Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Штучного інтелекту
(повна назва)

# КВАЛІФІКАЦІЙНА РОБОТА
## Пояснювальна записка

рівень вищої освіти другий (магістерський)

Розробка рекомендаційної системи для інтернет-магазину з використанням гібридних підходів
(тема)

Виконав:
здобувач другого року навчання
групи ДСм-23-1
Антіпін В. С.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма Науки про дані (Data Science)
(повна назва спеціалізації)
Керівник доц. Дейнеко А. О.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ О. В. Золотухін
(підпис) (прізвище, ініціали)

2025 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

Рівень вищої освіти другий (магістерський)

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

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

Освітня програма Науки про дані (Data Science)
(повна назва)

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

# ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Антіпіну Вадиму Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка рекомендаційної системи для інтернет-магазину з використанням гібридних підходів

затверджена наказом університету від 22 листопада 20 24 р. № 1238Ст

2. Термін подання студентом роботи до екзаменаційної комісії 23 січня 20 25 р.

3. Вихідні дані до роботи набори даних з реальних ecommerce-платформ, користувацькі дані, моделі для побудови рекомендацій, алгоритм контентного фільтрування, алгоритм колаборативного фільтрування, алгоритм гібридного фільтрування, алгоритм пошуку, програмне забезпечення та бібліотеки, метрики для оцінки ефективності

4. Перелік питань, що потрібно опрацювати в роботі
1) Теоретичні основи
2) Методи та засоби дослідження
3) Розробка рекомендаційної системи
4) Есперементальні дослідження
5) Перспективи розвитку системи

**КАЛЕНДАРНИЙ ПЛАН**

| № | Назва етапів роботи | Терміни виконання етапів роботи | Примітка |
|---|---|---|---|
| 1 | Отримання завдання на кваліфікаційну роботу | 25.11.2024 | виконано |
| 2 | Аналіз завдання та формування структури роботи | 26.11.2024 – 27.11.2024 | виконано |
| 3 | Збір вихідних даних та огляд теоретичних основ | 28.11.2024 – 29.11.2024 | виконано |
| | рекомендаційних систем | | |
| 4 | Розробка методики побудови рекомендаційної | 30.11.2024 – 02.12.2024 | виконано |
| | системи | | |
| 5 | Підготовка даних для навчання моделей | 03.12.2024 – 04.12.2024 | виконано |
| 6 | Реалізація контентного та колаборативного | 05.12.2024 – 06.12.2024 | виконано |
| | фільтрування | | |
| 7 | Розробка гібридної системи рекомендацій | 07.12.2024 – 08.12.2024 | виконано |
| 8 | Тестування та оцінка якості моделей за метриками | 09.12.2024 – 10.12.2024 | виконано |
| 9 | Аналіз продуктивності системи та порівняння з | 11.12.2024 – 12.12.2024 | виконано |
| | альтернативними підходами | | |
| 10 | Підготовка графічного матеріалу та візуалізація | 13.12.2024 | виконано |
| | результатів | | |
| 11 | Оформлення текстової частини роботи | 14.12.2024 – 15.12.2024 | виконано |
| 12 | Перевірка та доопрацювання кваліфікаційної роботи | 16.12.2024 | виконано |
| 13 | Завершення виконання роботи | 16.12.2024 | виконано |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

Дата видачі завдання 25 листопада 20 24 р.

Здобувач _____
(підпис)

Керівник роботи _____ доц. Дейнеко А. О.
(підпис) (посада, прізвище, ініціали)

# РЕФЕРАТ

Пояснювальна записка: 121 с., 12 рис., 3 табл., 1 дод., 36 джерел.

ГІБРИДНА СИСТЕМА, ЕЛЕКТРОННА КОМЕРЦІЯ, КОЛАБОРАТИВНЕ ФІЛЬТРУВАННЯ, КОНТЕНТНЕ ФІЛЬТРУВАННЯ, ПОШУК ТОВАРІВ, ПРОДУКТИВНІСТЬ СИСТЕМИ, РЕКОМЕНДАЦІЙНА СИСТЕМА, ТОЧНІСТЬ РЕКОМЕНДАЦІЙ, ФІЛЬТРАЦІЯ ТОВАРІВ.

Об’єкт дослідження – процес персоналізації рекомендацій для користувачів інтернет-магазину з використанням сучасних алгоритмів та підходів.

Предмет дослідження – методи та алгоритми побудови рекомендаційних систем, включаючи контентне, колаборативне та гібридне фільтрування, а також методи оцінки ефективності системи.

Мета роботи – розробка гібридної рекомендаційної системи для інтернет-магазину, яка забезпечує високу точність, релевантність і адаптивність рекомендацій для різних типів користувачів та сценаріїв, включаючи «холодний старт».

Методи дослідження – теоретичний аналіз існуючих підходів до побудови рекомендаційних систем, моделювання алгоритмів на основі контентного та колаборативного фільтрування, тестування системи на реальних наборах даних, використання метрик Precision, Recall, F1, MAP, MRR, NDCG, RMSE для оцінки ефективності, а також експериментальний аналіз впливу вагових коефіцієнтів на якість рекомендацій.

# ABSTRACT

Master’s thesis contains: 121 pp., 12 fig., 3 tabl., 1 ann., 36 references.

COLLABORATIVE FILTERING, CONTENT FILTERING, ELECTRONIC COMMERCE, HYBRID SYSTEM, PERFORMANCE OF THE SYSTEM, PRECISION OF RECOMMENDATIONS, PRODUCT FILTERING, PRODUCT SEARCH, RECOMMENDATION SYSTEM.

Object of the study – the process of personalizing recommendations for users of an e-commerce platform using modern algorithms and approaches.

Subject of the study – methods and algorithms for building recommendation systems, including content-based, collaborative, and hybrid filtering, as well as methods for evaluating system efficiency.

Purpose of the work – to develop a hybrid recommendation system for an e-commerce platform that ensures high precision, relevance, and adaptability of recommendations for various user types and scenarios, including the «cold start» problem.

Research methods – theoretical analysis of existing approaches to building recommendation systems, modeling algorithms based on content-based and collaborative filtering, testing the system on real datasets, using metrics such as Precision, Recall, F1, MAP, MRR, NDCG, and RMSE to evaluate efficiency, as well as experimental analysis of the impact of weight coefficients on recommendation quality.

# ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів ..... 8
Вступ ..... 9
1 Теоретичні основи ..... 11
1.1 Огляд рекомендаційних систем ..... 11
1.2 Гібридні підходи в рекомендаційних системах ..... 13
1.3 Сучасні тенденції та виклики ..... 16
2 Методи та засоби дослідження ..... 24
2.1 Аналіз методів для побудови рекомендаційних систем ..... 24
2.1.1 Контентне фільтрування ..... 24
2.1.2 Колаборативне фільтрування ..... 26
2.1.3 Гібридний підхід ..... 29
2.1.4 Рекомендація категорій товарів ..... 31
2.1.5 Пошук і фільтрація товарів ..... 32
2.2 Інструменти та технології ..... 33
2.3 Розробка методології експериментів ..... 35
2.3.1 Джерела даних ..... 35
2.3.2 Методи підготовки даних ..... 37
2.3.3 Методика тестування моделей ..... 38
2.3.4 Оцінка продуктивності системи ..... 43
3 Розробка рекомендаційної системи ..... 46
3.1 Архітектура системи ..... 46
3.2 Реалізація контентного фільтрування ..... 49
3.3 Реалізація колаборативного фільтрування ..... 53
3.4 Реалізація гібридного підходу для рекомендації товарів ..... 57
3.5 Реалізація гибридного підходу для рекомендації категорій ..... 62
3.6 Реалізація механізму пошуку та фільтрації товарів ..... 66
3.7 Побудова API для взаємодії з веб-додатком ..... 68
3.8 Реалізація оновлення рекомендацій ..... 72

4 Експериментальні дослідження ..... 77
4.1 Постановка задачі для експериментального дослідження ..... 77
4.2 Підготовка даних для експериментів ..... 79
4.3 Оцінка точності системи рекомендацій товарів ..... 84
4.3.1 Оцінка точності контентної системи ..... 85
4.3.2 Оцінка точності колаборативної системи ..... 87
4.3.3 Оцінка точності гібридної системи ..... 89
4.4 Оцінка ефективності рекомендацій категорій товарів..... 95
4.5 Оцінка ефективності пошуку та фільтрації товарів ..... 99
4.6 Аналіз продуктивності системи ..... 101
4.7 Порівняння з альтернативними підходами ..... 103
4.8. Висновки за результатами експериментального дослідження..... 106
5 Перспективи розвитку системи..... 109
5.1 Порівняння з існуючими системами рекомендацій ..... 109
5.2 Перспективи розвитку розробленої системи ..... 111
Висновки..... 114
Перелік джерел посилання ..... 116
Додаток А Відомість кваліфікаційної роботи..... 121

# ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ШІ – штучний інтелект;

AI – Artificial Intelligence – штучний інтелект;

ALS – Alternating Least Squares – метод чергування найменших квадратів;

AP – Average Precision – середня точність;

AR – Augmented Reality – доповнена реальність;

BERT – Bidirectional Encoder Representations from Transformers – двонапрямлені подання кодувальника на основі трансформерів;

CSR – Compressed Sparse Row – стиснутий розріджений ряд;

Explainable AI (XAI) – Explainable Artificial Intelligence – пояснюваний штучний інтелект;

GNN – Graph Neural Networks – графові нейронні мережі;

MAE – Mean Absolute Error – середня абсолютна помилка;

MAP – Mean Average Precision – середня точність усереднена за всіма класами;

MRR – Mean Reciprocal Rank – середній обернений ранг;

NDCG – Normalized Discounted Cumulative Gain – нормалізований зважений кумулятивний приріст;

NCF – Neural Collaborative Filtering – нейронна колаборативна фільтрація;

REST – Representational State Transfer – передача репрезентативного стану;

RMSE – Root Mean Square Error – середньоквадратична помилка;

RNN – Recurrent Neural Networks – рекурентні нейронні мережі;

SVD – Singular Value Decomposition – сингулярний розклад матриці;

TF-IDF – Term Frequency – Inverse Document Frequency – частота терміну – обернена частота документа.

# ВСТУП

У сучасному світі електронна комерція стала невід'ємною частиною глобальної економіки, забезпечуючи мільйонам користувачів доступ до широкого асортименту товарів та послуг. Зростання конкуренції у цій галузі вимагає від платформ не лише розширення асортименту, але й удосконалення взаємодії з клієнтами. Одним із ключових інструментів, що підвищує ефективність онлайн-продажів та покращує користувацький досвід, є рекомендаційні системи [1].

Рекомендаційні системи дозволяють автоматично аналізувати величезні обсяги даних про товари та поведінку користувачів, щоб пропонувати персоналізовані рекомендації. Вони сприяють швидкому пошуку релевантних товарів, підвищують конверсію та середній чек покупок, а також допомагають утримувати клієнтів. Однак ефективність таких систем залежить від вибору алгоритмів, структури даних та методів їх обробки.

Особливу складність у роботі рекомендаційних систем становлять сценарії «холодного старту», коли інформації про нових користувачів чи товари недостатньо для формування релевантних пропозицій [6]. Також важливо враховувати різні типи користувачів: нових, які шукають популярні товари, та активних, які очікують персоналізованих рекомендацій на основі своїх вподобань. Тому завдання створення універсальної та гнучкої системи є надзвичайно актуальним у контексті сучасної електронної комерції.

Метою роботи є розробка рекомендаційної системи для інтернет-магазину з використанням гібридних підходів, які поєднують контентне та колаборативне фільтрування. Це дозволяє забезпечити високу точність і релевантність рекомендацій, враховуючи як атрибути товарів, так і поведінкові дані користувачів. У роботі також оцінюється ефективність

рекомендацій категорій товарів, пошуку та фільтрації, а також продуктивність системи у сценаріях «холодного старту».

Основні завдання дослідження включають аналіз існуючих підходів до побудови рекомендаційних систем, вибір алгоритмів і інструментів для реалізації, тестування системи на реальних даних та оцінку її продуктивності за ключовими метриками. Результати роботи спрямовані на створення універсальної рекомендаційної системи, здатної адаптуватися до різних сценаріїв використання та забезпечувати високий рівень задоволення користувачів.

Таким чином, робота є важливим внеском у розвиток сучасних інструментів персоналізації для платформ електронної комерції, забезпечуючи оптимальний баланс між точністю, швидкістю роботи та адаптивністю системи.

