Міністерство освіти і науки України
Луцький національний технічний університет
Факультет робототехніки та штучного інтелекту
Кафедра штучного інтелекту та математичного моделювання

КВАЛІФІКАЦІЙНА РОБОТА ЗА СТУПЕНЕМ
ВИЩОЇ ОСВІТИ «БАКАЛАВР»

ДОСЛІДЖЕННЯ ТА РОЗРОБКА СИСТЕМИ РЕКОМЕНДАЦІЙ ТОВАРІВ НА ОСНОВІ КОЛАБОРАТИВНОЇ ФІЛЬТРАЦІЇ ТА SPARK MLLIB.

RESEARCH AND DEVELOPMENT OF A PRODUCT RECOMMENDATION SYSTEM USING COLLABORATIVE FILTERING AND APACHE SPARK MLLIB

Спеціальність 113 Прикладна математика
(шифр і назва спеціальності)

освітня програма «Штучний інтелект та аналіз масивів даних»
(назва освітньої програми)

Виконав: здобувач вищої освіти
Групи ПРМ-41
Томак Акім Володимирович
_____
(підпис)

Керівник:
к.т.н., доцент
Бондарський Олександр Георгійович
_____
(підпис)

Кваліфікаційну роботу
допущено до захисту
«_» _____ 20__ р.
к.т.н., доцент
Гарант освітньої програми:
Приходько Олексій Сергійович
_____
(підпис)

Луцьк – 2026 року

ЛУЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Факультет *архітертури, будівництва та дизайну*
Кафедра *прикладної математики та механіки*
Ступінь вищої освіти: *бакалавр*
Галузь знань: *11 Математика і статистика*
Спеціальність *113 Прикладна математика*
Освітня програма *Штучний інтелект та аналіз масивів даних*

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Мікуліч О.А.
«___» _____2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

Томак Акім Володимирович
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи
Дослідження та розробка системи рекомендацій товарів на основі колаборативної фільтрації та Spark MLlib.
Керівник роботи: *Бондарський Олександр Георгійович*
затверджені наказом закладу вищої освіти від «31» грудня 2025 р. № 557/01-02

2. Строк подання здобувачем вищої освіти кваліфікаційної роботи
«___»_____ 2026 р.

3. Вихідні дані до роботи

4. Зміст пояснювальної записки (перелік питань, що потрібно розробити):

5. Перелік графічного (ілюстративного) матеріалу:

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис	
		завдання видав	завдання прийняв
1 розділ	*Бондарський О.Г., доцент кафедри*		
2 розділ	*Бондарський О.Г., доцент кафедри*		
3 розділ	*Бондарський О.Г., доцент кафедри*		
4 розділ	*Бондарський О.Г., доцент кафедри*		
Висновки	*Бондарський О.Г., доцент кафедри*		

7. Дата видачі завдання «___» _____ 202__ р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи бакалавра	Строк виконання етапів роботи	Примітка
1.	*Огляд літератури із досліджуваної проблеми*	*до 01.03.2026*	
2.	*Перший розділ*	*до 05.03.2026*	
3.	*Другий розділ*	*до 01.042026*	
4.	*Третій розділ*	*до 10.04.2026*	
5.	*Четвертий розділ*	*до 18.04.2026*	
6.	*Висновки*	*до 29.04.2026*	
7.	*Формування списку використаних джерел*	*до 05.05.2026*	
8.	*Оформлення ілюстративного матеріалу*	*до 10.05.2026*	
9.	*Нормоконтроль*	*до 20.05.2026*	
10.	*Інструментальна перевірка на академічний плагіат*	*до 02.06.2026*	*Показник запозичень тексту ___%*
11.	*Представлення кваліфікаційної роботи бакалавра до захисту*	*до 12.06.2026*	

Здобувач вищої освіти _____ (Томак А.В.)
(підпис) (прізвище, ініціали)

Керівник кваліфікаційної роботи _____ (Бондарський О.Г.)
(підпис) (прізвище, ініціали)

АНОТАЦІЯ

Томак А. О. Дослідження та розробка системи рекомендацій товарів на основі колаборативної фільтрації та Spark Mllib. Рукопис. Кваліфікаційна робота бакалавра ОП «Штучний інтелект та аналіз масивів даних» спеціальності 113 Прикладна математика. Луцький національний технічний університет. Луцьк, 2026.

У кваліфікаційній роботі досліджено методи побудови рекомендаційних систем на основі колаборативної фільтрації з метою підвищення якості персоналізації товарних рекомендацій у сфері електронної комерції. Об’єктом дослідження є процес формування персоналізованих рекомендацій товарів на основі взаємодії користувачів із системою. Предметом дослідження є алгоритми колаборативної фільтрації, методи матричної факторизації та технології машинного навчання, що використовуються для створення рекомендаційних систем.

У роботі проведено аналіз предметної галузі та сучасних підходів до побудови рекомендаційних систем, розглянуто принципи роботи колаборативної фільтрації та алгоритму ALS. Досліджено методи обробки даних користувачів, виконано підготовку датасету та реалізовано модель рекомендаційної системи засобами Apache Spark MLlib. Проведено експерименти з оцінювання точності роботи моделі та аналіз ефективності сформованих рекомендацій.

Практична цінність роботи полягає у розробці програмного модуля рекомендаційної системи, який може бути використаний у системах електронної комерції для автоматичного формування персоналізованих товарних рекомендацій. Запропоноване рішення дозволяє підвищити якість взаємодії користувачів із сервісом, покращити релевантність рекомендацій та збільшити ефективність онлайн-платформ. Ключові слова: рекомендаційна система, колаборативна фільтрація, ALS, матрична факторизація, машинне навчання, Apache Spark, персоналізація, електронна комерція, аналіз даних.

ANNOTATION

Tomak A. O. Research and Development of a Product Recommendation System Based on Collaborative Filtering and Spark MLlib. Manuscript. Bachelor's qualification thesis in the field of study 113 Applied Mathematics, educational program "Artificial Intelligence and Data Analysis". Lutsk National Technical University, Lutsk, 2026.

The qualification thesis is devoted to the research and development of a product recommendation system based on collaborative filtering methods aimed at improving the quality of personalized recommendations in e-commerce systems. The object of the research is the process of generating personalized product recommendations based on user interaction with the system. The subject of the research includes collaborative filtering algorithms, matrix factorization methods, and machine learning technologies used in recommendation systems.

The paper analyzes the subject area and modern approaches to the construction of recommendation systems. The principles of collaborative filtering and the ALS (Alternating Least Squares) algorithm are considered. Methods of user data processing were investigated, the dataset was prepared and analyzed, and a recommendation model was implemented using Apache Spark MLlib tools. Experimental studies were conducted to evaluate the accuracy and effectiveness of the developed recommendation system.

The practical significance of the work lies in the development of a software module for generating personalized product recommendations that can be integrated into e-commerce platforms. The proposed solution improves the relevance of recommendations, enhances the quality of user interaction, and increases the efficiency of online services.

Keywords: recommendation system, collaborative filtering, ALS, matrix factorization, machine learning, Apache Spark MLlib, personalization, e-commerce, data analysis.

ЗМІСТ

ВСТУП

Сьогодні в електронній комерції відбувається інформаційний вибух, і кількість продуктів, доступних користувачам, постійно зростає. Існують також сотні інтернет-магазинів і маркетплейсів, які продають мільйони товарів, що ускладнює процес визначення того, що потрібно покупцям. У цьому сенсі системи рекомендацій, які дозволяють користувачам автоматично бачити найкращі продукти на основі їхніх вподобань, історії покупок та поведінки інших користувачів, також набули більшої популярності.

Системи рекомендацій використовуються багатонаціональними компаніями в електронній комерції, кінотеатрах, музиці та соціальних мережах. Використовуючи ці інструменти, можна покращити користувацький досвід, задоволеність клієнтів, прибутковість за допомогою персональних рекомендацій.

Колаборативний підхід часто використовується для розробки систем рекомендацій. Ця техніка спирається на дані про взаємодію користувачів з продуктами та пошук схожих користувачів або схожих продуктів.

Колаборативна фільтрація не вимагає детального опису атрибутів продукту, і рекомендації можуть бути зроблені на основі історичних даних про рейтинги, перегляди або покупки. Як результат, вони повинні використовувати інструменти для швидкої та ефективної обробки даних, оскільки обсяги даних зростають.

Apache Spark, ймовірно, є однією з найбільш часто використовуваних платформ для обробки великих даних. Завдяки обчисленням у пам'яті платформа дозволяє виконувати розподілені завдання на кластері комп'ютерів для ефективної обробки інформації.

У проекті використовується бібліотека Spark MLlib, яка включає інструменти для аналізу класифікації даних, кластеризації, систем рекомендацій та інших операцій обробки інформації для алгоритмів машинного навчання в рамках сценарію реалізації Spark. Spark MLlib особливо зосереджена на

алгоритмі чергування найменших квадратів (ALS), який є однією з найпотужніших стратегій колаборативної фільтрації для аналізу великих обсягів даних. Метод також дозволяє факторизацію матриці взаємодії користувач/продукт, а також прогнозування рейтингів для товарів, з якими користувач ще не взаємодіяв.

Диплом також надає можливість реалізувати ефективні системи персоналізації для сектора електронної комерції, а також широке використання інновацій великих даних та можливостей машинного навчання для обробки великих обсягів даних.

Apache Spark та бібліотека MLlib можуть бути використані для реалізації масштабованої системи рекомендацій, що працює на великих наборах даних і отримує швидкі результати.

Метою дипломної роботи є дослідження технік колаборативної фільтрації та створення системи рекомендацій продуктів у рамках проекту розробки згідно з Apache Spark MLlib.

Для досягнення мети це включає наступне:

1. Вивчення сучасних стратегій розробки систем рекомендацій;
2. Ознайомлення з функціональними можливостями колаборативної фільтрації;
3. Аналіз того, що робить Apache Spark з точки зору архітектури та можливостей;
4. Досвід роботи з інструментами, що використовуються в бібліотеці Spark MLlib для створення систем рекомендацій;
5. Підготовка та обробка набору даних;
6. Використання структури рекомендацій продуктів на основі алгоритму ALS;
7. Проведення експериментальної роботи та перевірка якості наданих рекомендацій;
8. Проведення аналізу того, що ми виявили в роботі системи.

Фокус дослідження Метою дослідження є створення рекомендацій продуктів, специфічних для користувача, для торгівлі на електронних платформах. Дослідження зосереджено на колаборативній фільтрації та алгоритмах для побудови системи рекомендацій з Spark MLlib. Методи дослідження включають: наукові джерела, методи машинного навчання, методи колаборативної фільтрації та статистичний аналіз даних, а також програмні інструменти Apache Spark та Spark MLlib.

Отже, практична значимість досягнутих результатів полягає в можливості підвищення ефективності інтернет-магазинів за допомогою запропонованої системи рекомендацій, підвищення персоналізації та якості обслуговування клієнтів, а також покращення вибору продуктів клієнтами.

Структура дипломної роботи включає вступ, основні розділи, висновок, список джерел для довідкових матеріалів та додатки. Ця робота спрямована на вивчення теоретичних основ системи рекомендацій, методів колаборативної фільтрації рекомендацій, можливостей платформи Apache Spark та бібліотеки MLlib разом, а також реалізацію та валідацію розробленої системи рекомендацій продуктів.

