Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Системотехніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

другий (магістерський)
(рівень вищої освіти)

(позначення документа)

Розробка методів персоналізованих рекомендацій для електронної B2C комерції

(тема)

Виконав: здобувач групи ІТПм-21-1

Спеціальності

122 Комп'ютерні науки

(код і повна назва спеціальності)

Освітньої програми

Інформаційні технології проектування

(повна назва освітньої програми)

Мещан К.А.
(прізвище, ініціали)

Керівник проф. Міщеряков Ю.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри системотехніки _____ Гребеннік І.В.
(підпис) (прізвище, ініціали)

2022 р.

Я як студент(ка) ХНУРЕ розумію і підтримую політику закладу із академічної доброчесності. Я не надавав(-ла) і не одержував(-ла) недозволену допомогу під час підготовки кваліфікаційної роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

21.12.2022  Мещан

(дата, підпис, прізвище студента/-ки)

Кваліфікаційна робота не містить відомостей заборонених до відкритого опублікування.
Керівник кваліфікаційної роботи проф. Міщеряков Ю.В.

Кваліфікаційна робота виконана у відповідності до стандартів, що діють в Україні.
Керівник кваліфікаційної роботи проф. Міщеряков Ю.В.

Попередній захист проведено 21 грудня 2022 р.
Керівник кваліфікаційної роботи проф. Міщеряков Ю.В.

Харківський національний університет радіоелектроніки

Факультет Комп’ютерних наук

Кафедра Системотехніки

Рівень вищої освіти другий (магістерський)

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

Освітня програма Інформаційні технології проектування

ЗАТВЕРДЖУЮ:
Зав. кафедри

(підпис)

«21» грудня 2022 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачові Мещан Катерині Андріївні
(прізвище, ім’я, по батькові)

1.Тема роботи Розробка методів персоналізованих рекомендацій для електронної B2C комерції
затверджена наказом по університету від « 21 » листопада 2022 р. № 1504 Ст
2.Термін подання здобувачем роботи 22.12.2022 р.
3.Вихідні дані до роботи рекомендації товару новому користувачу системи
4.Перелік питань, що потрібно опрацювати в роботі Вступ. Аналіз предметної області. Огляд методів і технологій. Розробка системи для подолання проблеми холодного старту. Опис прийнятих проектних рішень при розробці системи. Висновки
5.Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп’ютерних ілюстрацій (слайдів) навігація по сайту muzinstr.com.ua, перевантажений деталями сайт jam.ua, радна система на сайті soundmaster.ua, рекомендації на сайті issaplus.com, феномен довгого хвоста, тестові дані для таблиці продуктів, логічна модель бази даних, фізична модель бази даних, результат рекомендацій по не зовсім коректній формулі, рекомендації для ігрового ноутбуку, пошук товарів за окремими характеристиками в базі даних, рекомендації для дорослого користувача

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів	Примітка
1	Отримання завдання на виконання роботи	10.10.2022	
2	Визначення проблеми	11-15.10.2022	
3	Дослідження існуючих методів розв’язання	16-25.10.2022	
4	Розробка плану вирішення проблеми	26.10-01.11.2022	
5	Аналіз предметної області	02-10.11.2022	
6	Вибір засобів розробки програмного забезпечення	11-12.11.2022	
7	Створення та заповнення бази даних	13-20.11.2022	
8	Розробка системи	21-30.11.2022	
9	Оформлення пояснювальної записки	01-16.12.2022	
10	Представлення на рецензування	17.12.2022	

Дата видачі завдання 10 жовтня 2022 р.

Здобувач  Мещан К.А.
(підпис)

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

РЕФЕРАТ

Пояснювальна записка до магістерської кваліфікаційної роботи: 60 с., 6 табл., 22 рис., 2 додатка, 37 джерел інформації.

JAVASCRIPT, БАЗА ДАНИХ, СУБД, MYSQL, ІНФОРМАЦІЙНА СИСТЕМА, РАДНА СИСТЕМА, ПРОБЛЕМА ХОЛОДНОГО СТАРТУ, ДЕМОГРАФІЧНА ФІЛЬТРАЦІЯ

Об’єктом досліджень є процес електронної комерції інтернет-магазину з обліком замовлень та радною системою.

Предметом дослідження кваліфікаційної роботи є інформаційні технології та програмні методи створення серверної частини інформаційної системи обліку продажів, що надає можливість автоматизувати облік продажів та спростити роботу клієнта з системою.

Мета досліджень: розробка компонентів інформаційної системи інтернет-магазина.

Методи дослідження – системний підхід, методи структурного аналізу й моделювання реляційних баз даних.

У роботі проведено проектування та реалізація серверної частини інформаційної системи інтернет-магазину. Розроблена база даних. Проведено аналіз існуючих проблем схожих крамниць і прийнято рішення розробити систему для вирішення проблеми холодного старту. Проведено аналіз і розроблено рекомендації щодо подальшого використання компонентів системи.

Галузь застосування розробки – рекомендаційні системи в електронній B2C комерції.

ABSTRACT

Explanatory note to the master's qualification work: 60 p., 6 tables, 22 figures, 2 appendices, 37 sources of information.

JAVASCRIPT, DATABASE, SUBD, MYSQL, INFORMATION SYSTEM, ADVISORY SYSTEM, COLD START PROBLEM, DEMOGRAPHIC FILTERING

The object of research is the process of e-commerce online store with order accounting and advisory system.

The subject of the qualification work is information technology and software methods of creating the server part of the information system of sales accounting, which provides an opportunity to automate sales accounting and simplify the client's work with the system.

Purpose of research: development of components of the information system of online store.

Research methods – system approach, methods of structural analysis and modeling of relational databases.

The work carried out the design and implementation of the server part of the information system of the online store. The database is developed. The analysis of existing problems of similar stores and decided to develop a system to solve the problem of cold start. The analysis was carried out and recommendations for further use of the system components were developed.

Scope of the development – recommendation systems in electronic B2C commerce.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

ІС – інформаційна система;

БД – база даних;

СУБД – совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных;

SQL – structured query language, декларативна мова програмування, застосовувана для створення, модифікації та управління даними в реляційній базі даних, керованої відповідною системою управління базами даних;

IMDB – Internet Movie Database, веб-сайт із вільно редагованою та найбільшою у світі базою даних про кінематограф, яку можна вільно редагувати;

TF.IDF (TF – term frequency, IDF – inverse document frequency) – статистична міра, яку використовують для оцінювання важливості слова в контексті документа, що є частиною колекції документів або корпусу.

ВСТУП

В даний час покупки в інтернет-магазинах перестали бути чимось що виходить за межі нормального, адже і ціни в подібних магазинах зазвичай нижче, і асортимент більше. В даний час на просторах інтернету можна замовити різноманітні товари. Величезна кількість продукції всіх складнощів та напрямків, безліч оригінальних розробників, професійні та оперативні послуги. Обороти на ринку електронної комерції зростають щодня рекордними темпами.

З ростом популярності інтернет-магазинів зростає кількість магазинів і збільшується їх асортимент. Виникають проблеми, які негативно позначаються як на магазинах, так і на покупцях. Великий асортимент – це дуже добре, але з ним є і деякі труднощі. Покупець, побачив перед собою багато товарів і їх параметрів, може не знайти серед них той, що потрібен. На відміну від звичайного магазину тут, як правило, немає продавця-консультанта. Тому для збільшення зручності користування інтернет-магазинами і для збільшення прибутку, стали вводити нові функції. А саме, різні персоналізовані рекомендації.

Але у цих системах рекомендацій є деякі проблеми з роботою – проблема холодного старту користувача, холодного старту контенту.

Об’єктом досліджень є процес надання персоналізованих рекомендацій в інтернет-додатках.

Предметом дослідження кваліфікаційної роботи є інформаційні технології та програмні методи створення персоналізованих рекомендацій.

Метою роботи є розробка системи зменшення проблеми холодного старту в інтернет-магазині.

Методами дослідження є системний підхід, методи структурного аналізу й моделювання реляційних баз даних.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз предметної області

Проведемо аналіз інтернет-магазинів з продажу різних товарів.

Для початку розглянемо вузькоспрямовані магазини, а саме магазини музичних інструментів:

– muzinstr.com.ua. У цьому магазині дуже зручна навігація по сайту (рис. 1.1), але зовсім нема радної системи;

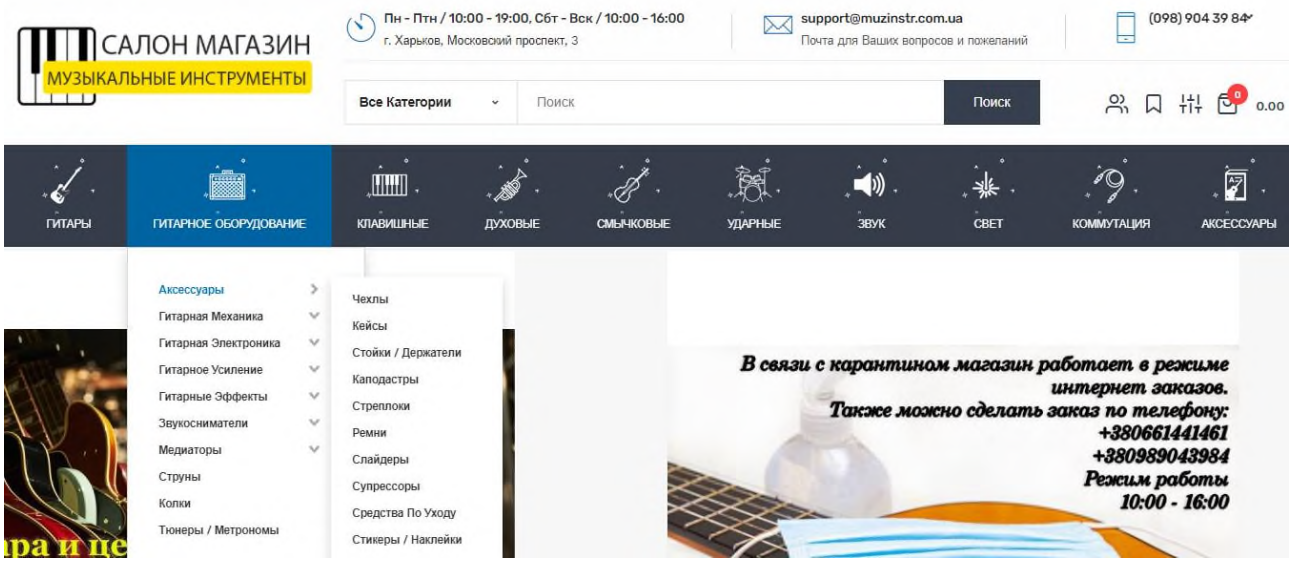

Рисунок 1.1 – Навігація по сайту muzinstr.com.ua

– soundmaster.ua. Цей магазин теж має зручну навігацію (рис. 1.2), але він має ще і радну систему розподілену на декілька розділів (рис. 1.3). У цій радній системі можна обрати потрібний розділ з шести типів товару (струни, чохли, комбопідсилювачі і т.д.) та буде показано список товарів обраного типу, які найчастіше продаються разом з інструментом, який ви переглядаєте;

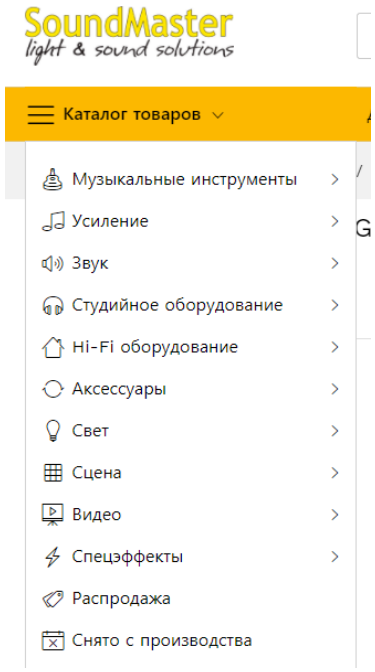

Рисунок 1.2 – Навігація на сайті soundmaster.ua

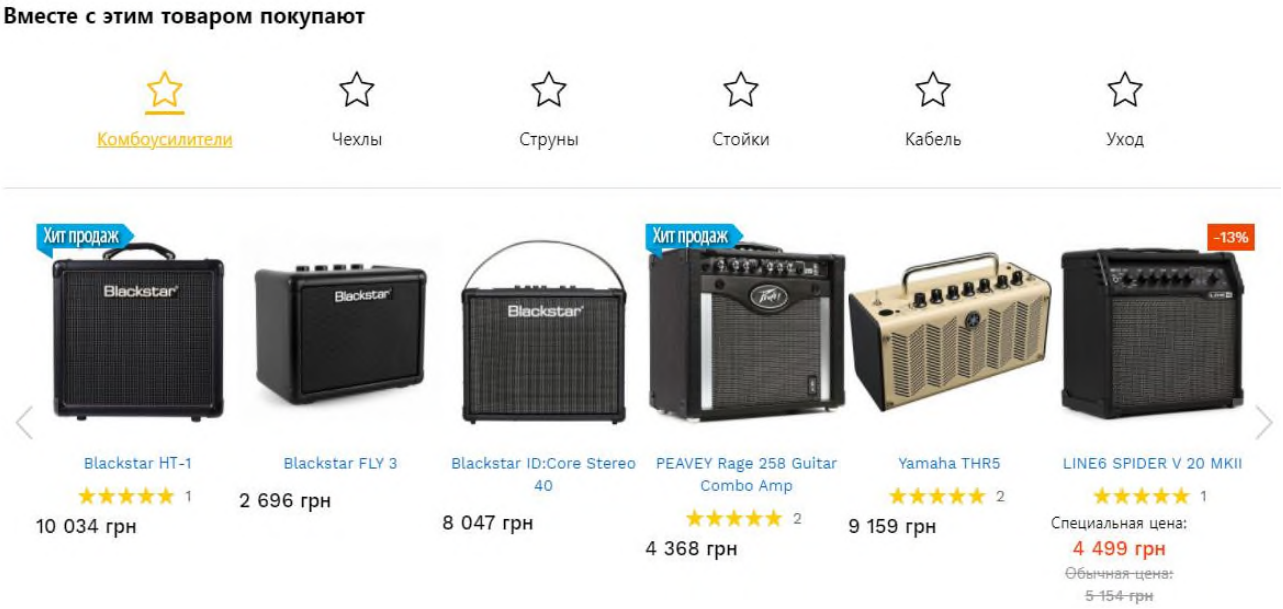

Рисунок 1.3 – Радна система на сайті soundmaster.ua

– у магазині muzline.ua розробники радної системи пішли ще далі. Як і в попередньому магазині у цьому є радна система з розділами, але це не все. Спочатку сайт радить придбати інструмент з ще трьома позиціями (чохол, струни та тюнер) зі знижкою (рис. 1.4). Далі цей інструмент представлено у різних кольорах (рис. 1.5). А в кінці представлено схожі за типом інструменти, які можна відсортувати за брендом (рис. 1.6);