# 1 ТЕОРЕТИЧНІ ОСНОВИ

## 1.1 Огляд рекомендаційних систем

Рекомендаційні системи виникли як відповідь на проблему обробки великого обсягу інформації, що зросла з розвитком інтернету та цифрових технологій. У 1990-х роках з'явилися перші алгоритми, які базувалися на фільтруванні за контентом і колаборативному підході [31]. Одними з перших прикладів їхнього використання стали платформи для продажу книг і музики, такі як Amazon і Pandora, де рекомендації допомагали користувачам знаходити товари на основі їхніх попередніх дій.

Подальший розвиток припав на 2000-ті роки, коли зростання обчислювальних потужностей і обсягу даних дозволило вдосконалити алгоритми. Почали активно використовувати техніки кластеризації, машинного навчання та обробки великих даних. Платформи, такі як Netflix і Spotify, стали прикладами успішного застосування рекомендаційних систем у масштабах мільйонів користувачів.

Сьогодні рекомендаційні системи охоплюють різноманітні галузі, включаючи електронну комерцію, стрімінгові сервіси, медицину та освіту. Їх еволюція пов’язана з інтеграцією новітніх технологій, таких як глибоке навчання, персоналізовані моделі та хмарні обчислення [1], [9].

Сучасні рекомендаційні системи поділяються на кілька основних типів залежно від методів і джерел даних, які вони використовують [17]:

– контентні рекомендаційні системи;
– колаборативні рекомендаційні системи;
– гібридні рекомендаційні системи;
– моделі на основі знань.

Контентні рекомендаційні системи базуються на властивостях об'єктів, які рекомендуються. Для кожного об'єкта (наприклад, товару, пісні чи фільму) створюється профіль, що описує його характеристики.

Рекомендації формуються шляхом порівняння цих характеристик з інтересами користувача. До переваг можна віднести гарну точність для нових користувачів, якщо є достатньо інформації про об'єкти. З недоліків присутня проблема «холодного старту» для нових об'єктів, недостатність диверсифікації рекомендацій [6].

Колаборативні рекомендаційні системи грунтуються на взаємодії користувачів із системою. Методика порівнює вподобання одного користувача з вподобаннями інших, щоб передбачити, які об'єкти можуть зацікавити. З переваг, вона добре працює, коли є багато даних про взаємодії. До недоліків можна віднести потребу у великій кількості даних, проблему «холодного старту» для нових користувачів [16].

Гібридні рекомендаційні системи поєднують різні підходи, наприклад, контентний і колаборативний, для досягнення кращої точності. Це дозволяє компенсувати недоліки одного методу за рахунок переваг іншого. З очевидних переваг – універсальність, висока точність. Недоліки – складність реалізації, збільшення обчислювальних витрат.

Моделі на основі знань використовують експертні знання про предметну область для створення рекомендацій. Цей підхід часто застосовується у вузькоспеціалізованих системах, наприклад, у медицині чи наукових дослідженнях.

Рекомендаційні системи відіграють ключову роль у формуванні сучасного цифрового середовища, сприяючи задоволенню потреб користувачів, зростанню доходів компаній, оптимізації інформаційного потоку та підтримці лояльності аудиторії. Завдяки персоналізованим рекомендаціям користувачі швидше знаходять потрібний контент, що підвищує їхню задоволеність від взаємодії із сервісом. У сфері електронної комерції такі системи допомагають збільшувати продажі, формуючи пропозиції, які відповідають інтересам клієнтів, і тим самим сприяють зростанню середнього чека.

Крім того, рекомендаційні системи вирішують проблему інформаційного перевантаження, пропонуючи тільки релевантний контент. Це значно полегшує користувачам навігацію в інформаційному середовищі, де обсяг даних постійно зростає. Завдяки персоналізованому підходу платформи, які використовують рекомендаційні системи, посилюють довіру користувачів і формують довгострокову лояльність, що є ключовим фактором їхньої конкурентоспроможності на ринку.

Рекомендаційні системи стали невід'ємною частиною багатьох платформ завдяки своїй здатності адаптуватися до інтересів користувачів і впливати на їхні рішення. Основними викликами для подальшого розвитку є проблеми масштабованості, етичні аспекти (наприклад, прозорість алгоритмів) і персоналізація для малих даних. У наступних розділах буде розглянуто, як сучасні підходи та технології, такі як гібридні методи, допомагають вирішувати ці виклики.

## 1.2 Гібридні підходи в рекомендаційних системах

Гібридні рекомендаційні системи виникли як відповідь на обмеження окремих підходів, таких як контентний і колаборативний. Ці системи поєднують переваги різних методів, зменшуючи їх недоліки, що дозволяє досягати більш високої точності та ефективності в рекомендаціях. В основі гібридних систем лежить ідея комбінування декількох алгоритмів таким чином, щоб кінцевий результат був кращим, ніж у кожного алгоритму окремо.

Гібридизація може відбуватися на різних етапах роботи рекомендаційної системи. По-перше, це інтеграція на рівні даних, коли різні типи інформації (наприклад, профілі користувачів, історії взаємодій і метадані об'єктів) комбінуються ще до застосування алгоритмів. По-друге, об'єднання може відбуватися на рівні алгоритмів, коли декілька методів застосовуються паралельно, а їх результати згодом комбінуються. Нарешті,

інтеграція може бути виконана на рівні рекомендацій, де кожен алгоритм формує свій список пропозицій, а система обирає оптимальні рекомендації з усіх отриманих варіантів.

Одним із ключових типів гібридних підходів є послідовне комбінування, коли результати одного алгоритму використовуються як вхідні дані для іншого. Наприклад, контентний підхід може попередньо відфільтрувати об'єкти, що відповідають інтересам користувача, а колаборативний алгоритм надасть рекомендації з відібраного списку, враховуючи вподобання подібних користувачів. Такий метод дозволяє уникнути проблеми «холодного старту», характерної для контентних систем, і водночас забезпечує високий рівень персоналізації [6].

Інший поширений підхід – паралельне комбінування, коли декілька алгоритмів працюють одночасно, і результати їхньої роботи об'єднуються. Для інтеграції результатів часто застосовуються методи зваженого голосування, ранжування або середнього арифметичного. Наприклад, якщо контентна система оцінює об'єкт у 80%, а колаборативна – у 60%, підсумкова оцінка може бути розрахована як середнє значення, скориговане на основі важливості кожного методу.

Ще одним цікавим варіантом є вибіркове комбінування, коли система адаптується до поточного стану і вибирає найбільш релевантний алгоритм залежно від доступних даних чи характеристик користувача. Наприклад, для нових користувачів, про яких немає історії взаємодії, може використовуватися контентний метод, а для постійних клієнтів – колаборативний.

Гібридизація також дозволяє експериментувати з новітніми алгоритмами. Наприклад, сучасні гібридні системи інтегрують методи глибокого навчання, такі як згорткові або рекурентні нейронні мережі, з традиційними алгоритмами. Глибоке навчання дозволяє витягати складні патерни з даних, а традиційні методи забезпечують базову надійність і простоту реалізації. Одним із прикладів таких підходів є використання

автоенкодерів для попередньої обробки даних, а потім застосування колаборативного фільтрування для формування рекомендацій.

Гібридні підходи особливо корисні в умовах великої варіативності даних. Наприклад, в інтернет-магазинах дані про покупки, перегляди товарів, оцінки та навіть коментарі клієнтів можуть використовуватися в комбінації для формування рекомендацій. У цьому випадку гібридизація дозволяє врахувати не лише явні, але й приховані уподобання користувачів, що значно підвищує якість рекомендацій.

На рисунку 1.1 зображено приклад роботи гібридної системи рекомендацій, яка поєднує дані про користувача, інформацію спільноти, особливості товарів та моделі знань для формування персоналізованого списку рекомендацій

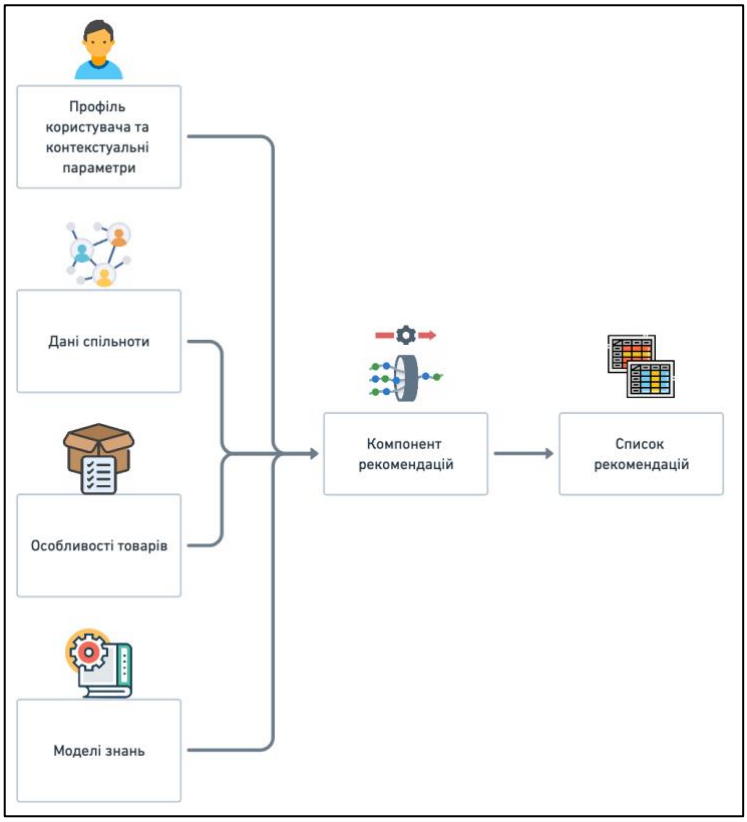


Рисунок 1.1 – Приклад роботи гібридної системи рекомендацій

Однак використання гібридних методів має і свої виклики. Поєднання декількох алгоритмів значно ускладнює реалізацію системи та підвищує її обчислювальну вартість. Крім того, необхідно враховувати потенційний конфлікт між результатами різних підходів і забезпечувати їхнє гармонійне інтегрування.

Таким чином, гібридні рекомендаційні системи представляють собою найбільш універсальний і гнучкий інструмент для створення персоналізованих рекомендацій. Їх ефективність залежить від правильного вибору методів інтеграції, оптимізації алгоритмів та адаптації до конкретних задач. У наступних розділах буде розглянуто, як саме ці підходи реалізуються в рекомендаційній системі для інтернет-магазину.

## 1.3 Сучасні тенденції та виклики

Сучасні рекомендаційні системи є важливим інструментом в умовах швидкого зростання обсягів даних та еволюції технологій штучного інтелекту. Їхній розвиток тісно пов'язаний з новітніми підходами до обробки інформації, інтеграцією інноваційних методів і вирішенням нових викликів, що виникають у зв’язку з масштабованістю, персоналізацією та етикою.

Електронна комерція (ecommerce) є однією з найдинамічніших і найприбутковіших галузей сучасної економіки [9], [32]. Завдяки глобальному доступу до інтернету та постійному зростанню кількості користувачів онлайн-магазинів, конкуренція на ринку ecommerce суттєво посилюється. Покупці сьогодні мають доступ до величезного асортименту товарів, що створює нові виклики для онлайн-ритейлерів, пов'язані з наданням персоналізованого досвіду користувачам. Однією з головних проблем є швидке й точне знаходження товарів, які найбільше відповідають інтересам та потребам клієнтів, оскільки обсяг товарної номенклатури може досягати сотень тисяч або мільйонів найменувань.

На рисунку 1.2 зображено динаміку зростання продажів електронної комерції як відсоток від загального обсягу роздрібного продажу за період з Q3-2019 по Q3-2024, протягом якого помітно стабільне збільшення частки ecommerce, що підкреслює зростаючу роль онлайн-продажів у світовій економіці.