РОЗДІЛ 1. АНАЛІЗ РЕКОМЕНДАЦІЙНИХ СИСТЕМ

1.1 Поняття рекомендаційних систем

Цифрові дані зростають з вражаючою швидкістю в сучасному інформаційному суспільстві. Усі онлайн-продажі та послуги, соціальні мережі та інші онлайн-магазини й інформаційні сховища щодня збирають величезну кількість інформації про використання та поведінку користувачів. У таких умовах клієнтам стає важче знайти те, що, на їхню думку, найкраще відповідає їхнім інтересам і потребам. Для вирішення цієї проблеми впроваджуються рекомендаційні системи, які автоматизують процес відбору відповідного контенту та пропонують індивідуально підібрані рішення для кожного користувача.

Рекомендаційна система – це комп'ютерна програма або алгоритм, призначений для прогнозування того, що користувач захоче (або не захоче) придбати в певному продукті, послузі чи інформаційному об'єкті. Основна мета рекомендаційних систем – надавати користувачам найбільш релевантні пропозиції на основі аналізу їхньої поведінки (як вони взаємодіють і що вже мають). Рекомендаційні системи з'явилися у відповідь на проблему перевантаження інформацією.

Електронна комерція спричинила експоненційне зростання кількості продуктів, які можна придбати. Наприклад, великі онлайн-ритейлери можуть запропонувати мільйони товарних позицій, що ускладнює клієнту пошук найкращих продуктів під час перегляду. Те ж саме відбувається з мультимедійним контентом: користувачі можуть переглядати та отримувати доступ до безлічі фільмів, музичних композицій, книг та інших інформаційних ресурсів.

Рекомендаційні системи сприяють зменшенню часу пошуку та покращенню користувацького досвіду. Вони обробляють відомі дані та здатні автоматично рекомендувати об'єкти, які можуть зацікавити конкретного

користувача. Це робить рекомендаційні системи потужним механізмом для персоналізованих інформаційних послуг сьогодні.

Функціонування рекомендаційних систем базується на зборі, аналізі та валідації даних про взаємодію користувача з широким спектром об'єктів. Ці дані можуть включати рейтинги продуктів, історію покупок, перегляди сторінок, кліки на кнопки, додавання певних продуктів до списків бажань, коментарі та інші дії користувача. Система вписує ці дані в математичну модель, яка дозволяє прогнозувати уподобання користувача в майбутньому.

Сьогодні рекомендаційні системи стали звичними в різних аспектах діяльності. Найчастіше їх використовують у сфері електронної комерції. Рекомендаційні алгоритми, що використовуються в онлайн-магазинах, формують персоналізовані пропозиції продуктів для онлайн-магазинів, що може створити обсяг продажів, підвищити задоволеність споживачів тощо.

Крім того, рекомендаційні системи застосовуються у відеосервісах, де користувачі можуть обирати фільми та серіали, на музичних платформах, які пропонують музичні композиції, на соціальних платформах для показу релевантного контенту та новинних ресурсах для персоналізації стрічки новин. Основна перевага рекомендаційних систем полягає в тому, що вони можуть автоматично враховувати індивідуальні характеристики користувачів. На відміну від традиційних методів пошуку, які залежать від активного пошуку інформації користувачем, рекомендаційні системи самостійно пропонують найбільш релевантні об'єкти, тим самим значно підвищуючи ефективність взаємодії з інформаційними ресурсами.

Рекомендаційні системи бувають різних типів залежно від використовуваних методів аналізу даних. Це в основному включає системи, засновані на контенті, колаборативну фільтрацію та кілька гібридних моделей. Системи, засновані на контенті, аналізують атрибути об'єктів і створюють рекомендації на основі цих подібностей.

Спільний пошук поведінки багатьох користувачів приносить подібні уподобання через колаборативну фільтрацію. Гібридні системи використовують

ці переваги кількох підходів для покращення рекомендацій. Ця робота особливо зацікавлена в колаборативній фільтрації, яка є одним з найефективніших і універсальних методів побудови рекомендаційних систем. Її передумова полягає в тому, що подібні інтереси користувачів призведуть до подібних уподобань у майбутньому. Система може вважати, що якщо два користувачі дали позитивні оцінки схожому продукту, то інші подібні продукти також, ймовірно, будуть такими ж.

Переваги є всі, але так само є й недоліки рекомендаційних систем. Можливо, найвідомішою з цих проблем є холодний старт, ситуація, яка виникає, коли нові споживачі або новий продукт не мають достатньо даних для ефективних пропозицій.

По-друге, існує проблема розрідженості даних, коли більшість користувачів взаємодіють лише з невеликою кількістю об'єктів. Рекомендаційні системи також повинні мати справу з великими обсягами даних, що є ще однією характеристикою тих, що надають послуги.

Такі великі інформаційні системи можуть похвалитися мільйонами користувачів або їхніх продуктів. Відповідно, тут потрібні технології високої продуктивності обробки даних. Тому багато традиційних рекомендаційних систем реалізуються на розподілених обчислювальних платформах, таких як Apache Spark та бібліотека Spark MLlib.

Отже, в сучасних інформаційних технологіях системи рекомендацій є важливою частиною інформаційних технологій, які принесуть розумне рішення проблеми пошуку інформації з постійним збільшенням обсягів даних. Впровадження таких систем допомагає покращити якість обслуговування користувачів, ефективність процесів у бізнес-середовищі та розвиток технологій персоналізації в різних галузях промисловості.

1.2 Види рекомендаційних систем

У сучасних інформаційних технологіях системи рекомендацій є важливим інструментом для надання персоналізованої інформації окремим клієнтам. Існує кілька типів систем рекомендацій, однак, заснованих на техніках аналізу даних і концепціях формування рекомендацій. Кожен з них має унікальні характеристики, переваги та недоліки, які визначають область, в якій він корисний.

Контентно-орієнтовані рекомендаційні системи

Контентно-орієнтовані (Content-Based) рекомендаційні системи формують рекомендації на основі характеристик об'єктів та попередніх уподобань користувача. Основна ідея такого підходу полягає в тому що користувачу рекомендуються об'єкти, схожі на ті які він позитивно оцінював або використовував раніше.

Для роботи системи аналізуються атрибути товарів або контенту. Наприклад, у системі рекомендації фільмів можуть враховуватися жанр, режисер, акторський склад та рік випуску. Якщо користувач часто переглядає науково-фантастичні фільми, система рекомендуватиме інші фільми цього жанру.

Основними перевагами контентно-орієнтованого підходу є незалежність від інших користувачів та можливість рекомендувати нові об'єкти. Водночас такий метод має недоліки, серед яких обмежена різноманітність рекомендацій та необхідність наявності детального опису характеристик об'єктів.

Рекомендаційні системи на основі колаборативної фільтрації

Колаборативна фільтрація (Collaborative Filtering) є одним із найпопулярніших підходів до побудови рекомендаційних систем. Вона базується на аналізі поведінки великої кількості користувачів та пошуку схожостей між ними.

Основна ідея методу полягає в тому, що користувачі зі схожими вподобаннями можуть бути зацікавлені в однакових товарах або послугах. Якщо

двоє користувачів позитивно оцінювали схожі товари в минулому, існує висока ймовірність того, що їхні майбутні вподобання також будуть подібними.

Колаборативна фільтрація поділяється на два основні підходи:

1. User-Based Collaborative Filtering;
2. Item-Based Collaborative Filtering.

У першому випадку система знаходить користувачів зі схожими вподобаннями та формує рекомендації на основі їхньої поведінки. У другому випадку аналізується схожість між самими товарами або об'єктами.

Головною перевагою колаборативної фільтрації є можливість формування якісних рекомендацій без необхідності аналізувати характеристики товарів. Недоліками є проблема холодного старту та зниження ефективності при недостатній кількості даних.

Гібридні рекомендаційні системи

Гібридні рекомендаційні системи поєднують декілька методів формування рекомендацій з метою підвищення точності результатів. Найчастіше поєднуються контентно-орієнтований підхід та колаборативна фільтрація.

Використання гібридного підходу дозволяє компенсувати недоліки окремих методів. Наприклад, проблема холодного старту може бути частково вирішена завдяки використанню характеристик товарів навіть за відсутності історії взаємодій користувача.

Гібридні системи широко застосовуються у великих комерційних сервісах, де необхідно забезпечити високу якість рекомендацій для мільйонів користувачів. Вони вважаються одним із найефективніших напрямів розвитку сучасних рекомендаційних технологій.

Демографічні рекомендаційні системи

Демографічні рекомендаційні системи використовують інформацію про характеристики користувачів, такі як вік, стать, місце проживання, рівень освіти або професія. На основі цих даних система визначає групи користувачів зі схожими характеристиками та пропонує їм відповідні товари або послуги.

Перевагою такого підходу є можливість формування рекомендацій навіть за відсутності історії взаємодій. Проте точність рекомендацій значною мірою залежить від повноти та актуальності демографічних даних.

Порівняльна характеристика рекомендаційних систем

Кожен із розглянутих підходів має власні особливості та сферу застосування. Контентно-орієнтовані системи ефективні при наявності детального опису об'єктів, колаборативна фільтрація демонструє високу якість рекомендацій за наявності великої кількості користувацьких даних, а гібридні системи дозволяють поєднати переваги кількох методів одночасно.

Таблиця 1.1 - Види рекомендаційних систем, їх переваги та недоліки

Вид системи	**Принцип роботи**	**Переваги**	**Недоліки**
Контентно-орієнтована	Рекомендує об'єкти, схожі на ті, які подобались користувачу раніше	Не залежить від інших користувачів, працює для нових товарів	Обмежена різноманітність рекомендацій
Колаборативна фільтрація	Використовує вподобання схожих користувачів	Висока точність рекомендацій	Проблема холодного старту, розрідженість даних
Демографічна	Використовує вік, стать, місце проживання та інші характеристики	Простота реалізації	Низька персоналізація
Знаннєво-орієнтована	Формує рекомендації на основі правил та баз знань	Висока точність для складних товарів	Складність підтримки бази знань
Гібридна	Поєднує декілька підходів	Висока якість рекомендацій	Складність реалізації

1.3 Переваги та недоліки рекомендаційних систем

Рекомендаційні системи є невід'ємною складовою сучасних інформаційних платформ та сервісів електронної комерції. Їх використання

дозволяє автоматизувати процес пошуку релевантної інформації та забезпечити персоналізовану взаємодію з користувачами. Завдяки розвитку технологій машинного навчання та аналізу даних рекомендаційні системи стали одним із найефективніших інструментів підвищення якості обслуговування клієнтів і збільшення прибутковості бізнесу. Проте поряд із численними перевагами вони мають і певні обмеження та недоліки.

Переваги рекомендаційних систем

Однією з головних переваг рекомендаційних систем є персоналізація контенту. Система аналізує поведінку користувача, його інтереси, історію покупок або переглядів та на основі отриманих даних формує індивідуальні рекомендації. Це дозволяє кожному користувачу отримувати пропозиції, які найбільше відповідають його потребам та вподобанням.

Ще однією важливою перевагою є зменшення інформаційного перевантаження. У сучасних умовах користувачі мають доступ до величезної кількості товарів, послуг та інформаційних ресурсів. Самостійний пошук потрібного продукту серед тисяч або навіть мільйонів варіантів може займати значний час. Рекомендаційні системи допомагають швидко знаходити найбільш релевантні пропозиції та суттєво спрощують процес вибору.

Використання рекомендаційних систем позитивно впливає на показники діяльності компаній. Персоналізовані рекомендації сприяють збільшенню кількості покупок, підвищенню середнього чека та покращенню конверсії. Користувачі частіше купують товари, які відповідають їхнім інтересам, тому рекомендаційні системи є важливим інструментом підвищення прибутковості бізнесу.

Суттєвою перевагою є також підвищення рівня задоволеності користувачів. Отримуючи корисні та актуальні рекомендації, користувачі витрачають менше часу на пошук потрібної інформації та отримують більш позитивний досвід взаємодії з платформою. Це сприяє формуванню лояльності клієнтів та збільшенню ймовірності повторного використання сервісу.