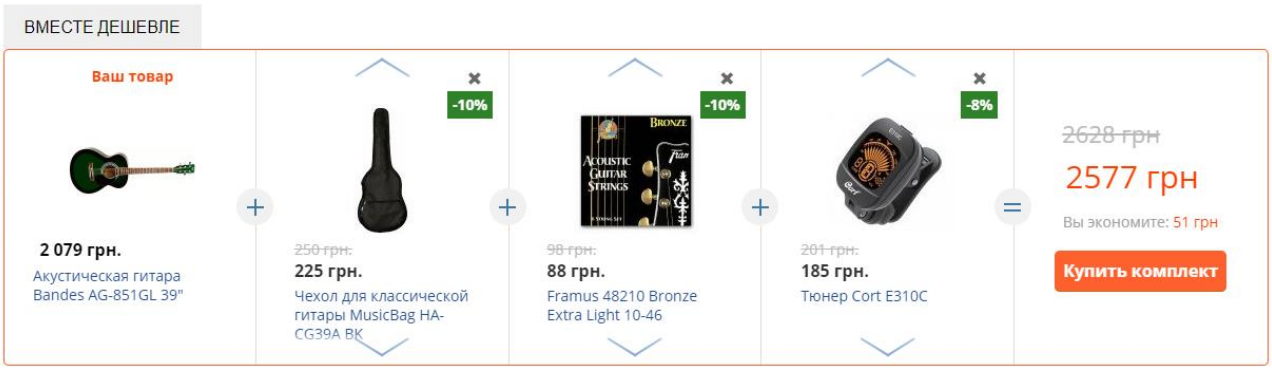

Рисунок 1.4 – Товар з ще трьома позиціями зі знижкою на сайті muzline.ua

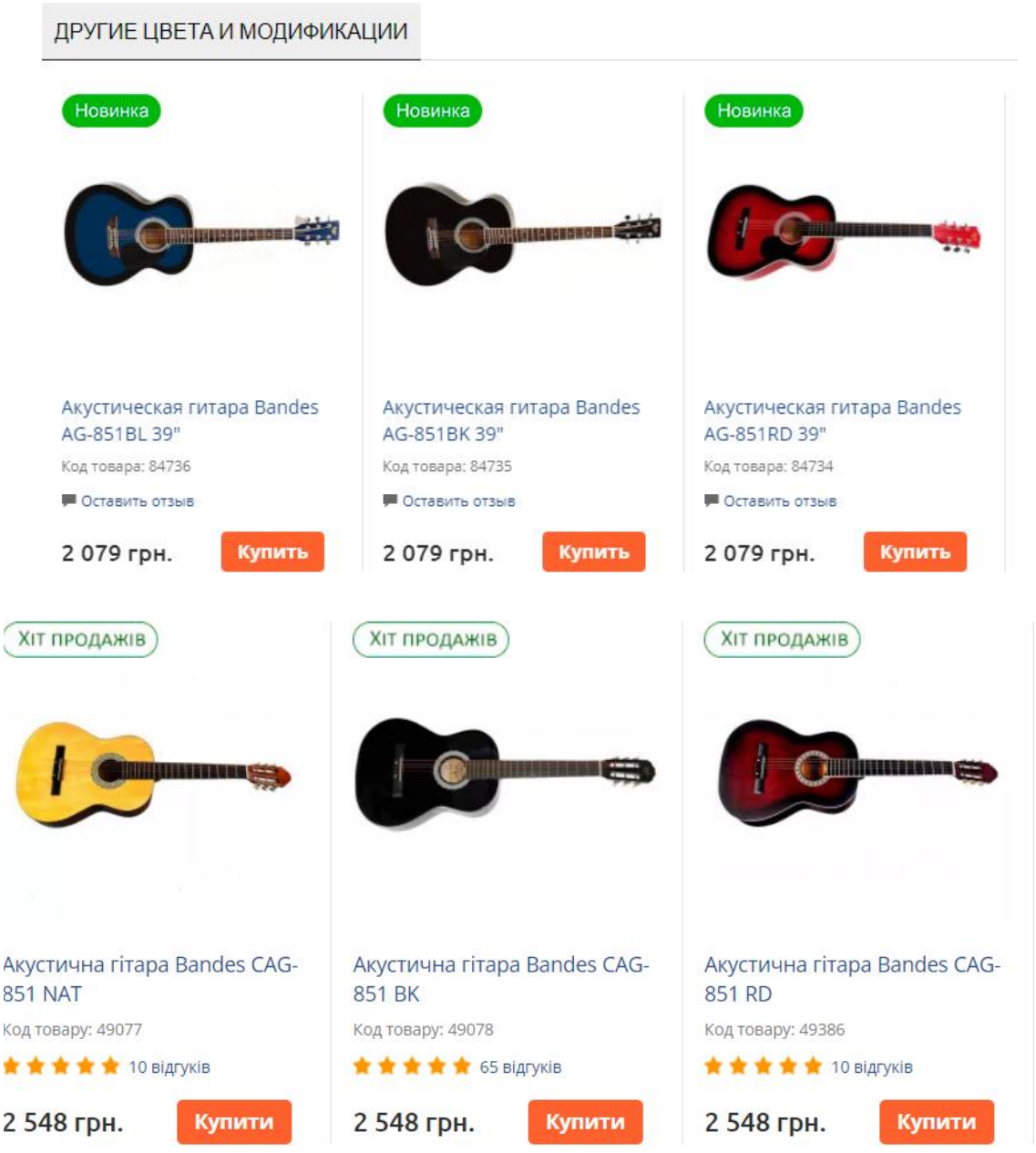

Рисунок 1.5 – Товар у різних кольорах на сайті muzline.ua

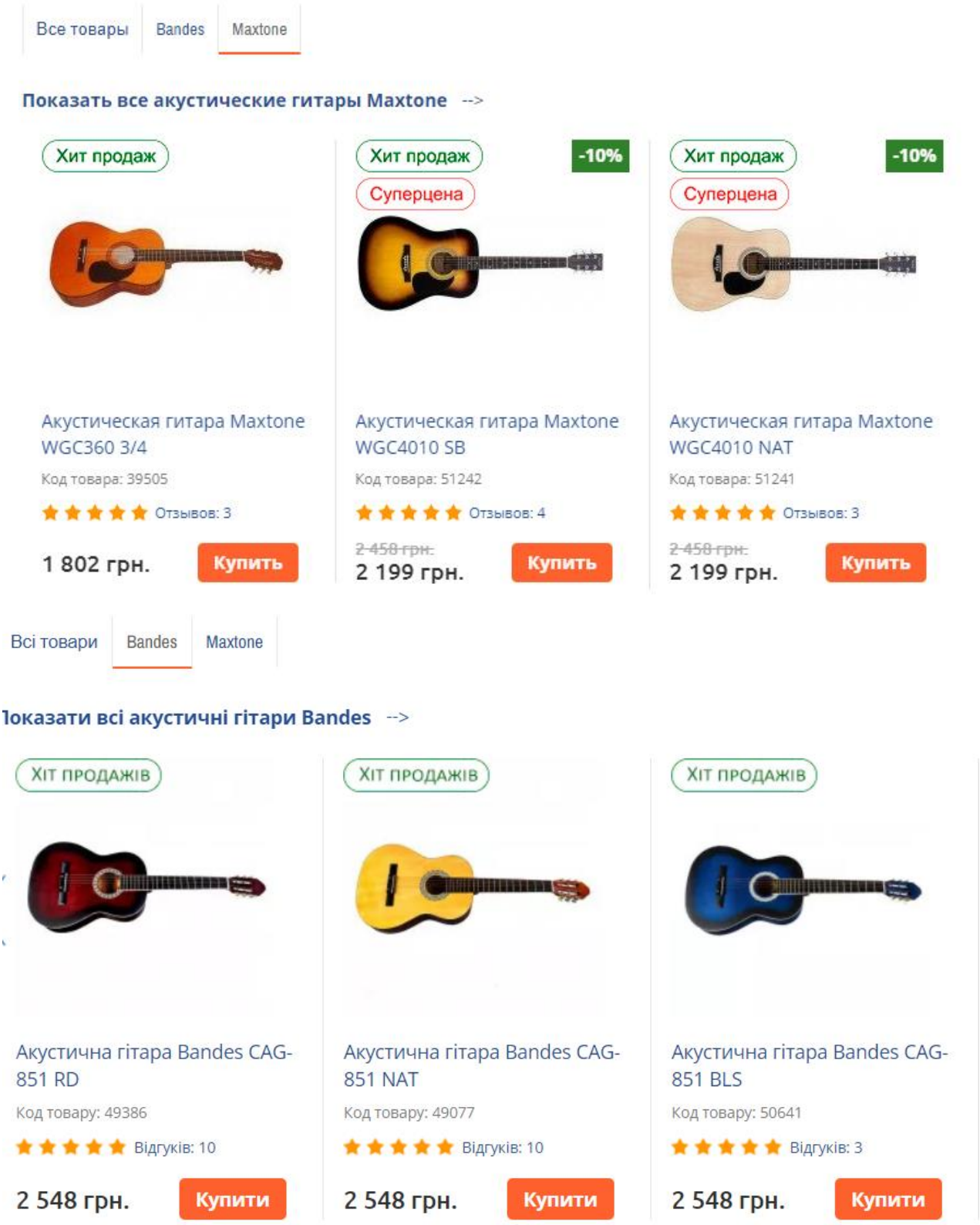

Рисунок 1.6 – Схожі за типом інструменти на сайті muzline.ua

– jam.ua. Цей магазин можна приводити у приклад як не треба робити. Він перевантажений деталями так, що іноді важко відрізнити рекламу від товару, який продається у цьому магазині (рис. 1.7).

Рисунок 1.7 – Перевантажений деталями сайт jam.ua

Далі було проведено аналіз магазину одягу issaplus.com. Сайт має дуже зручну навігацію про усім категоріям (рис. 1.8). Було обрано один товар (плаття), та до нього рекомендують інший (взуття), як доповнення образу (рис. 1.9). Також, є секції «Схожий товар» та «Разом придбають». Отже, можна зробити висновок, що цей інтернет-магазин зручний для користувачів.

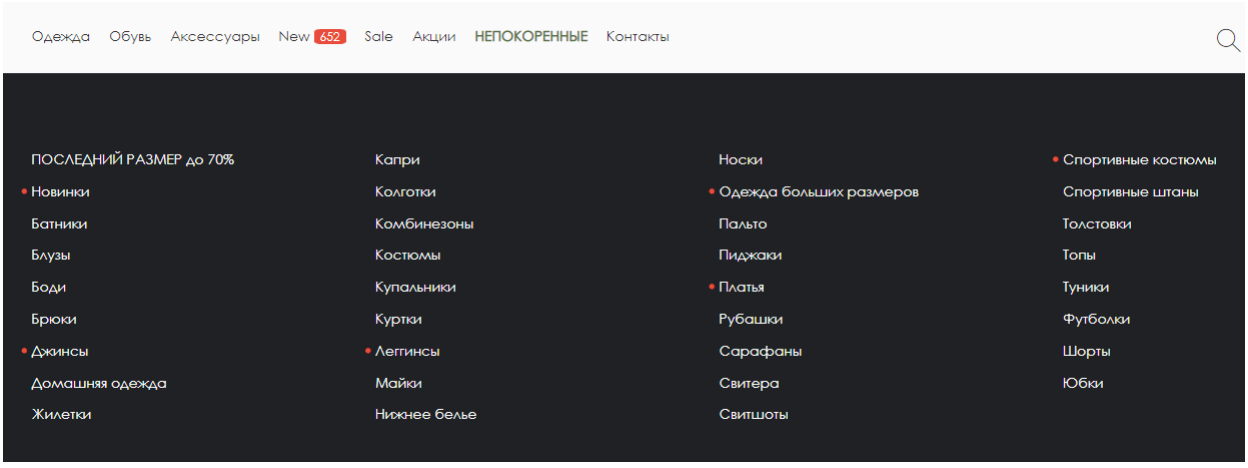

Рисунок 1.8 – Зручна навігація на сайті issaplus.com

Рисунок 1.9 – Рекомендації на сайті issaplus.com

Далі було проведено аналіз інтернет-магазину Aliexspress. Я зайшла на сторінку чохла для телефону та дивилася на роботу радної системи. У результаті була найдена помилка. Радна система запропонувала мені декілька схожих чохлів для телефону та машинку для стрижки волосся (рис. 1.10), якою я ніколи не цікавилася та не переглядала.

Рисунок 1.10 – Помилка в роботі радної системи на сайти aliexspress.ru

Аналізуючи ці інтернет-магазини можна зробити декілька висновків, якими функціями повинен володіти сайт .

Клієнт заходить на сайт з метою знайти товар який йому б підійшов і замовити його. Для цього сайт повинен бути інтуїтивно зрозумілим, простим та не перенавантаженим деталями, інакше – клієнт швидше за все відразу ж зачинить його і перейде на більш зручний йому. Тому перша функція – зробити сайт інтуїтивно зрозумілим і простим для використання.

Наступна функція – додавання пошуку товару по всім доступним критеріям. Було помічено при аналізі інтернет-магазинів, що вони всі мають такий пошук, тобто це друга за важливістю функція такого магазину.

Третє. Після покупки, клієнти зазвичай залишають інтернет-магазин, але ж йому можна запропонувати докупити щось ще до своєї покупки. Тому деякі магазини додали пропозиції до купівлі товарів разом з тим, що вони вибрали. Але магазини пропонують не випадкові товари, а тільки ті, які потрібні саме з їх купівлею або ті, які часто купують разом з нею.

Останнє. У таких магазинах немає продавця-консультанта, нікому запропонувати щось клієнту. Часто буває ситуація, коли клієнту начебто подобається товар але його все одно щось бентежить. Але щоб подивитися схожий товар йому доведеться довго шукати. Це знижує прибуток – клієнт може просто не знайти "свій" товар. Тому деякі інтернет-магазини додали функцію, яка "радить" схожий товар.

Останню проблему можна вирішити реалізувавши радну систему. Такі системи дозволяють користувачеві вибрати серед всіх доступних об'єктів саме ті, які будуть йому цікаві. Ці системи обробляють інформацію про різні об'єкти, а також про те, які користувачі які об'єкти купили, подивилися, послухали і т.д. Прикладами таких сервісів є last.fm, hunch.com, youtube.com та інші. Існує декілька методів для реалізації персоналізованих рекомендацій, які будуть розглянуті у цій роботі.

1.2 Постановка задачі

При докладному дослідженні рекомендуючих систем було виявлено деякі їх проблеми:

– холодний старт користувача. Завдання видати рекомендації новому користувачеві – це велика проблема, тому що ми нічого про нього не знаємо, він ще не робив жодних дій на сайті;

– холодний старт контенту. Ще одна проблема – новий товар. Якщо до нього не було ще достатнього інтересу користувачів, щоб зробити прогноз. Таке можливо, якщо об'єкт тільки з'явився, або може бути винна закільцьована проблема непопулярності: поки контент не популярний, його не рекомендують, а отже, йому нема звідки взяти популярність для рекомендацій;

– старіння контенту. Деякі товари можуть старіти з часом, наприклад, на їх місце можуть прийти нові моделі;

– врахування сезонності. Деякі товари можуть бути популярнішими у певну пору року. Це теж варто враховувати;

– швидкість видачі рекомендацій, навантаження на сервер. Щоб видати рекомендації, потрібно спочатку проаналізувати інформацію по всіх товарах та всім користувачам. Це колосальний обсяг даних, і він може займати багато часу.

Проблему холодного старту користувача можна вирішити декількома способами. При реєстрації на сайті можна попросити користувача ввести деякі свої дані, такі як місто, вік та попросити оцінити товари з різних категорій. Але мало який користувач з радістю буде це робити – людина хоче швидше продовжити переглядати товар, а може вже і купувати. Тому можна просто рекомендувати йому найпопулярніші товари, а далі вже дивитися, що його зацікавить.

У цій роботі потрібно дослідити розробку методів персоналізованих рекомендацій для електронної B2C комерції та розробити підхід для зменшення проблеми холодного старту.

2 ОГЛЯД МЕТОДІВ І ТЕХНОЛОГІЙ

Існує великий клас веб-додатків, які передбачають прогнозування реакції користувача на варіанти вибору. Такі засоби називаються рекомендаційними системами. Почнемо розділ з огляду найбільш важливих прикладів таких систем. Однак, для того, щоб зосередити увагу на проблемі, наведемо два хороших приклади рекомендаційних систем:

– Пропонування новинних статей читачам інтернет-газети на основі прогнозування читацьких інтересів;

– Пропонування клієнтам інтернет-магазину пропозицій щодо того, що вони можуть захотіти купити, виходячи з їхньої минулої історії покупок та/або пошуку товарів.

У додатку рекомендаційної системи є два класи сутностей, які будемо називати користувачами та об'єктами. Користувачі мають вподобання щодо певних об'єктів, і ці вподобання повинні бути витягнуті з даних. Самі дані представляються у вигляді матриці корисності, що дає для кожної пари користувач-об'єкт значення, яке представляє що відомо про ступінь переваги цього користувача для цього товару. Значення походять з впорядкованої множини, наприклад, цілі числа 1-5, що представляють кількість зірок які користувач поставив у якості оцінки для цього пункту. Ми припускаємо, що матриця є розріджена, тобто більшість елементів є «невідомими». Невідомий рейтинг означає що ми не маємо чіткої інформації про те, як користувач віддає перевагу тому чи іншому товару.

Наприклад, у таблиці 2.1 наведено приклад матриці корисності, яка представляє оцінки користувачів на фільми за шкалою від 1 до 5, де 5 є найвищою оцінкою. Пропуски представляють ситуацію, коли користувач не оцінив фільм. Назви фільмів наступні HP1, HP2 та HP3 – «Гаррі Поттер» 1, 2 та 3, TW - "Сутінки" та SW1, SW2 і SW3 - "Зоряні війни" для епізодів 1, 2 і 3. Користувачі представлені великими літерами від A до D.

Таблиця 2.1 – Матриця корисності з оцінками користувачів на фільми

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Можна звернути увагу на те, що більшість пар користувач-фільм мають порожні місця, тобто користувач не оцінив фільм. На практиці матриця буде ще більш розрідженою, з типовою користувацькою оцінкою лише крихітної частки всіх доступних фільмів.

Метою системи рекомендацій є прогнозування пропусків у матриці корисності. Наприклад, чи сподобається користувачеві A SW2? З крихітної матриці у таблиці 2.1 мало що видно. Ми могли б розробити систему рекомендацій з урахуванням властивостей фільмів, таких як їх продюсер, режисер, зірки або навіть схожість їх назв. Якщо це так, то ми можемо відзначити схожість між SW1 і SW2, а потім зробити висновок, що оскільки A не сподобався SW1, то навряд чи йому сподобається і SW2. З іншого боку, маючи набагато більше даних, ми могли б помітити, що люди, які оцінювали як SW1, так і SW2, мали тенденцію давати їм схожі оцінки. Таким чином, ми можемо зробити висновок, що A також дасть SW2 низьку оцінку, подібно до того, як A оцінює SW1.

Слід також пам'ятати про дещо іншу мету, яка має сенс у багатьох додатках. Не обов'язково передбачати кожну порожню клітинку в матриці корисності. Скоріше, необхідно лише виявити деякі записи в кожному рядку, які, ймовірно, можуть бути високими. У більшості додатків система рекомендацій не пропонує користувачам рейтинг всіх елементів, а скоріше пропонує кілька, які користувач повинен оцінити високо. Можливо, навіть не потрібно шукати всі позиції з найвищими очікуваними рейтингами, а лише знайти велику підмножину тих, що мають найвищі рейтинги.

Перш ніж обговорювати основні сфери застосування рекомендаційних систем, давайте поміркуємо над феноменом довгого хвоста, який робить рекомендаційні системи необхідними. Фізичні системи доставки

характеризуються дефіцитом ресурсів. Традиційні магазини мають обмежену площу полиць і можуть показати покупцеві лише невелику частину всіх існуючих товарів. З іншого боку, інтернет-магазини можуть зробити доступним для покупця все, що існує. Так, фізичний книжковий магазин може мати на своїх полицях кілька тисяч книг, але Amazon пропонує мільйони книг. Фізична газета може друкувати кілька десятків статей на день, в той час як он-лайн служби новин пропонують тисячі на день.