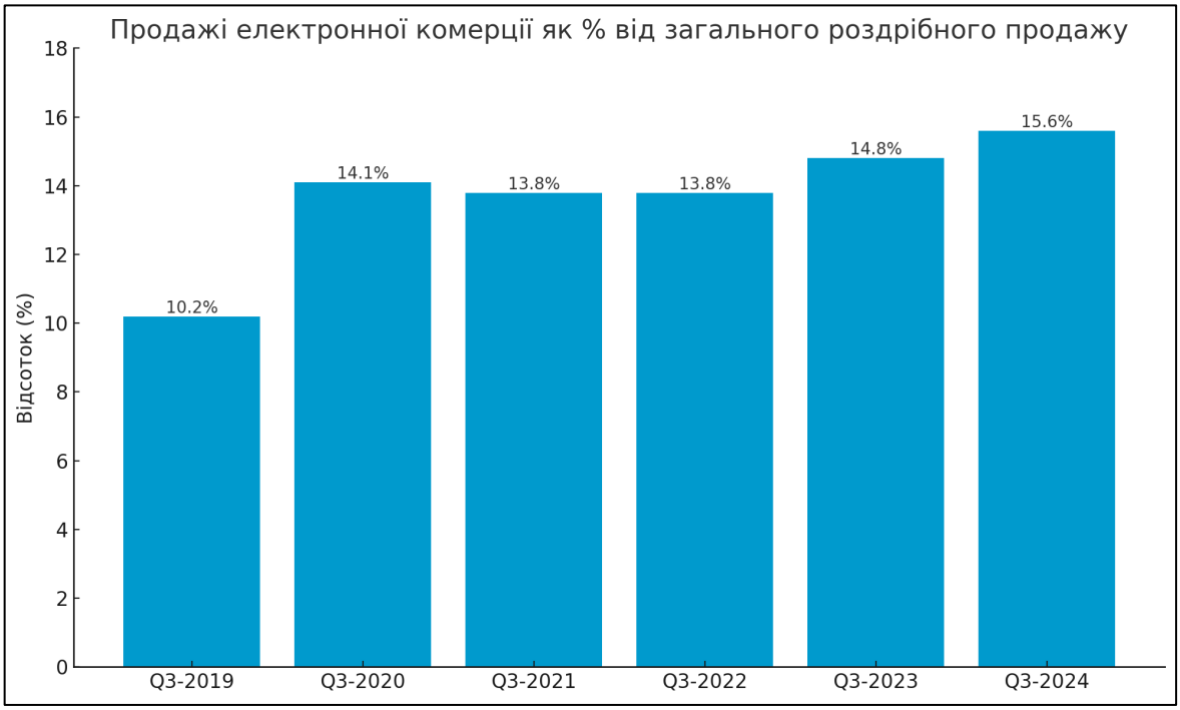


Рисунок 1.2 – Динаміка зростання продажів електронної коммерції [9]

У Q3-2019 частка електронної комерції становила 10.2%, але вже у Q3-2020 відбулося значне зростання до 14.1%, що можна пов'язати зі світовими змінами в поведінці споживачів, зокрема з пандемією COVID-19, яка значно прискорила перехід покупок в онлайн-середовище. Надалі спостерігається стабільний розвиток: хоча у Q3-2021 та Q3-2022 темпи росту дещо сповільнилися, показник тримається на рівні 13.8%, що свідчить про закріплення нових звичок покупців.

Починаючи з Q3-2023, відновлюється позитивна динаміка, і у Q3-2024 частка електронної комерції досягає 15.6%. Це підкреслює стійку тенденцію до зростання популярності онлайн-покупок та ефективність ecommerce як

основного каналу для роздрібної торгівлі. Значне і стабільне збільшення частки електронної комерції підтверджує її важливість як ключової ніші в сучасній економіці.

На рисунку 1.3 зображено частку пошуку товарів онлайн та в магазинах у листопаді 2023 року за різними категоріями серед дорослого населення світу. Кожна категорія товарів демонструє унікальні особливості у споживчій поведінці, що обумовлені як природою продукту, так і звичками покупців.

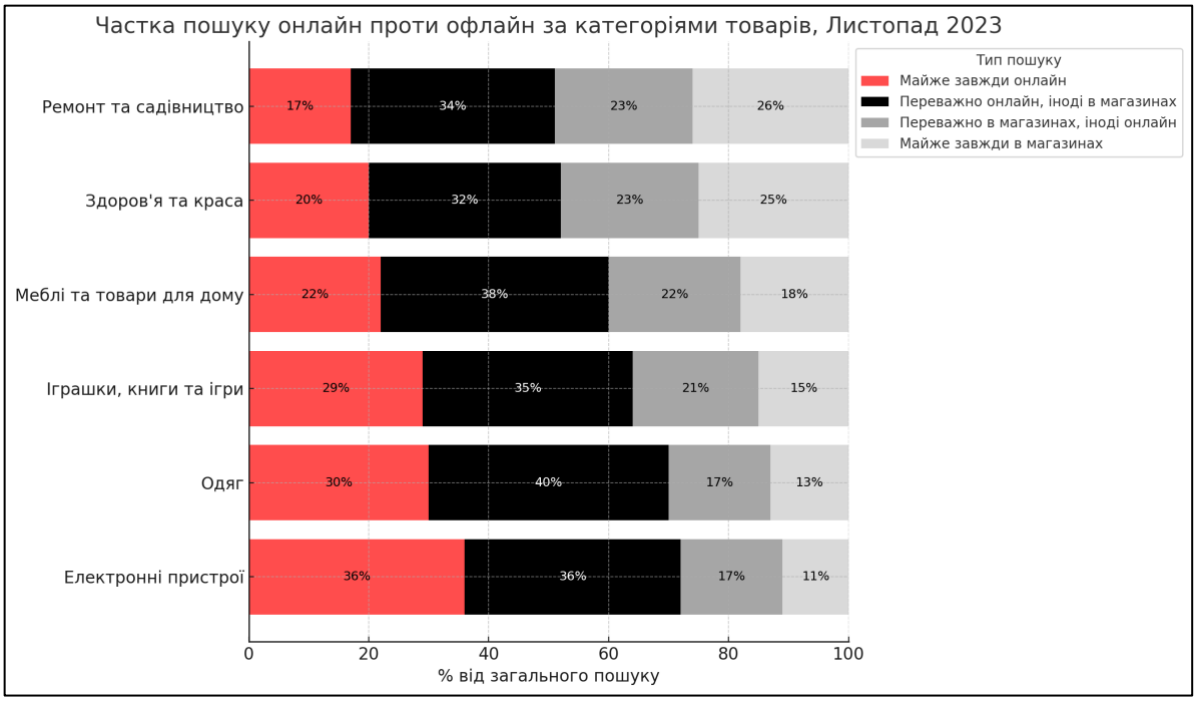


Рисунок 1.3 – Частка покупок онлайн та офлайн за категоріями товарів [32]

Графік ілюструє, як онлайн- і офлайн-пошук розподіляються між різними категоріями товарів. Найбільш показовою є категорія електроніки, де 72% споживачів віддають перевагу онлайн-каналам для пошуку та порівняння товарів завдяки зручності доступу до характеристик, оглядів і цін. Однак одяг залишається більш збалансованим: 70% користуються онлайн-пошуком, проте значна частина споживачів обирає магазини для примірки та оцінки матеріалів.

Схожа ситуація спостерігається в категоріях іграшок, книг і товарів для дому. Онлайн-платформи зручні для пошуку та порівняння асортименту, але фізичний огляд товарів все ще відіграє важливу роль, особливо для прийняття остаточного рішення. Для меблів та товарів для ремонту онлайн-канали часто використовуються на етапі пошуку, тоді як магазини дозволяють оцінити розмір, якість і функціональність товарів. Загалом, онлайн-пошук домінує завдяки зручності та широкому вибору, але фізичні магазини залишаються важливими у категоріях, де критичним є тактильний контакт і індивідуальна оцінка товару.

Важливість рекомендаційних систем в ecommerce обумовлена необхідністю підвищення ефективності продажів та утримання клієнтів шляхом надання індивідуальних пропозицій. Споживачі очікують, що онлайн-магазини зможуть передбачити їхні потреби та запропонувати найбільш релевантні товари в зручний час. Це є важливим не лише для підвищення задоволеності клієнтів, а й для мінімізації часу, який користувач витрачає на пошук необхідного товару. Відсутність персоналізованих рекомендацій або їх низька точність можуть призводити до втрати клієнтів, зменшення прибутків і переходу покупців до конкурентів.

Ключовим викликом є обробка великого обсягу даних про користувачів, товари та їх взаємодію в режимі реального часу. Ecommerce-системи постійно генерують дані про історію переглядів, кліки, покупки, оцінки та відгуки, які необхідно ефективно обробляти для побудови рекомендацій. При цьому слід враховувати як явні дані, такі як оцінки товарів, так і неявні, зокрема поведінкові сигнали – час перебування на сторінці, додавання товару до кошика чи списку бажань. Такі обчислення потребують продуктивних алгоритмів, що забезпечують швидке й точне формування рекомендацій навіть при великому потоці користувачів.

Окрім персоналізації, важливою задачею є підвищення конверсій і середнього чека покупця. Рекомендаційні системи вирішують цю проблему шляхом пропозиції супутніх товарів, які найчастіше купують разом, а також

просування популярних чи нових товарів, які відповідають поточним інтересам користувача. Таким чином, вони дозволяють не лише задовольняти потреби клієнтів, а й стимулювати додаткові покупки.

Однією з найважливіших сучасних тенденцій є використання глибокого навчання у рекомендаційних системах. Згорткові нейронні мережі (CNN) дозволяють аналізувати мультимедійні дані, такі як зображення та відео, для покращення рекомендацій. Рекурентні нейронні мережі (RNN) і трансформери, такі як двоспрямовані кодувальні представлення з трансформерів (BERT), успішно застосовуються для моделювання послідовностей даних, наприклад, історії взаємодій користувача з платформою. Глибоке навчання забезпечує рекомендаціям глибший рівень персоналізації, дозволяючи враховувати складні залежності між даними.

Останнім часом популярності набули графові нейронні мережі (GNN), які використовуються для моделювання складних відношень між користувачами та об'єктами. Наприклад, графи дозволяють відображати соціальні зв'язки або зв'язки між товарами на основі їхніх атрибутів. Це дає змогу створювати рекомендації, які враховують багаторівневі зв'язки в даних і їхній контекст.

Сучасні системи все частіше орієнтуються на рекомендації в реальному часі, щоб враховувати динамічну зміну інтересів користувачів. Це особливо актуально для платформ потокового контенту, електронної комерції та соціальних мереж. Рішення для реального часу часто базуються на потоковій обробці даних, що потребує розподілених обчислювальних інфраструктур, таких як Apache Kafka або Apache Flink.

Ще однією важливою тенденцією є використання мультимодальних даних, які об'єднують текст, зображення, відео та аудіо для створення більш точних рекомендацій. Наприклад, у моді чи електроніці текстові описи товарів доповнюються аналізом зображень, щоб краще розуміти потреби

користувачів. Такі підходи потребують складної інтеграції різних типів даних, але значно підвищують якість рекомендацій.

Зі зростанням популярності рекомендаційних систем важливим викликом стало питання етичності та прозорості алгоритмів. Виникає потреба пояснювати, як саме система формує рекомендації, особливо у сферах, які впливають на прийняття важливих рішень, наприклад, у медицині чи кредитуванні. Відсутність прозорості може знизити довіру до системи, тому дослідники активно працюють над розробкою Explainable AI (XAI) – алгоритмів, які можуть пояснювати свої рішення у зрозумілий для користувача спосіб.

Зростання обсягів даних і кількості користувачів створює виклик масштабованості систем. Алгоритми повинні бути здатними обробляти мільйони взаємодій у режимі реального часу. Це вимагає розподілених обчислень і оптимізації пам'яті. Наприклад, використання графових баз даних або фреймворків, таких як TensorFlow Extended, дозволяє досягати високої продуктивності навіть у великих системах.

Проблема «холодного старту» залишається однією з найбільших перепон для рекомендаційних систем [6]. У випадку нових користувачів система не має даних про їхні вподобання, що значно ускладнює надання точних рекомендацій. Для вирішення цієї проблеми використовуються демографічні дані, контекстна інформація чи інтеграція зі сторонніми джерелами даних, наприклад, соціальними мережами.

Ще одним викликом є забезпечення різноманітності рекомендацій. Надто вузькі чи повторювані пропозиції можуть швидко набриднути користувачам. У відповідь на це системи починають враховувати новизну та різноманітність під час формування списку рекомендацій, наприклад, через зважування рідше використовуваних об'єктів або впровадження алгоритмів, що сприяють відкриттю нового контенту.

У сучасних інтернет-магазинах спостерігається зростання інтересу до технологій доповненої реальності (AR). Наприклад, системи, які

дозволяють приміряти одяг або візуалізувати меблі у своєму просторі, можуть використовувати рекомендаційні алгоритми для кращої персоналізації пропозицій. Інтеграція AR додає новий вимір взаємодії, роблячи рекомендації більш інтуїтивними та корисними.

Сучасні платформи все частіше використовують рекомендаційні системи як інструмент для підвищення лояльності користувачів. Рекомендації створюються не лише для збільшення продажів, а й для формування позитивного досвіду користувача, що спонукає до повернення на платформу. Це включає рекомендації контенту на основі попередніх дій або індивідуальні пропозиції з урахуванням поточних трендів.

На рисунку 1.4 демонструється динаміка зростання ринку рекомендаційних систем у США за період з 2018 по 2024 роки, та прогноз до 2028 року, розподіленого за типами фільтрації: фільтрація на основі співпраці, контентна фільтрація та гібридні рекомендації.