Рекомендаційні системи дозволяють ефективно просувати нові або менш популярні товари. За допомогою відповідних алгоритмів система може рекомендувати користувачам об'єкти, які вони навряд чи знайшли б самостійно. Це сприяє збільшенню різноманітності вибору та розширенню асортименту продукції, що привертає увагу покупців.

Ще однією перевагою є можливість автоматизації процесу прийняття рішень. Сучасні рекомендаційні системи працюють у режимі реального часу та здатні швидко аналізувати великі обсяги даних. Це дозволяє мінімізувати участь людини у процесі формування рекомендацій та забезпечити високу швидкість роботи сервісів.

Недоліки рекомендаційних систем

Незважаючи на значну кількість переваг, рекомендаційні системи мають низку проблем та обмежень, які можуть впливати на якість їхньої роботи.

Однією з найбільш відомих проблем є проблема холодного старту (Cold Start). Вона виникає у випадках, коли новий користувач або новий товар ще не мають достатньої кількості даних для аналізу. Через відсутність історії взаємодій система не може сформувати якісні рекомендації, що призводить до зниження їхньої точності.

Іншим суттєвим недоліком є розрідженість даних (Data Sparsity). У великих системах користувачі зазвичай взаємодіють лише з невеликою частиною доступних товарів або контенту. У результаті матриця взаємодій містить значну кількість порожніх значень, що ускладнює роботу алгоритмів та може негативно впливати на точність рекомендацій.

Проблемою також є масштабованість системи. Зі збільшенням кількості користувачів і товарів обсяг даних стрімко зростає. Для обробки таких масивів інформації необхідні високопродуктивні обчислювальні ресурси та спеціалізовані платформи, зокрема Apache Spark. Без використання сучасних технологій продуктивність рекомендаційної системи може значно знижуватися.

Ще одним недоліком є ризик виникнення так званого «інформаційного міхура» (Filter Bubble). У цьому випадку система постійно пропонує користувачу

лише ті товари або матеріали, які відповідають його попереднім інтересам. Це обмежує різноманітність контенту та може призвести до звуження кола доступної інформації.

Важливою проблемою залишається захист персональних даних користувачів. Для побудови якісних рекомендацій система повинна збирати та аналізувати значну кількість інформації про поведінку користувачів. Неправильне використання або недостатній рівень захисту таких даних може призвести до порушення конфіденційності та втрати довіри клієнтів.

Також рекомендаційні системи можуть бути вразливими до маніпуляцій. Недобросовісні користувачі або конкуренти можуть штучно змінювати рейтинги товарів або створювати фальшиві профілі з метою впливу на результати рекомендацій. Такі дії здатні погіршити якість роботи системи та знизити її ефективність.

Аналіз переваг і недоліків

Ефективність рекомендаційної системи значною мірою залежить від якості вихідних даних, вибраного алгоритму та особливостей предметної області. У більшості випадків переваги рекомендаційних систем суттєво переважають їхні недоліки, що пояснює їх широке використання у сучасних цифрових сервісах.

Для мінімізації існуючих проблем використовуються різноманітні підходи, зокрема гібридні алгоритми, методи машинного навчання та технології обробки великих даних. Одним із ефективних рішень є використання алгоритму ALS у середовищі Apache Spark MLlib, який дозволяє працювати з великими наборами даних та забезпечує високу масштабованість системи.

Таким чином, рекомендаційні системи є потужним інструментом персоналізації інформаційних сервісів та електронної комерції. Незважаючи на наявність певних недоліків, їх використання дозволяє значно підвищити якість обслуговування користувачів, покращити ефективність бізнес-процесів та забезпечити швидкий доступ до релевантної інформації.

1.4 Аналіз сучасних рекомендаційних платформ

У сучасному цифровому середовищі рекомендаційні системи стали невід'ємною частиною багатьох інформаційних сервісів. Вони активно використовуються в електронній комерції, стримінгових платформах, соціальних мережах та новинних ресурсах. Основною метою таких систем є надання користувачам персоналізованих рекомендацій, які відповідають їхнім інтересам і потребам. Завдяки використанню алгоритмів машинного навчання та аналізу великих даних сучасні рекомендаційні платформи здатні обробляти інформацію про мільйони користувачів та формувати високоточні рекомендації в режимі реального часу.

Однією з найвідоміших платформ, що активно використовує рекомендаційні технології, є Amazon. Компанія вважається одним із піонерів впровадження рекомендаційних систем у сфері електронної комерції. Алгоритми Amazon аналізують історію покупок користувачів, переглянуті товари, пошукові запити та поведінкові фактори. На основі отриманих даних система формує персоналізовані пропозиції товарів, які можуть зацікавити покупця.

Особливістю рекомендаційної системи Amazon є використання колаборативної фільтрації на основі схожості товарів. Якщо користувач переглядає певний товар, система пропонує інші товари, які часто купували разом із ним. Такий підхід дозволяє збільшити кількість продажів та покращити досвід користувачів під час здійснення покупок.

Іншою відомою платформою є Netflix, яка спеціалізується на наданні відеоконтенту. Рекомендаційна система Netflix вважається однією з найефективніших у світі. Вона аналізує історію переглядів, оцінки фільмів та серіалів, тривалість перегляду, жанрові вподобання користувача та інші параметри.

Основою роботи рекомендаційної системи Netflix є поєднання колаборативної фільтрації, контентного аналізу та методів машинного навчання. Завдяки цьому сервіс здатний пропонувати користувачам фільми та серіали, які

з високою ймовірністю викличуть у них зацікавленість. За оцінками компанії, значна частина переглядів на платформі здійснюється саме завдяки рекомендаціям системи.

Широке використання рекомендаційних технологій також спостерігається на платформі YouTube. Щоденно сервіс обробляє величезні обсяги інформації про перегляди відео, підписки, лайки, коментарі та інші дії користувачів. На основі цих даних формується персоналізована стрічка рекомендованих відеороликів.

Особливістю рекомендаційної системи YouTube, є використання складних нейронних мереж та алгоритмів глибокого навчання. Система аналізує не лише попередню активність користувача, а й характеристики відео, поведінку схожих користувачів та поточні тренди. Це дозволяє підтримувати високий рівень залучення аудиторії та збільшувати тривалість перегляду контенту.

Важливу роль рекомендаційні системи відіграють і в соціальних мережах. Наприклад, Facebook, Instagram та TikTok використовують складні алгоритми персоналізації для формування стрічки новин та рекомендації контенту. Система аналізує взаємодію користувача з публікаціями, перегляди відео, коментарі, підписки та інші поведінкові показники.

Особливо показовим є приклад з TikTok, де рекомендаційна система фактично визначає більшість контенту, який бачить користувач. Алгоритми платформи постійно аналізують реакцію користувачів на відео та швидко адаптуються до змін їхніх інтересів. Це дозволяє забезпечувати високий рівень персоналізації та утримання аудиторії.

В Україні рекомендаційні системи активно використовуються великими інтернет-магазинами. Наприклад, маркетплейси застосовують алгоритми для рекомендації схожих товарів, популярних покупок та персоналізованих пропозицій. Такі рішення допомагають користувачам швидше знаходити потрібні товари та підвищують ефективність роботи електронних торговельних платформ.

Аналіз сучасних рекомендаційних платформ показує, що більшість із них використовують комбінацію декількох підходів до формування рекомендацій. Найчастіше застосовуються колаборативна фільтрація, контентно-орієнтовані методи, машинне навчання та нейронні мережі. Використання гібридних алгоритмів дозволяє підвищити точність рекомендацій та зменшити вплив таких проблем, як холодний старт або розрідженість даних.

Таким чином, сучасні рекомендаційні платформи є важливим інструментом персоналізації цифрових сервісів. Їх використання дозволяє покращити користувацький досвід, збільшити залученість аудиторії та підвищити ефективність бізнес-процесів. Аналіз існуючих рішень свідчить про високу ефективність алгоритмів колаборативної фільтрації, що обумовлює доцільність їх використання у даній дипломній роботі для побудови системи рекомендацій товарів на основі Apache Spark MLlib.

1.5 Висновки до розділу 1

У першому розділі дипломної роботи обговорювалися теоретичні основи рекомендаційних систем та їх роль у сучасних інформаційних технологіях. Було перевірено, що такі рекомендаційні системи є важливим засобом персоналізації контенту (з метою кращого вирішення проблеми перевантаження інформацією користувачів). Обговорювалися основні цілі та принципи рекомендаційних систем, а також базові принципи їх роботи.

Основним завданням рекомендаційних систем, на основі аналізу дій користувачів та доступних даних про об'єкти, було визначено генерацію персоналізованих рекомендацій. У рамках дослідницького процесу було розглянуто основні типи рекомендаційних систем, включаючи системи на основі контенту, колаборативну фільтрацію, демографічні, на основі знань та гібридні системи.

Системи колаборативної фільтрації є найбільш використовуваними на практиці, оскільки вони надають якісні рекомендації без необхідності глибокого аналізу характеристик продуктів.

Також були досліджені переваги та недоліки рекомендаційних систем. До переваг належать: персоналізований контент, кращий сервіс для користувачів, підвищення ефективності бізнес-процесів та скорочення часу на пошук інформації. Однак основними недоліками є: холодний старт, розрідженість даних, складність масштабування та проблеми захисту персональних даних.

Отримані результати надають корисну інформацію для дослідження протоколів колаборативної фільтрації та можливості розробки систем рекомендацій продуктів. Таким чином, у наступному розділі буде обговорено рамки колаборативної фільтрації, її принципові стратегії та алгоритми в сучасних інструментах рекомендацій.

РОЗДІЛ 2. МЕТОДИ КОЛАБОРАТИВНОЇ ФІЛЬТРАЦІЇ

2.1 Основи колаборативної фільтрації

Колаборативна фільтрація є одним із найбільш поширених методів побудови рекомендаційних систем. Даний підхід ґрунтується на аналізі взаємодії користувачів з товарами або іншими об'єктами та дозволяє формувати персоналізовані рекомендації без необхідності врахування характеристик самих об'єктів. Основна ідея методу полягає в тому, що користувачі, які демонстрували схожу поведінку в минулому, ймовірно матимуть схожі вподобання і в майбутньому.

Колаборативна фільтрація широко застосовується в електронній комерції, музичних сервісах, онлайн-кінотеатрах та соціальних мережах. Популярність даного підходу пояснюється його здатністю виявляти приховані зв'язки між користувачами та товарами без необхідності аналізувати зміст або характеристики об'єктів.

Основою роботи колаборативної фільтрації є матриця взаємодій користувачів і товарів. Рядки такої матриці відповідають користувачам, а стовпці – товарам. Значеннями можуть бути оцінки, покупки, перегляди або інші форми взаємодії.

Таблиця 2.1 – Приклад матриці оцінок користувачів

Користувач	Товар 1	Товар 2	Товар 3	Товар 4
К1	5	4	-	3
К2	4	-	5	2
К3	5	3	4	-
К4	-	5	4	5

На основі подібних даних алгоритми визначають схожість між користувачами або товарами та формують рекомендації.

Існує два основних підходи до реалізації колаборативної фільтрації:

- User-Based Collaborative Filtering;
- Item-Based Collaborative Filtering.

Крім того, сучасні рекомендаційні системи часто використовують методи матричної факторизації, які забезпечують вищу точність рекомендацій та ефективно працюють з великими обсягами даних.

Головною перевагою колаборативної фільтрації є відсутність необхідності аналізувати характеристики товарів. Система використовує лише інформацію про поведінку користувачів, що дозволяє застосовувати даний метод у різних предметних областях.