Рекомендація у фізичному світі досить проста. По-перше, не неможливо підігнати магазин під кожного окремого покупця. Таким чином, вибір того, що пропонується, регулюється лише сукупними показниками. Як правило, в книгарні виставлятимуть лише ті книги, які користуються найбільшою популярністю, а газета друкує лише ті статті, які, на її думку, зацікавлять найбільшу кількість людей. У першому випадку цифри продажів визначають вибір, у другому – редакційне судження.

Різниця між фізичним та он-лайн світом отримала назву "феномен довгого хвоста" і представлена на рис. 2.1.

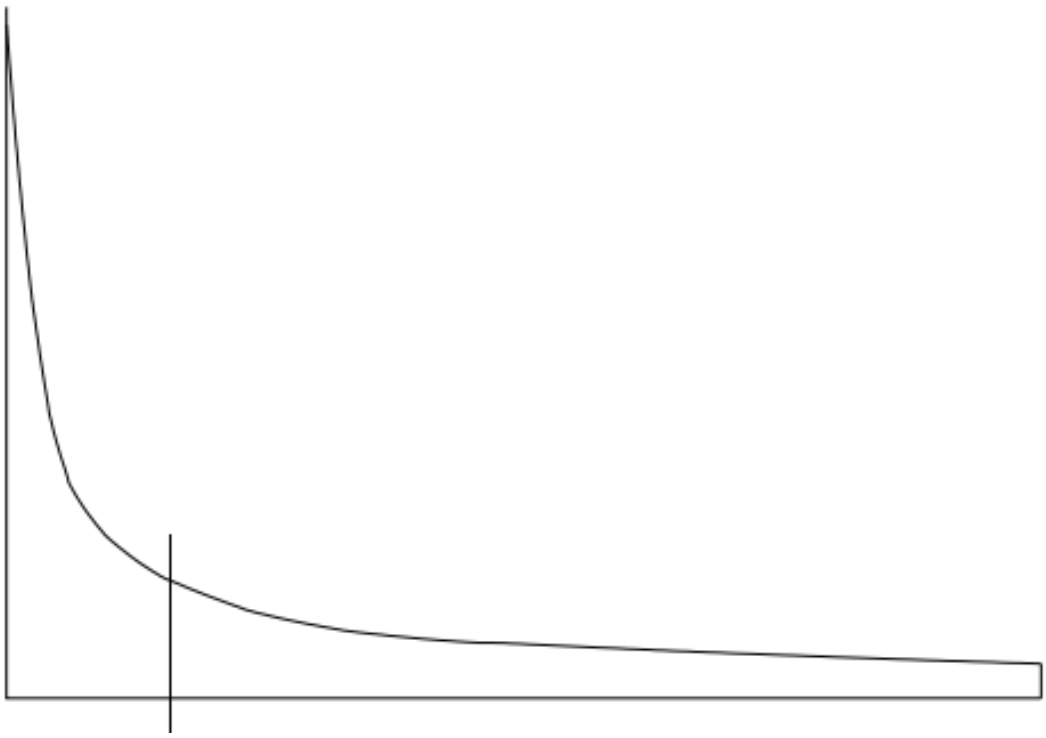

Рисунок 2.1 – Феномен довгого хвоста: фізичні заклади можуть надавати лише те, що користується популярністю, в той час як он-лайн заклади можуть зробити доступним все

По вертикальній осі відкладено популярність (кількість разів, коли обирається товар). На горизонтальній осі товари впорядковані відповідно до їх популярності. Фізичні заклади надають лише найпопулярніші позиції зліва від вертикальної лінії, тоді як відповідні он-лайн заклади надають весь асортимент позицій: хвіст а також популярні позиції.

Феномен «довгого хвоста» змушує онлайнові установи рекомендувати товари окремим користувачам. Неможливо представити користувачеві всі наявні предмети, так, як це можуть зробити фізичні установи. Ми також не можемо очікувати, що користувачі почують про кожну з позицій, які їм можуть сподобатися.

Вже було згадано кілька важливих застосувань рекомендаційних систем, але тут об'єднаний перелік в одному місці:

– Рекомендації до товарів: Мабуть, найбільш важливе використання систем рекомендацій – це використання в інтернет-магазинах. Ми відзначили, як Amazon або подібні онлайнові продавці намагаються представити кожному користувачеві, який повертається, деякі пропозиції щодо продуктів, які він міг би придбати. Ці пропозиції є не випадкові, а базуються на рішеннях про покупку, прийнятих подібними покупцями, або на інших методах, які будуть розглянуті далі;

– Рекомендації до фільмів: Netflix пропонує своїм користувачам рекомендації щодо фільмів, які можуть їм сподобатися. Ці рекомендації базуються на оцінках, наданих користувачами, подібно до оцінок, запропонованих у прикладі матриці корисності у таблиці 2.1. Важливість точного прогнозування рейтингів настільки висока, що компанія Netflix запропонувала приз в один мільйон доларів за перший алгоритм, який зможе перевершити її власну систему рекомендацій на 10%. Приз був нарешті виграний у 2009 році групою дослідників під назвою "Bellkor's Pragmatic Chaos", після більш ніж трьох років змагань;

– Новинні статті: Служби новин намагаються визначити статті, що становлять інтерес для читачів, на основі статей, які вони читали в минулому. Подібність може ґрунтуватися на схожості важливих слів у документах або на статтях, які читають люди зі схожими читацькими смаками. Ті ж принципи

застосовуються для рекомендації блогів з мільйонів доступних блогів, відео на YouTube або інших сайтів, де контент надається регулярно.

Яскравим прикладом того, як довгий хвіст разом з добре продуманою системою рекомендацій може впливати на події, є історія, розказана Крісом Андерсоном про книгу під назвою «Дотик до порожнечі» (Touching the Void). Ця книга про альпінізм свого часу не дуже добре продавалася, але через багато років після її виходу в світ вийшла ще одна книга на ту ж тему під назвою «У розрідженому повітрі» ("Into Thin Air"). Система рекомендацій Amazon помітила кількох людей, які купили обидві книги, і почала рекомендувати «Дотик до порожнечі» людям, які купили або розглядали можливість купити «У розрідженому повітрі». Якби не було книжкового інтернет-магазину, «Дотик до порожнечі», можливо, ніколи б не побачили потенційні покупці, але в онлайновому світі «Дотик до порожнечі» зрештою стала дуже популярною сама по собі, фактично, навіть більше, ніж «У розрідженому повітрі».

Без матриці корисності практично неможливо рекомендувати елементи. Однак отримання даних, на основі яких можна побудувати матрицю корисності, часто є складним завданням. Існує два загальних підходи до визначення оцінок, які користувачі надають товарам:

– Ми можемо попросити користувачів оцінити об’єкти. Зазвичай таким чином отримують рейтинги фільмів, а деякі інтернет-магазини таким чином намагаються отримати оцінки від своїх покупців. Сайти, що надають контент, такі як деякі новинні сайти або YouTube, також просять користувачів оцінювати матеріали. Цей підхід є обмеженим у своїй ефективності, оскільки, як правило, користувачі не бажають надавати відповіді, а інформація від тих хто це робить, може бути необ'єктивною через сам факт, що вона надходить від людей, які бажають надавати оцінки;

– Ми можемо робити висновки з поведінки користувачів. Очевидно, що якщо користувач купує товар на Amazon, дивиться фільм на YouTube або читає новину статтю, то можна сказати, що користувачеві «подобається» цей товар. Треба зауважити, що такого роду система оцінювання насправді має лише одне значення: 1 означає, що користувачеві подобається елемент. Часто ми зустрічаємо матрицю корисності з такими даними, в якій замість пропусків стоять 0, якщо користувач не купив або не переглянув товар. Однак у цьому

випадку 0 не є нижчою оцінкою, ніж 1; це не оцінка взагалі. У більш загальному випадку, можна зробити висновок про зацікавленість на основі поведінки, відмінної від купівлі. Наприклад, якщо клієнт Amazon переглядає інформацію про товар, ми можемо зробити висновок, що він зацікавлений в цьому товарі, навіть якщо вони не купують його.

Персоналізовані системи можуть надавати такі рекомендації, як:

– рекомендації на основі продуктів (наприклад, Amazon, Booking.com);

– рекомендації на основі вмісту (наприклад, Netflix, Spotify, TikTok, Instagram).

Базуючись на методах і функціях, які використовуються для прогнозування того, яким елементам віддадуть перевагу користувачі, існує три основні підходи до створення систем рекомендацій, а саме:

– фільтрація на основі вмісту (content-based filtering), яка генерує прогнози шляхом аналізу атрибутів елементів і пошуку подібності між ними;

– спільна фільтрація (collaborative filtering), яка генерує прогнози, аналізуючи поведінку користувачів і зіставляючи користувачів зі схожими смаками;

– гібридна фільтрація (hybrid filtering), яка об'єднує дві або більше моделей.

Модель фільтрації на основі вмісту надає рекомендації, використовуючи конкретні атрибути елементів шляхом пошуку схожості. Такі системи створюють профілі даних на основі описової інформації, яка може містити характеристики елементів або користувачів. Потім створені профілі використовуються для рекомендації предметів, подібних до тих, які користувач любив/купував/дивився/слухав у минулому.

Найпоширеніша модель, спільна фільтрація, надає відповідні рекомендації на основі взаємодії різних користувачів із цільовими елементами. Такі системи рекомендацій збирають інформацію про минулу поведінку користувачів, а потім викопують її, щоб вирішити, які елементи відображати іншим активним користувачам із подібними смаками. Це може бути що завгодно: від пісень, які користувачі слухали, або продуктів, які вони додали в кошик, до оголошень, які користувачі натискали, і фільмів, які вони раніше оцінювали, тощо. Ідея такої

системи полягає в спробі передбачити, як людина відреагує на предмети, з якими вона ще не взаємодіяла.

Гібридна фільтрація була створена, щоб усунути проблеми та обмеження чистих моделей систем рекомендацій. Гібридні моделі використовують кілька методів рекомендацій під одним дахом, щоб отримати вищу точність рекомендацій із меншою кількістю недоліків будь-якої окремої. Як правило, це спільна фільтрація, яка змішується з іншими методами, щоб подолати проблему холодного запуску. Але не виключно, оскільки підходи можуть поєднуватися різними способами.

2.1 Фільтрація на основі вмісту (Content-Based Filtering)

На основі попередніх відповідей, наданих користувачем, система вчиться давати рекомендації, аналізуючи схожість характеристик між елементами. Наприклад, на основі оцінки користувача для різних жанрів фільмів система навчиться рекомендувати жанр, який був позитивно оцінений користувачем. Система рекомендацій на основі вмісту створює профіль користувача на основі попередньо оцінених елементів користувача. Профіль користувача представляє інтереси користувача та також може адаптуватися до нових інтересів. Зіставлення профілю користувача з характеристиками об’єкта вмісту – є основою процесу рекомендації. Результатом цього процесу є судження, яке вказує на інтерес користувача до об'єкта. Точність профілю інтересів користувача призведе до підвищення корисності пропозицій. Наприклад, його можна використовувати для фільтрації веб-результатів, визначаючи, чи зацікавлений користувач у певній сторінці чи ні.

Процес рекомендації складається з 3 модулів, кожен з яких виконується окремо:

– Аналізатор вмісту: якщо дані неструктуровані, потрібна попередня обробка, щоб отримати релевантну інформацію. Основний обов’язок аналізатора вмісту полягає в представленні вмісту, що надходить із джерела, у відповідній формі для наступних кроків обробки. Для зміни структури елемента з оригінальної на цільову можуть використовуватися різні методи перетворення

(наприклад, веб-сторінки, представлені у вигляді векторів ключових слів). Цільове представлення може бути використано іншими компонентами;

– Навчальний компонент: цей модуль створює профіль користувача, узагальнюючи дані, отримані з попереднього компонента. Методи машинного навчання використовуються для вивчення стратегії узагальнення, яка дозволяє побудувати модель на основі попередніх уподобань користувача, як позитивних, так і негативних. Розробник системи повинен забезпечити релевантний зворотній зв'язок який об'єднує позитивні та негативні відгуки в прототипний вектор, що представляє профіль користувача;

– Компонент фільтрації: цей модуль використовує профіль користувача для отримання пов'язаних елементів зі списку можливих рекомендацій. Це робиться шляхом зіставлення профілю разом із елементами, які будуть рекомендовані. На основі показників подібності відповідне судження виробляється або бінарним, або безперервним.

Системи рекомендацій вмісту отримують ідею рекомендації з минулих даних користувача на основі того, які товари користувач придбав або які йому сподобалися. І атрибути користувача, і атрибути елемента однаково важливі з точки зору створення прогнозу. Розглянемо приклад рекомендувача новин, такі функції, як категорії (фінанси, спорт, здоров'я, технології, політика, розваги, автомобілі тощо) або місцезнаходження (місцеве, національне чи міжнародне) тощо, необхідні для визначення індексу подібності між новинами.

У системі, заснованій на змісті, ми повинні побудувати для кожного елемента профіль, який є запис або набір записів, що представляють важливі характеристики цього елемента. У простих випадках профіль складається з деяких характеристик об'єкта які легко виявляються. Наприклад, розглянемо характеристики фільму, які можуть мати відношення до системи рекомендацій:

– Набір акторів фільму. Деякі глядачі віддають перевагу фільмам зі своїми улюбленими акторами;

– Режисер. Деякі глядачі віддають перевагу творчості певних режисерів;

– Рік, в якому був знятий фільм. Деякі глядачі віддають перевагу старим фільмам, інші дивляться лише останні новинки;

– Жанр або загальний тип фільму. Деяким глядачам подобаються тільки комедії, іншим – драми або мелодрами.

Є багато інших особливостей фільмів, які також можна було б використати. За винятком останньої, жанрової, інформація легко доступна з описів фільмів. Жанр – поняття розпливчасте. Однак, в оглядах фільмів, як правило, визначають жанр з набору загальновживаних термінів. Наприклад, Internet Movie Database (IMDB) присвоює жанр або жанри кожному фільму.

Багато інших класів товарів також дозволяють отримати характеристики з наявних даних, навіть якщо ці дані в певний момент повинні бути введені вручну. Наприклад, товари часто мають описи, написані виробником, що містять характеристики, які стосуються цього класу товарів (наприклад, розмір екрану та колір корпусу телевізора). Книги мають описи, подібні до описів фільмів, тому ми можемо отримати такі характеристики, як автор, рік видання та жанр. Музичні продукти, такі як компакт-диски та завантаження в форматі MP3, мають такі характеристики, як виконавець, композитор і жанр.

Є й інші класи предметів, де не відразу видно, якими мають бути значення ознак. Розглянемо два з них: колекції документів та зображення. З документами виникають особливі проблеми.

Існує багато видів документів, для яких система рекомендацій може бути корисною. Наприклад, щодня публікується багато новин, і ми не можемо прочитати їх усі. Система рекомендацій може запропонувати статті на теми, які цікавлять користувача, але як розрізнити теми? Веб-сторінки – це також колекція документів. Чи можемо ми запропонувати сторінки, які користувач може захотіти переглянути? Аналогічно, блоги можна було б рекомендувати зацікавленим користувачам, якби ми могли класифікувати блоги за темами.

На жаль, ці класи документів, як правило, не мають легкодоступних функцій надання інформації. Заміною, яка виявилася корисною на практиці, є ідентифікація слів, що характеризують тему документа. По-перше, усунути стоп-слова – кілька сотень найпоширеніших слів, які, як правило, мало що говорять про тему документа. Для слів, що залишилися, обчислюємо показник TF.IDF для кожного слова в документі. Слова з найвищими показниками – це ті слова які характеризують документ.

Тоді ми можемо взяти за ознаки документа n слів з найвищими оцінками TF.IDF. Можна вибрати n однаковим для всіх документів, або нехай n буде фіксованим відсотком слів у документі. Ми також можемо зробити так, щоб усі

слова, чиї оцінки TF.IDF перевищують заданий поріг, були частиною набору ознак.

Зараз документи представлені наборами слів. Інтуїтивно ми очікуємо, що ці слова виражають тематику або основні ідеї документа. Наприклад, у новинній статті ми очікуємо, що слова з найвищою оцінкою TF.IDF включатимуть імена людей, про яких йдеться в статті, незвичайні властивості описуваної події, а також місце, де вона відбувається. Для вимірювання схожості двох документів, ми можемо використовувати кілька природних мір відстані:

– Ми могли б використати відстань Жаккара між наборами слів;

– Ми могли б використати косинусну відстань між множинами, які розглядаються як вектори.

Щоб обчислити косинусну відстань, уявіть собі набір слів з високим TF.IDF як вектор, з однією компонентою для кожного можливого слова. Вектор має значення 1, якщо слово є в наборі, і 0, якщо немає. Оскільки між двома документами існує лише скінченна кількість слів серед двох їхніх наборів, нескінченна розмірність векторів не має значення. Майже всі компоненти дорівнюють 0 в обох, і 0 не впливають на значення точкового добутку. Якщо бути точним, то точковий добуток – це розмір перетину двох наборів слів, а довжини векторів – це квадратні корені з кількості слів у кожному наборі. Цей розрахунок дозволяє обчислити косинус кута між векторами як добуток точкового добутку на добуток довжин векторів.