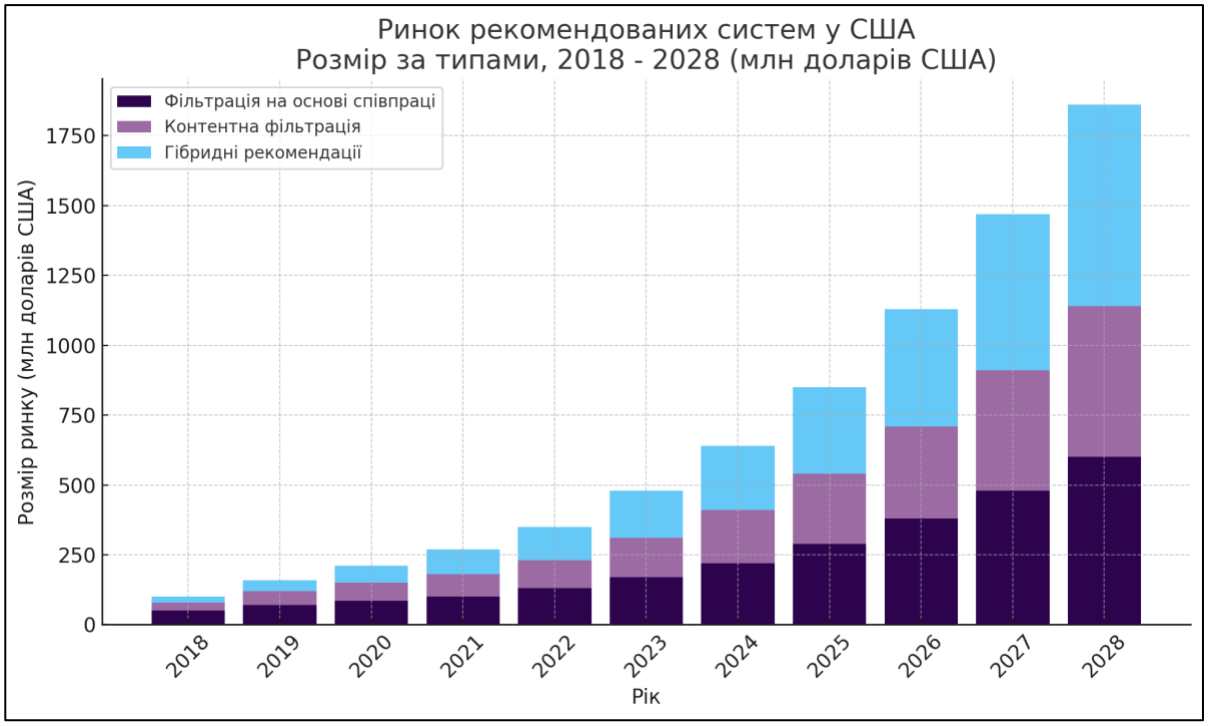


Рисунок 1.4 – Динаміка зростання ринку рекомендаційних систем у США

Загальна тенденція свідчить про стрімке зростання ринку, що досягне майже 1.75 мільярдів доларів США до 2028 року. Найшвидший приріст спостерігається у сегменті гібридних рекомендацій, що відзначаються поєднанням переваг колаборативного та контентного підходів. Уже в 2023 році гібридні системи починають домінувати на ринку, демонструючи їх ефективність для вирішення сучасних викликів персоналізації в ecommerce.

Сучасні тенденції у розробці рекомендаційних систем вказують на поступове ускладнення алгоритмів, інтеграцію інноваційних технологій і зростання вимог до їхньої адаптивності, прозорості й масштабованості. Виклики, такі як забезпечення етичності, вирішення проблеми «холодного старту» та використання мультимодальних даних, стимулюють розвиток нових методів і стратегій. Усе це створює потужну основу для вдосконалення рекомендаційних систем і розширення їхнього застосування в різних сферах.

# 2 МЕТОДИ ТА ЗАСОБИ ДОСЛІДЖЕННЯ

## 2.1 Аналіз методів для побудови рекомендаційних систем

### 2.1.1 Контентне фільтрування

Контентне фільтрування є одним із базових методів рекомендаційних систем, що використовує характеристики об'єктів для пошуку схожих товарів. У контексті розробки системи рекомендацій для інтернет-магазину цей підхід був застосований для аналізу атрибутів продуктів, з подальшим обчисленням подібностей між товарами.

Контентне фільтрування передбачає створення профілів об'єктів на основі їхніх характеристик [24]. У даному випадку кожен продукт характеризується:

– текстовим описом, що є основним джерелом семантичної інформації;

– брендом, який дозволяє враховувати подібність товарів від одного виробника;

– ціною, що забезпечує можливість порівняння товарів за їхньою вартістю.

Для обчислення подібностей між товарами було застосовано векторизацію текстових даних, а також обробку числових атрибутів з подальшим об'єднанням результатів у загальну метрику подібності [24].

Першим етапом реалізації контентного фільтрування є підготовка текстових даних. Для кожного продукту текстові описи були попередньо оброблені, за допомогою видалення зайвих символів та стоп-слів, та збагачення описів технічними характеристиками товарів за допомогою додаткового сервісу. Це дозволяє отримати більш повний семантичний опис кожного продукту.

Другим етапом є векторизація текстів. Для перетворення текстових описів у числові вектори використовується модель на основі SentenceTransformer. Ця модель забезпечує семантичну подібність текстів, оскільки векторизує їх у просторі, де близькі за значенням описи мають подібні координати.

Третім етапом є обчислення подібності. Подібність між векторами текстових описів обчислюється за допомогою косинусної подібності здіно з формулою (2.1).

$$Cosine\ Similarity = \frac{A*B}{\|A\|*\|B\|} = \frac{\sum_{i=1}^{n} A_i*B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}*\sqrt{\sum_{i=1}^{n} B_i^2}}, \quad (2.1)$$

де $A_i$ та $B_i$ – компоненти векторів текстових описів двох продуктів.

Четвертим етапом є обробка числових атрибутів. Для порівняння брендів використовувалася бінарна матриця подібності, де збіги брендів позначаються значенням 1, а відмінності – 0. Ціни продуктів були нормалізовані у діапазоні від 0 до 1 за допомогою MinMaxScaler. Подібність цін розраховується як обернена різниця між нормалізованими значеннями згідно з формулою (2.2).

$$Price\ Similarity = 1 - |p_i - p_j|, \quad (2.2)$$

де $p_i$ і $p_j$ – нормалізовані ціни двох продуктів.

Останнім п’ятим етапом є комбінування результатів. Загальна подібність між двома продуктами обчислюється як зважена сума. Формулу наведено у (2.3).

$$Combined\ Similarity = w_1 * S_{description} + w_2 * S_{brand} + w_3 * S_{price}, \quad (2.3)$$

де $S_{description}$ – подібність текстових описів;

$S_{brand}$ – подібність брендів;

$S_{price}$ – подібність цін.

Підхід має кілька важливих переваг. По-перше, він забезпечує гнучкість у додаванні нових характеристик. Система легко інтегрує не лише текстові описи, а й інші атрибути товарів, такі як колір, розмір чи рейтинг, що дозволяє значно підвищити якість рекомендацій. По-друге, використання семантичного порівняння текстів сприяє досягненню високої точності рекомендацій для товарів зі схожими описами. Крім того, контентне фільтрування базується виключно на характеристиках товарів, що зменшує залежність від користувацьких даних і вирішує проблему «холодного старту» для нових продуктів.

Незважаючи на переваги, контентне фільтрування має певні обмеження. Метод фокусується лише на характеристиках продуктів і не враховує історію взаємодій конкретного користувача. Неякісні або неповні описи товарів можуть призводити до некоректних рекомендацій. Система може рекомендувати занадто схожі товари, що знижує різноманітність рекомендацій.

Контентне фільтрування стало одним із ключових компонентів системи рекомендацій, дозволяючи аналізувати семантичну подібність продуктів на основі їхніх текстових описів, бренду та ціни. Завдяки використанню векторизації та вагового комбінування характеристик цей метод забезпечує точні й релевантні рекомендації, особливо для нових товарів у системі. У поєднанні з колаборативним фільтруванням він формує основу для гібридного підходу, що дозволяє компенсувати його обмеження та покращувати загальну ефективність системи.

### 2.1.2 Колаборативне фільтрування

Колаборативне фільтрування є одним із основних методів побудови рекомендаційних систем, що ґрунтується на аналізі взаємодій користувачів

із об’єктами. Його основна ідея полягає у виявленні схожості між користувачами або продуктами на основі історичних даних, таких як оцінки, перегляди, кліки та покупки. На відміну від контентного фільтрування, яке аналізує лише атрибути об’єктів, колаборативний підхід дозволяє враховувати приховані патерни у вподобаннях користувачів, що робить його ефективним для більш складних сценаріїв [7].

У цій роботі колаборативне фільтрування було реалізовано двома підходами: обробкою експліцитних оцінок і використанням імпліцитних даних про взаємодії. Поєднання цих двох підходів дозволяє враховувати як явні вподобання користувачів, так і непрямі сигнали їхньої активності в системі.

Експліцитне колаборативне фільтрування реалізоване за допомогою методу Singular Value Decomposition (SVD), який є одним із найбільш популярних алгоритмів у цій категорії. SVD використовує матрицю оцінок користувачів, де рядки відповідають користувачам, а стовпці – продуктам, для виявлення латентних факторів, що впливають на вибір товарів. Розкладання матриці дозволяє визначити приховані взаємозв’язки між користувачами та об’єктами, що неможливо зробити за допомогою простих метрик подібності. Для покращення якості рекомендацій пропущені значення у матриці заповнюються глобальним середнім значенням оцінок, що дозволяє мінімізувати вплив неповних даних на результат.

Паралельно з експліцитним методом було реалізовано імпліцитне колаборативне фільтрування з використанням алгоритму Alternating Least Squares (ALS). Цей підхід обробляє непрямі взаємодії користувачів, такі як перегляди, кліки та покупки, об’єднуючи їх у єдину оцінку. Обчислення ваги для кожного типу взаємодії виконується на основі їх значущості: перегляди мають найменшу вагу, тоді як покупки отримують максимальну оцінку [22]. Для стабілізації даних використовується логарифмічне масштабування, що зменшує вплив великих значень і вирівнює розподіл. Додатково розріджена матриця взаємодій користувач-продукт обробляється

за допомогою ALS для побудови латентних факторів, що дозволяє моделювати уподобання користувачів навіть за відсутності явних оцінок.

Комбінування результатів експліцитного та імпліцитного фільтрування реалізується через зважене об’єднання отриманих матриць подібності. Обидва алгоритми нормалізуються до єдиного діапазону значень для забезпечення коректного підсумовування. Таким чином, комбінована матриця враховує як оцінки користувачів, так і їхні непрямі дії, що дозволяє отримати більш точні рекомендації. Для коректного обчислення оцінок взаємодії користувачів різні типи активності були агреговані у псевдооцінки, які формувалися за формулою (2.4) [22].

$$\text{Псевдооцінка} = 0.5 * \text{Перегляди} + 0.7 * \text{Кліки} + 1.0 * \text{Покупки}. \quad (2.4)$$

Використання логарифмічного масштабування дозволило стабілізувати розподіл значень у матриці, мінімізуючи вплив викидів.

Особливо важливим аспектом є обробка неповних даних, які є характерними для колаборативного фільтрування [34]. У випадку відсутності оцінок для деяких користувачів або товарів використовується підхід із заповненням матриці середніми значеннями або невеликим випадковим шумом. Це дозволяє уникнути «холодного старту» для нових користувачів або продуктів, мінімізуючи їх вплив на загальний результат.

Колаборативне фільтрування є ефективним у сценаріях, де система володіє великою кількістю даних про користувацькі взаємодії. Застосування SVD і ALS у цій роботі дозволяє обробляти як явні, так і неявні вподобання, а їх поєднання забезпечує точні та адаптивні рекомендації. Незважаючи на свої переваги, метод має певні обмеження: він залежить від наявності великого обсягу даних, а також потребує періодичного оновлення моделей для врахування нових взаємодій.

Таким чином, колаборативне фільтрування забезпечує ефективне виявлення зв'язків між користувачами та об’єктами на основі їхніх дій, а

його комбінування з контентним підходом у гібридній системі дозволяє компенсувати обмеження кожного з методів окремо.

### 2.1.3 Гібридний підхід

Гібридний підхід у рекомендаційних системах поєднує різні методи для досягнення більшої точності та усунення обмежень, характерних для окремих підходів. У цій роботі комбінуються контентне та колаборативне фільтрування, що дозволяє враховувати як характеристики продуктів, так і вподобання користувачів. Основна ідея гібридизації полягає у комбінуванні результатів цих двох методів з використанням вагових коефіцієнтів, щоб отримати підсумковий показник подібності або релевантності товарів для конкретного користувача.