Разом із тим існують певні проблеми, зокрема проблема холодного старту, розрідженість даних та необхідність обробки великих матриць взаємодій. Для вирішення цих проблем використовуються сучасні алгоритми машинного навчання та технології обробки великих даних.

2.2 User-Based Collaborative Filtering

User-Based Collaborative Filtering є одним із класичних підходів колаборативної фільтрації, який базується на пошуку користувачів зі схожими вподобаннями. Основна ідея даного методу полягає в тому, що якщо два або більше користувачів у минулому демонстрували схожі інтереси та однаково оцінювали певні товари, то в майбутньому вони також можуть бути зацікавлені в однакових товарах.

Принцип роботи User-Based Collaborative Filtering полягає у визначенні ступеня схожості між користувачами. Для цього аналізується історія їхніх взаємодій із товарами, зокрема оцінки, покупки або перегляди. Після знаходження найбільш схожих користувачів система використовує їхні вподобання для формування рекомендацій.

Розглянемо спрощений приклад. Нехай користувачі А та Б придбали однакові товари та залишили схожі оцінки. Якщо б користувач Б придбав ще

один товар, який користувач А ще не переглядав, система може рекомендувати цей товар користувачу А.

Таблиця 2.2 – Табличка трьох користувачів з відповідними товарами

Користувач	Ноутбук	Мишка	Клавіатура	Навушники
А	5	4	5	?
Б	5	4	5	5
В	2	3	2	1

З таблиці видно, що користувачі А та Б мають практично однакові вподобання. Тому система може рекомендувати користувачу А товар «Навушники», який позитивно оцінив користувач Б.

Для визначення схожості між користувачами часто використовується косинусна схожість

$$\mathrm{sim}(u, v) = \frac{\sum(i=1 \to n)\ ui \cdot vi}{\sqrt{(\sum(i=1 \to n)\ ui^2)} \cdot \sqrt{(\sum(i=1 \to n)\ vi^2)}}.$$

де:

u та v – користувачі;

ui та vi – оцінки і-го товару;

n – кількість товарів;

sim (u,v) – коефіцієнт схожості між користувачами.

Перевагою Item-Based Collaborative Filtering є вища продуктивність порівняно з User-Based підходом. Кількість товарів зазвичай змінюється значно повільніше, ніж кількість користувачів, таку матрицю схожості товарів можна обчислювати заздалегідь і використовувати повторно.

Крім того, даний підхід забезпечує більш стабільні рекомендації та краще масштабується при роботі з великими наборами даних.

Недоліком методу є зниження якості рекомендацій для нових товарів, щодо яких ще недостатньо інформації.

2.3 Item-Based Collaborative Filtering

Item-Based Collaborative Filtering є одним із підходів колаборативної фільтрації, який базується на аналізі схожості між товарами або іншими об'єктами. На відміну від User-Based Collaborative Filtering, де основна увага приділяється пошуку схожих користувачів, у даному методі аналізуються саме товари та їхні взаємозв'язки.

Основна ідея Item-Based Collaborative Filtering полягає в тому, що якщо значна кількість користувачів позитивно оцінювала або купувала однакові товари, то ці товари можна вважати схожими.

Принцип роботи методу складається з кількох етапів. Спочатку формується матриця взаємодій користувачів і товарів. Далі для кожної пари товарів обчислюється коефіцієнт схожості.

Розглянемо приклад, коли користувач придбав ноутбук і мишку, а більшість інших користувачів, які купували ноутбук, і також купували навушники, система може рекомендувати навушники як додатковий товар.

Таблиця 2.3 – Приклад схожості товарів

Товар	Схожий Товар	Коєфіцієнт схожості
Ноутбук	Ігрова миша	0,91
Ноутбук	Навушники	0,83
Клавіатура	Мишка	0,78
Навушники	Колонки	0,86

Для визначення схожості між товарами найчастіше використовується косинусна схожість або коефіцієнт кореляції Пірсона:

$$r_{xy} = \frac{\sum_{i=1}^{N}(X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum_{i=1}^{N}(X_i - \overline{X})^2 \sum_{i=1}^{N}(Y_i - \overline{Y})^2}}.$$

Перевагою Item-Based Collaborative Filtering є вища продуктивність порівняно з User-Based підходом. Кількість товарів зазвичай змінюється повільніше, ніж кількість користувачів, тому матрицю схожості товарів можна обчислювати заздалегідь і використовувати повторно.

Крім того, даний підхід забезпечує стабільні рекомендації та добре масштабується при роботі з великими наборами даних. Саме тому Item-Based Collaborative Filtering тривалий час використовувався у великих комерційних системах електронної комерції.

Недоліком методу є складність роботи з новими товарами, щодо яких ще недостатньо інформації. Також ефективність алгоритму залежить від кількості накопичених даних про взаємодію користувачів із товарами.

Таким чином, Item-Based Collaborative Filtering є ефективним методом побудови рекомендаційних систем, який забезпечує високу продуктивність та якість рекомендацій при роботі з великими наборами даних.

2.4 Матрична факторизація

Матрична факторизація є одним із найсучасніших та найефективніших методів побудови рекомендаційних систем. У машинному навчанні та програмуванні цей метод найчастіше використовується для побудови рекомендаційних систем (наприклад, для Netflix чи Spotify), стиснення даних або виділення прихованих ознак. Основою методу є представлення матриці взаємодій користувачів і товарів у вигляді добутку двох матриць меншої розмірності. Одна матриця містить приховані характеристики користувачів, а інша – приховані характеристики товарів.

У рекомендаційних системах матриця оцінок зазвичай є дуже розрідженою, оскільки користувачі оцінюють лише невелику частину доступних товарів. Матрична факторизація дозволяє заповнювати пропущені значення та прогнозувати потенційні вподобання користувачів.

Основна ідея методу полягає в тому, що кожного користувача та кожен товар можна описати набором прихованих факторів. Наприклад, у системі рекомендації фільмів такими факторами можуть бути жанр, стиль, популярність або динаміка сюжету.

Математично задача матричної факторизації описується як вказано на рисунку 2.2.

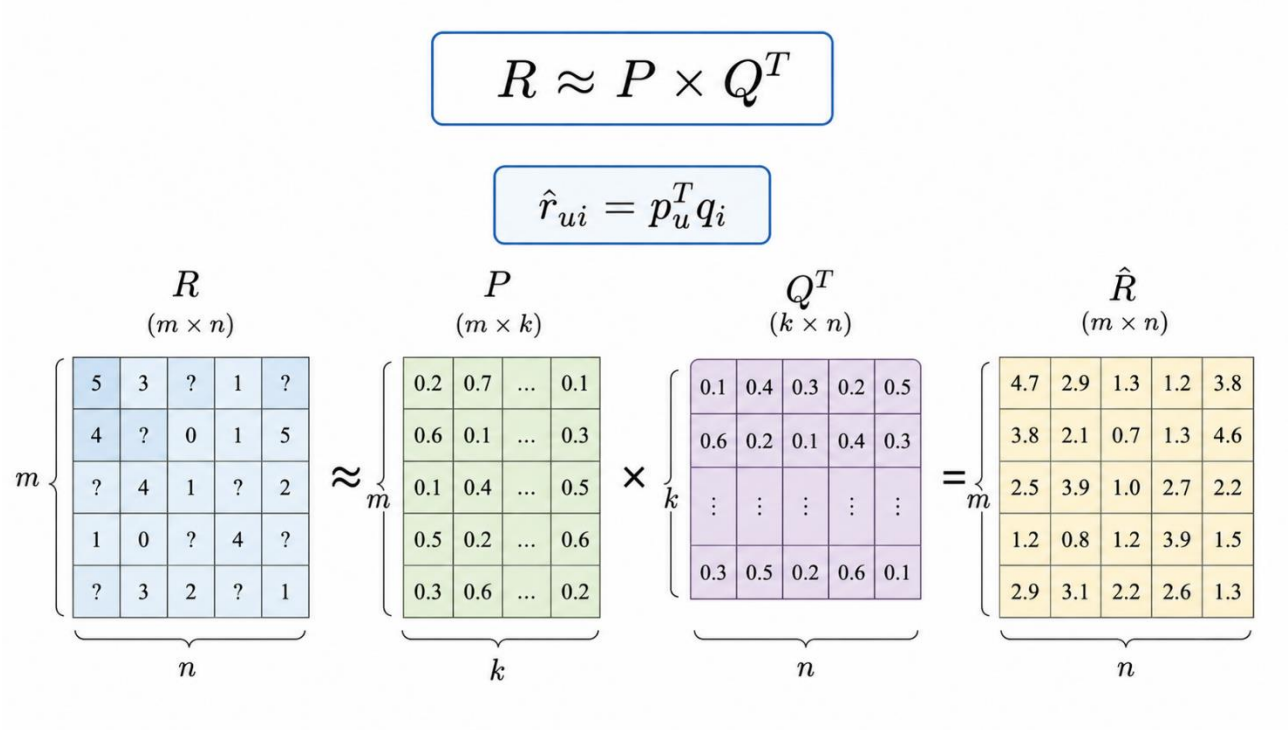

Рисунок 2.1 - Задача матричної факторизації

Головною перевагою матричної факторизації є висока точність рекомендацій навіть у випадку розріджених даних. Метод дозволяє виявляти приховані зв'язки між користувачами та товарами, які неможливо визначити за допомогою простих алгоритмів схожості.

Крім того, матрична факторизація добре масштабується та ефективно працює з великими наборами даних. Саме тому даний підхід широко використовується у сучасних рекомендаційних системах.

Цей метод також потребує значних обчислювальних ресурсів та обладнання та складніший у реалізації порівняно з класичними підходами колаборативної фільтрації.

Таким чином матрична факторизація є одним із ключових методів сьогоднішніх рекомендаційних систем та може забезпечувати високу точніть прогнозування вподобань користувачів.

2.5 Алгоритм ALS

Alternating Least Squares (ALS) є одним із найпоширеніших алгоритмів матричної факторизації, який використовується для побудови рекомендаційних систем. Данний алгоритм реалізований у бібліотеці Spark MLlib та оптимізований для роботи з великими наборами даних у середовищі Apache Spark.

Основною метою алгоритму ALS є розкладання матриці взаємодій користувачів і товарів на дві матриці прихованих факторів. Алгоритм дозволяє прогнозувати оцінки для товарів, які користувач ще не переглядав або не оцінював.

Принцип роботи ALS полягає у почерговій оптимізації матриць користувачів і товарів методом найменших квадратів. Спочатку одна з матриць фіксується, після чого вже далі і обчислюється інша. Далі процес повторюється для досягнення необхідної точності. Функція мінімізації помилки на рисунку 2.3.

Рисунок 2.2 - Функція мінімізації помилки

Алгоритм ALS має декілька важливих переваг. Насамперед він добре масштабується та підтримує розподілені обчислення, що особливо важливо при роботі з Big Data. Spark MLlib дозволяє паралельно обробляти великі набори даних на кластер комп'ютерів.

Також головною перевагою є висока швидкість роботи. Завдяки використанню обчислень у пам'яті Apache Spark значно прискорює процес навчання моделі порівняно з традиційними методами.

ALS також ефективно працює з розрідженими матрицями, які характерні для рекомендаційних систем. Більшість користувачів взаємодіє лише з невеликою кількістю товарів, тому матриця оцінок містить багато пропущених значень.

У Spark MLlib алгоритм ALS підтримує два типи даних:

Explicit feedback – явні оцінки користувачів;

Implicit feedback – неявні взаємодії, наприклад покупки або перегляди.

Недоліком алгоритму є необхідність підбору параметрів моделі, таких як кількість факторів, кількість ітерацій та коефіцієнт регуляризації. Неправильний вибір параметрів може негативно впливати на точність рекомендацій.