Розглянемо базу даних зображень як приклад того, як були отримані ознаки для елементів. Проблема з зображеннями полягає в тому, що їх дані, як правило масив пікселів, не говорять нам нічого корисного про їх особливості. Ми можемо обчислити прості властивості пікселів, такі як середня кількість червоного кольору на зображенні, але мало хто з користувачів шукає червоні зображення або особливо любить червоні зображення.

Можна отримати інформацію про особливості об'єктів, пропонуючи користувачам позначати об'єкти, вводячи слова або фрази, що описують об'єкт. Таким чином, одна фотографія з великою кількістю червоного кольору може бути позначена як «Площа Тяньаньмень», а інша – як «Захід сонця в Малібу». Ця відмінність не може бути виявлена за допомогою існуючих програм аналізу зображень.

Практично будь-який вид даних може мати свої особливості, що описуються тегами. Однією з перших спроб тегування великих масивів даних був сайт del.icio.us, пізніше куплений компанією Yahoo!, який пропонував користувачам тегувати веб-сторінки. Метою цього тегування було зробити доступним новий метод пошуку, коли користувачі вводили набір тегів як пошуковий запит, а система отримувала веб-сторінки, які були позначені таким чином. Однак, також можливе використання тегів в якості рекомендаційної системи. Якщо помічено, що користувач переглядає або додає до закладок багато сторінок з певним набором тегів, то можна рекомендувати інші сторінки з такими ж тегами.

Проблема з тегуванням як підходом до виявлення особливостей полягає в тому, що процес працює тільки в тому випадку, якщо користувачі готові взяти на себе клопоти по створенню тегів, і є достатньо тегів, щоб випадкові помилкові не спотворювали систему занадто сильно.

Цікавим напрямком заохочення до тегування є «ігровий» підхід започаткований Луїсом фон Аном. Він дозволив двом гравцям співпрацювати над тегом для зображення. У раундах вони пропонували тег, і обмінювалися тегами. Якщо вони погоджувалися, то «вигравали», а якщо ні, то грали ще один раунд з тим же зображенням, намагаючись одночасно узгодити тег. Хоча це інноваційний напрямок, який варто спробувати, є сумнівним, чи достатній інтерес громадськості, щоб створити достатню кількість вільної роботи щоб задовольнити потреби в маркованих даних.

У кінці треба сказати пару слів про переваги рекомендацій на основі вмісту:

– Незалежність користувача — система рекомендацій на основі вмісту створює профіль користувача лише на основі оцінки або покупки, зробленої користувачем у минулому. Жоден сусід не розглядається для створення профілю користувача, який має такі ж інтереси, як і користувач;

– Прозорість — функція пояснення системи рекомендацій на основі вмісту є прозорою для користувача, що означає, що вона надає пояснення до рекомендацій;

– Новий продукт — він не страждає від проблеми першого оцінювача, що означає, що якщо товар не оцінено жодним користувачем, він усе ще може рекомендуватися користувачеві.

Недоліки рекомендацій на основі вмісту:

– Обмежений аналіз вмісту. Одним із недоліків системи рекомендацій на основі вмісту є обмеженість вмісту, пов’язаного з елементом, щодо кількості ознак і типу ознак. Знання про предметну область також має вирішальне значення для надання рекомендації. Наприклад, створення системи рекомендацій фільмів вимагає знання акторів і режисерів фільму. Якщо доступних даних недостатньо, неможливо правильно розрізнити те, що подобається користувачам, і те, що не подобається. Представлення іноді здатне охопити лише певні аспекти вибору користувача, але не всі. Наприклад, веб-сторінки, техніки виділення властивостей із тексту повністю пропускають візуальні властивості та додаткову мультимедійну інформацію;

– Надмірна спеціалізація. Система рекомендацій на основі вмісту не має жодного спеціального методу дослідження чогось непередбачуваного. Система може рекомендувати лише ті елементи, які дають високу оцінку під час збігу з профілем користувача. Це також називається проблемою випадковості, яка показує обмеження рекомендацій, які можна зробити на основі вмісту. «Ідеальна» техніка, заснована на контенті, навряд чи буде надавати щось нове, обмежуючи діапазон застосувань, для яких це було б корисно;

– Новий користувач. Щоб створити систему рекомендацій та дізнатися про вподобання користувача, потрібно зібрати достатню кількість оцінок. Система не може надавати надійні рекомендації новим користувачам, оскільки минулі дані відсутні.

2.2 Спільна фільтрація (Collaborative Filtering)

Замість того, щоб використовувати характеристики товарів для визначення їх схожості, ми зосередимося на схожості користувацьких оцінок для двох товарів. Тобто, замість вектора «товар-профіль» для товару ми використовуємо його стовпчик у матриці корисності. Далі, замість того, щоб замість того, щоб вигадувати вектор профілю для користувачів, ми представляємо їх рядками в

матриці корисності. Користувачі є схожими, якщо їх вектори близькі за деякою мірою відстані, такою як відстань Жаккара або косинусна відстань. Рекомендація для користувача *U* потім робиться шляхом перегляду користувачів, які найбільш схожі на *U* в цьому сенсі, і рекомендує елементи, які подобаються цим користувачам. Процес визначення схожих користувачів і рекомендації того, що подобається схожим користувачам, називається спільною фільтрацією.

Цей підхід використовує «поведінку користувача» для рекомендацій. У цьому підході немає характеристики, яка відповідає користувачам або елементам. Він використовує матрицю корисностей і найчастіше використовується в промисловості, оскільки не залежить від будь-якої додаткової інформації.

Обмеження системи рекомендацій на основі вмісту можна подолати за допомогою спільного підходу, наприклад, він може зробити прогноз для тих елементів, для яких вміст недоступний. Він використовує відгуки інших користувачів, щоб рекомендувати такі товари. Ці системи оцінюють якість товару на основі експертної оцінки. Він також може пропонувати продукти з різним вмістом, якщо інші користувачі виявили інтерес до вмісту.

Перше питання, яке ми повинні вирішити, полягає в тому, як виміряти схожість користувачів або елементів з їх рядків або стовпців у матриці корисності. Ми відтворили таблицю 2.1 у вигляді таблиці 2.2.

Таблиця 2.2 – Матриця корисності з оцінками користувачів на фільми

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Ці дані занадто малі, щоб зробити якісь надійні висновки, але їх невеликий розмір зробить зрозумілими деякі підводні камені у виборі міри відстані. Потрібно звернути увагу на користувачів *A* і *C*. Вони оцінили два спільних фільми, але, схоже, вони мають майже діаметрально протилежні думки про ці

фільми. Можна було б очікувати, що хороша міра відстані зробить їх досить далекими один від одного. Ось кілька альтернативних мір для розгляду:

Ми могли б ігнорувати значення в матриці і зосередитись лише на наборах позицій, що отримали оцінку. Якби матриця корисності відображала лише покупки, цей показник був би хорошим вибором. Однак, коли комунальні послуги є більш детальними рейтингами, відстань Жаккара втрачає важливу інформацію.

Ми можемо розглядати пропуски як значення 0. Цей вибір є сумнівним, оскільки він має ефект відсутності оцінки як більш схожої на те, що фільм не сподобався, ніж сподобався.

Ми могли б спробувати усунути очевидну схожість між фільмами, які користувач оцінює високо і низько оціненими фільмами шляхом округлення оцінок. Наприклад, ми могли б вважати оцінки 3, 4 та 5 за «1», а оцінки 1 та 2 вважати не оціненими. Матриця корисності тоді виглядатиме як у таблиці 2.3.

Таблиця 2.3 – Матриця корисності, де оцінки 3, 4 та 5 замінено на 1, а оцінки 1 та 2 пропущено

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	1			1			
B	1	1	1				
C					1	1	
D		1					1

Якщо ми нормалізуємо рейтинги, віднімаючи від кожного рейтингу середній рейтинг цього користувача, ми перетворимо низькі оцінки на від'ємні числа, а високі – на позитивні числа. Якщо ми потім візьмемо косинусну відстань, то виявимо, що користувачі з протилежними поглядами на фільми, які вони переглядали разом, матимуть вектори майже в протилежних напрямках, і їх можна вважати максимально віддаленими один від одного. Однак користувачі зі схожими думками про фільми, які вони оцінювали спільно, матимуть відносно невеликий кут між ними.

Матрицю корисності можна розглядати як таку, що розповідає нам про користувачів або про предмети, або і про те, і про інше. Важливо розуміти, що

будь-який з методів, описаних вище для пошуку схожих користувачів, може бути використаний на стовпчиках матриці корисності для пошуку схожих елементів. На практиці симетрія порушується двома способами.

Ми можемо використовувати інформацію про користувачів для рекомендації товарів. Тобто, маючи користувача, ми можемо знайти деяку кількість найбільш схожих користувачів. Ми можемо базувати нашу рекомендацію на рішеннях, прийнятих цими схожими користувачами, наприклад, рекомендувати товари, які найбільша кількість з них придбала або високо оцінила. Однак, симетрія відсутня. Навіть якщо ми знаходимо пари схожих товарів, нам потрібно зробити додатковий крок для того, щоб рекомендувати товари користувачам.

Існує різниця в типовій поведінці користувачів і предметів, оскільки вона стосується схожості. Інтуїтивно, предмети, як правило, класифікуються в простих термінами. Наприклад, музика має тенденцію належати до одного жанру. Неможливо, наприклад, щоб музичний твір був одночасно і роком 60-х років, і бароко 1700-х років. З іншого боку, є люди, яким подобається і рок 60-х, і бароко 1700-х років, і які купують зразки обох типів музики. Наслідком цього є те, що легше виявити предмети, які схожі, тому що вони належать до одного жанру, ніж виявити, що два користувачі схожі, тому що вони віддають перевагу одному спільному жанру, в той час як кожному з них також подобаються деякі жанри, які іншому не подобаються.

Як було запропоновано вище, один із способів прогнозування значення елемента матриці корисності для користувача U та товару I полягає в тому, щоб знайти n користувачів (для деякого заздалегідь визначеного числа n), найбільш схожих на U, та усереднити їх оцінки для товару I, враховуючи лише тих серед n подібних користувачів, які оцінили I. Як правило, краще спочатку нормалізувати матрицю. Тобто для кожного з n користувачів відняти їх середню оцінку для пунктів з їхньої оцінки для i. Усереднити різницю для тих користувачів, які оцінили I, а потім додати це середнє значення до середньої оцінки, яку U дає для всіх пунктів. Ця нормалізація коригує оцінку у випадку, якщо U має тенденцію давати дуже високі або дуже низькі оцінки, або велика частка подібних користувачів, які оцінили I (яких може бути лише декілька), є користувачами, які мають тенденцію ставити дуже високі або дуже низькі оцінки.

Ми можемо використовувати схожість товарів для оцінки запису для користувача *U* та товару *I*. Знайдемо *m* товарів, найбільш схожих на *I*, для деякого *m*, і візьмемо середню оцінку серед *m* товарів з оцінок, які поставив *U*. Що стосується схожості між користувачами, ми розглядаємо лише ті товари з *m*, які оцінив *U*, і, ймовірно, доцільно спочатку нормалізувати рейтинги товарів.

Треба зауважити, що який би підхід до оцінки записів у матриці корисності ми не використовували, недостатньо знайти лише одну позицію. Для того, щоб рекомендувати товари користувачеві *U*, ми повинні оцінити кожну позицію в рядку матриці корисності для *U*, або, принаймні, знайти всі або більшість позицій в цьому рядку, які є порожніми, але мають високу оціночну вартість. Існує компроміс щодо того, чи повинні ми працювати з подібними користувачами або подібними товарами:

– Якщо ми знайдемо схожих користувачів, то нам потрібно буде виконати процес лише один раз для користувача *U*. З набору подібних користувачів ми можемо оцінити всі пропуски в матриці корисності для *U*. Якщо ми працюємо з подібними елементами, ми повинні обчислити подібні елементи майже для всіх елементів, перш ніж ми зможемо оцінити рядок для *U*;

– З іншого боку, схожість елементів часто надає більш достовірну інформацію через явище, про яке йшлося вище, а саме те, що легше знайти матеріали одного жанру, ніж знайти користувачів, яким подобаються тільки лише товари одного жанру.

Який би метод ми не обрали, ми повинні заздалегідь розрахувати бажані елементи для кожного користувача, а не чекати, поки нам потрібно буде приймати рішення. Оскільки матриця корисності змінюється повільно, то, як правило, достатньо обчислювати її нечасто і припускати, що вона залишається фіксованою між переобчисленнями.

Важко виявити схожість між об'єктами або користувачами, тому що ми маємо мало інформації про пари користувач-товар у розрідженій матриці корисності. Навіть якщо два елементи належать до одного жанру, існує ймовірність, що буде дуже мало користувачів, які купили або оцінили обидва. Аналогічно, навіть якщо двом користувачам подобається жанр або жанри, вони, можливо, не купили жодної спільного товару.

Одним із способів подолання цієї пастки є кластеризація об'єктів та/або користувачів. Можна обрати будь-яку з мір відстані, запропонованих вище, і використовувати її для виконання кластеризації, скажімо, об'єктів. Однак, ми побачимо, що може бути мало підстав для того, щоб намагатися одразу розбити дані на невелику кількість кластерів. Скоріше, ієрархічний підхід, коли ми залишаємо багато кластерів не об'єднаними, може бути достатнім як перший крок. Наприклад, ми можемо залишити вдвічі менше кластерів, ніж є товарів.

Наприклад, у таблиці 2.4 показано, що станеться з матрицею корисності таблиці 2.2, якщо нам вдасться об'єднати три фільми про Гаррі Поттера в один кластер, позначений HP, а також об'єднати три фільми про «Зоряні війни» в один кластер SW.

Таблиця 2.4 – Матриця корисності для користувачів та кластерів елементів

	HP	TW	SW
A	4	5	1
B	4,67		
C		2	4,5
D	3		3

Згрупувавши елементи до певної міри, ми можемо переглянути матрицю корисності таким чином, щоб стовпці представляли кластери елементів, а запис для користувача *U* та кластера *C* – це середня оцінка, яку користувач *U* дав тим членам кластера *C*, які він оцінив. Треба зауважити, що *U* може не оцінити жодного з членів кластера, і в цьому випадку запис для *U* та *C* залишається порожнім.

Ми можемо використовувати цю переглянуту матрицю корисності для кластеризації користувачів, знову ж таки, використовуючи міру відстані, яку ми вважаємо найбільш прийнятною. Потрібно використовувати алгоритм кластеризації, який знову залишає багато кластерів, наприклад, вдвічі менше кластерів, ніж є користувачів. Далі потрібно переглянути матрицю корисності так, щоб рядки відповідали кластерам користувачів, так само, як стовпці відповідали кластерам стовпці – кластерам товарів. Як і для кластерів елементів,

потрібно обчислити запис для кластеру користувачів шляхом усереднення оцінок користувачів у кластері.

Тепер цей процес можна повторити кілька разів, якщо ми захочемо. Тобто, ми можемо кластеризувати елементи кластерів і знову об'єднати стовпці матриці корисності, які належать до одного кластеру. Потім ми можемо знову звернутися до користувачів і кластеризувати кластери користувачів. Процес може повторюватися до тих пір, поки ми не отримаємо інтуїтивно обґрунтовану кількість кластерів кожного виду.

Після того, як ми згрупували користувачів та/або предмети до бажаного ступеня та обчислили матрицю корисності кластер-кластер, ми можемо оцінити записи у вихідній матриці корисності наступним чином. Припустимо, що ми хочемо спрогнозувати вхід для користувача *U* та товару *I,* тоді для цього треба:

– Знайти кластери, до яких належать *U* та *I*, скажімо, кластери *C* та *D* відповідно;

– Якщо запис у матриці корисності кластера для *C* та *D* є чимось відмінний від порожнього, використовувати це значення як оціночне значення для запису *U-I* у вихідній матриці корисності;

– Якщо запис для *C-D* не заповнений, то використовувати метод, описаний вище для оцінки цієї позиції, розглядаючи кластери, подібні до *C* або *D*. Використовувати отриману оцінку як оцінку для позиції *U-I*.

Далі перейдемо до проблем спільної фільтрації.

Що станеться, якщо в набір даних буде додано нового користувача або новий елемент? Існує термін, відомий як холодний старт, який буває двох типів:

– Холодний старт користувача. Коли новий споживач додається до бази знань стає важко запропонувати йому будь-який продукт, оскільки система не має попередніх даних користувача. Щоб вирішити цю проблему, рекомендуються найпопулярніші продукти в цілому чи регіоні;

– Холодний запуск продукту. Коли новий товар з’являється на ринку або передається системі, для визначення його вартості потрібні дії користувача. Чим вищі відгуки та оцінки продукт отримує від користувачів, тим простіше системі запропонувати відповідного споживача.

2.3 Порівняння рекомендацій на основі пам’яті (Memory-based) та на основі моделі (Model-based)

2.3.1 Рекомендації на основі пам’яті (Memory-based)

Переваги колаборативної фільтрації на основі пам’яті (Memory-based):

– Простота. Підходи на основі сусідства є інтуїтивно зрозумілими, а також простими у впровадженні. Лише один параметр потребує налаштування, а саме, яку кількість сусідів слід враховувати для остаточного оцінювання;

– Обґрунтованість. Підходи до спільної фільтрації на основі пам'яті також пропонують чітке пояснення розрахункових результатів. Методи на основі елементів можуть просто використовуватися для пояснення того, чому надається та чи інша рекомендація;

– Ефективність. З точки зору ефективності, система на основі сусідства є кращою. У порівнянні з іншими, системи на основі моделі користувача або об’єкту рекомендації не потребують дорогих етапів навчання, а також мають низьке споживання пам'яті;

– Стабільність. Системи спільної фільтрації на основі пам'яті дещо страждають від безперервного введення споживачів, продуктів та рейтингів. Наприклад, після обчислення подібності між товарами модель може генерувати пропозиції. Після додавання нового рейтингу потрібно обчислювати лише схожість між позиціями.