У контексті реалізації системи контентне фільтрування відповідає за аналіз атрибутів продуктів, таких як текстовий опис, бренд і ціна, та обчислення подібності між товарами. Воно добре працює для нових продуктів або категорій, коли немає історії взаємодій користувачів. Колаборативне фільтрування, навпаки, фокусується на історичних даних про взаємодію користувачів, таких як оцінки, перегляди або покупки, і дозволяє виявляти приховані патерни у вподобаннях. Однак цей метод стикається з проблемою «холодного старту», коли нові користувачі або товари ще не мають достатньо даних [6].

Комбінування контентного та колаборативного підходів у гібридній системі відбувається на рівні результатів, які обчислюються обома методами. На етапі обчислення контентна частина формує матрицю подібності товарів, де кожна комірка відповідає мірі схожості між парою продуктів. Колаборативна частина генерує матрицю, що показує, наскільки користувачі схильні до взаємодії з певними товарами.

Для об’єднання результатів обидві матриці вирівнюються по всіх товарах, після чого комбінуються за допомогою методу лінійного

комбінування результатів із використанням вагових коефіцієнтів. Цей метод є простим у реалізації, дозволяє гнучко налаштовувати вплив кожного компонента системи та забезпечує високу точність у комбінованих сценаріях. Під час експериментів було обґрунтовано, що оптимальне значення ваг залежить від характеристик даних: контентний компонент є більш ефективним для нових товарів та користувачів, а колаборативний компонент забезпечує вищу персоналізацію для активних користувачів.

Підсумковий показник для кожного товару обчислюється згідно з формулою (2.5).

$$\text{Гібридна оцінка} = w_{content} * S_{content} + w_{collaborative} * S_{collaborative}, \quad (2.5)$$

де $S_{content}$ – результат контентного фільтрування;

$S_{collaborative}$ – результат колаборативного фільтрування;

$w_{content}$ – ваговий коефіцієнт контентного фільтрування;

$w_{collaborative}$ – ваговий коефіцієнт колаборативного фільтрування.

У цій роботі перевага надається контентному фільтруванню, що відображено у більших значеннях відповідного коефіцієнта. Це пояснюється тим, що контентна частина забезпечує базову точність рекомендацій завдяки атрибутам товарів, а колаборативна частина підсилює результат на основі поведінкових патернів користувачів.

Важливим аспектом гібридизації є забезпечення узгодженості між двома матрицями подібності. Для цього використовується механізм вирівнювання товарів: усі продукти, що були враховані хоча б в одній із матриць, об’єднуються у єдиний список, а відсутні значення заповнюються нулями. Це дозволяє уникнути втрати даних і забезпечує коректне комбінування результатів.

Гібридний підхід є гнучким і адаптивним, оскільки дозволяє враховувати як явні дані у вигляді оцінок, так і приховані зв’язки між товарами. Завдяки цьому система генерує рекомендації, що є не лише

точними, але й релевантними до конкретних інтересів користувачів. Окрім того, використання вагового об’єднання дає змогу налаштовувати баланс між методами залежно від наявності даних або особливостей конкретного застосування.

Хоча гібридний підхід забезпечує високу якість рекомендацій, він потребує значних обчислювальних ресурсів, особливо на етапі комбінування великих матриць подібності. Для оптимізації роботи системи результати обчислень кешуються у базі даних, що дозволяє забезпечити швидкий доступ до рекомендацій у режимі реального часу. Таким чином, гібридизація є ключовим методом у даній роботі, оскільки вона дозволяє поєднати переваги контентного фільтрування та колаборативного підходу, компенсуючи їхні індивідуальні обмеження та підвищуючи загальну ефективність рекомендаційної системи.

### 2.1.4 Рекомендація категорій товарів

Окрім рекомендації окремих продуктів, сучасні системи також використовують методи для рекомендації категорій товарів, які є рівнем ієрархії в електронній комерції. Такі підходи дозволяють користувачам орієнтуватися у великій кількості товарів, спрощуючи навігацію та підвищуючи ймовірність здійснення покупки.

Методи рекомендації категорій поєднують два важливі аспекти: глобальну популярність категорій серед усіх користувачів та індивідуальні інтереси конкретного користувача. Популярність категорії визначається на основі загальної активності користувачів, такої як перегляди, кліки та покупки, що агрегуються для кожної категорії. Інтерес користувача, своєю чергою, обчислюється на основі його індивідуальних взаємодій з товарами у межах конкретних категорій.

Ці два фактори комбінуються за допомогою вагових коефіцієнтів, що дозволяє врахувати як загальні тренди платформи, так і персональні

вподобання користувача. У результаті система генерує ранжований список категорій, які мають найбільший потенціал для зацікавлення користувача. Такий підхід є ефективним для нових користувачів, які ще не мають достатньої історії взаємодій, а також для користувачів, які прагнуть дослідити нові напрями або категорії товарів.

### 2.1.5 Пошук і фільтрація товарів

Системи пошуку та фільтрації є важливим компонентом рекомендаційних механізмів, оскільки вони забезпечують гнучкий інструмент навігації, що дозволяє користувачам самостійно впливати на результати. На відміну від класичних рекомендацій, які формуються автоматично, пошук дозволяє користувачу вводити конкретні запити, а фільтрація дає змогу звузити результати на основі категорій, цінових діапазонів або інших параметрів.

Текстовий пошук у системі реалізується на основі Term Frequency - Inverse Document Frequency (TF-IDF), що є одним із найпоширеніших методів обробки текстових даних. TF-IDF дозволяє виявити ключові слова у текстових описах товарів та зіставити їх із запитом користувача для обчислення косинусної подібності. Цей підхід забезпечує швидке й точне ранжування товарів за релевантністю до введеного запиту.

Фільтрація результатів здійснюється на основі структурованих параметрів товарів, таких як категорія або ціна. Комбінування пошуку та фільтрації дозволяє користувачам отримувати результати, що найбільше відповідають їхнім очікуванням. Наприклад, користувач може виконати пошук «ноутбук для ігор» та застосувати фільтри для обмеження результатів за категорією «ноутбуки» та діапазоном цін.

Використання такого підходу значно покращує навігацію на платформі та підвищує задоволеність користувачів, оскільки вони отримують більший контроль над результатами рекомендацій. Це є

особливо корисним у випадках, коли користувач має конкретний намір здійснити покупку.

## 2.2 Інструменти та технології

Для реалізації рекомендаційної системи було обрано набір інструментів і технологій, які забезпечують високу продуктивність, масштабованість та інтеграцію різних методів фільтрування. Рішення про вибір конкретних інструментів ґрунтувалося на вимогах до швидкості обробки даних, точності рекомендацій та ефективності розгортання системи в реальному часі.

Python є основною мовою програмування для побудови рекомендаційної системи завдяки широкому спектру бібліотек для роботи з даними та машинного навчання [30]. Під час обробки великих обсягів інформації Python забезпечує ефективну реалізацію алгоритмів, зокрема для колаборативного та контентного фільтрування. У рамках роботи було використано числові обчислення з NumPy та маніпуляцію табличними даними за допомогою Pandas, що спрощує обробку структурованих даних про взаємодії користувачів і характеристики продуктів.

Для моделювання експліцитного колаборативного фільтрування застосовується бібліотека Surprise, яка надає інструменти для роботи з матрицями оцінок та алгоритмами факторизації, такими як SVD. Використання Surprise дозволяє реалізувати моделі, що працюють з розрідженими матрицями оцінок, знаходячи приховані патерни у даних.

Для імпліцитного фільтрування застосовується бібліотека Implicit, яка реалізує метод ALS. Це ефективне рішення для обробки неявних взаємодій, таких як перегляди, кліки або покупки. Алгоритм працює з розрідженими матрицями у форматі Compressed Sparse Row (CSR) і забезпечує високу продуктивність при обчисленні латентних факторів для великих наборів даних.

Підготовлені результати колаборативного та контентного фільтрування комбінуються у фінальну гібридну модель, що працює з ваговими коефіцієнтами. Для забезпечення узгодженості даних використовуються методи нормалізації, такі як MinMaxScaler із бібліотеки Scikit-learn, що дозволяє привести значення різних підходів до єдиного діапазону [27], [35]. Вирівнювання матриць подібності забезпечується завдяки гнучкій обробці індексів продуктів і користувачів, що дозволяє ефективно поєднувати результати різних методів у єдину систему.

Для обробки матриць взаємодій використовується SciPy, зокрема для створення розріджених матриць у форматі csr_matrix. Цей формат дозволяє ефективно працювати з великими розрідженими даними, зменшуючи обсяг пам’яті та час обчислень.

Для обробки текстових описів товарів було обрано SentenceTransformer – сучасний інструмент для семантичної векторизації текстів. Модель «all-MiniLM-L6-v2» забезпечує високу якість подібностей завдяки ефективній архітектурі, що поєднує точність та швидкість роботи. Вибір цього алгоритму обґрунтовується його здатністю виконувати семантичну векторизацію текстів, що дозволяє системі краще розуміти зміст текстових описів товарів і виявляти їхню подібність. Порівняно із класичними підходами, SentenceTransformer забезпечує більш якісне представлення текстових даних, враховуючи їхній контекст. Це особливо важливо для контентного фільтрування, де основним джерелом даних є текстові атрибути товарів. Збереження обчислених рекомендацій відбувається у базі даних SQLite, яка забезпечує легкий доступ до готових результатів та знижує обчислювальне навантаження під час запитів у режимі реального часу. Використання SQLite як локальної бази дозволяє мінімізувати затримки при отриманні рекомендацій, що особливо важливо для веб-додатків із великою кількістю користувачів [26].

Для забезпечення швидкої інтеграції рекомендаційної системи з зовнішньою системою інтернет магазину використовується фреймворк

Flask для створення REST API [12]. Python-модуль регулярно генерує нові рекомендації на основі даних про взаємодії користувачів, а Flask забезпечує обмін даними між модулями та їх доступ до веб-застосунку. Кожні 30 хвилин модуль виконує обчислення, оновлюючи результати у базі даних, що дозволяє підтримувати актуальність рекомендацій для користувачів.

Для спрощення розгортання та підтримки системи використовується Docker, що дозволяє контейнеризувати Python-модуль та забезпечити його стабільну роботу незалежно від середовища виконання. Контейнеризація спрощує масштабування системи, а також полегшує інтеграцію у загальну інфраструктуру додатка.

Таким чином, обрані інструменти та технології забезпечують продуктивність і надійність рекомендаційної системи. Використання Python для обробки даних та побудови моделей, SQLite для збереження результатів та Flask для інтеграції дозволяє системі працювати швидко й ефективно. Це поєднання забезпечує як обчислювальну ефективність на етапі генерації рекомендацій, так і швидку обробку запитів у реальному часі, що відповідає вимогам сучасних веб-застосунків.

## 2.3 Розробка методології експериментів

### 2.3.1 Джерела даних

Модуль рекомендаційної системи було розроблено з урахуванням можливості роботи з реальними користувацькими даними, що надходять із зовнішнього веб-додатка. У робочому середовищі система має отримувати актуальну інформацію про взаємодії користувачів із продуктами, зокрема оцінки, перегляди, кліки та покупки, а також детальну інформацію про товари, що зберігається у базі даних веб-застосунку. Передавання цих даних відбувається через REST API, що забезпечує швидкий обмін між веб-додатком та Python-модулем для обчислення рекомендацій.

На етапі тестування та валідації система була налаштована на роботу з відкритими наборами даних, що походять із реальних e-commerce платформ. Використання таких даних дозволило максимально наблизити умови тестування до реальних сценаріїв роботи у продакшені. Було обрано кілька датасетів, що містять анонімізовану інформацію про взаємодії користувачів із товарами, включаючи неявні дії, такі як перегляди, кліки та покупки, а також явні оцінки продуктів. Окрім цього, набори даних містили детальну інформацію про товари: текстові описи, бренди, категорії та цінові атрибути.

Обрані датасети було об’єднано, структуровано та імпортовано у базу даних веб-додатка, який став джерелом даних для рекомендаційної системи. Це дозволило зімітувати реальні умови роботи системи, забезпечивши сценарій, де Python-модуль отримує актуальні дані із зовнішнього додатка через API. Таким чином, веб-додаток діє як посередник, надаючи Python-модулю структуровані дані, що включають взаємодії користувачів та характеристики продуктів.