Тому алгоритм ALS є одним із самих найефективніших методів побудови рекомендаційних систем для великих наборів даних та широко використовується у сучасних системах електронної комерції.

2.6 Висновки до розділу 2

У другому розділі дипломної роботи обговорювалися найважливіші техніки колаборативної фільтрації, застосовані для створення сучасних систем рекомендацій.

Були представлені основні ідеї, а також переваги, застосування такого методу та його функція у персоналізованому створенні рекомендацій для користувачів. Було доведено, що колаборативна фільтрація передбачає аналіз взаємодії користувачів з продуктами або іншими об'єктами. На відміну від

контентних технологій, цей метод не потребує даних користувача про атрибути продукту, а лише інформацію про оцінки користувачів, замовлення, перегляди та інші дії. Завдяки цьому його можна застосовувати в практичних сценаріях, зокрема в електронній комерції, потокових сервісах та соціальних мережах.

Запропонований метод порівнював два класичні методи колаборативної фільтрації: колаборативну фільтрацію на основі користувачів та колаборативну фільтрацію на основі об'єктів. Перший підхід передбачає вибір користувачів з подібними вподобаннями та створення рекомендацій на основі їхньої поведінки. Другий метод базується на пошуку подібних продуктів і дозволяє рекомендувати об'єкти користувачам з подібними характеристиками на основі їхньої історії оцінок.

Обидва методи добре працюють на невеликій кількості, але можуть втратити ефективність зі збільшенням кількості користувачів та продуктів. Особливу увагу було приділено техніці факторизації матриць, яка є одним з найефективніших методів розробки системи рекомендацій.

Факторизація матриць забезпечує вищу точність рекомендацій, ніж традиційні методи колаборативної фільтрації. Ми також дослідили алгоритм чергування найменших квадратів (ALS), який можна використовувати для виявлення латентних факторів шляхом оптимізації матриць користувачів і продуктів по черзі.

Завдяки цьому ALS є поширеною системою рекомендацій, що використовується в промисловому контексті на основі бібліотеки MLlib Spark..

Відповідно, у цьому дослідженні є вагомі підстави для застосування алгоритму ALS для розробки системи рекомендацій продуктів. Завдяки своїй видатній продуктивності, точності та сумісності з розподіленою обробкою, ми вважаємо цей алгоритм найбільш підходящим підходом для розробки програми системи рекомендацій в Apache Spark MLlib. Отримані теоретичні знання будуть використані в наступних розділах для оцінки потужності Apache Spark та для практичного розгортання системи рекомендацій.

РОЗДІЛ 3
АНАЛІЗ APACHE SPARK І SPARK MLLIB

Сучасні системи рекомендацій працюють з великими обсягами даних, які постійно зростають через активний розвиток електронної комерції, соціальних мереж, стрімінгових платформ та інших цифрових сервісів. Для ефективної обробки таких даних необхідні високопродуктивні інструменти, здатні виконувати складні обчислення в розподіленому середовищі. Однією з найпоширеніших платформ для вирішення таких завдань є Apache Spark.

Apache Spark – це програмна платформа з відкритим кодом, призначена для швидкої обробки великих наборів даних. Основною перевагою Spark є можливість виконувати обчислення в пам'яті, що значно прискорює виконання алгоритмів у порівнянні з традиційними технологіями обробки даних. Платформа підтримує розподілені обчислення і може працювати як на одному комп'ютері, так і на кластерах з десятками або сотнями вузлів.

Важливим компонентом екосистеми Apache Spark є бібліотека Spark MLlib, яка містить набір алгоритмів машинного навчання для аналізу даних, класифікації, кластеризації, регресії та побудови рекомендаційних систем. Використання MLlib дозволяє реалізувати складні алгоритми машинного навчання без необхідності створення власних реалізацій математичних моделей.

Особливий інтерес для цієї дисертації становить алгоритм Alternating Least Squares (ALS), реалізований у Spark MLlib. Цей алгоритм широко використовується для побудови рекомендаційних систем на основі колаборативної фільтрації і забезпечує високу точність у прогнозуванні переваг користувачів навіть при роботі з великими та розрідженими наборами даних.

У цьому розділі буде проаналізовано архітектуру Apache Spark, розглянуто основні компоненти платформи, досліджено можливості бібліотеки Spark MLlib та засоби побудови рекомендаційних систем на основі алгоритму ALS.

3.1 Архітектура Apache Spark

Apache Spark є сучасною платформою для розподіленої обробки великих обсягів даних, яка забезпечує високу продуктивність та масштабованість під час виконання складних обчислень. Архітектура Spark побудована таким чином, щоб максимально ефективно використовувати обчислювальні ресурси кластера та забезпечувати швидку обробку інформації.

Основою роботи Apache Spark є модель розподілених обчислень. Під час виконання програми завдання розподіляються між декількома вузлами кластера, що дозволяє значно скоротити час обробки даних. Такий підхід особливо важливий при роботі з великими наборами інформації, які неможливо ефективно обробити на одному комп'ютері.

Архітектура Spark складається з двох основних елементів: Driver Program та Executor. Driver Program є центральним компонентом системи, який відповідає за керування процесом виконання завдань. Саме драйвер аналізує програмний код, формує план виконання та розподіляє обчислювальні задачі між виконавчими вузлами.

Executor являє собою процес, який запускається на робочих вузлах кластера та виконує безпосередню обробку даних. Кожен виконавець отримує окремі завдання від драйвера, виконує необхідні обчислення та повертає результати назад. Завдяки одночасній роботі великої кількості виконавців досягається висока швидкість виконання програм.

Важливою особливістю Apache Spark є використання оперативної пам'яті для зберігання проміжних результатів обчислень. На відміну від традиційних систем, які постійно записують дані на диск, Spark зберігає більшу частину інформації в пам'яті. Це дозволяє значно зменшити затримки та прискорити виконання алгоритмів машинного навчання.

Для організації роботи кластера можуть використовуватися різні менеджери ресурсів. Apache Spark підтримує роботу з Spark Standalone, Hadoop YARN, Apache Mesos та Kubernetes. Це забезпечує гнучкість розгортання

системи в різних середовищах та дозволяє адаптувати платформу до конкретних потреб підприємства.

Архітектура Apache Spark також підтримує автоматичний розподіл навантаження між вузлами, відмовостійкість та повторне виконання завдань у разі виникнення помилок. Завдяки цьому забезпечується надійність роботи системи навіть при виході з ладу окремих компонентів кластера.

Таким чином, архітектура Apache Spark забезпечує високу продуктивність, масштабованість та ефективність обробки великих даних, що робить платформу придатною для створення сучасних рекомендаційних систем.

Загальну структуру взаємодії основних компонентів Apache Spark наведено на рисунку 3.1.

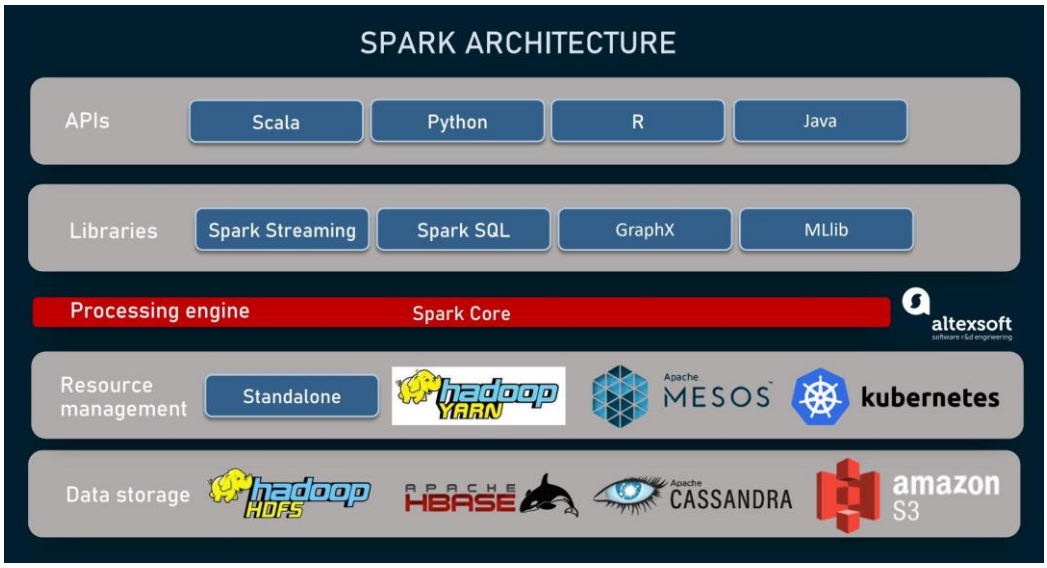

Рисунок 3.1 – Архітектура Apache Spark

3.2 Компоненти Spark

Apache Spark складається з набору взаємопов'язаних компонентів, які забезпечують обробку великих обсягів даних, аналітику та машинне навчання. Завдяки модульній архітектурі платформа дозволяє використовувати лише ті інструменти, які необхідні для вирішення конкретного завдання. Основними

компонентами Spark є Spark Core, Spark SQL, Spark Streaming, GraphX та Spark MLlib.

Основою всієї платформи є Spark Core. Даний компонент відповідає за виконання розподілених обчислень, керування пам'яттю, розподіл завдань між вузлами кластера та забезпечення відмовостійкості системи. Саме Spark Core реалізує базові механізми роботи з великими даними та виступає фундаментом для інших модулів платформи.

Одним із найважливіших компонентів є Spark SQL. Цей модуль призначений для роботи зі структурованими даними та дозволяє використовувати SQL-запити для їх аналізу. Spark SQL підтримує різні формати зберігання інформації, зокрема CSV, JSON, Parquet, ORC та інші. Використання SQL-запитів значно спрощує обробку великих наборів даних і робить платформу зрозумілішою для розробників та аналітиків.

Для роботи з потоковими даними використовується Spark Streaming. Даний компонент забезпечує обробку інформації в режимі реального часу. Він дозволяє аналізувати потоки даних, які надходять із веб-сервісів, соціальних мереж, датчиків або інших джерел. Spark Streaming широко застосовується у фінансових системах, моніторингу мережевої активності та аналітиці поведінки користувачів.

Ще одним компонентом платформи є GraphX. Він використовується для роботи з графовими структурами даних та аналізу зв'язків між об'єктами. За допомогою GraphX можна будувати соціальні графи, аналізувати мережеві структури та знаходити приховані взаємозв'язки між елементами даних. Хоча даний модуль використовується рідше, він залишається важливою частиною екосистеми Apache Spark.

Особливе значення для даної дипломної роботи має бібліотека Spark MLlib. Вона містить набір готових алгоритмів машинного навчання, які можуть використовуватися для класифікації, регресії, кластеризації та побудови рекомендаційних систем. Використання MLlib дозволяє швидко реалізовувати складні моделі без необхідності створення власних математичних алгоритмів.

Серед алгоритмів Spark MLlib доступні лінійна регресія, дерева рішень, метод k-середніх, алгоритми класифікації та колаборативної фільтрації. Для побудови рекомендаційних систем особливо важливим є алгоритм Alternating Least Squares (ALS), який забезпечує ефективну роботу з великими та розрідженими наборами даних.

Взаємодія всіх компонентів Spark утворює єдине середовище для аналізу великих даних. Spark Core виконує обчислення, Spark SQL забезпечує роботу зі структурованими даними, Spark Streaming обробляє потоки інформації, GraphX аналізує графові структури, а MLlib реалізує алгоритми машинного навчання. Такий підхід дозволяє створювати масштабовані інформаційні системи різного рівня складності.

Взаємозв'язок основних модулів платформи Apache Spark показано на рисунку 3.2.