Підводячи підсумки можна сказати, що:

– ці системи повністю залежать від оцінок, які надають користувачами;

– ці системи не здатні обробляти розріджені дані, що призводить до погіршення продуктивності;

– рекомендація не може бути надана для нових клієнтів та нових продуктів;

– система не масштабується.

Далі ця система рекомендації поділяється на засновану на об’єктах (Item-based) та засновану на користувачах (User-based).

Для того, щоб зробити вибір між цими системами рекомендацій, необхідно врахувати п'ять пунктів. Ці пункти є наступними:

– Точність. Співвідношення користувачів та об'єктів, як правило, відповідає за точність системи рекомендацій щодо сусідства. У системі рекомендацій на основі користувачів, схожість між двома користувачами розраховується шляхом аналізу оцінок, виставлених користувачами за один і той самий об'єкт. Підходи на основі об'єктів обчислюють схожість між двома об'єктами шляхом аналізу оцінок, виставлених одним і тим же користувачем;

– Ефективність. Ефективність з точки зору пам'яті та обчислювальної потужності також залежить від співвідношення користувачів і предметів. Так, якщо користувачів більше, ніж об'єктів, що трапляється в більшості випадків, то системи рекомендацій на основі товарів є більш надійними з точки зору пам'яті та часу, необхідного для обчислення схожості. З іншого боку, часова складність однакова для обох оскільки залежить від кількості користувачів та кількості елементів;

– Стабільність. Стабільність системи, заснованої на користувачах та елементах, пов'язана з появою та зміною кількості користувачів та елементів у системі. Якщо об'єкти є статичними, то ми повинні використовувати систему рекомендацій на основі об'єктів, оскільки схожість ваги об'єктів можна обчислити через нерегулярні проміжки часу. З іншого боку, якщо список об'єктів змінюється, то системи, засновані на користувачах, є найбільш прийнятними;

– Обґрунтованість. Залежно від уподобань, чи потрібне обґрунтування, чи ні, обирається система рекомендацій, що базується на елементах або на користувачах. Методи на основі елементів можуть просто використовуватися для пояснення того, чому робиться рекомендація. В якості пояснення користувачеві може бути показаний набір характеристик з вказанням міри вагових коефіцієнтів подібності або список сусідніх об'єктів та їх вагові коефіцієнти подібності, які використовуються для надання рекомендації. Користувач також може брати участь у процесі, змінюючи сусідів. З іншого боку, користувацькі методи не можуть пояснити процес формування рекомендацій, оскільки користувач не знає про переваги інших;

– Випадковість. Проблема випадковості виникає в системі рекомендацій на основі товарів, оскільки вона рекомендує користувачам тільки ті товари, які користувачеві сподобалися в минулому. Наприклад, у системі рекомендацій фільмів користувачеві будуть рекомендовані лише ті фільми, жанр або актори яких збігаються з тими, що йому сподобалися раніше. І навпаки, система на

основі користувачів може дати несподівану рекомендацію, проаналізувавши сусідів, які поставили таку ж оцінку товару, як і користувач, а також перевіривши оцінки різних товарів, виставлені сусідніми користувачами, які ще не оцінені користувачем.

У методі User-User визначається схожість між користувачами і в якості рекомендацій користувачеві видається *n* товарів, які купляються найчастіше, *k* покупцями, які найбільше на нього схожі. Для оцінки ступеня схожості користувачів в плані їх переваг можуть використовуватися різні функції подібності (метрики). Найбільш популярними серед них є: евклідова відстань, косинусна міра, відстань Хеммінга, коефіцієнт кореляції Пірсона, коефіцієнт Танімото, Манхеттенська відстань і деякі інші. Визначення рекомендацій методом User-User передбачає побудову матриці активності користувачів, кожен рядок якої описує дії конкретного користувача стосовно до конкретного об'єкта (категорія, товар, послуга) на сайті. Дії користувачів можуть позначатися самими різними способами. Наприклад, це може бути бінарна інформація про відвідування або не відвідування заданого ресурсу даним користувачем, частота (або число) користувань ресурсу *r* користувачем *u*, вартість або рейтинг, проставлений користувачем *u* для ресурсу *r* і т.д. Таким чином, кожен рядок матриці активності являє собою вектор оцінок, які відповідають різним категоріям товарів (тематичний профіль користувача). Профіль користувача характеризує ступінь його інтересу до кожної групи товарів. Для кожної пари «Користувач-об'єкт (товар, послуга, дія)» в матриці активності обчислюється міра близькості з використанням обраної метрики.

Метод Item-Item історично з'явився як альтернатива методу User-User, покликана підвищити продуктивність рекомендаційних систем для тих магазинів, де число покупців істотно перевищує кількість найменувань товарів в каталозі. Спочатку даний метод був запропонований компанією Amazon для вирішення наступних основних проблем підходу User-User: проблема холодного старту і проблема частого поновлення даних про активність користувача. Проблема холодного старту істотно знижує якість роботи рекомендаційної системи внаслідок відсутності даних про переваги нових (або мало активних) користувачів. Проблема частого поновлення даних про активність користувача

(в разі компанії Amazon йдеться про мільйони покупців) різко знижує продуктивність рекомендаційної системи в цілому.

Основна ідея методу Item-Item полягає в угрупуванні інформаційних одиниць (товари, послуги, дії) які мають подібні оцінки користувачів (рейтинги). Рекомендації виробляються за наступним принципом: користувач який високо оцінить об'єкт *X* буде запропонований об'єкт *Y*, який високо оцінили інші користувачі, які також високо оцінили і об'єкт *X*. Використання методу Item-Item дозволяє підвищити якість рекомендацій для нових користувачів (немає критичної залежності від даних про переваги користувача), а також значно підвищує продуктивність рекомендаційної системи в разі, коли кількість користувачів перевищує кількість об'єктів (характеристики об'єктів змінюються рідше). При цьому якість рекомендацій в середньому вище, ніж в разі використання підходу, заснованого на аналізі профілів. Для обчислення попарної близькості інформаційних одиниць можуть використовуватися ті ж метрики, що і у випадку з парами «користувач-об’єкт» (часто використовується косинусна або модифікована косинусна міри). Для пошуку рекомендацій на підставі матриці об'єктів часто використовуються вагові функції та методи регресійного аналізу. Одним з перспективних методів вирішення задачі Item-Item є метод Item2Vec. Проте для більшості інтернет-магазинів підхід, пов'язаний з рекомендаціями щодо рейтингів, не дуже ефективний в силу відсутності можливості мотивувати користувачів визначати рейтинг інформаційних одиниць (покупці приходять з пошукових систем і товарних каталогів, роблять потрібну їм покупку і йдуть, щоб більше ніколи не повернутися). І постає завдання, як в таких умовах зробити рекомендацію (інформаційна пропозиція), на яку відгукнеться користувач.

2.3.2 Рекомендації на основі моделі (Model-based)

Підхід на основі моделі (model based). Ця модель представляє користувачів та предмети за допомогою матриці корисності, тобто матриця корисності містить матриці A та B, де A позначає користувача, а B позначає предмети. Для декомпозиції матриці використовуються різні методи. Розраховується рейтинг

кожного товару та рекомендується товар з найвищим рейтингом. Ця модель є корисна, коли наявні дані є великими за обсягом.

Цей підхід далі поділяється на 3 підтипи:

– алгоритм на основі кластеризації;

– алгоритм на основі матричної факторизації;

– глибинне навчання, нейронні мережі.

Переваги колаборативної фільтрації на основі моделей (Model-Based):

– це допомагає вирішити проблему розрідженості та масштабованості;

– точність прогнозування краще.

Підводячи підсумки можна сказати, що:

– вартість впровадження є високою;

– існує баланс між масштабованістю та ефективністю моделі;

– за рахунок методів зменшення розмірності може бути втрачена цінна інформація.

2.4 Гібридна система рекомендацій

При цій системі поєднуються дві або більше методик, наприклад, контент-орієнтована та спільна фільтрація. Обмеження одного методу можуть бути подолані за допомогою іншого методу. Існує кілька класів гібридних рекомендаційних систем:

– Змішані – у цьому випадку декілька різних методів об'єднуються разом для розробки системи. Тут згенеровані списки елементів кожного методу додаються для отримання остаточного списку рекомендованих елементів. Рекомендації, надані всіма методами, об'єднуються, і кінцевий результат представляється користувачеві;

– Зважені – у цьому випадку зважені лінійні функції використовуються для обчислення рангу продукції шляхом агрегування вихідних даних рангів усіх систем. P-Tango є першою зваженою гібридною системою для рекомендаційної системи інтернет-газет;

– Каскадні – ці системи працюють поетапно. Перший метод використовується для складання грубого рейтингу продуктів, а потім

сформований список уточнюється другим методом. У цих системах велике значення має порядок процесів;

– Розширення функцій – у системах розширення функцій, вихід однієї системи діє як вхід для іншої системи. Вони також чутливі до порядку;

– Перемикання – при цьому система перемикається між різними рекомендованими методами в залежності від деяких умов. Наприклад, метод CF-CBF може переключитися на рекомендацію на основі контенту, якщо метод спільної фільтрації не дає достатньо надійних результатів.

Переваги гібридних рекомендаційних систем:

– подолання обмежень спільної фільтрації, контент-орієнтованих та інших систем;

– покращують результати рекомендацій;

– можуть працювати з розрідженими даними.

Підводячи підсумки можна сказати, що:

– витрати збільшено;

– підвищено рівень складності;

– потрібна зовнішня інформація, яка не завжди доступна.

2.5 Демографічна фільтрація

Демографічний підхід базується на припущенні, що різні демографічні групи мають різні смаки в товарах. Тому система рекомендує товари користувачам на основі їхніх демографічних профілів, таких як вік, стать, мова та місцезнаходження.

Багато сучасних рекомендаційних систем використовують гібридизацію, яка поєднує два або більше методів рекомендацій для отримання кращої продуктивності, ніж коли системи реалізуються окремо. Демографічна фільтрація здебільшого використовується в гібридних системах разом з іншими типами рекомендаційних методів з метою підвищення точності прогнозування.

Сильною стороною методу демографічної фільтрації є те, що проблема нового користувача не стосується цього типу рекомендаційних систем, оскільки їм не потрібен список оцінок від нового користувача для надання рекомендацій. Однак, згідно з попередніми дослідженнями, основна проблема з

демографічними системами полягає в тому, що демографічні дані в поєднанні з рейтингами товарів важко отримати.

Демографічна система може бути з контрольованою класифікацією та з неконтрольованою.

Контрольована класифікація означає, що мітки для точок даних, з яких складається навчальна вибірка, відомі заздалегідь. Алгоритми класифікації для керованого навчання включають, серед іншого, метод найближчого сусіда, дерева рішень та байєсівські класифікатори.

Неконтрольована класифікація означає, що мітки для кожної точки даних, що використовується для навчання, невідомі до того, як модель була побудована. Таким чином, завдання полягає в тому, щоб організувати і згрупувати ці точки даних за їхніми ознаками в осмислений спосіб, створюючи кластери.

2.6 Інші рекомендаційні системи

Фільтрація на основі знань. Ця система використовує знання про користувачів та їх вимоги/вподобання для формування пропозицій. Системи на основі обмежень належать до систем, заснованих на знаннях, які рекомендують продукти, що рідко купуються, такі як автомобіль, будинок тощо.

2.7 Виклик Netflix (The Netflix Challenge)

Значний поштовх дослідженням рекомендаційних систем було надано, коли компанія Netflix запропонувала приз у розмірі 1 000 000 доларів США першій людині або команді, яка перевершить власний алгоритм рекомендацій під назвою CineMatch на 10%. Після більш ніж трьох років роботи, приз був вручений у вересні 2009 року.

Виклик Netflix складався з опублікованого набору даних, що давав рейтинги від приблизно півмільйона користувачів на (як правило, невеликі підмножини) приблизно 17,000 фільмів. Ці дані були відібрані з більшого набору даних, і запропоновані алгоритми були протестовані на їхню здатність передбачати рейтинги в секретному залишку більшого набору даних. Інформація

для кожної пари (користувач, фільм) в опублікованому наборі даних включала рейтинг (1-5 зірок) і дату, на яку рейтинг було виставлено.

Для вимірювання продуктивності алгоритмів використовували RMSE (середньоквадратичне відхилення). CineMatch має RMSE приблизно 0,95; тобто, типова оцінка буде відставати майже на одну зірку. Щоб виграти приз, необхідно було, щоб алгоритм мав RMSE не більше 90% від RMSE CineMatch.

Можна згадати деякі цікаві алгоритми-переможці і, можливо неінтуїтивно зрозумілі факти про задачу:

а) CineMatch був не дуже хорошим алгоритмом. Насправді, рано було виявлено що очевидний алгоритм передбачення, для оцінки користувачем *u* на фільм *m*, середнє значення:

1) середня оцінка, яку *u* поставив усім рейтинговим фільмам та
2) середня оцінка фільму *m* від усіх користувачів, які оцінили фільм

виявилося лише на 3% гіршим за CineMatch.

б) Алгоритм УФ-розкладання, був знайдений трьома студентами (Майклом Харрісом, Джеффрі Вангом та Девідом Камом), що він дає 7% покращення порівняно з CineMatch, у поєднанні з нормалізацією та деякими іншими трюками;

в) Робота-переможець фактично являла собою комбінацію декількох різних алгоритмів, які були розроблені незалежно один від одного. Друга команда, яка подала роботу, перемогла б, якби була подана на кілька хвилин раніше, також являла собою комбінацію незалежних алгоритмів. Ця стратегія – комбінування різних алгоритмів – використовувалася раніше в ряді складних задач, і про неї варто пам'ятати;

г) Було зроблено кілька спроб використати дані, що містяться в Інтернет-базі даних про фільми IMDB, щоб зіставити назви фільмів з виклику Netflix з їхніми назвами в IMDB, і таким чином витягти корисну інформацію, яка не міститься в самих даних Netflix. IMDB містить інформацію про акторів та режисерів, а також класифікує фільми за одним або кількома з 28 жанрів. Було виявлено, що жанрова та інша інформація не є корисною. Одна з можливих причин полягає в тому, що алгоритми машинного навчання і так змогли виявити відповідну інформацію, а друга – в тому, що сутнісне вирішення проблеми

зіставлення назв фільмів, наведених у даних Netflix та IMDB не так вже й легко вирішити на практиці;

д) Час рейтингування виявився корисним. Виявляється, є фільми, які з більшою ймовірністю оцінять люди, які ставлять оцінку одразу після перегляду, ніж ті, хто чекає деякий час, а потім оцінює його. «Патч Адамс» був наведений як приклад такого фільму. І навпаки, є й інші фільми, які не сподобалися тим, хто оцінював його одразу, але були й ті, хто краще оцінили його через деякий час; як приклад наводився фільм «Мементо». Хоча з даних не можна виокремити інформацію про те, скільки часу пройшло між переглядом і оцінкою, загалом можна припустити, що більшість людей дивляться фільм невдовзі після його виходу на екрани. Таким чином, можна дослідити рейтинги будь-якого фільму, щоб побачити, чи мають його рейтинги висхідний або низхідний нахил з часом.

3 РОЗРОБКА СИСТЕМИ ДЛЯ ПОДОЛАННЯ ПРОБЛЕМИ ХОЛОДНОГО СТАРТУ

3.1 Обґрунтування вибору підходу для подолання проблеми

У цій роботі буде вирішуватися проблема холодного старту користувача.

Найпопулярніший спосіб подолання цієї проблеми – рекомендування найпопулярніших продуктів в цілому чи регіоні. Це гарний підхід але його можна покращити за допомогою демографічної фільтрації.

Статистично, новий користувач найвірогідніше зайде в інтернет-магазин на сторінку конкретного товару, а не на головну сторінку. Це можна використовувати для подолання проблеми холодного старту. Можна відслідковувати переглянуті товари користувачем, та рекомендувати йому схожі. Систему рекомендацій можна зробити кількома способами але було обрано демографічну фільтрацію.

Можна групувати користувачів за демографічними ознаками – вік, стать, освіта, регіон, мова, професія, сімейний стан, наявність дітей тощо. А далі рекомендувати їм те, що найбільше користується популярністю у цій групі користувачів.

Демографічна система може бути з контрольованою класифікацією та з неконтрольованою. Було обрано систему з контрольованою класифікацією – демографічні ознаки для навчальної вибірки будуть братися одразу з бази даних. Тобто до товарів прив'язуватимуться категорії користувачів, які найчастіше купують цей товар. Далі інформація про товари може уточнюватися менеджерами магазину у самій базі даних.

З такою системою є шанс помилитися з рекомендаціями – наприклад, користувач може шукати товар на подарунок і може більше не цікавитися ним після придбання. Але у кожної системи є свої плюси і мінуси. Далі можна це покращити.

3.2 Розробка системи на основі демографічної фільтрації

Для початку потрібно додати тестові дані до бази даних.

В якості демографічних ознак було обрано:

– Стать – чоловіча, жіноча та невизначена, що позначає, що цей товар купують однаково чоловіки та жінки (далі те ж самий принцип);

– Вік – підлітки, дорослі та невизначений вік;

– Професія – може бути багато, але для цієї тестової таблиці були обрані художник, будівельник, перукар та невизначена професія;

– Працевлаштування – працевлаштований та невизначений тип;

– Наявність дітей – так, ні та невизначено.

