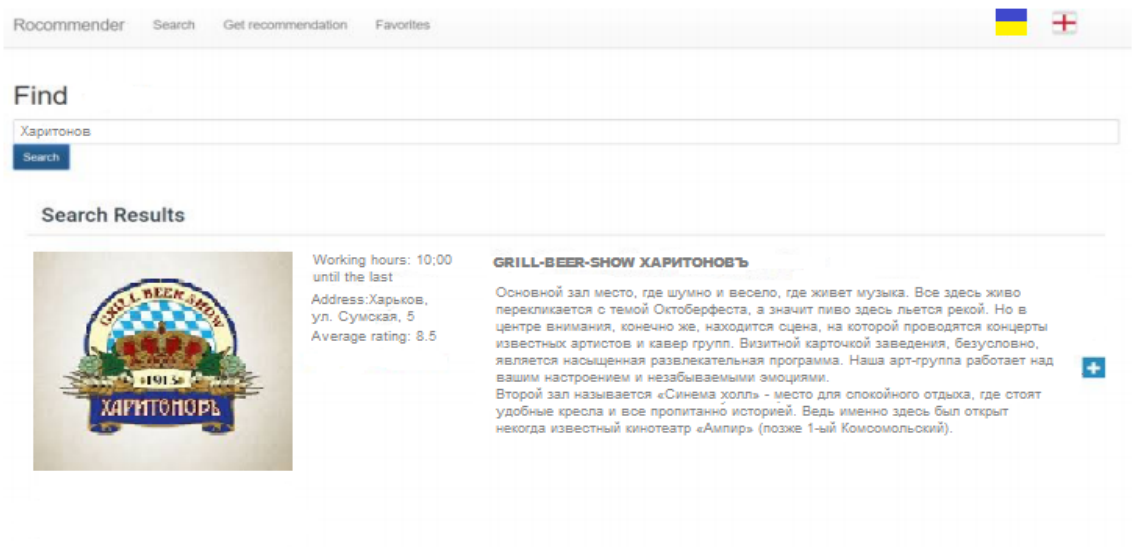

Рисунок 3.3 Сторінка пошуку місця відпочинку

. Далі за допомогою javascript відбувається розбір цих даних і подання користувачеві. Переглянуті місця відпочинку можна відразу оцінити, тим самим підвищити якість майбутніх рекомендацій, а незнайомі, але цікаві до перегляду можна додати в спеціальний список. Місця відпочинку, додані в даний список, доступні в розділі favorites.

Отримати рекомендацію можна в розділі getrecommendation. Виконується асинхронний GET запит з передачею id поточного користувача. Рекомендоване місце відпочинку повертається в форматі JSON. У користувача є три варіанти дій: оцінити заклад і запросити наступну рекомендацію, запросити наступну рекомендацію без оцінки місця, додати місце в розділ favorites.