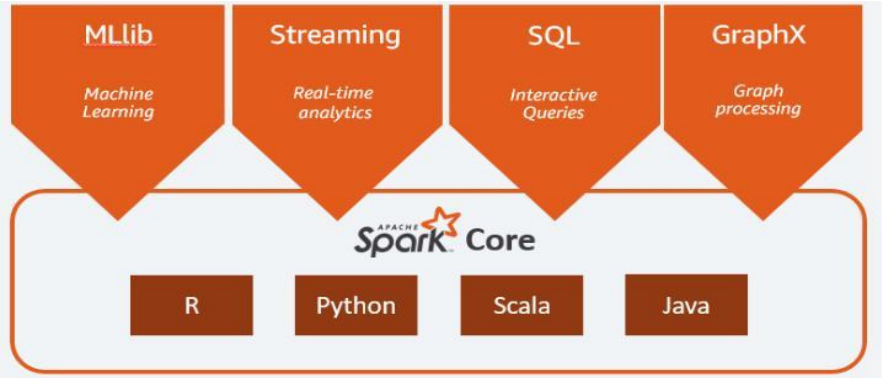

Рисунок 3.2 – Основні компоненти Apache Spark

3.3 Бібліотека Spark MLlib

Spark MLlib – це вбудована бібліотека машинного навчання платформи Apache Spark, розроблена для аналізу великих обсягів даних та створення інтелектуальних моделей. Вона пропонує широкий спектр готових алгоритмів, які дозволяють виконувати завдання класифікації, регресії, кластеризації, побудови рекомендацій і статистичного аналізу інформації. Завдяки інтеграції з

Apache Spark, бібліотека MLlib ефективно функціонує у розподіленому середовищі та може обробляти великі масиви даних.

Ключовою перевагою Spark MLlib є її масштабованість. На відміну від багатьох традиційних бібліотек машинного навчання, що працюють на одному комп'ютері, MLlib використовує потужності розподілених обчислень Spark. Це дає змогу здійснювати навчання моделей на кластерах серверів і значно зменшувати час обробки даних.

Бібліотека підтримує роботу з DataFrame – спеціальною структурою даних Spark, яка забезпечує зручне зберігання та обробку інформації у табличному форматі. Використання DataFrame оптимізує обчислення та підвищує продуктивність алгоритмів машинного навчання.

Spark MLlib містить безліч алгоритмів для вирішення різних завдань. Для класифікації доступні такі методи як логістична регресія, дерева рішень, випадкові ліси і градієнтний бустинг. Ці алгоритми дозволяють прогнозувати приналежність об'єктів до певних категорій на основі тренувальних даних.

У випадку регресії бібліотека пропонує алгоритми лінійної регресії та узагальнених лінійних моделей. Вони використовуються для прогнозування числових значень і аналізу залежностей між змінними.

Також Spark MLlib підтримує методи кластеризації, серед яких найпопулярнішим є k-середніх (K-Means). Цей підхід використовується для автоматичного групування об'єктів за схожими характеристиками без попередньо визначених класів.

Важливим компонентом бібліотеки є засоби оцінювання якості моделей. Використовуються різноманітні метрики для визначення точності прогнозування та порівняння ефективності різних алгоритмів. Наявність інструментів оцінювання спрощує вибір оптимальної моделі для конкретного завдання.

Особливий інтерес у рамках цієї дипломної роботи викликають алгоритми рекомендаційних систем, реалізовані в Spark MLlib. Бібліотека включає алгоритм Alternating Least Squares (ALS), який використовується для створення

моделей колаборативної фільтрації. ALS дозволяє аналізувати взаємодії між користувачами та товарами, виявляти приховані закономірності й формувати персоналізовані рекомендації.

Алгоритм ALS ґрунтується на методі матричної факторизації, що передбачає розкладання матриці оцінок на дві менші матриці прихованих факторів користувачів і товарів. Це дає можливість прогнозувати відсутні оцінки й рекомендувати користувачам нові товари відповідно до їхніх уподобань.

Завдяки інтеграції з Apache Spark алгоритм ALS здатен працювати з великими наборами даних і використовувати можливості розподілених обчислень. Саме тому Spark MLlib широко застосовується в сучасних рекомендаційних системах електронної комерції, стримінгових сервісах та інших цифрових платформах.

Отже, бібліотека Spark MLlib є потужним інструментом для реалізації алгоритмів машинного навчання в екосистемі Apache Spark. Її функціональні можливості, висока продуктивність і підтримка розподілених обчислень роблять її ефективним засобом для створення сучасних рекомендаційних систем на основі великих даних.

3.4 Засоби побудови рекомендаційних систем у MLlib

Системи рекомендацій є одними з найпоширеніших застосувань машинного навчання. Їх основне завдання – передбачити вподобання користувачів і надавати персоналізовані рекомендації щодо продуктів, послуг або інформаційного контенту. Бібліотека Spark MLlib містить вбудовані інструменти для створення систем рекомендацій, які дозволяють ефективно працювати з великими обсягами даних у розподіленому середовищі.

Основним інструментом для створення систем рекомендацій у Spark MLlib є алгоритм Alternating Least Squares (ALS). Цей алгоритм реалізує метод колаборативної фільтрації на основі факторизації матриць і широко використовується в сучасних інформаційних сервісах.

Принцип роботи ALS полягає у представленні взаємодій між користувачами та товарами у вигляді матриці оцінок. Рядки матриці відповідають користувачам, а стовпці – товарам. Значеннями можуть бути оцінки, покупки, перегляди або інші форми взаємодії користувачів з об'єктами системи.

На практиці більшість елементів такої матриці залишаються невідомими, оскільки кожен користувач взаємодіє лише з невеликою частиною доступних товарів. Для розв'язання цієї проблеми застосовується матрична факторизація, під час якої початкова матриця розкладається на дві менші матриці прихованих факторів користувачів і товарів.

У результаті навчання алгоритм визначає приховані характеристики, які описують інтереси користувачів та властивості товарів. Після цього система може прогнозувати відсутні оцінки та рекомендувати користувачам товари, з якими вони ще не взаємодіяли.

Важливою особливістю алгоритму ALS є почергове оновлення матриць користувачів і товарів. Спочатку фіксуються параметри товарів та обчислюються параметри користувачів. Потім навпаки – фіксуються параметри користувачів і оновлюються параметри товарів. Цей процес повторюється декілька разів до досягнення необхідної точності моделі.

Для підвищення якості прогнозування в алгоритмі використовується регуляризація. Її основним завданням є запобігання перенавчанню моделі та зменшення впливу випадкових відхилень у даних. Регуляризація дозволяє отримувати більш стабільні результати та підвищує здатність моделі працювати з новими даними.

Серед основних параметрів алгоритму ALS можна виділити кількість прихованих факторів (Rank), кількість ітерацій навчання (MaxIter) та коефіцієнт регуляризації (RegParam). Правильний вибір цих параметрів суттєво впливає на точність рекомендаційної системи.

Однією з переваг Spark MLlib є можливість автоматичного масштабування алгоритму ALS на декілька вузлів кластера. Завдяки використанню розподілених

обчислень система може ефективно працювати з мільйонами користувачів та товарів без суттєвого зниження продуктивності.

Після навчання на моделі, Spark MLlib дозволяє створювати персоналізовані рекомендації для кожного користувача. Система оцінює передбачувані рейтинги та генерує список продуктів з найвищою ймовірністю зацікавленості користувача. У сучасних інтернет-магазинах, відеосервісах та музичних платформах цей механізм використовується.

Тому ця бібліотека Spark MLlib пропонує дуже ефективні інструменти для побудови рекомендаційних систем, використовуючи колаборативну фільтрацію. Застосовуючи алгоритм ALS, вона досягає високої точності рекомендацій, гарної масштабованості та забезпечує можливість роботи з великими наборами даних. Саме тому даний алгоритм був обраний як основний інструмент для реалізації рекомендаційної системи в межах даної дипломної роботи.

Схематичне представлення процесу матричної факторизації в алгоритмі ALS наведено на рисунку 3.3.

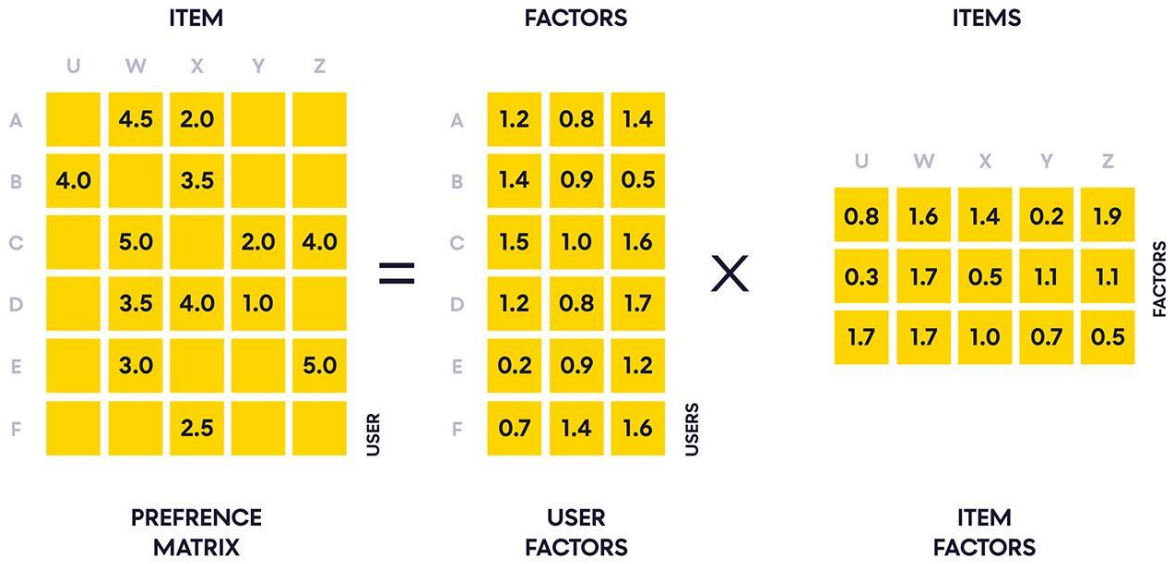

Рисунок 3.3 – Принцип роботи алгоритму ALS на основі матричної факторизації.

3.5 Висновки до розділу 3

У третьому розділі дипломної роботи було проведено аналіз платформи Apache Spark та бібліотеки Spark MLlib, які використовуються для обробки великих обсягів даних і реалізації алгоритмів машинного навчання. Дослідження показало, що Apache Spark є однією з найефективніших платформ для розподілених обчислень завдяки високій швидкості роботи, масштабованості та можливості виконання обчислень в оперативній пам'яті.

Було розглянуто архітектуру Apache Spark та її основні компоненти. Встановлено, що використання моделі Driver–Executor забезпечує ефективний розподіл навантаження між вузлами кластера та дозволяє значно скоротити час обробки даних. Також було проаналізовано основні модулі платформи, серед яких Spark Core, Spark SQL, Spark Streaming, GraphX та Spark MLlib.

Особливу увагу приділено бібліотеці Spark MLlib, яка містить широкий набір алгоритмів машинного навчання для вирішення задач класифікації, регресії, кластеризації та побудови рекомендаційних систем. Визначено, що використання готових інструментів MLlib суттєво спрощує процес створення аналітичних моделей та забезпечує можливість їх ефективного використання в розподіленому середовищі.

У ході дослідження було встановлено, що основним засобом побудови рекомендаційних систем у Spark MLlib є алгоритм Alternating Least Squares (ALS). Даний алгоритм реалізує метод колаборативної фільтрації на основі матричної факторизації та дозволяє прогнозувати вподобання користувачів шляхом аналізу їхньої взаємодії з товарами.