Приклад таблиці продуктів наведений на рисунку 3.1.

id_product	name	price	quantity	picture	gender	age	profession	employment	having_children
1	Gaming laptop	30000.00	345	...	male	teens	unknown	employed	unknown
2	Fishing rod	2000.00	356	...	male	adults	unknown	unknown	yes
3	Blender	500.00	45	https://yoer.pl/images/yoer/4000-5000/Blen...	female	adults	unknown	unknown	yes
4	Tent	3500.00	23	...	unknown	unknown	unknown	unknown	no
5	Paint set	450.00	82	https://m.media-amazon.com/images/I/819a...	unknown	unknown	artist	employed	no
6	Screwdriver	150.00	340	...	male	adults	builder	employed	unknown
7	Hammer	200.00	576	...	male	adults	builder	employed	unknown
8	Hairdryer	600.00	38	...	female	unknown	unknown	unknown	unknown
9	Hairdressing scissors	700.00	400	...	female	adults	hairdresser	employed	unknown
10	Makeup kit	2000.00	65	...	female	adults	unknown	unknown	unknown

Рисунок 3.1 – Тестові дані для таблиці продуктів

Далі потрібно продумати як буде працювати алгоритм фільтрації. Система повинна надавати рекомендації вже на першому товарі. Тобто потрібно збирати інформацію про вподобання користувача одразу же як він вперше увійде до системи (найчастіше одразу на сторінку перегляду товару).

Реалізацію цього масиву з переглянутими об’єктами можна зробити декількома методами:

– Найпростіший – зберігати цю інформацію у системі поки клієнт не закриє браузер (сесію);

– Середній по складності – зберігати інформацію на пристрої користувача використовуючи кеши;

– Більш складний – записувати інформацію до бази даних, наприклад, до таблиці «Неавторизований користувач» поки він не зареєструється (і потім теж враховувати його вподобання, продовжувати слідкувати за ним). Але для цього

методу потрібно буде подбати про оптимізацію системи, бо вона буде працювати з великою кількісті даних.

У роботі буде використано перший метод.

Далі треба якось збирати інформацію про демографічні характеристики товарів, які користувач переглядав. Тут також можна використати три різних метода як для реалізації масиву з переглянутими товарами. Будемо зберігати цю інформацію поки тільки у системі.

Цей об’єкт на старті системи буде виглядати як представлено у таблиці 3.1.

Таблиця 3.1 – Початковий об’єкт з демографічними характеристиками

Стать	Чоловіча	0
	Жіноча	0
Вік	Підлітки	0
	Дорослі	0
Професія	Художник	0
	Будівельник	0
	Перукар	0
Працевлаштування	Працевлаштований	0
Наявність дітей	Так	0
	Ні	0

Коли користувач буде активно переглядати товар, цей об’єкт повинен заповнюватися цифрами. На цій основі система буде рекомендувати йому такі товари, які користуються популярністю у людей зі схожими демографічними ознаками.

Система повинна враховувати ті характеристики, у яких цифра в об’єкті з ознаками максимальна. Тут також можна зробити по-різному – можна враховувати саму значущу характеристику – тільки максимальне число, а можна враховувати його і ще декілька близьких до нього.

Наприклад, маємо об’єкт з демографічними характеристиками як представлено у таблиці 3.2.

Таблиця 3.2 – Приклад об’єкта з демографічними характеристиками

Стать	Чоловіча	5
	Жіноча	14
Вік	Підлітки	10
	Дорослі	2
Професія	Художник	0
	Будівельник	4
	Перукар	0
Працевлаштування	Працевлаштований	5
Наявність дітей	Так	6
	Ні	12

Бачимо, що максимальне число 14 є у характеристики стать – жіноча. Але ще бачимо, що є ще деякі значення, наближені до максимального. Це відсутність дітей, та вік – підліток.

Тобто можна будувати систему двома методами:

– Враховувати лише максимальне число з демографічних ознак;

– Враховувати максимальне та ще декілька чисел найближчого до максимального.

У роботі було обрано перший метод – врахування лише одного максимального числа.

Далі після того, як були визначені переважаючі демографічні характеристики, потрібно вирішити як шукати по ним товари для рекомендацій.

Рекомендації користувачу будуть розраховуватися за формулою 3.1. Множина з товарами буде фільтруватися і братимуться лише ті товари у яких є демографічна характеристика y (переважаюча демографічна характеристика, визначена раніше), за якою фільтрується.

$$R_y = \{x_j, y \in C_{x_j}\}, \tag{3.1}$$

де R_y – множина відібраних товарів для рекомендації з y демографічною характеристикою, X – множина товарів, y – переважаюча демографічна

характеристика користувача, j – елемент множини товарів X, C_{x_j} – множина демографічних характеристик заданих для товару x_j.

Якщо у користувача буде декілька явних демографічних ознак, то розрахунки будуть робитися на основі формули 3.2. За цією формулою будуть фільтруватися такі товари, які будуть мати всі шукані ознаки в собі разом. Тобто будуть братися товари, у яких множина демографічних характеристик буде такою ж як і множина ознак, за якою фільтрується.

$$R_Y = \left\{x_j, Y = C_{x_j}\right\}, \tag{3.2}$$

де R_Y – множина відібраних товарів для рекомендації з Y підмножиною демографічних характеристик, X – множина товарів, Y – підмножина переважаючих демографічних характеристик користувача, j – елемент множини товарів X, C_{x_j} – множина демографічних характеристик заданих для товару x_j.

За попередньою формулою товарів може знайтися мало. Якщо це відбудеться, можна буде використовувати формулу 3.3, за якою буде проводитися пошук товару за окремими характеристиками окремо. Тобто будуть шукатися такі товари, для яких множина Y переважаючих характеристик буде підмножиною для множини C демографічних ознак окремого товару. Та при умові, що перетин підмножини Y та множини C не буде пустою множиною, тобто між множинами Y та C буде щонайменше одна спільна демографічна характеристика.

$$R_Y = \{x_j, Y \subseteq C_{x_j}, Y \cap C_{x_j} \neq \emptyset\}, \tag{3.3}$$

де R_Y – множина відібраних товарів для рекомендації з Y підмножиною демографічних характеристик, X – множина товарів, Y – підмножина переважаючих демографічних характеристик користувача, j – елемент множини товарів X, C_{x_j} – множина демографічних характеристик заданих для товару x_j.

3.3 Ідеї для подальшого покращення системи

Кожна система не є ідеальною, тож і цю можна покращити.

Наприклад, можна збирати дані про регіон користувача через його API адресу та рекомендувати йому релевантний товар у його регіоні. Але людина може користуватися VPN та це зменшує вірогідність коректного визначення регіону. В цьому випадку можна давати змогу користувачу самому задавати регіон, в якому він знаходиться.

Можна використовувати декілька персоналізованих систем. Наприклад, додати до системи ще одну, яка б рекомендувала товар, схожий на той, що переглядається в даний час.

Ще до існуючої системи можна додати ваги товарам – враховувати скільки разів користувач переглядав товар. Якщо багато – рекомендувати йому схожі. Але це в свою чергу може привести до циклічності рекомендацій. Користувачу рекомендуватиметься один і той самий тип товару по колу.

4 ОПИС ПРИЙНЯТИХ ПРОЕКТНИХ РІШЕНЬ ПРИ РОЗРОБЦІ СИСТЕМИ

4.1 Обґрунтування вибору мови програмування

В якості мови програмування для реалізації основних елементів проекту була обрана мова java script (JS).

Переваги JS:

– затребуваність. На даний час це дуже затребувана мова програмування, програмуючи на цій мові можна достатньо легко знайти роботу;

– дружелюбність. Цю мову програмування підтримують усі популярні браузери і тільки вона може використовуватися для створення інтерактивних веб-сторінок;

– простота. JS простий для вивчення. Також можна почати писати на цій мові прямо з браузера, не установлюючи середу розробки;

– універсальність. Ця мова використовується для створення практично чого завгодно;

– перспективність. Останні роки показують, що і користувачі, і, відповідно, розробники все більше концентруються на веб-проектах і сервісах. При цьому і додатки для смартфонів, планшетів все частіше і легше реалізуються на JS. Рейтинги популярності мов програмувань доводять перспективність мови, показуючи, що JS займає перше місце випереджаючи друге з великим відривом.

4.2 Обґрунтування вибору платформи СУБД

Для розробки бази даних було використано програмний засіб MySQL Workbench.

MySQL Workbench — інструмент для візуального проектування баз даних, що інтегрує проектування, моделювання, створення й експлуатацію БД в єдине безкоштовне оточення для системи баз даних MySQL.

Було вибрано це програмний засіб за його можливості, а саме:

– можливість наочно представити модель бази даних в графічному вигляді;

– можливість користування наочним і функціональним механізмом установки зв'язків між таблицями, в тому числі «багато до багатьох» із створенням таблиці зв'язків;

– можливість роботи з Reverse Engineering — відновлення структури таблиць з вже існуючої на сервері БД;

– можливість користування зручним редактором SQL запитів, що дозволяє відразу ж відправляти їх серверові і отримати відповідь у вигляді таблиці;

– можливість редагування даних у таблиці в візуальному режимі.

4.3 Опис архітектури розробленої системи

В якості архітектури системи було обрано трирівневу архітектуру «клієнт-сервер». В такій архітектурі є три компонента: клієнт, сервера додатків та сервера баз даних.

Клієнт – це інтерфейси, які надаються кінцевому користувачу. На цей рівень зазвичай виноситься тільки найпростіша бізнес-логіка: інтерфейс авторизації, алгоритми шифрування, перевірка значень, які вводяться, на допустимість і відповідність формату, нескладні операції з даними (сортування, угруповання, підрахунок значень), вже завантаженими на термінал.

Сервер додатку – це зв'язуючий шар програмного додатка, де реалізується велика частина бізнес-логіки.

Сервер бази даних – забезпечує зберігання даних, в нього входять, наприклад, збережені процедури і тригери бази даних.

4.4 Створення бази даних

При проектуванні БД для інформаційної системи було виділено такі сутності:

– «Товар» – сутність, яка зв’язана із сутністю «Кошик» (кошик має інструменти), має інформацію про товари: ім’я, ціна, кількість, малюнок та інформацію для системи рекомендацій: стать, вік, професія, працевлаштування та наявність дітей;

– «Замовлення» – сутність, що організовує основний бізнес-процес – оформлення замовлень, містить інформацію про замовлення: клієнт, менеджер, вартість доставки, дата, статус, кур’єр, коментар клієнта, адреса;

– «Кошик» – проміжна сутність, що зв’язує таблиці «Інструмент» та «Замовлення»;

– «Статуси» – сутність, яка є залежною від сутності «Замовлення» (замовлення мають статуси). Вона містить перелік всіх доступних статусів замовлень;

– «Менеджер» – сутність яка зв’язана із сутністю «Замовлення» (менеджери коректують замовлення) та має інформацію про данні менеджера: ім’я, прізвище, по батькові, гендер, телефон, дата прийому на роботу, id і серія паспорту, зарплата, додаткова інформація, id користувача;

– «Клієнт» – сутність яка зв’язана із сутністю «Замовлення» (клієнти здійснюють замовлення) та має інформацію про данні клієнта: ім’я, прізвище, по батькові, телефон, день народження, додаткова інформація, адреса для замовлень, id користувача;

– «Кур’єр» – сутність яка зв’язана із сутністю «Замовлення» (кур’єри доставляють замовлення) та має інформацію про данні кур’єра: ім’я, прізвище, по батькові, телефон, дата прийому на роботу, id і серія паспорту, зарплата, додаткова інформація, id користувача;

– «Користувач» – сутність яка зв’язана із сутностями «Менеджер», «Клієнт» та «Кур’єр» (менеджер/клієнт/кур’єр є користувачі) та має інформацію про данні користувачів: роль, логін та пароль;

– «Роль» – сутність, яка є залежною від сутності «Користувач» (користувачі поділяються на ролі). Вона містить перелік усіх доступних ролей користувачів.

Логічна модель бази даних представлена на рисунку 4.1.

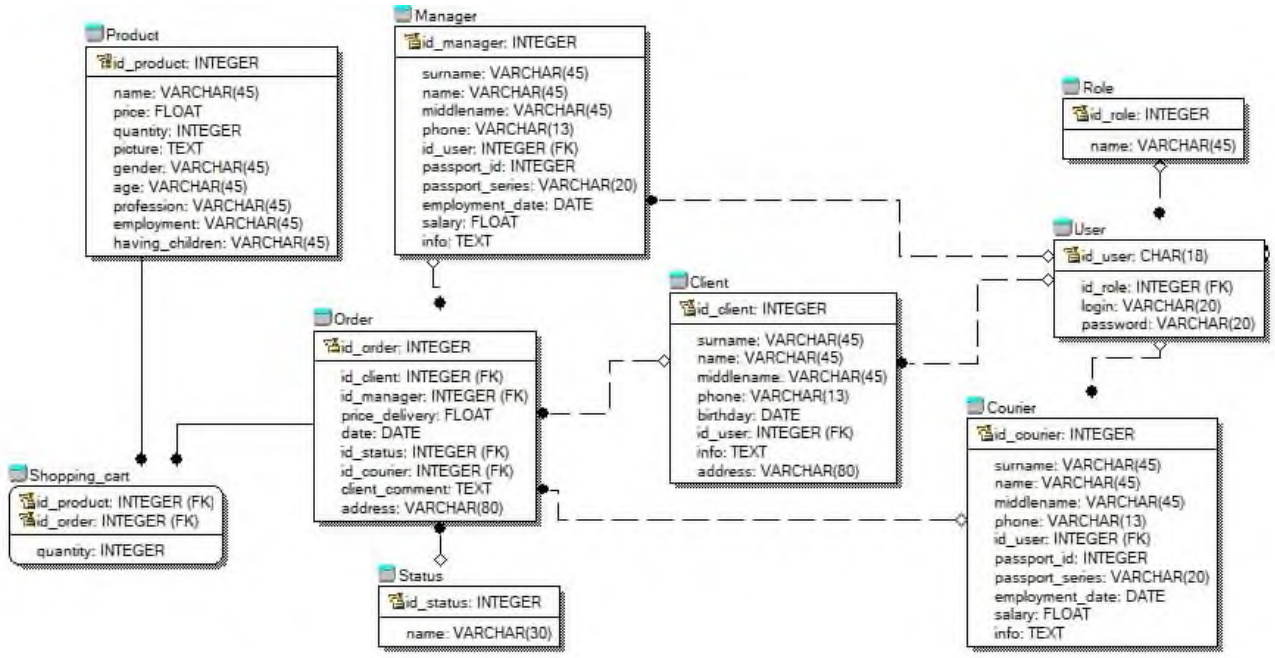

Рисунок 4.1– Логічна модель бази даних

Фізична модель бази даних представлена на рисунку 4.2.

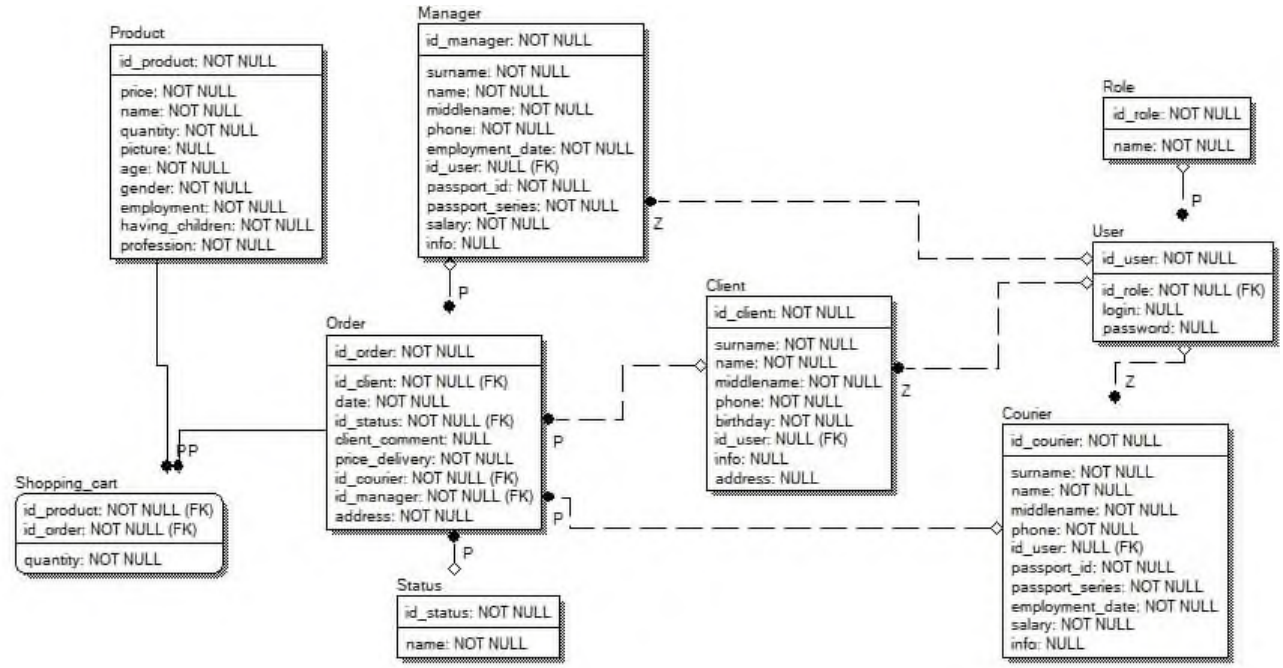

Рисунок 4.2 – Фізична модель бази даних

Фрагмент схеми бази даних для зберігання усіх характеристик по користувачам та покупкам з демографічними ознаками представлено на рисунку 4.3. Ця схема також дозволяє зберігати кількість разів коли товар був куплений по кожній демографічній характеристиці.