Такий підхід є переконливим і обґрунтованим із двох причин. По-перше, використання відкритих реальних даних дозволяє забезпечити якісну апробацію системи ще до її впровадження у продакшен, оскільки дані відображають реальну поведінку користувачів на e-commerce платформах. По-друге, імпортування цих даних у веб-додаток дозволяє перевірити повний цикл роботи системи, від запиту на отримання даних до їхньої обробки та повернення результатів.

Таким чином, система спроєктована таким чином, щоб працювати із живими даними зовнішнього додатка, але для тестування та валідації використовувалися реальні відкриті датасети, адаптовані під поточний сценарій. Це дозволяє оцінити продуктивність і точність системи, а також забезпечує готовність до роботи з реальними користувацькими даними у подальших етапах розробки та впровадження.

### 2.3.2 Методи підготовки даних

Підготовка даних є ключовим етапом у побудові рекомендаційної системи, оскільки якість і точність результатів безпосередньо залежать від коректності оброблених даних [28]. На етапі тестування та валідації було використано відкриті датасети, які містять дані про взаємодії користувачів із товарами, а також атрибути самих продуктів. Ці дані були імпортовані у веб-додаток і передані Python-модулю для подальшої обробки. Процес підготовки даних включав обробку неявних взаємодій, роботу з неповними даними та трансформацію текстових і числових атрибутів продуктів у відповідні формати.

Для обробки неявних взаємодій, таких як перегляди, кліки та покупки, було розроблено методику генерації псевдооцінок. Оскільки імпліцитні дані не містять явних числових значень, їм присвоюються ваги на основі значущості кожного типу взаємодії. Перегляди мають найменший вплив на фінальну оцінку, оскільки вони є найпоширенішим, але найменш релевантним сигналом, тоді як покупки отримують максимальну вагу, оскільки вони чітко відображають зацікавленість користувача товаром. Псевдооцінки обчислюються як лінійна комбінація взаємодій, що дозволяє сформувати єдину числову міру інтересу користувача до продукту.

Особливу увагу приділено обробці неповних даних, які є типовими для реальних датасетів. Відсутні оцінки в матриці користувач-продукт були заповнені глобальним середнім значенням, що мінімізує вплив пропусків на роботу алгоритмів колаборативного фільтрування. Такий підхід дозволяє зберегти структуру даних і забезпечити стабільність алгоритмів на етапі навчання моделі. У випадках, коли користувачі або продукти не мали жодних взаємодій, до даних було додано невеликий випадковий шум, що дозволяє уникнути ситуації «холодного старту» для нових елементів.

Текстові дані про товари були піддані попередній обробці для забезпечення коректного функціонування контентного фільтрування.

Тексти описів були очищені від зайвих символів, стоп-слів та лемматизовані для зменшення розмірності даних і підвищення семантичної якості. Для подальшої обробки використовувалася векторизація, що конвертує текстові описи у числові вектори для подальшого обчислення подібності між продуктами.

Числові атрибути, такі як ціни, були нормалізовані у діапазоні від 0 до 1 для забезпечення їхньої сумісності з іншими компонентами системи. Нормалізація виконувалася за допомогою MinMaxScaler, що дозволяє зберегти відносні співвідношення між значеннями і водночас спростити їх інтеграцію у фінальну метрику подібності. Бренди продуктів оброблялися як категоріальні дані, де їхня подібність визначалася на основі бінарного збігу значень.

Після обробки всі типи даних були об’єднані в узгоджений формат, де кожен користувач мав повний набір взаємодій, а кожен продукт – коректно підготовлені атрибути. Це забезпечило можливість створення матриць для колаборативного фільтрування та подальшого обчислення подібностей у контентному фільтруванні. Додатково було забезпечено вирівнювання індексів продуктів і користувачів між різними компонентами системи, що є критичним для гібридного підходу.

Таким чином, методи підготовки даних охоплювали обробку неявних і явних взаємодій, трансформацію текстових і числових атрибутів продуктів та узгодження структури даних для подальшої обробки. Ці кроки дозволили створити чистий і повний набір даних, що став основою для побудови якісних рекомендацій у гібридній системі.

### 2.3.3 Методика тестування моделей

Методика тестування моделей є важливим етапом у процесі розробки рекомендаційної системи, оскільки дозволяє оцінити точність, продуктивність і адаптивність використаних методів у різних сценаріях.

Тестування здійснювалося для окремих компонентів системи, таких як колаборативне, контентне та гібридне фільтрування, щоб визначити ефективність кожного методу та їх комбінованого результату.

На початковому етапі тестування було сформовано контрольний набір даних, що складався із відкритих датасетів, збагачених атрибутами товарів і неявними взаємодіями користувачів. Набір даних був розділений на навчальну та тестову вибірки для забезпечення об’єктивної перевірки моделей. Для колаборативного фільтрування навчання моделей виконувалося на 80% даних, тоді як решта 20% використовувалася для перевірки якості прогнозування оцінок користувачів. Для контентного фільтрування тестування ґрунтувалося на обчисленні подібностей між товарами та оцінці релевантності отриманих рекомендацій щодо заданого еталонного продукту.

У випадку колаборативного фільтрування було застосовано крос-валідацію для стабільної перевірки результатів. Тестування моделей експліцитного фільтрування з використанням SVD полягало у порівнянні передбачених оцінок користувачів із фактичними значеннями на тестовій вибірці. Для обчислення точності використовувалися метрики помилок, такі як середньоквадратична помилка (RMSE), згідно з формулою (2.6), та середня абсолютна помилка (MAE), згідно з формулою (2.7), що дозволяють кількісно оцінити різницю між прогнозами моделі та реальними оцінками. Імпліцитне фільтрування, реалізоване на основі ALS, перевірялося шляхом ранжування рекомендацій для кожного користувача. Якість результатів оцінювалася за допомогою метрик, орієнтованих на ранжування, таких як Precision@K, Recall@K, F1-міра, середня точність усереднена за всіма класами (MAP), середній обернений ранг (MRR), та NDCG@K, згідно формул (2.8), (2.9), (2.10), (2.11), (2.13), та (2.14) відповідно.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}}, \tag{2.6}$$

де $y_i$ – реальна оцінка;

$\hat{y}_i$– прогнозована оцінка;

$n$ – кількість оцінок.

$$MAE = \frac{\sum_{i=1}^{n}|y_i - \hat{y}_i|}{n}. \quad (2.7)$$

$$Precision@K = \frac{\text{Кількість релевантних товарів у } Top\text{–}K}{K}, \quad (2.8)$$

де K – кількість рекомендованих об’єктів.

$$Recall@K = \frac{\text{Кількість релевантних товарів у } Top\text{–}K}{\text{Загальна кількість релевантних товарів}}. \quad (2.9)$$

$$F1@K = 2 * \frac{Precision@K * Recall@K}{Precision@K + Recall@K}. \quad (2.10)$$

$$MAP = \frac{1}{Q}\sum_{q=1}^{Q} AP_q, \quad (2.11)$$

де $Q$ – кількість рекомендаційних списків, для яких обчислюється метрика;

$AP_q$ – середня точність (AP) для конкретного запиту q, розраховується згідно з формулою (2.12).

$$AP_q = \frac{1}{R_q} * \sum_{k=1}^{N} P(k) * rel_q(k), \quad (2.12)$$

де $R_q$ – загальна кількість релевантних документів (або елементів) для запиту q;

N – загальна кількість елементів у списку результатів (або верхня межа, наприклад, топ-10);

$P(k)$ – точність (Precision) на позиції k;

$rel_q(k)$ – індикатор релевантності, який дорівнює 1, якщо документ на позиції k є релевантним для запиту q, і 0 в іншому випадку.

$$MRR = \frac{1}{Q}\sum_{q=1}^{Q}\frac{1}{rank_q}, \quad (2.13)$$

де $Q$ – загальна кількість рекомендаційних списків, для яких обчислюється метрика;

$rank_q$ – позиція (ранг) першого релевантного елемента для запиту q.

$$NDCG@K = \frac{DCG@K}{IDCG@K}, \quad (2.14)$$

де $DCG@K$ – зважена сума релевантностей для фактичного ранжування, розраховується згідно з формулою (2.11);

$IDCG@K$ – зважена сума релевантностей для ідеального ранжування, розраховується згідно з формулою (2.12);.

$$DCG@K = \sum_{i=1}^{K}\frac{rel_i}{\log_2(i+1)}, \quad (2.15)$$

де $rel_i$ – релевантність елемента на позиції іі (визначається як числова оцінка товару);

$i$ – позиція елемента у списку (починаючи з 1);

$\log_2(i+1)$ – логарифмічний дільник, що забезпечує дисконтування релевантності: чим далі елемент знаходиться у списку, тим менший внесок він робить у загальну суму.

$$IDCG@K = \sum_{i=1}^{K}\frac{rel_i^{ideal}}{\log_2(i+1)}, \quad (2.16)$$

де $rel_i^{ideal}$ – релевантність елементів, упорядкованих за спаданням значення.

Контентне фільтрування перевірялося шляхом обчислення подібностей між продуктами на основі їхніх текстових описів, бренду та ціни. Методика тестування передбачала вибір контрольного товару, для якого система генерувала список найбільш схожих продуктів. Оцінка якості рекомендацій виконувалася шляхом візуального порівняння схожих продуктів і аналізу того, наскільки вони відповідають очікуванням щодо категорії, бренду чи функціональності. Окрім цього, для підвищення об’єктивності було виконано порівняння результатів системи із загальними тенденціями у вибірці, що дозволило перевірити ефективність моделі на рівні семантичної подібності.

Для гібридної моделі тестування передбачало комбінування результатів контентного та колаборативного фільтрування з використанням вагових коефіцієнтів. У процесі тестування досліджувалося, як зміна ваг впливає на якість рекомендацій. У результаті було визначено оптимальні значення коефіцієнтів для конкретної задачі, що забезпечують баланс між характеристиками продуктів і поведінковими даними користувачів. Для оцінки результатів гібридної системи застосовувалися ті ж метрики, що й для колаборативного фільтрування, такі як Precision@K та Recall@K, оскільки вони дозволяють виміряти, наскільки система успішно рекомендує товари, що відповідають інтересам користувача.

На додаток до оцінки точності було проведено тестування продуктивності системи. Було виміряно час, необхідний для генерації рекомендацій на основі нових даних, а також час відповіді при отриманні запитів через API у режимі реального часу. Продуктивність системи оцінювалася на основі швидкодії Python-модуля під час обробки великих обсягів даних і часового проміжку, потрібного для оновлення рекомендацій у базі даних. Особлива увага приділялася оптимізації роботи з

розрідженими матрицями та використанню ефективних обчислювальних методів для зниження ресурсомісткості.

Методика тестування також передбачала аналіз впливу розміру даних на якість рекомендацій. Було перевірено, як зміна кількості користувачів, товарів та взаємодій впливає на ефективність алгоритмів, а також чи спроможна система масштабуватися без втрати якості. Отримані результати підтвердили стійкість гібридної системи до збільшення розміру даних та її здатність адаптуватися до нових даних за рахунок регулярних оновлень.

Таким чином, методика тестування моделей включала оцінку точності прогнозів, аналіз ранжування рекомендацій, вимірювання продуктивності та перевірку адаптивності системи до змін у даних. Використання різних метрик та підходів забезпечило повноцінну перевірку якості роботи окремих компонентів системи та їх комбінованого результату у гібридній моделі.

### 2.3.4 Оцінка продуктивності системи

Оцінка продуктивності рекомендаційної системи є необхідним етапом для визначення ефективності її роботи у реальних умовах. Продуктивність системи аналізувалася за двома ключовими параметрами: швидкістю обробки даних на етапі генерації рекомендацій та часом відповіді під час отримання результатів у реальному часі. Основна увага приділялася оптимізації обчислень і зменшенню затримок, оскільки швидкість роботи є критично важливою для забезпечення зручності користувача та відповідності вимогам сучасних веб-додатків.

Під час тестування на етапі генерації рекомендацій було досліджено ефективність роботи Python-модуля при обробці великих обсягів даних про взаємодії користувачів та атрибути продуктів. У середовищі тестування модуль виконував обчислення кожні 30 хвилин, що забезпечувало актуальність рекомендацій. Було зафіксовано час, необхідний для

виконання кожного етапу обробки: створення матриць подібності для колаборативного та контентного фільтрування, їхнє вирівнювання та комбінування у гібридну модель. Для прискорення обчислень використовувалися оптимізовані алгоритми, такі як ALS для обробки імпліцитних даних і SVD для експліцитного фільтрування, які ефективно працюють із розрідженими матрицями. Наявність розподілених обчислень у бібліотеці Implicit дозволила мінімізувати витрати часу при обробці великих матриць користувач-продукт.