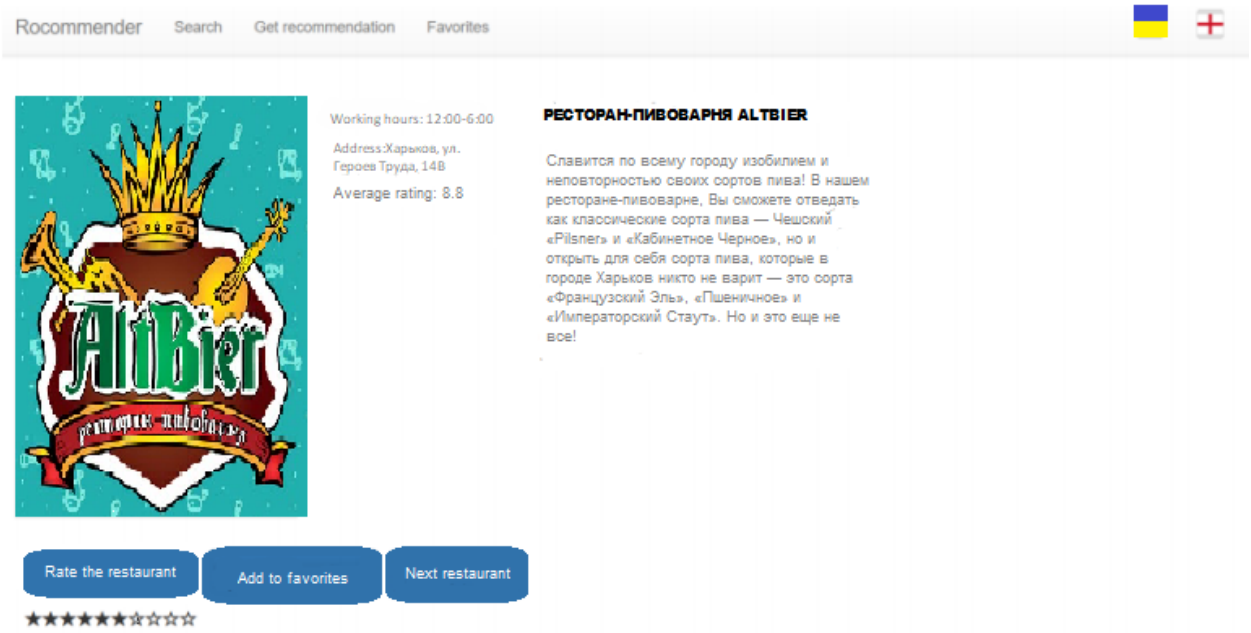

Рисунок 3.4 Сторінка з рекомендаціями

3.3 Оцінка якості моделі

Для оцінки якості моделі необхідно розділити вхідний набір даних на тестову частину і навчальну. Для початку зафіксуємо обсяг вибірки на рівні 1000000 оцінок. Вхідним параметром є відсоток тестової вибірки. На виході: метрика RMSE і час виконання алгоритму. Коефіцієнт швидкості навчання і

кількість ітерацій градієнтного спуску задаються відповідно до оптимальної збіжності градієнтного спуску. Результати представлені в таблиці 3.1.

Таблиця 3.1 – Метрика в залежності від об'єму тестової вибірки

% тестової вибірки	RMSE
5	0.908
10	0.952
20	1.032
35	1.113
50	1.195
70	1.561
95	2.674

З таблиці випливає, що якість рекомендацій падає зі зменшенням обсягу навчальної вибірки. Даний результат узгоджується з теорією з чого можна зробити висновок, що рекомендаційна система працює правильно.

Оцінимо поведінку моделі при зміні обсягу вибірки. Зафіксуємо відсотків тестової вибірки на рівні 10%.

Вхідні дані: обсяг вибірки.

Вихідні: метрика RMSE і час виконання алгоритму.

Результати представлені в таблиці 3.2.

Таблиця 3.2 – Час виконання роботи алгоритму

Обсяг вибірки	RMSE	Час,с
1000	2.780	0.7
10000	1.871	1
100000	1.186	4.2
1000000	0.952	34

Більший обсяг даних дозволяє більш правильно навчитися параметрам і більш чітко виділити латентні характеристики закладів і користувачів. Це показує не випадковість проставлених оцінок і наявність кореляції в моделі.

3.4 Оцінка якості кластеризації

В результаті обробки даних, для кожного користувача в матриці активності було зроблено число звернень до конкретного місця відпочинку. Таким чином, кожен рядок матриці активності користувачів представляє собою вектор оцінок, які відповідають різним категоріям товарів (тематичний профіль користувача). Профіль користувача характеризує ступінь його інтересу до кожної групи місць відпочинку. На рисунку 3.5 представлений фрагмент матриці активності користувачів.