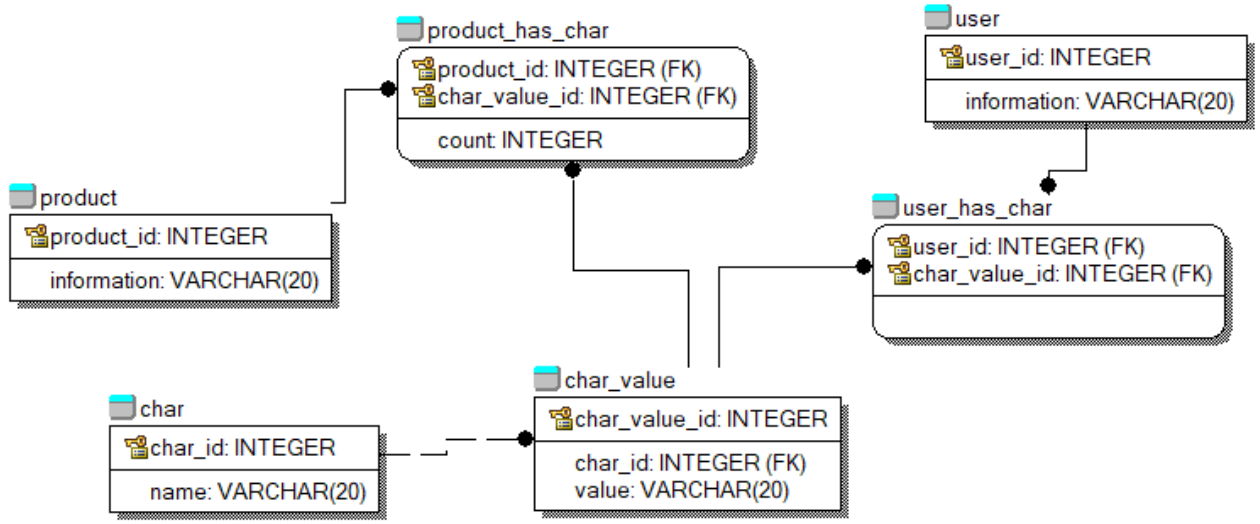

Рисунок 4.3 – Фрагмент схеми бази даних

4.5 Реалізація основної частини системи

Для реалізації системи було обрано мову програмування JavaScript, а саме сервер на базі Node.js. Node.js — платформа з відкритим кодом для виконання високопродуктивних мережевих застосунків, написаних мовою JavaScript. Якщо раніше Javascript застосовувався для обробки даних в браузері користувача, то Node.js надав можливість виконувати JavaScript-скрипти на сервері та відправляти користувачеві результат їхнього виконання. Платформа Node.js перетворила JavaScript на мову загального використання з великою спільнотою розробників.

Під час реалізації системи було використано наступні бібліотеки:

– Express – бібліотека на базі HTTP модуля, яка дозволяє створити сервер;

– Mysql – бібліотека для використання бази даних;

– Nodemon – сервіс, який автоматично перезбирає веб-сервер при внесенні змін до відповідних файлів або відповідних розширень файлів;

– Handlebars – це шаблонний процесор, який динамічно генерує HTML-сторінку.

4.6 Розробка системи і експериментальне дослідження

Для початку був створений об’єкт з демографічними ознаками, представлений у лістингу 4.1, який буде використовуватися для підрахування відповідності користувача до різних груп.

Лістинг 4.1 – Об’єкт з демографічними ознаками

```
const demographic_characteristics = {
  gender: {
    male: 0,
    female: 0
  },
  age: {
    teens: 0,
    adults: 0
  },
  profession: {
    artist: 0,
    builder: 0,
    hairdresser: 0
  },
  employment: {
    employed: 0
  },
  having_children: {
    yes: 0,
    no: 0
  }
};
```

Далі потрібно розробити алгоритм фільтрування.

Вказані вище демографічні ознаки будуть використовуватися для підрахування відповідності користувача до різних груп.

Також збиратиметься інформація щодо відвідуваних продуктів до спеціального масиву. І тут виявилась перша проблема – чи потрібно враховувати кількість разів переглядання товару користувачем? Для цього потрібен додатковий функціонал системи, тож було вирішено не враховувати кількість відвідувань одного й того самого товару, а працювати тільки над демографічною фільтрацією.

Систему було зроблено так, що як тільки користувач перейшов на сторінку перегляду товару, система одразу починає за ним слідкувати – товар вноситься до масиву переглянутих (якщо людина його ще не переглядала), беруться демографічні характеристики цього товару і приписуються користувачеві (якщо ознака характеристики для товару визначена). Людина одразу на першому товарі в системі може бачити рекомендації на основі характеристик цього товару.

Тут була виявлена друга проблема – за формулами 3.1-3.3 користувач може бачити в рекомендаціях і той товар, який він зараз переглядає (рис. 4.4).

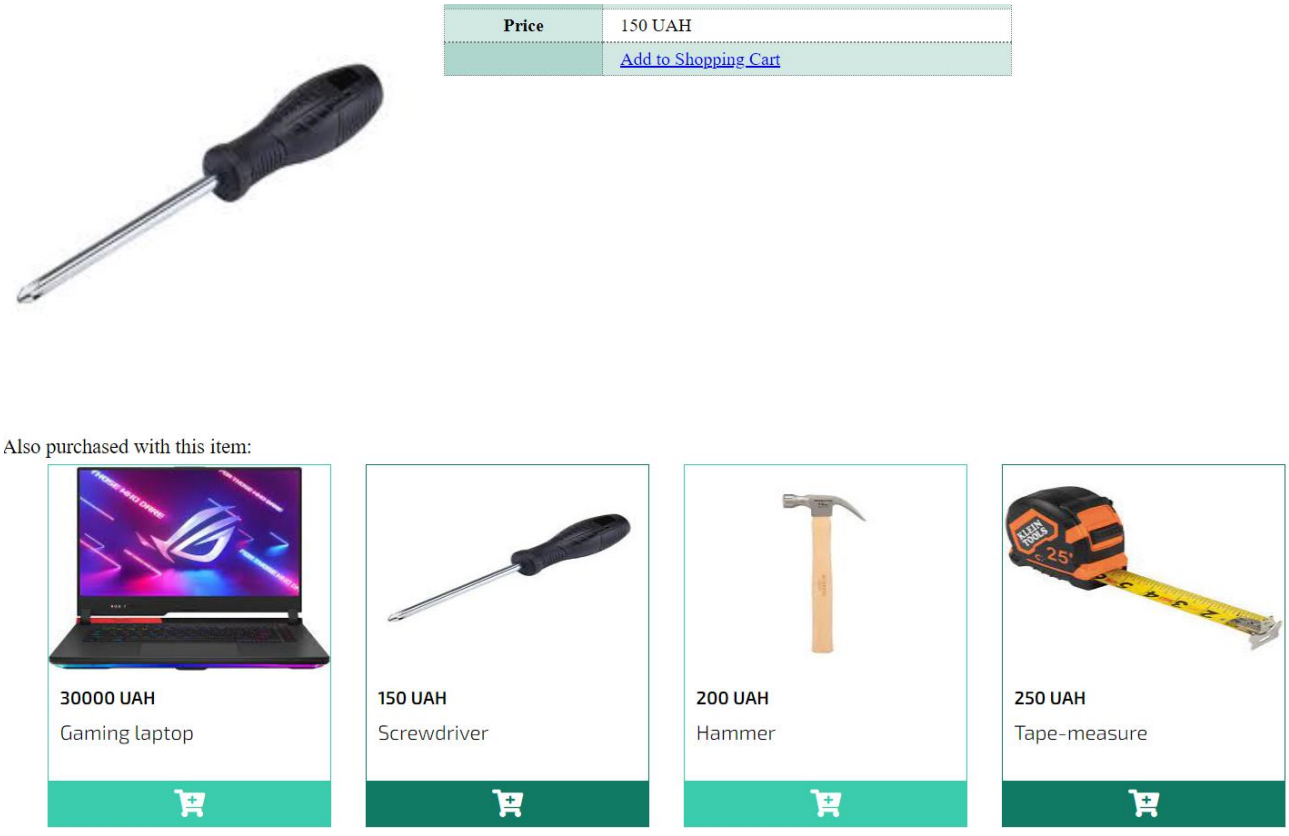

Рисунок 4.4 – Результат рекомендацій по не зовсім коректній формулі

Отже, потрібно трохи переробити формули виключивши з рекомендації товар, на якому користувач зараз знаходиться.

Рекомендації, які будуть показуватися на якомусь конкретному товарі, будуть розраховуватися за формулою 4.1. Тобто це та ж сама формула 3.1, тільки з виключенням з підмножини рекомендацій товару, на якому ці рекомендації і будуть показані.

$$R_y = \left\{x_j, y \in C_{x_j}\right\} - x_b, \tag{4.1}$$

де R_y – множина відібраних товарів для рекомендації з y демографічною характеристикою, x_b – товар, на якому користувач знаходиться, на якому виводяться рекомендації, X – множина товарів, y – переважаюча демографічна характеристика користувача, j – елемент множини товарів X, C_{x_j} – множина демографічних характеристик заданих для товару x_j.

Фільтрування за декількома явними демографічними ознаками будуть робитися на основі формули 4.2. За цією формулою будуть фільтруватися такі товари, які будуть мати всі шукані ознаки в собі разом виключаючи товар, на якому користувач зараз знаходиться.

$$R_Y = \left\{x_j, Y = C_{x_j}\right\} - x_b, \tag{4.2}$$

де R_Y – множина відібраних товарів для рекомендації з Y підмножиною демографічних характеристик, x_b – товар, на якому користувач знаходиться, на якому виводяться рекомендації, X – множина товарів, Y – підмножина переважаючих демографічних характеристик користувача, j – елемент множини товарів X, C_{x_j} – множина демографічних характеристик заданих для товару x_j.

Пошук товару за окремими характеристиками окремо буде проводитися за формулою 4.3.

$$R_Y = \{x_j, Y \subseteq C_{x_j}, Y \cap C_{x_j} \neq \emptyset\} - x_b, \tag{4.3}$$

де R_Y – множина відібраних товарів для рекомендації з Y підмножиною демографічних характеристик, x_b – товар, на якому користувач знаходиться, на якому виводяться рекомендації, X – множина товарів, Y – підмножина переважаючих демографічних характеристик користувача, j – елемент множини товарів X, C_{x_j} – множина демографічних характеристик заданих для товару x_j.

Змоделюємо роботу системи. Наприклад, користувач вперше перейшов на сторінку товару на сайті і це ігровий ноутбук. У системі зазначено, що найчастіше його купують чоловіки, працевлаштовані, підлітки (рис. 4.5).

	id_product	name	price	quantity	picture	gender	age	profession	employment	having_children
▶	1	Gaming laptop	30000.00	345	...	male	teens	unknown	employed	unknown

Рисунок 4.5 – Значення по ігровому ноутбуку у базі даних

Система бере цю інформацію і рекомендує товар, який найчастіше купують чоловіки, працевлаштовані, підлітки (рис. 4.6).

Title	Gaming laptop
Price	30000 UAH
	Add to Shopping Cart

Also purchased with this item:

2000 UAH
Fishing rod

150 UAH
Screwdriver

200 UAH
Hammer

250 UAH
Tape-measure

Рисунок 4.6 – Рекомендації для ігрового ноутбуку

Після цього масив з демографічними ознаками буде виглядати, як представлено у лістингу 4.2.

Лістинг 4.2 – Масив з демографічними ознаками користувача, який починає переглядати товари

```
{
  gender: {
    male: 1,
    female: 0
  },
  age: {
    teens: 1,
    adults: 0
  },
  profession: {
    artist: 0,
    builder: 0,
    hairdresser: 0
  },
  employment: {
    employed: 1
  },
  having_children: {
    yes: 0,
    no: 0
  }
}
```

Далі система визначає максимальне значення з цього об’єкту і якщо це значення тільки у одної характеристики, то посилає запит на базу даних тільки з цією характеристикою та не враховує товар, на якому користувач зараз знаходиться (лістинг 4.3). Запит написано на основі формули 4.1.

Лістинг 4.3 – Запит на базу даних на товар по одній характеристиці

```
SELECT id_product, name, price, picture, gender, age, profession, employment, having_children
```

```
FROM product
WHERE id_product != 1
AND gender LIKE 'male'
```

Складніше, якщо ця максимальне значення є у декількох характеристик. Спочатку система намагається знайти продукти, які будуть релевантними по всіх цих характеристиках.

У даному випадку з ноутбуком у нас є три максимальні характеристики – одиниця. Тобто користувач може однаково відноситися до трьох демографічних груп – чоловіків, працевлаштованих та підлітків. Система буде шукати товари за всіма цими характеристиками (лістинг 4.4). Запит написано на основі формули 4.2.

Лістинг 4.4 – Запит на базу даних на товар за декількома характеристиками разом

```
SELECT id_product, name, price, picture, gender, age, profession, employment, having_children
FROM product
WHERE id_product != 1
AND gender LIKE 'male'
AND age LIKE 'teens'
AND employment LIKE 'employed'
```

Таких товарів не було знайдено (рис. 4.7).

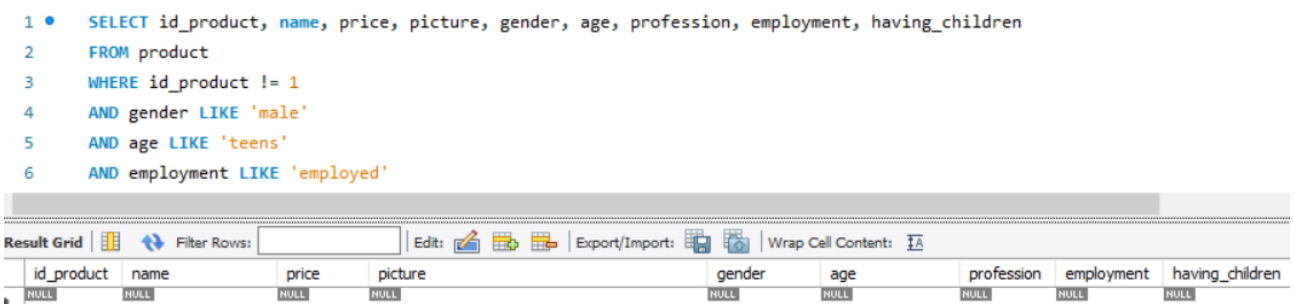

Рисунок 4.7 – Товарів релевантних по всіх характеристиках не було знайдено

Далі, якщо було знайдено менш, ніж 3 товари, то потрібно щось робити, не показувати ж користувачу тільки один-два товари у рекомендаціях. Тож було

використано формулу 4.3 для запита – система починає шукати товари за окремими характеристиками поєднуючи всі результати у кінці та не враховуючи поточний товар (лістинг 4.5).

Лістинг 4.5 – Запит на базу даних для пошуку товарів за окремими характеристиками

```
SELECT id_product, name, price, picture, gender, age, profession, employment, having_children
FROM product
WHERE id_product != 1
AND gender LIKE 'male'
UNION
SELECT id_product, name, price, picture, gender, age, profession, employment, having_children
FROM product
WHERE id_product != 1
AND age LIKE 'teens'
UNION
SELECT id_product, name, price, picture, gender, age, profession, employment, having_children
FROM product
WHERE id_product != 1
AND employment LIKE 'employed'
```

Таким методом було знайдено декілька товарів (рис. 4.8, 4.9), що дає змогу продовжувати надавати рекомендації користувачам.

id_product	name	price	picture	gender	age	profession	employment	having_children
2	Fishing rod	2000.00	...	male	adults	unknown	unknown	yes
6	Screwdriver	150.00	...	male	adults	builder	employed	unknown
7	Hammer	200.00	...	male	adults	builder	employed	unknown
15	Tape-measure	250.00	...	male	adults	builder	employed	unknown
16	Cupboard	13000.00	...	male	adults	unknown	employed	unknown
18	Ring with a diamond	30000.00	...	male	adults	unknown	employed	no
20	Helicopter	3000.00	...	male	teens	unknown	unknown	no
21	Joystick	600.00	...	male	teens	unknown	unknown	no
22	VR glasses	2000.00	...	male	teens	unknown	unknown	no
24	Dumbbells	500.00	...	male	adults	unknown	unknown	unknown
28	Smartphone	34000.00	https://encrypted-tbn0.gstatic.com/images?q=...	male	adults	unknown	employed	unknown
11	Color contact lenses	600.00	...	female	teens	unknown	unknown	no
5	Paint set	450.00	https://m.media-amazon.com/images/I/819aUI...	unknown	unkn...	artist	employed	no
9	Hairdressing scissors	700.00	...	female	adults	hairdresser	employed	unknown
13	Kick scooter	1500.00	https://encrypted-tbn0.gstatic.com/images?q=...	unknown	adults	unknown	employed	yes
17	Oven	40000.00	...	unknown	adults	unknown	employed	unknown
30	Multicooker	24000.00	...	female	adults	unknown	employed	unknown

Рисунок 4.8 – Пошук товарів за окремими характеристиками в базі даних

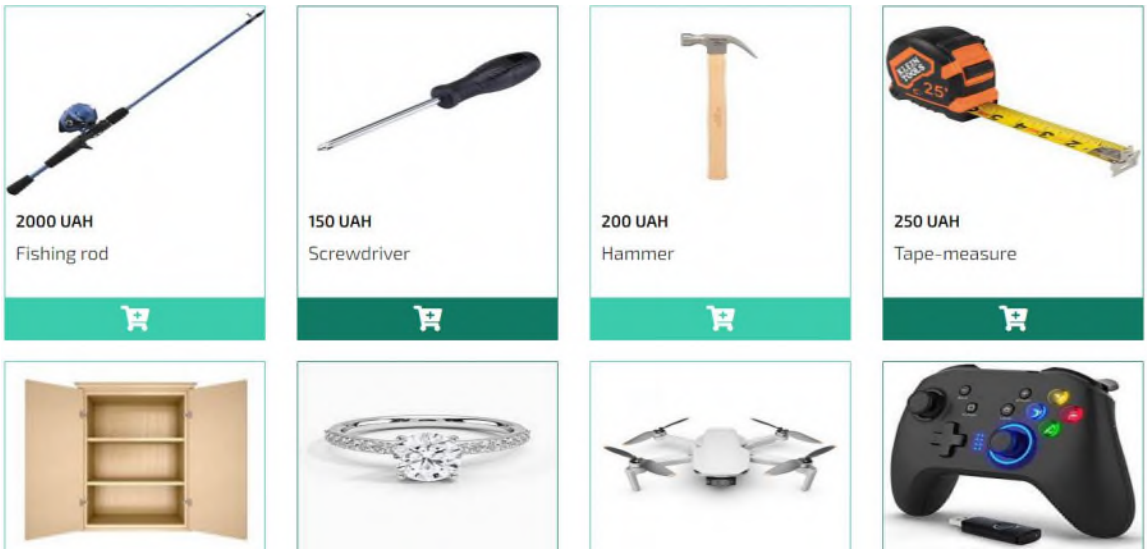

Рисунок 4.9 – Рекомендації користувачу для першого переглянутого товару на сайті – ігровому ноутбуку

Коли користувач буде активно переглядати товари, навчальний об’єкт буде уточнюватися і рекомендації будуть теж більш релевантими. Наприклад, об’єкт який наведено у лістингу 4.6.