На етапі підготовки даних було досягнуто додаткової оптимізації завдяки використанню методів лінійної алгебри з бібліотеки NumPy та розріджених матриць із SciPy. Це дозволило суттєво знизити обчислювальне навантаження і забезпечити стабільну продуктивність системи навіть за збільшення обсягів вхідних даних. Також було протестовано вплив розміру матриць на швидкість роботи алгоритмів: результати підтвердили, що система є стійкою до зростання кількості користувачів і продуктів завдяки ефективному використанню пам'яті та обчислювальних ресурсів.

Другий аспект продуктивності системи стосувався часу відповіді при запитах у реальному часі. Під час взаємодії з користувачами веб-додаток передавав запит до Python-модуля через REST API на базі Flask, який повертав готові результати з бази даних SQLite. Завдяки попередньо обчисленим і кешованим рекомендаціям час відповіді був мінімальним. У процесі тестування було виміряно середній час обробки запитів, який становив кілька мілісекунд, що відповідає вимогам до сучасних веб-додатків. Також було перевірено стабільність відповіді системи за умов підвищеного навантаження, що досягалося шляхом паралельного надсилання великої кількості запитів до Flask API. Результати тестування показали, що система витримує навантаження без суттєвого зростання часу відповіді завдяки легковажній архітектурі та ефективній обробці даних у SQLite.

Додатково було оцінено вплив частоти оновлень рекомендацій на загальну продуктивність системи. Автоматичне оновлення даних кожні 30 хвилин забезпечило баланс між актуальністю рекомендацій і ресурсозатратністю обчислень. Під час цього процесу Python-модуль виконував повний цикл обробки даних, включаючи генерацію матриць подібності, їх комбінування у гібридний результат та запис у базу даних. Оптимізація цих операцій дозволила зберегти продуктивність, не створюючи критичного навантаження на інфраструктуру.

Таким чином, оцінка продуктивності системи підтвердила її ефективність як на етапі обробки великих обсягів даних, так і при відповіді у режимі реального часу. Завдяки оптимізованим алгоритмам, використанню розріджених матриць та кешуванню готових рекомендацій система демонструє високу швидкість роботи й стабільну продуктивність, що робить її готовою до використання в умовах реального навантаження.

# 3 РОЗРОБКА РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ

## 3.1 Архітектура системи

Архітектура рекомендаційної системи була спроєктована як модульна, щоб забезпечити її гнучкість, продуктивність і можливість інтеграції з веб-застосунком. Рішення було реалізовано з урахуванням розділення відповідальності між компонентами, що дозволяє кожному модулю ефективно виконувати свою функцію без дублювання завдань.

Центральним компонентом архітектури є Python-модуль, який відповідає за обробку даних, генерацію рекомендацій і їхнє оновлення. Його робота організована у два основні етапи: регулярна підготовка рекомендацій на основі вхідних даних і повернення готових результатів на запит веб-додатка. Архітектура побудована так, щоб мінімізувати затримки на етапі отримання рекомендацій, оскільки всі необхідні обчислення виконуються заздалегідь.

Для збору та передачі даних використовується веб-додаток, який виконує роль джерела інформації про користувачів, товари та їхні взаємодії. веб-додаток передає дані у структурованому форматі через REST API на базі Flask, що забезпечує взаємодію між веб-застосунком і Python-модулем. Дані включають неявні взаємодії користувачів, такі як перегляди, кліки та покупки, а також інформацію про продукти, зокрема текстові описи, бренди та ціни. Передані дані обробляються у Python-модулі для побудови рекомендацій.

На етапі обробки даних Python-модуль формує матриці подібності для різних підходів фільтрування. Контентне фільтрування використовує векторизовані текстові описи продуктів, обчислює їхню подібність і враховує інші атрибути товарів, такі як ціна та бренд. Колаборативне фільтрування будує матриці на основі взаємодій користувачів, що дозволяє виявити приховані патерни у їхніх вподобаннях. Обидві матриці

комбінуються у єдиний гібридний результат за допомогою вагових коефіцієнтів, які визначають баланс між контентним і колаборативним підходами.

Результати обчислень записуються у базу даних SQLite, що діє як локальне сховище рекомендацій. Збереження попередньо обчислених рекомендацій дозволяє знизити навантаження на сервер та забезпечити швидке отримання результатів. Кожні 30 хвилин Python-модуль оновлює рекомендації, виконавши повний цикл обробки нових даних, що дозволяє зберігати актуальність результатів навіть при зміні вподобань користувачів або додаванні нових товарів.

У архітектурі також передбачено швидкий доступ до рекомендацій у режимі реального часу. Коли користувач відкриває сторінку певного продукту у веб-додатку, веб-додаток надсилає запит до Python-модуля через REST API. Python-модуль витягує готові рекомендації із бази даних SQLite і повертає їх у відповідь на запит. Завдяки цій архітектурі система працює швидко, оскільки основні обчислення виконуються заздалегідь, а запити до API обмежуються читанням даних зі сховища.

Архітектура системи є масштабованою, оскільки Python-модуль контейнеризовано за допомогою Docker, що забезпечує стабільну роботу незалежно від середовища виконання. Контейнеризація дозволяє легко масштабувати обчислювальні ресурси у разі збільшення кількості користувачів або обсягів даних. Окрім цього, регулярні оновлення рекомендацій автоматизовані за допомогою cron, що дозволяє мінімізувати ручне втручання у роботу системи [14].

Архітектура рекомендаційної системи передбачає розподіл функціональних компонентів на кілька модулів для забезпечення гнучкості, продуктивності та масштабованості. Окрім основного модуля, що відповідає за гібридну рекомендацію товарів, архітектура включає додаткові сервіси, які розширюють можливості системи: рекомендацію категорій товарів та пошук і фільтрацію товарів.

Модуль рекомендацій категорій товарів інтегрований як окремий компонент системи, що працює паралельно з основним гібридним підходом для товарів. Його основне завдання – формування списку категорій, що є найбільш релевантними для конкретного користувача. Цей функціонал ґрунтується на комбінуванні глобальної популярності категорій та індивідуальних інтересів користувача, обчислених на основі його взаємодій із платформою. Цей компонент системи є важливим для підвищення зручності навігації, особливо для нових користувачів або тих, хто ще не визначився з конкретними товарами.

Модуль пошуку та фільтрації товарів інтегровано для надання користувачеві можливості взаємодіяти з системою в інтерактивному режимі. Цей функціонал дозволяє виконувати точний пошук товарів на основі текстового запиту та застосовувати додаткові фільтри, такі як категорія товару чи ціновий діапазон.

У архітектурі системи цей модуль виконує кілька ключових завдань. Він здійснює підготовку TF-IDF матриці на основі текстових описів товарів, що використовується для обчислення подібності з текстовими запитами. Також модуль відповідає за фільтрацію товарів відповідно до параметрів, переданих у запиті, таких як категорія або цінові обмеження. Після цього формується список результатів пошуку та фільтрації, який ранжується за рівнем релевантності.

Пошук реалізується у реальному часі з використанням TF-IDF та косинусної подібності, що забезпечує високу швидкість відповіді. Фільтрація виконується на рівні структурованих параметрів товарів, після чого результати об'єднуються та повертаються веб-додатку у вигляді списку відфільтрованих і релевантних товарів.

На рисунку 3.1 зображено діаграму архітектури системи.

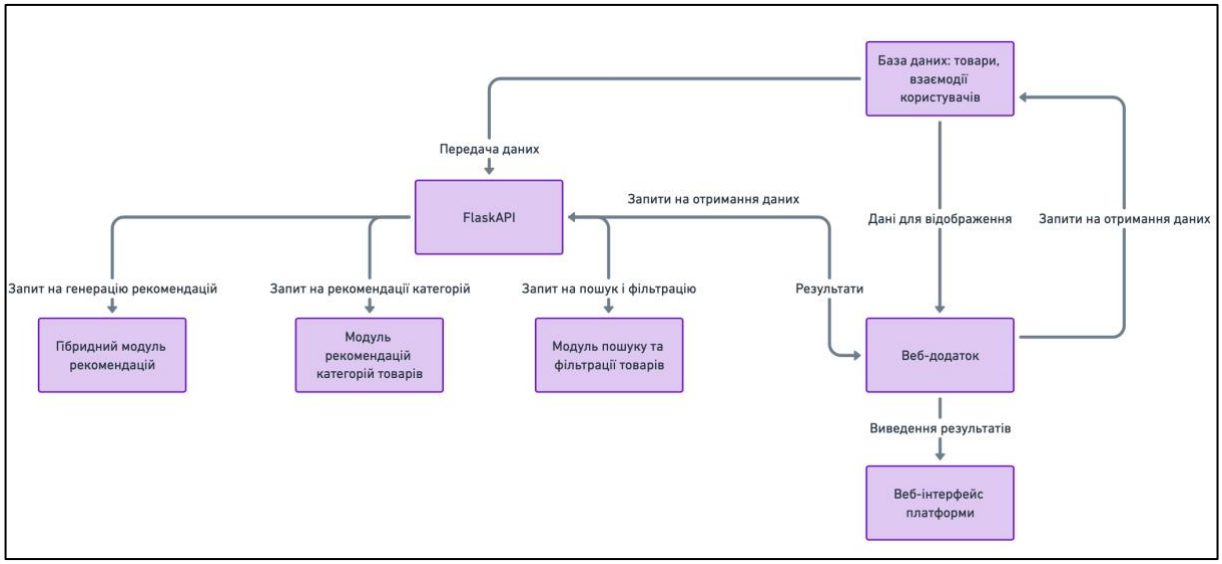


Рисунок 3.1 – Діаграма архітектури системи

Таким чином, архітектура системи поєднує у собі зручність, продуктивність і надійність. Python-модуль відповідає за генерацію та оновлення рекомендацій, тоді як веб-додаток забезпечує їх передачу кінцевим користувачам. Використання попередньо обчислених результатів і локального сховища дозволяє мінімізувати затримки, а контейнеризація системи забезпечує її гнучкість і готовність до масштабування у продакшені.

## 3.2 Реалізація контентного фільтрування

Реалізація контентного фільтрування у розробленій системі базується на використанні текстових описів товарів, їхніх цін і брендів для визначення подібності між продуктами. Це дозволяє генерувати рекомендації, що враховують семантичну та атрибутивну схожість товарів, навіть за відсутності історії взаємодій користувачів.

На першому етапі текстові описи товарів були очищені та доповнені додатковою інформацією, що підвищує якість рекомендацій. Для цього було інтегровано модуль ContentEnhancer, який збагачує текстові описи

технічними характеристиками. Фрагмент наведений у лістингу 3.1 показує, як текстові описи збагачуються технічними характеристиками для створення більш точних рекомендацій. У результаті кожен продукт отримує «збагачений» опис, який слугує основою для подальшої векторизації.

Лістинг 3.1 – Код функції підготовки описів товарів

```
def prepare_descriptions(self, products):
    for product in products:
        enriched_description =
self.enhancer.enrich_description(
            product['description'],
product.get('technical_specs', {})
        )
        product['enriched_description'] =
enriched_description
```

Текстові дані було перетворено у числові вектори за допомогою моделі SentenceTransformer, яка здатна генерувати семантичні вектори на основі архітектури all-MiniLM-L6-v2. Це забезпечує високу якість подібності між описами товарів. Для обчислення подібності між векторами застосовується косинусна подібність, що є основною метрикою для визначення схожості текстів. Код наведено у лістингу 3.2.

Лістинг 3.2 – Код функції визначення схожості текстів товарів

```
self.prepare_descriptions(products)
descriptions = [product['enriched_description'] for
product in products]
embeddings = self.model.encode(descriptions,
convert_to_tensor=True)
description_sim = util.cos_sim(embeddings,
embeddings).cpu().numpy()
return description_sim, products
```

Для підвищення точності рекомендацій враховуються числові та категоріальні атрибути, такі як ціна та бренд. Ціни продуктів нормалізуються у діапазон від 0 до 1 за допомогою MinMaxScaler, а подібність брендів визначається на основі їхнього збігу. Код у лістингу 3.3 показує комбінування трьох компонентів подібності: текстової, цінової та брендової. Кожному з них призначено ваговий коефіцієнт, що дозволяє налаштувати їхній вплив на фінальний результат.