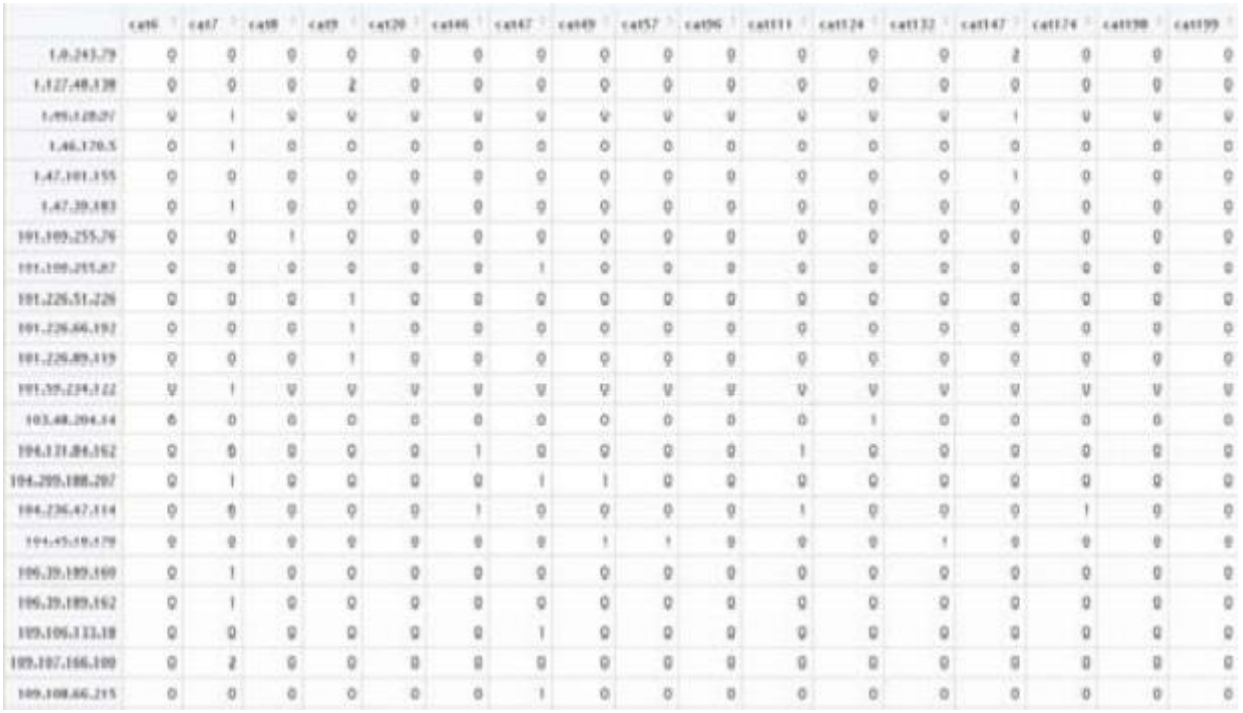

Рисунок 3.5- Матриця активності користувачів

Попереднім етапом перед виявленням груп користувачів зі схожими характеристиками методами кластеризації є обчислення попарних відстаней між елементами матриці активності користувачів. Як метрики відстані

(функція подібності) було використано відстань Евкліда (геометрична відстань в багатовимірному просторі), яка є однією з найбільш простих для реалізації і часто використовуваних на практиці на сьогоднішній день. На рисунку 3.6 представлена гістограма відстаней, отримана в результаті обробки матриці з попарними відстанями між об'єктами. Для побудови гістограми використовувався статистичний пакет R. Попередній розгляд результатів дозволяє зробити висновок, що переваги (вектора активності) користувачів не сильно відрізняються (тобто швидше за все більшість користувачів цікавляться подібними місцями відпочинку). Даний висновок можна зробити виходячи з того, що близько 90% даних на гістограмі зосереджено на відрізку відстаней від 0 до 5.