Лістинг 4.6 – Навчальний об’єкт з характеристиками активного користувача

```
{
  gender: {
    male: 4,
    female: 4
```

```
  },
  age: {
    teens: 2,
    adults: 9
  },
  profession: {
    artist: 1,
    builder: 1,
    hairdresser: 0
  },
  employment: {
    employed: 7
  },
  having_children: {
    yes: 3,
    no: 3
  }
}
```

Можна побачити, що користувач активно переглядав товар у системі та можна зробити висновок, що найвірогідніше це доросла працевлаштована людина. Йому буде рекомендовано товари, які найчастіше купують дорослі люди, бо товари для дорослих він переглядав 9 разів, а товари для працевлаштованих – 7, що все-таки трохи менше.

Вибірка рекомендацій для цього користувача буде як представлено на рисунку 4.10 (з врахуванням того, що товари для дітей теж купують дорослі).

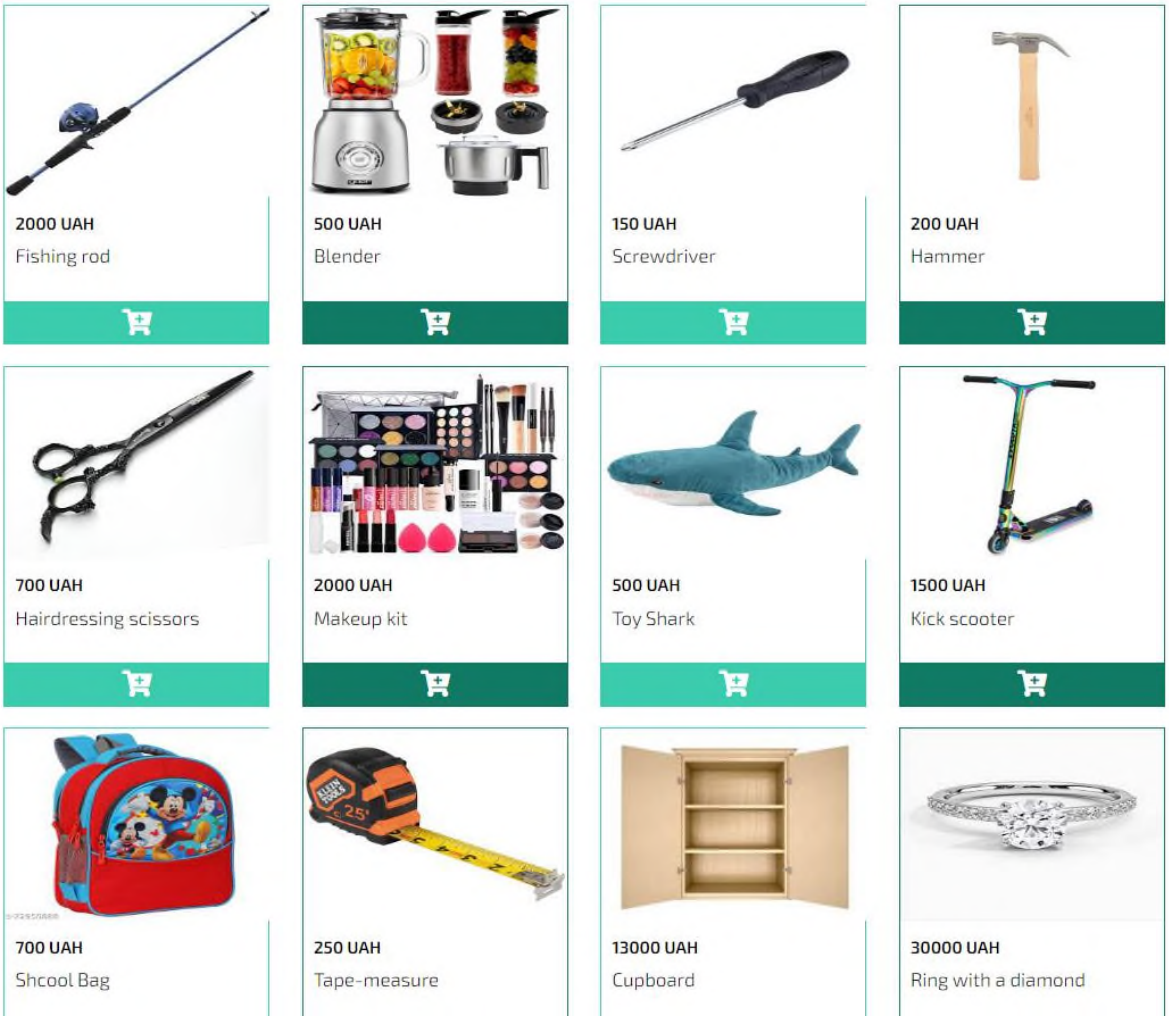

Рисунок 4.10 – Рекомендації для дорослого користувача

На даному етапі була виявлена ще одна проблема – рекомендації найчастіше бувають занадто об’ємними. Щоб позбутися цього, вибірку було обмежено до дванадцяти товарів (лістинг 4.7).

Лістинг 4.7 – Запит на базу даних на товар за декількома характеристиками разом з обмеженням вибірки до дванадцяти елементів

```
SELECT id_product, name, price, picture, gender, age, profession, employment, having_children
FROM product
WHERE id_product != 1
AND gender LIKE 'male'
AND age LIKE 'teens'
AND employment LIKE 'employed'
LIMIT 12
```

У подальшому систему можливо покращити декількома способами.

Можна брати до уваги максимальне число у навчальному об’єкті з характеристиками та декілька наближених до нього, щоб система мала змогу не так критично надавати рекомендації користувачеві.

Наприклад, у людини об’єкт з характеристиками як представлено на лістингу 4.8.

Лістинг 4.8 – Приклад об’єкта з характеристиками активного користувача

```
{
  gender: {
    male: 20,
    female: 45
  },
  age: {
    teens: 89,
    adults: 9
  },
  profession: {
    artist: 15,
    builder: 34,
    hairdresser: 7
  },
  employment: {
    employed: 56
  },
  having_children: {
    yes: 48,
    no: 98
  }
}
```

Можна побачити, що домінуючих характеристик тут декілька – підліток та без дітей. В такому випадку можна було би надавати йому рекомендації на основі

цих ознак, бо вони хоча і не однакові по значенню але близькі настільки, що це можна і треба враховувати.

Далі можна розширити масив з демографічними характеристиками. Наприклад, можна додати освіту, регіон, мову, сімейний стан тощо. Регіон можна брати з API адреси користувача, але надати можливість йому самому обрати регіон. Бо система може помилитися якщо людина користується VPN.

Ще можна попрацювати над оптимізацією системи в подальшому бо така система найчастіше повинна працювати з великими об’ємами даних – тисячі товарів і тисячі користувачів зі своїми власними рекомендаціями.

ВИСНОВКИ

Було проведено дослідження предметної області – методів фільтрації даних в рекомендаційних системах. Існує два найбільш популярних базових підходи: коллаборативна фільтрація (collaborative filtering, CF) і фільтрація на основі змісту (content-based filtering, CbF). Метод останньої фільтрації фокусується на виявленні об'єктів зі схожими характеристиками по відношенню до тих об'єктів, які вже зацікавили користувача. В основі методу спільної фільтрації лежать припущення про консервативність користувацьких переваг (тобто користувачі, які однаково оцінюють певні об'єкти, швидше за все аналогічним чином будуть оцінювати і нові об'єкти з подібними характеристиками).

Було виявлено, що є деякі проблеми при реалізації методів фільтрації даних в рекомендаційних системах, а саме:

– холодний старт користувача;
– холодний старт контенту;
– старіння контенту;
– врахування сезонності;
– швидкість видачі рекомендацій, навантаження на сервер.

Було прийняте рішення реалізовувати такий метод, щоб подолати проблему холодного старту користувача. Її можна було вирішити декількома способами. Наприклад, при реєстрації на сайті можна попросити користувача ввести деякі свої дані, такі як місто, вік та попросити оцінити товари з різних категорій. А можна просто рекомендувати йому найпопулярніші товари, а далі вже дивитися, що його зацікавить.

Було вирішено покращити існуючі системи для вирішення проблеми холодного старту методом демографічної фільтрації.

Були визначені деякі демографічні ознаки та додані до БД до таблиці товарів, по яким проводиться фільтрація. Це дало змогу побудувати математичну модель.

Побудовано математичну модель для надання рекомендацій на підставі демографічних ознак, що надало можливість покращити рекомендації для користувачів, що вперше зайшли на сайт.

Було розроблено БД та систему, та проведено експериментальне дослідження, які показали доцільність використання запропонованого підходу.

В подальшому систему можна по-різному покращувати, зокрема, оптимізацію роботи з великими обсягами даних.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Катерина Мещан, Юрій Міщеряков, Сучасні Методи Надання Персоналізованих Рекомендацій в Інформаційних Системах // Інформаційні системи та технології: матеріали 11-ї Міжнародної науково-технічної конференції. Частина 2, Харків, 22-25 листопада 2022 року / наук. ред. В.В. Безкоровайний, L. Petryshyn, В.Г. Кобзєв. – Х.: ХНУРЕ, 2022. – 48-49 с.

2. Мещан К.А. СУЧАСНІ МЕТОДИ НАДАННЯ ПЕРСОНАЛІЗОВАНИХ РЕКОМЕНДАЦІЙ В ІНФОРМАЦІЙНИХ СИСТЕМАХ / Мещан К.А., Міщеряков Ю.В. // Автоматизація, електроніка та робототехніка. Стратегії розвитку та інноваційні технології / Мещан К.А., Міщеряков Ю.В.

3. Маклаков, С.В. BPwin и ERwin [Текст] : CASE-средства разработки информационных систем/С.В. Маклаков — М.:Диалог-МИФИ, 1999. — 256 с.

4. Веллінг Л., Томсон Л. MySQL. Навчальний посібник: Пер. з англ. - М .: Видавничий дім "Вільямс", 2015. - 304с.

5. Руководство по Node.js [Електронний ресурс] – Режим доступу до ресурсу: https://metanit.com/web/nodejs/.

6. Greg Linden, Brent Smith and Jeremy York Amazon.com recommendations: Item-to-Item Collaborative Filtering // Industry Report, IEEE INTERNET COMPUTING, 2003.

7. Брейкин Е. А. Рекомендательная система на основе коллаборативной фильтрации // Молодой ученый. — 2015. — №13. — С. 31-33.

8. Fleder D., Hosanagar K. Blockbuster Culture's Next Rise or Fall: The Impact of Recommender Systems on Sales Diversity // Management Science, Vol. 55, No. 5, May 2009, pp. 697-712.

9. Рекомендательные системы в ритейле [Електронний ресурс] – Режим доступу до ресурсу: https://www.simbirsoft.com/blog/rekomendatelnye-sistemy-v-riteyle/.

10. Косолапов К. Введение в рекомендательные системы [Електронний ресурс] / Кирилл Косолапов. – 2019. – Режим доступу до ресурсу: https://habr.com/ru/post/476222/.

11. Николенко С. Рекомендательные системы: user-based и item-based [Електронний ресурс] / Сергей Николенко. – 2012. – Режим доступу до ресурсу: https://habr.com/ru/company/surfingbird/blog/139518/.

12. Badrul M. Sarwar, George Karypis, Joseph A. Konstan, John Riedl: Item-based collaborative filtering recommendation algorithms. WWW 2001: 285—295(англ.).

13. D. Billsus and M. Pazzani. Learning collaborative information filterings. In AAAI Workshop on Recommender Systems, 1998.

14. J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In Fourteenth Conference on Uncertainty in AI. Morgan Kaufmann, July 1998.

15. J. Herlocker, J. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In Proc. of Research and Development in Information Retrieval, 1999.

16. K. Honda, N. Sugiura, H. Ichihashi, and S. Araki. Collaborative filtering using principal component analysis and fuzzy clustering. In Web Intelligence, number 2198 in Lecture Notes in Artificial Intelligence, pages 394–402. Springer, 2001.

17. D. M. Pennock and E. Horvitz. Collaborative filtering by personality diagnosis: A hybrid memory- and model-based approach. In IJCAI-99, 1999.

18. P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In Proc. ACM Computer Supported Cooperative Work, pages 175–186, 1994.

19. B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl. Itembased collaborative filtering recommender algorithms. In WWW10, 2001.

20. B.M. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Incremental svd-based algorithms for highly scaleable recommender systems. In ICCIT’02, 2002.

21. S. Vucetic and Z. Obradovic. A regression-based approach for scaling-up personalized recommender systems in e-commerce. In WEBKDD ’00, 2000.

22. S.M. Weiss and N. Indurkhya. Lightweight collaborative filtering method for binary encoded data. In PKDD ’01, 2001.

23. Барсегян А.А., Куприянов М.С., Степаненко В.В., Холод И.И. Методы и модели анализа данных: OLAP и Data Mining // СПб.: БХВПетербург, 2004. — 336 с.

24. Barkan O., Koenigstein N. Item2Vec: Neural Item Embedding for Collaborative Filtering // arXiv preprint arXiv:1603.04259, Mar 2016.

25. Grebennik I., Semenets V., Hubarenko Y. (2020) Information Technologies for Assessing the Impact of Climate Change and Natural Disasters in Socio-Economic Systems. In: Murayama Y., Velev D., Zlateva P. (eds) Information Technology in

Disaster Risk Reduction. ITDRR 2019. IFIP Advances in Information and Communication Technology, vol 575. Springer, Cham.

26. Recommender Systems: Behind the Scenes of Machine Learning-Based Personalization [Електронний ресурс]. – 2021. – Режим доступу до ресурсу: https://www.altexsoft.com/blog/recommender-system-personalization.

27. Recommender System with Machine Learning and Artificial Intelligence / Sachi Nandan Mohanty, Jyotir Moy Chatterjee, Sarika Jain та ін.

28. M.Sridevi. DECORS: A Simple and Efficient Demographic Collaborative Recommender System for Movie Recommendation / M.Sridevi, Dr .R.Rajeswara Rao.

29. Alva Liu. Using Demographic Information to Reduce the New User Problem in Recommender Systems / Alva Liu, Jonah Callvik.

30. Cory Maklin. Memory Based Collaborative Filtering — User Based [Електронний ресурс] / Cory Maklin – Режим доступу до ресурсу: https://medium.com/@corymaklin/memory-based-collaborative-filtering-user-based-42b2679c6fb5.

31. S.H.S. Chee, J Han, K. Wang. Rectree: An efficient collaborative filtering method. Lecture Notes in Computer Science, 2114, 2001.

32. M. O. Conner and J. Herlocker. Clustering Items for Collaborative Filtering. In Proceedings of the ACM SIGIR Workshop on Recommender Systems, Berkeley, CA, August 1999.

33. A. Said, T. Plumbaum, W. E. De Luca, and S. Albayrak, “A comparison of how demographic data affects recommendation,” in Proc. 19th international conference on User modeling, adaption, and personalization, 2011.

34. Kantor, P. B., Rokach, L., Ricci, F., & Shapira, B. (2011). Recommender systems handbook. Springer.

35. J.B.Schafer, D.Frankowski, J.Herlocker, S.Sen, Collaborative filtering recommender systems. The Adaptive Web, 291-324, 2007.

36. Yuanyuan Wang, Stephen Chi-fai Chan and Grace Ngai Applicability of Demographic Recommender System to Tourist Attractions: A Case Study on Trip Advisor, 2012 IEEE DOI 10.1109/WI-IAT.2012.133.

37. L. H. Ungar and D. P. Foster. Clustering Methods for Collaborative Filtering. In Proc. Workshop on Recommendation Systems at the 15th National Conf. on Artificial Intelligence. Menlo Park, CA: AAAI Press.1998.