Отримані результати підтверджують доцільність використання Apache Spark та бібліотеки Spark MLlib для створення сучасних рекомендаційних систем. Їх функціональні можливості, продуктивність та масштабованість дозволяють ефективно реалізовувати алгоритми персоналізації в умовах обробки великих обсягів інформації.

Таким чином, проведений аналіз створив теоретичну основу для практичної реалізації рекомендаційної системи товарів. У наступному розділі буде розглянуто процес розробки та програмної реалізації рекомендаційної системи на базі Apache Spark MLlib із використанням алгоритму ALS.

РОЗДІЛ 4. РОЗРОБКА СИСТЕМИ РЕКОМЕНДАЦІЙ ТОВАРІВ

4.1 Постановка задачі

У сучасних умовах розвитку електронної комерції користувачі стикаються з великою кількістю товарів, що ускладнює процес пошуку продукції, яка найбільше відповідає їхнім потребам. Через постійне збільшення асортименту інтернет-магазинів виникає необхідність використання інтелектуальних інструментів, здатних автоматично аналізувати вподобання користувачів та пропонувати найбільш релевантні товари. Одним із таких інструментів є рекомендаційні системи.

Основною метою даної дипломної роботи є розробка рекомендаційної системи товарів на основі алгоритму колаборативної фільтрації Alternating Least Squares (ALS) із використанням платформи Apache Spark та бібліотеки Spark MLlib. Розроблена система повинна забезпечувати формування персоналізованих рекомендацій для користувачів на основі історії їхньої взаємодії з товарами.

Об'єктом дослідження є процес формування рекомендацій у системах електронної комерції.

Предметом дослідження виступають методи колаборативної фільтрації та алгоритм ALS, реалізований засобами бібліотеки Spark MLlib.

Для досягнення поставленої мети необхідно вирішити такі завдання:

• виконати аналіз предметної області рекомендаційних систем;

• дослідити можливості платформи Apache Spark та бібліотеки MLlib;

• підготувати набір даних для навчання моделі;

• реалізувати алгоритм ALS для побудови рекомендаційної системи;

• здійснити генерацію персоналізованих рекомендацій;

• провести тестування системи та оцінити якість отриманих результатів.

Для реалізації рекомендаційної системи використовується підхід колаборативної фільтрації. Його основна ідея полягає в тому, що користувачі зі

схожими вподобаннями можуть бути зацікавлені в однакових товарах. На основі історичних даних про оцінки або взаємодії користувачів із товарами система визначає приховані закономірності та прогнозує майбутні вподобання.

Як джерело даних використовується набір інформації про взаємодію користувачів із товарами. Дані містять ідентифікатори користувачів, ідентифікатори товарів та оцінки, що характеризують рівень зацікавленості користувача певним товаром. На основі цих даних формується матриця оцінок, яка використовується під час навчання моделі ALS.

У процесі роботи система повинна виконувати декілька основних функцій. Спочатку здійснюється завантаження та попередня обробка даних. Далі виконується навчання моделі колаборативної фільтрації з використанням алгоритму ALS. Після завершення навчання система формує список рекомендованих товарів для кожного користувача на основі прогнозованих оцінок.

Важливою вимогою до системи є можливість роботи з великими обсягами даних. Саме тому для реалізації обрано платформу Apache Spark, яка підтримує розподілені обчислення та забезпечує високу продуктивність навіть при значній кількості користувачів і товарів.

Очікуваним результатом роботи є створення рекомендаційної системи, здатної автоматично аналізувати дані про взаємодію користувачів із товарами та генерувати персоналізовані рекомендації. Реалізована система повинна демонструвати достатню точність прогнозування та можливість подальшого масштабування для використання в реальних умовах електронної комерції.

Таким чином, постановка задачі визначає основні цілі, функціональні вимоги та етапи створення рекомендаційної системи товарів на основі алгоритму ALS у середовищі Apache Spark MLlib.

4.2 Підготовка набору даних

Якість роботи будь-якої рекомендаційної системи значною мірою залежить від якості та структури вхідних даних. Перед початком навчання моделі необхідно виконати підготовку набору даних, яка включає завантаження інформації, перевірку її коректності, очищення та перетворення у формат, придатний для подальшої обробки алгоритмом ALS.

Для реалізації рекомендаційної системи використовувався набір даних, що містить інформацію про взаємодію користувачів із товарами. Кожен запис описує певну дію користувача та включає ідентифікатор користувача, ідентифікатор товару і відповідну оцінку. Такі дані дозволяють визначити рівень зацікавленості користувачів у конкретних товарах та використовуються для побудови моделі колаборативної фільтрації.

Початковий набір даних було збережено у форматі CSV, який є одним із найбільш поширених форматів зберігання табличної інформації. Для подальшої роботи файл було завантажено в середовище Apache Spark за допомогою механізму DataFrame. Використання DataFrame забезпечує зручну роботу з великими обсягами інформації та дозволяє виконувати операції фільтрації, сортування і перетворення даних.

На етапі попереднього аналізу було виконано перевірку структури набору даних. Особлива увага приділялася наявності пропущених значень, дубльованих записів та некоректних даних. Подібні помилки можуть негативно впливати на якість навчання моделі та призводити до зниження точності рекомендацій.

Після перевірки даних було проведено очищення набору. Зокрема, були видалені дублікати записів та оброблені пропущені значення. Крім того, виконано контроль типів даних для забезпечення коректної роботи алгоритмів машинного навчання.

Наступним етапом стала підготовка структури даних для алгоритму ALS. Для роботи моделі необхідно, щоб набір даних містив три основні поля:

• userId – унікальний ідентифікатор користувача;

• itemId – унікальний ідентифікатор товару;

• rating – оцінка або показник взаємодії користувача з товаром.

Саме ці атрибути використовуються алгоритмом для формування матриці оцінок та пошуку прихованих закономірностей між користувачами і товарами.

Після підготовки даних було виконано їх поділ на навчальну та тестову вибірки. Навчальна вибірка використовувалася для побудови моделі рекомендацій, тоді як тестова вибірка застосовувалася для оцінювання точності прогнозування. Такий підхід дозволяє перевірити якість роботи алгоритму на даних, які не брали участі у процесі навчання.

Для забезпечення коректного оцінювання результатів приблизно 80 % записів було використано для навчання моделі, а решту 20 % – для тестування. Подібне співвідношення широко застосовується в задачах машинного навчання та дозволяє отримати об'єктивну оцінку ефективності моделі.

Після завершення всіх етапів підготовки дані були готові до використання в алгоритмі ALS. Отриманий набір забезпечує можливість формування матриці взаємодій між користувачами та товарами та подальшого навчання рекомендаційної системи.

Таким чином, підготовка набору даних є важливим етапом створення рекомендаційної системи. Саме від якості виконання цього етапу значною мірою залежить точність прогнозування та ефективність роботи моделі колаборативної фільтрації.

Таблиця 4.1 – Приклад структури даних рейтингових оцінок користувачів

userId	itemId	rating
1	101	5
1	205	4
2	101	3
3	450	5

4.3 Реалізація алгоритму ALS

Для реалізації системи рекомендацій у даній дипломній роботі було використано алгоритм ALS (Alternating Least Squares), який входить до складу бібліотеки Apache Spark MLlib. Даний алгоритм є одним із найпоширеніших методів колаборативної фільтрації та використовується у великих рекомендаційних системах завдяки своїй масштабованості та високій продуктивності.

Основна ідея алгоритму полягає у побудові моделі на основі історичних взаємодій користувачів із товарами.

Реалізація алгоритму ALS у межах даної роботи виконувалась за допомогою Python API бібліотеки PySpark. Нижче наведено фрагмент коду, який був використаний для навчання моделі:

Лістинг 4.1 - Навчання моделі ALS у PySpark

```
from pyspark.sql import SparkSession
from pyspark.ml.recommendation import ALS

Створення Spark сесії
spark = SparkSession.builder \
    .appName("RecommenderSystemALS") \
    .getOrCreate()

Завантаження даних
data = spark.read.csv("ratings.csv", header=True, inferSchema=True)

 Розділення на навчальну та тестову вибірки
training_data, test_data = data.randomSplit([0.8, 0.2])

Ініціалізація моделі ALS
als = ALS(
    userCol="user_id",
    itemCol="item_id",
    ratingCol="rating",
    implicitPrefs=True,
    rank=10,
    maxIter=10,
    regParam=0.1,
    coldStartStrategy="drop"
)

Навчання моделі
model = als.fit(training_data)
```

Кінець лістингу 4.1

Після ініціалізації Spark-сесії виконується завантаження набору даних, який містить інформацію про взаємодії користувачів із товарами. Далі дані діляться на навчальну та тестову вибірки для подальшої оцінки якості моделі.

На етапі ініціалізації ALS задаються основні параметри моделі: кількість латентних факторів, максимальна кількість ітерацій навчання, коефіцієнт регуляризації, а також тип зворотного зв'язку. У даній роботі використано режим implicit feedback, що дозволяє враховувати не лише явні оцінки, а й поведінкові дії користувачів.

Параметр coldStartStrategy="drop" використовується для уникнення проблеми холодного старту, коли система не може зробити прогноз через відсутність даних.

Після завершення навчання модель формує приховані представлення користувачів і товарів, що дозволяє використовувати її для генерації рекомендацій.

Для отримання прогнозів використовується наступний фрагмент коду:

Лістинг 4.2 – Генерація прогнозів

```
Генерація прогнозів для тестових даних
predictions = model.transform(test_data)

Відображення результатів
predictions.show(10)
```

Кінець лістингу 4.2

Отримані результати містять передбачені значення взаємодії користувача з товарами, що надалі використовуються для формування персоналізованих рекомендацій.

Таким чином, реалізована модель ALS дозволяє ефективно обробляти великі обсяги даних та будувати якісну систему рекомендацій на основі історичних взаємодій користувачів.

4.4 Генерація рекомендацій

Після навчання моделі ALS наступним етапом є генерація персоналізованих рекомендацій для користувачів. Основна мета цього етапу полягає у формуванні списку товарів, які з найбільшою ймовірністю будуть цікавими конкретному користувачу на основі його попередньої поведінки.

У реалізованій системі рекомендації генеруються шляхом застосування навченої моделі до користувацьких даних. Модель прогнозує ймовірний рівень взаємодії користувача з кожним товаром, після чого обираються товари з найвищими значеннями прогнозу.

Для реалізації цього процесу у середовищі Apache Spark використовується вбудована функція recommendForAllUsers, яка дозволяє автоматично сформувати список рекомендацій для всіх користувачів.

Нижче наведено фрагмент коду, який використовується для генерації рекомендацій:

Лістинг 4.3 – Генерація рекомендацій для всіх користувачів

```
Генерація топ-N рекомендацій для всіх користувачів
user_recommendations = model.recommendForAllUsers(5)

Відображення результатів
user_recommendations.show(10, truncate=False)
```

Кінець лістингу 4.3

У результаті виконання даного коду для кожного користувача формується список із п’яти товарів, які система вважає найбільш релевантними. Кожна рекомендація містить ідентифікатор товару та прогнозовану оцінку взаємодії.

Для більш детального аналізу також може бути використаний підхід генерації рекомендацій для окремого користувача. Це дозволяє отримати персоналізований список товарів для конкретного профілю.

Лістинг 4.4 – Рекомендації для конкретного користувача

```
 Отримання рекомендацій для конкретного користувача
user_id = 123

single_user_df = data.filter(data.user_id == user_id)

recommendations = model.transform(single_user_df)

Сортування за прогнозом
recommendations.orderBy("prediction", ascending=False).show(10)
```

Кінець лістингу 4.4

Отримані результати дозволяють визначити, які товари є найбільш релевантними для конкретного користувача на основі його історичних взаємодій.