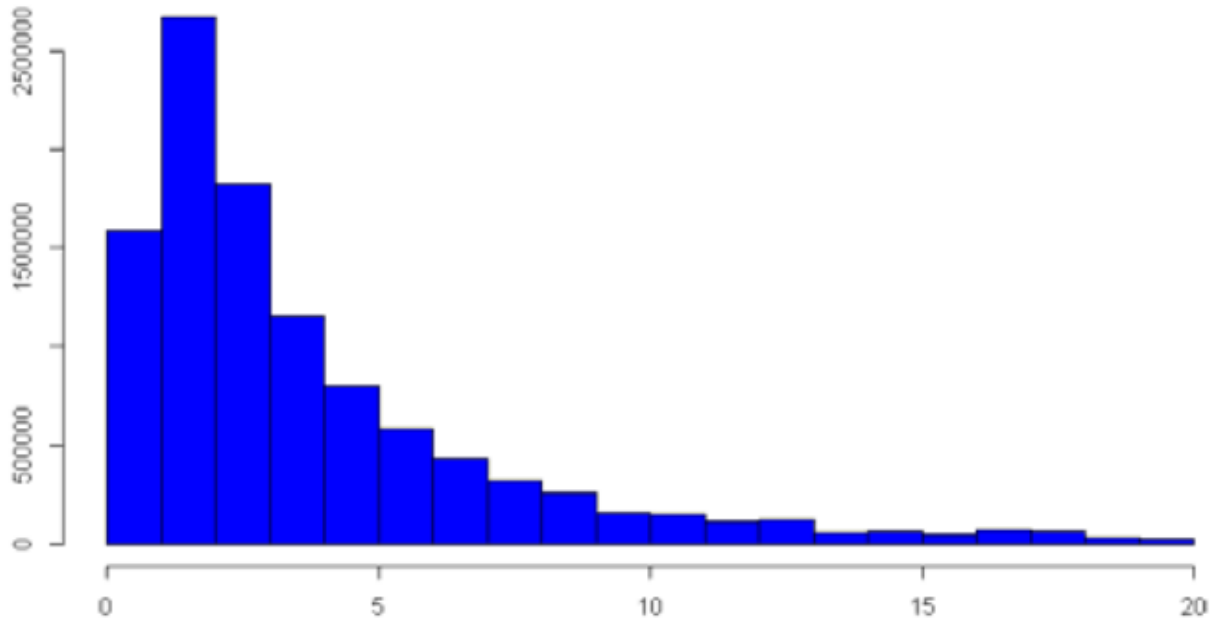

Рисунок 3.6 – Гістограма відстаней

Наступним кроком було визначення оптимального числа кластерів, необхідного для застосування методу кластеризації k-means. На рисунку 3.7 наведені результати розрахунку всередині кластерної суми квадратів відстаней по методу ліктя (Elbow method). Даний метод дав число в 30 кластерів.

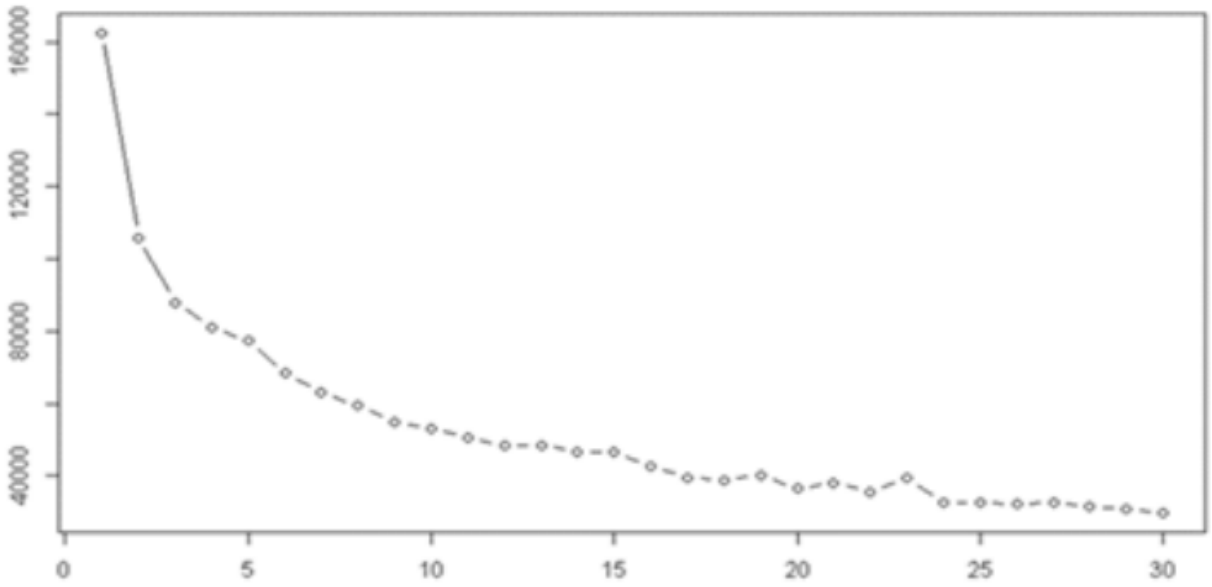

Рисунок 3.7- Аналіз кількості кластерів

Таким чином були отримані всі необхідні параметри і проведена кластеризація даних по матриці активності користувачів.

ВИСНОВОК

В ході даної роботи були досліджені основні принципи побудови рекомендаційних систем. Була обрана модель, яка дає більш якісні результати на теоретичному рівні. Були реалізовані всі розглянуті алгоритми, і проведена оцінка якості їх роботи за допомогою відповідних метрик.

Розроблена рекомендаційна система є сервісом, який допомагає користувачам знайти гарний заклад для відпочинку. Однак архітектура спроектована таким чином, що дані можуть бути замінені, або додані нові елементи для рекомендацій без втручання в інші частини проекту.

Для наочної демонстрації роботи системи був розроблений web-сервіс. При реєстрації користувач відповідає на ряд питань, які допомагають сформувати його соціальний образ, що має допомогти підвищити якість рекомендацій. Однак для істотного впливу необхідна велика кількість користувачів. Подальшим розвитком даного проекту є інтеграція в соціальні мережі. Вони містять у собі історію користувача протягом тривалого періоду часу, це дозволяє отримати більш об'єктивні дані про користувача і сформувати правильніший соціальний образ.

ПЕРЕЛІК ПОСИЛАНЬ

1. Жернакова О. Системи рекомендацій і пошуку відео контента // Телемультимедіа, 2012. [Електронний ресурс]. – Режим доступу http://old.telemultimedia.ru/art.php?id=464&rid=22

2. Recommendation Systems – How Companiesare Making Money [Електронний ресурс]. – Режим доступу https://sigmoidal.io/recommender-systems-recommendation-engine

3. Automatic movieratings prediction using machine learning [Електронний ресурс]. – Режим доступу http://www.csc.kth.se/~miksa/papers/AutomaticMovieRatingsPrediction_ MIPRO.pdf .

4. Statistical Analysis of K-Nearest Neighbor Collaborative Recommendation [Електронний ресурс]. – Режим доступу https://arxiv.org/pdf/1010.0499.pdf

5. Xiaoyuan Su, Taghi M. Khoshgoftaar A Survey of Collaborative Filtering Techniques. SuXiaoyuan, M. Khoshgoftaar Taghi [Електронний ресурс]. – Режим доступу: http://www.hindawi.com/journals/aai/2009/421425/

6. Models and methods for building web recommendation systems // Наукові журнали та конференції, 2018 [Електронний ресурс] / Режим доступу: http://science.lpnu.ua/uk/keywords-paper/kolaboratyvna-filtraciya .

7. Мазурік О.Ю. Покращення результатів роботи рекомендаційних алгоритмів за допомогою алгоритму SVD / Мазурік О.Ю. // International Scientific Journal. – 2015. – #9. – С. 61-65.

8. J. BenSchafer. Recommender Systems in E-Commerce / J. Ben Schafer, Joseph Konstan, John Riedl // Group Lens Research Project Book. – Minneapolis, Minnesota, USA: University of Minnesota, 2011. – С. 43 – 55.

9. Рекомендательные системы // Офіційний блог для розробників компанії IBM. [Електронний ресурс] / Режим доступа : https://www.ibm.com/developerworks/ru/library/os-recommender1/.

10. Мазурік О.Ю. Порівняльний аналіз моделей оцінювання в рекомендаційних системах / Мазурік О.Ю. // Системний аналіз та інформаційні технології : «САІТ-2016», 30 травня –2 червня 2016, Київ, Україна : матеріали. – К. : НТУУ «КПІ», 2016. – С. 113

11. Джонс М. Рекомендательные системы. [Електронний ресурс] / Режим доступу https://www.ibm.com/developerworks/ru/library/os-recommender1/index.html

12. Алгоритми кластеризації .Сергей Николенко[Електронний ресурс] / Режим доступу https://logic.pdmi.ras.ru/~sergey/teaching/ml/11-cluster.pdf

13. Методи кластеризації К.В.Воронцов[Електронний ресурс] / Режим доступуhttp://www.machinelearning.ru/wiki/images/archive/2/28/20150427184336%21Voron-ML-Clustering-slides.pdf

14. Алгоритм_k_средних_(k-means) Д.Бротиковская и Д.Зобнин [Електронний ресурс] / Режим доступу https://algowiki-project.org/ru/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_k_%D1%81%D1%80%D0%B5%D0%B4%D0%BD%D0%B8%D1%85_(k-means)

15. Matrix Factorization and Collaborative Filtering, Matt Gormley[Текст] Lecture 25 April 19, 2017

16. Градиентный спуск: всё, что нужно знать, Neurohive, Базовый курс [Електронний ресурс] / Режим доступу: https://neurohive.io/ru/osnovy-data-science/gradient-descent/ 20 листопада 2018р.

17. Воронцов К.В. Лекции по алгоритмам кластеризации и многомерного шкалирования [Електронний ресурс] / Режим доступу http://www.ccas.ru/voron/download/Clustering.pdf

18. T. Segaran. Programming Collective Intalligence [Текст] / T. Segaran // США, Себастополь: O'ReillyMedia, 2007 – с. 368

19. Adolf Proidl. История систем рекомендаций [Електронний ресурс] / Adolf Proidl/Режим доступу: http://www.telemultimedia.ru/art.php?id=582.

20. Aggarwal C.C. Recommender Systems [Текст] / Aggarwal C.C.//Springer, 2016 p. – 493 c.

21. Метрики качества ранжирования [Електронний ресурс] // HABR. – 2016 URL://m.habr.com/ru/company/econtenta/blog/303458/.

22. Matrix Factorization Techniques for recommender systems [Електронний ресурс] / Режим доступу: https://datajobs.com/data-science-repo/Recommender-Systems-[Netflix]. Pdf