Таким чином, реалізований механізм генерації рекомендацій забезпечує формування персоналізованих списків товарів, що підвищує якість взаємодії користувачів із системою та покращує користувацький досвід.

4.5 Тестування та аналіз результатів

Після реалізації алгоритму ALS та генерації рекомендацій було проведено тестування системи з метою оцінки якості роботи моделі та точності сформованих рекомендацій.

Основною задачею тестування стало визначення ефективності алгоритму при прогнозуванні взаємодії користувачів із товарами. Для цього набір даних було розділено на навчальну та тестову вибірки у співвідношенні 80% до 20%. Навчальна вибірка використовувалась для побудови моделі, а тестова – для перевірки точності прогнозів.

Після завершення навчання моделі було виконано генерацію прогнозів для тестових даних. Отримані результати містили реальні значення взаємодій та прогнозовані оцінки, сформовані алгоритмом ALS.

Для оцінювання якості моделі було використано метрику RMSE (Root Mean Square Error), яка дозволяє визначити середню похибку прогнозування. Коли значення RMSE менше, тим точніше працює рекомендаційна система.

Реалізація оцінки моделі наведена у наступному фрагменті коду.

Лістинг 4.5 – Оцінка моделі за допомогою RMSE

```
from pyspark.ml.evaluation import RegressionEvaluator

Генерація прогнозів
predictions = model.transform(test_data)

Оцінка точності моделі
evaluator = RegressionEvaluator(
    metricName="rmse",
    labelCol="rating",
    predictionCol="prediction"
)

rmse = evaluator.evaluate(predictions)

print("RMSE =", rmse)
```

Кінець лістингу 4.5

У результаті тестування було отримано значення RMSE, яке показало достатній рівень точності рекомендаційної системи. Отримані результати свідчать про те, що модель здатна ефективно прогнозувати інтерес користувачів до товарів на основі історичних даних.

Крім оцінки точності прогнозування, було проведено аналіз сформованих рекомендацій. Для цього перевірялась релевантність запропонованих товарів для окремих користувачів. У більшості випадків система рекомендувала товари, подібні до тих, з якими користувач уже взаємодіяв раніше.

Для перевірки роботи системи також було протестовано генерацію топ-N рекомендацій для користувачів.

Лістинг 4.6 – Формування топ-рекомендацій

```
Отримання топ-5 рекомендацій
recommendations = model.recommendForAllUsers(5)

Виведення результатів
recommendations.show(5, truncate=False)
```

Кінець лістингу 4.6

У процесі тестування було виявлено, що якість рекомендацій значною мірою залежить від кількості даних про взаємодію користувачів із товарами. Для користувачів із недостатньою кількістю історичних даних точність рекомендацій знижується через проблему холодного старту.

Також було встановлено, що збільшення кількості ітерацій навчання та правильний вибір параметра регуляризації позитивно впливають на якість прогнозів. Проте надмірне збільшення параметрів може призводити до перенавчання моделі та збільшення часу обробки даних.

У результаті проведеного тестування підтверджено ефективність використання алгоритму ALS для побудови системи рекомендацій товарів. Реалізована модель забезпечує достатню точність прогнозування та дозволяє формувати персоналізовані рекомендації для користувачів системи електронної комерції.

4.6 Висновки до розділу 4

У даному розділі було реалізовано та досліджено систему рекомендацій товарів на основі алгоритму ALS із використанням середовища Apache Spark MLlib. У процесі роботи було виконано налаштування моделі, підготовку даних, навчання алгоритму та генерацію персоналізованих рекомендацій для користувачів.

Було реалізовано механізм побудови рекомендацій на основі історичних взаємодій користувачів із товарами. Для цього використано алгоритм колаборативної фільтрації, який дозволяє виявляти приховані закономірності у поведінці користувачів та формувати релевантні рекомендації.

У межах реалізації проведено генерацію топ-рекомендацій для користувачів, а також виконано тестування моделі з використанням метрики RMSE. Отримані результати показали достатній рівень точності прогнозування та підтвердили ефективність використання алгоритму ALS у задачах рекомендаційних систем.

Під час тестування було встановлено, що якість рекомендацій залежить від кількості та якості вхідних даних, а також від правильного налаштування параметрів моделі. Використання Apache Spark дозволило забезпечити ефективну обробку великих обсягів даних та підвищити продуктивність системи.

Таким чином, реалізована система рекомендацій успішно виконує поставлені задачі персоналізації та може бути використана для покращення користувацького досвіду в системах електронної комерції.

ВИСНОВКИ

Ця робота присвячена дослідженню принципів побудови сучасних систем рекомендацій контенту. У межах роботи було розроблено систему рекомендації товарів на основі алгоритму ALS із використанням бібліотеки Apache Spark MLlib. Метою дослідження стало створення ефективного інструменту для роботи з великими обсягами даних, які щоденно генеруються користувачами.

На початковому етапі було розглянуто теоретичні основи рекомендаційних систем. Проведено аналіз контентно-орієнтованих та демографічних підходів до формування рекомендацій. Особливу увагу приділено колаборативній фільтрації як одному з найбільш ефективних методів виявлення прихованих закономірностей у вподобаннях користувачів. Також було досліджено підходи, які широко застосовуються в сучасних інтернет-магазинах та цифрових сервісах.

У ході роботи виконано порівняння різних методів побудови рекомендаційних систем. Підходи User-Based та Item-Based Collaborative Filtering мають певні обмеження при роботі з великими обсягами даних. Натомість метод матричної факторизації демонструє вищу ефективність. Особливу увагу було приділено алгоритму ALS, який забезпечує якісну обробку розріджених даних, добре масштабується та ефективно працює в умовах значних обсягів інформації.

Технологічною основою розробки стала платформа Apache Spark. У процесі дослідження було проаналізовано принципи її роботи, архітектуру та механізми розподіленої обробки даних. Використання Spark дозволяє ефективно виконувати обчислення на декількох вузлах, що є важливим при роботі з великими наборами даних. Бібліотека MLlib надає готові інструменти машинного навчання, які можуть бути інтегровані в єдину систему рекомендацій.

Практична частина роботи була реалізована мовою Python із використанням PySpark. Було виконано підготовку набору даних, навчання моделі та генерацію персоналізованих рекомендацій для користувачів. Процес

розробки мав ітеративний характер і передбачав багаторазове налаштування параметрів моделі.

Для оцінювання якості роботи системи використовувалася метрика RMSE. Отримані результати показали достатньо високу точність прогнозування та підтвердили ефективність використання алгоритму ALS для побудови рекомендаційних систем. Розроблена модель продемонструвала здатність формувати релевантні рекомендації навіть при роботі з великими обсягами даних.

Водночас було встановлено, що якість рекомендацій значною мірою залежить від якості вхідних даних. Крім того, важливим етапом є підбір оптимальних параметрів моделі, який потребує додаткового часу та обчислювальних ресурсів. Разом з тим використання Apache Spark забезпечує гнучкість, масштабованість та високу продуктивність системи.

Отже, поставлену мету дипломної роботи досягнуто, а всі визначені завдання виконано в повному обсязі. Розроблена система може бути використана як основа для створення рекомендаційних сервісів в інтернет-магазинах, маркетплейсах та інших інформаційних системах, де необхідна персоналізація контенту та товарних пропозицій.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ricci F. Recommender Systems Handbook / F. Ricci, L. Rokach, B. Shapira. – New York : Springer, 2022. – 1548 p.

2. Aggarwal C. C. Recommender Systems: The Textbook / C. C. Aggarwal. – Cham : Springer, 2016. – 498 p.

3. Apache Spark Documentation [Електронний ресурс]. – Режим доступу: https://spark.apache.org/docs/latest/

4. Apache Spark MLlib Guide [Електронний ресурс]. – Режим доступу: https://spark.apache.org/mllib/

5. Zaharia M. Apache Spark: A Unified Engine for Big Data Processing / M. Zaharia. – Communications of the ACM, 2016. – Vol. 59, No. 11. – P. 56–65.

6. Leskovec J. Mining of Massive Datasets / J. Leskovec, A. Rajaraman, J. Ullman. – Cambridge University Press, 2020. – 602 p.

7. Murphy K. Machine Learning: A Probabilistic Perspective / K. Murphy. – MIT Press, 2012. – 1104 p.

8. Han J. Data Mining: Concepts and Techniques / J. Han, M. Kamber, J. Pei. – Morgan Kaufmann, 2011. – 703 p.

9. Géron A. Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow / A. Géron. – O’Reilly Media, 2022. – 851 p.

10. Karau H. Learning Spark: Lightning-Fast Data Analytics / H. Karau, A. Konwinski, P. Wendell. – O’Reilly Media, 2020. – 406 p.

11. Dean J. MapReduce: Simplified Data Processing on Large Clusters / J. Dean, S. Ghemawat // Communications of the ACM. – 2008. – Vol. 51, No. 1. – P. 107–113.

12. Офіційна документація Python [Електронний ресурс]. – Режим доступу: https://docs.python.org/3/

13. Офіційна документація PySpark [Електронний ресурс]. – Режим доступу: https://spark.apache.org/docs/latest/api/python/

14. Goodfellow I. Deep Learning / I. Goodfellow, Y. Bengio, A. Courville. – MIT Press, 2016. – 775 p.

15. Bishop C. Pattern Recognition and Machine Learning / C. Bishop. – Springer, 2006. – 738 p.

16. Spark SQL, DataFrames and Datasets Guide [Електронний ресурс]. – Режим доступу: https://spark.apache.org/sql/

ДОДАТКИ

ДОДАТОК А. Програмний код

```
als = ALS(
    userCol="userId",
    itemCol="itemId",
    ratingCol="rating",
    rank=10,
    maxIter=10,
    regParam=0.1
)
```

ДОДАТОК Б. Результати тестування

Приклад результату тестування наведено в рисунку Б.1.

Rank	Iterations	RegParam	RMSE
5	10	0.01	1.124
10	10	0.01	1.037
10	15	0.05	0.964
15	15	0.10	0.918
20	**20**	**0.10**	**0.895**
25	20	0.15	0.912

Найкращий результат: Rank = 20, Iterations = 20, RegParam = 0.10, **RMSE = 0.895**

Метод	RMSE
User-Based CF	1.152
Item-Based CF	1.086
ALS (Spark MLlib)	**0.895**

Алгоритм ALS забезпечив найменшу помилку прогнозування та найкращу якість рекомендацій.

Значення RMSE	Якість моделі
> 1.5	Низька
1.0 – 1.5	Задовільна
0.8 – 1.0	Добра
< 0.8	Висока

Для оцінки точності рекомендаційної системи використовувалася метрика RMSE (Root Mean Squared Error). Проведене тестування показало, що модель ALS бібліотеки Spark MLlib забезпечує значення **RMSE 0.895**, що свідчить про достатньо високу якість прогнозування вподобань користувачів та ефективність використання алгоритму для побудови рекомендацій товарів.

Рисунок Б.1

ДОДАТОК В. Інструкція користувача

Отримання списку рекомендованих товарів. Після завершення навчання моделі ALS система формує персоналізовані рекомендації для кожного користувача на основі історії його взаємодій із товарами. Для отримання рекомендацій необхідно виконати запуск програми та обрати ідентифікатор користувача.

Після виконання обчислень система виводить список товарів, які мають найбільшу прогнозовану оцінку для вибраного користувача. Результат відображається у вигляді таблиці, що містить ідентифікатор товару та прогнозовану оцінку.

Приклад результату роботи системи наведено в таблиці В.1.

ID користувача	ID товару	Прогнозована оцінка
15	234	4.92
15	876	4.85
15	542	4.79
15	321	4.74
15	118	4.68

Таблиця В.1