Лістинг 3.3 – Код функції нормалізації параметрів та комбінування коефіцієнтів

```
    brands = np.array([p['brand'] for p in products])
    brand_sim = np.equal.outer(brands, brands).astype(int)
    prices = np.array([p['price'] for p in products])
    scaler = MinMaxScaler()
    normalized_prices = scaler.fit_transform(prices.reshape(-
1, 1)).flatten()
    price_diff = np.abs(normalized_prices[:, None] -
normalized_prices[None, :])
    price_sim = 1 - price_diff
    category = products[0]['category'] if products else None
    weights = self.get_weights(category) if category else
self.default_weights
    combined_similarity = (
        weights['description'] * description_sim +
        weights['brand'] * brand_sim +
        weights['price'] * price_sim
    )
    return combined_similarity, products
```

Фінальна матриця подібності використовується для формування списку рекомендованих товарів. Після обчислення подібностей система сортує продукти за значенням метрики та повертає найбільш схожі товари для заданого цільового продукту. Лістинг 3.4 реалізує функцію для повернення списку товарів, які є найбільш схожими на цільовий продукт.

Лістинг 3.4 – Код функції повернення схожих товарів

```
    combined_similarity, products =
self.compute_similarity_matrix(products)

    id_to_index = {p['id']: i for i, p in enumerate(products)}

    if target_product_id not in id_to_index:
        raise ValueError(f"Target product with ID
{target_product_id} not found in the products list.")

    target_index      = id_to_index[target_product_id]
    similarity_scores = combined_similarity[target_index]
    sorted_indices    = similarity_scores.argsort()[::-1]

    recommendations = [
        {
            "product_id": products[i]['id'],
            "similarity_score": similarity_scores[i]
        }
        for i in sorted_indices if i != target_index
    ]

    return recommendations
```

На рисунку 3.2 представлено теплову карту групування товарів, що наочно демонструє групування товарів за ступенем їхньої схожості, що є основою для подальшої генерації рекомендацій.

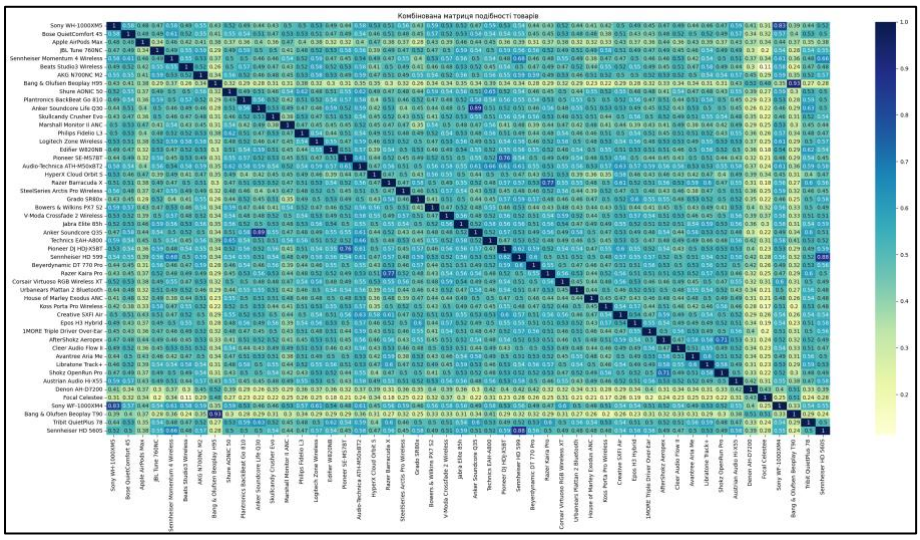


Рисунок 3.2 – Теплова карта групування товарів

Контентне фільтрування реалізовано як процес обчислення подібностей між продуктами на основі їхніх текстових описів, брендів і цін. Використання SentenceTransformer для семантичної векторизації, нормалізації числових атрибутів та комбінування результатів дозволило створити точну й ефективну систему рекомендацій. Ключовим результатом цього підходу є можливість генерувати рекомендації навіть для нових товарів, що не мають історії взаємодій, забезпечуючи гнучкість і надійність системи.

## 3.3 Реалізація колаборативного фільтрування

Колаборативне фільтрування у реалізованій системі використовується для генерації рекомендацій на основі взаємодій користувачів із товарами. Цей підхід фокусується на аналізі даних про явні оцінки та неявні взаємодії, такі як перегляди, кліки та покупки. У результаті система будує матриці подібності, що дозволяють виявити приховані патерни у поведінці користувачів і на цій основі передбачити їхні вподобання. Реалізація методу об’єднує два підходи: експліцитне колаборативне фільтрування на основі SVD та імпліцитне колаборативне фільтрування із використанням ALS. Окремо реалізовано механізм комбінування результатів для формування єдиної матриці рекомендацій.

Дані про неявні взаємодії користувачів з продуктами включають кількість переглядів, кліків та покупок. Для їхньої обробки обчислюються псевдооцінки, що відображають значущість кожного типу взаємодії. Оцінки формуються як лінійна комбінація взаємодій з різними вагами. Лістинг 3.5 відповідає за обробку імпліцитних даних і перетворення їх у єдині псевдооцінки, що дозволяють побудувати матрицю користувач-продукт для подальшого навчання моделей.

Лістинг 3.5 – Код функцій обробки імпліцитних даних та створення псевдооцінки

```
    def _calculate_interactions(self, ratings):
        ratings['views'] = ratings.get('views', 0)
        ratings['clicks'] = ratings.get('clicks', 0)
        ratings['purchases'] = ratings.get('purchases', 0)
        return ratings['views'] * 0.5 + ratings['clicks'] *
0.7 + ratings['purchases'] * 1.0

    def generate_pseudo_ratings(self, implicit_ratings):
        implicit_ratings['pseudo_rating'] =
self._calculate_interactions(implicit_ratings)
        return implicit_ratings[['user_id', 'product_id',
'pseudo_rating']]
```

Оскільки реальні дані можуть містити пропуски, матриця оцінок доповнюється відсутніми значеннями. Заповнення відбувається шляхом генерації повного набору комбінацій користувач-продукт і заміни пропущених оцінок глобальним середнім значенням. Код наведено у лістингу 3.6.

Лістинг 3.6 – Код функцій заповнення матриці оцінок

```
    def fill_missing_explicit_ratings(self, explicit_ratings,
all_items, global_default=3.0):
        users = explicit_ratings['user_id'].unique()
        all_combinations = pd.MultiIndex.from_product([users,
all_items], names=['user_id', 'product_id'])
        full_ratings =
pd.DataFrame(index=all_combinations).reset_index()
        full_ratings = full_ratings.merge(explicit_ratings,
on=['user_id', 'product_id'], how='left')
        full_ratings['rating'] =
full_ratings['rating'].fillna(global_default)
        return full_ratings[['user_id', 'product_id',
'rating']]

    def combine_ratings(self, explicit_ratings,
implicit_ratings, global_default=3.0):
        pseudo_ratings =
self.generate_pseudo_ratings(implicit_ratings)
        combined_ratings = pd.concat([
```

Продовження лістингу 3.6

```
            explicit_ratings[['user_id', 'product_id',
'rating']],
            pseudo_ratings.rename(columns={'pseudo_rating':
'rating'})
        ])
        all_items = combined_ratings['product_id'].unique()
        return
self.fill_missing_explicit_ratings(combined_ratings,
all_items, global_default)
```

Для роботи з явними оцінками використовується алгоритм SVD з бібліотеки Surprise. Модель навчається на навчальній вибірці та перевіряється на тестових даних. Лістинг 3.7 реалізує навчання моделі на основі SVD для точного прогнозування явних оцінок користувачів.

Лістинг 3.7 – Код функції навчання моделі на основі SVD

```
        def run_explicit_cf(self, explicit_ratings):
        reader = Reader(rating_scale=(1, 5))
        surprise_data =
Dataset.load_from_df(explicit_ratings[['user_id',
'product_id', 'rating']], reader)
        trainset, testset = train_test_split(surprise_data,
test_size=0.2)
        model = SVD(n_factors=50, reg_all=0.02, lr_all=0.005,
random_state=42)
        model.fit(trainset)
        predictions = model.test(testset)
        return model, predictions
```

Для обробки неявних взаємодій використовується ALS з бібліотеки Implicit. Метод дозволяє працювати з великими розрідженими матрицями. Код наведено у лістингу 3.8.

Лістинг 3.8 – Код функції обробки неявних взаємодій з використанням

```
    def run_implicit_cf(self, implicit_ratings):
        implicit_ratings['interactions'] =
self._calculate_interactions(implicit_ratings)
```

Продовження лістингу 3.8

```
        user_item_matrix =
implicit_ratings.pivot(index='user_id', columns='product_id',
values='interactions').fillna(0)
        user_item_matrix = np.log1p(user_item_matrix)
        sparse_matrix = csr_matrix(user_item_matrix.values)
        model = AlternatingLeastSquares(factors=10,
regularization=1.0, iterations=10)
        model.fit(sparse_matrix)
        return model, user_item_matrix, sparse_matrix
```

Отримані результати з обох методів комбінуються у єдину матрицю подібності з використанням вагових коефіцієнтів. Код наведено у лістингу 3.9.

Лістинг 3.9 – Код функції комбінування методів у єдину матрицю

```
    def run_combined_cf_matrix(self, explicit_ratings,
implicit_ratings):
        explicit_model, _ =
self.run_explicit_cf(explicit_ratings)
        all_users_explicit =
explicit_ratings['user_id'].unique()
        all_items_explicit =
explicit_ratings['product_id'].unique()
        explicit_scores = np.zeros((len(all_users_explicit),
len(all_items_explicit)))
        user_id_to_index_explicit = {user_id: idx for idx,
user_id in enumerate(all_users_explicit)}
        item_id_to_index_explicit = {item_id: idx for idx,
item_id in enumerate(all_items_explicit)}
        for user_id in all_users_explicit:
            for product_id in all_items_explicit:

explicit_scores[user_id_to_index_explicit[user_id],
item_id_to_index_explicit[product_id]] = \
                    explicit_model.predict(uid=user_id,
iid=product_id).est
```

Продовження лістингу 3.9

```
        scaler = MinMaxScaler(feature_range=(0, 1))
        explicit_scores =
scaler.fit_transform(explicit_scores)
        implicit_model, user_item_matrix, sparse_matrix =
self.run_implicit_cf(implicit_ratings)
        combined_scores = 0.3 * explicit_scores + 0.7 *
user_item_matrix.values
        return combined_scores, all_users_explicit,
all_items_explicit
```

Колаборативне фільтрування реалізовано як поєднання експліцитного та імпліцитного методів. Використання SVD для явних оцінок і ALS для неявних взаємодій забезпечило ефективну побудову рекомендацій для користувачів. Комбінування результатів двох підходів дозволило отримати точні рекомендації, що враховують як явні вподобання користувачів, так і їхню приховану активність у системі.

## 3.4 Реалізація гібридного підходу для рекомендації товарів

Гібридний підхід у реалізованій рекомендаційній системі поєднує результати контентного та колаборативного фільтрування для досягнення високої точності й універсальності рекомендацій. Основна ідея гібридизації полягає у комбінуванні матриць подібності, що отримуються окремо кожним методом, з використанням вагових коефіцієнтів. Це дозволяє враховувати як характеристики товарів, так і вподобання користувачів, компенсуючи обмеження кожного підходу.

Реалізація гібридної системи передбачає вирівнювання та об’єднання матриць подібності, обчислених у процесі контентного й колаборативного фільтрування. На виході формується єдина комбінована матриця, що використовується для побудови остаточних рекомендацій.

Гібридний рекоммендер ініціалізується з визначеними вагами для контентного та колаборативного підходів. Вагові коефіцієнти забезпечують баланс між методами, надаючи пріоритет одному з них залежно від поставлених вимог. У лістингу 3.10 наведено код ініціалізації гібридного рекомендера.

Лістинг 3.10 – Код ініціалізації гібридного рекомендера

```
class HybridRecommender:
    def __init__(self, weights=None):
        self.content_recommender =
ContentBasedRecommender()
        self.collaborative_recommender =
CollaborativeRecommender()
        self.weights = weights or {'content': 0.75,
'collaborative': 0.25}
        self.weights =
self.normalize_weights(self.weights)

    def normalize_weights(self, weights):
        total = sum(weights.values())
        return {k: v / total for k, v in weights.items()}
```

У цьому блоці визначено початкові параметри ваг та ініціалізовано обидва підходи: контентний та колаборативний. Ваги нормалізуються, щоб сума їх значень дорівнювала одиниці.

Для об’єднання результатів двох методів формується загальна матриця подібності, що враховує результати контентного фільтрування та колаборативного фільтрування. Обчислюються окремі матриці для кожного підходу, після чого вони вирівнюються на основі унікальних ідентифікаторів продуктів. Код наведено у лістингу 3.11.

Лістинг 3.11 – Код об’єднання результатів контентного та колабораційного методів

```
def recommend_matrix(self, products, explicit_ratings,
implicit_ratings):
```

Продовження лістингу 3.11

```
        content_matrix,          content_products          =
self.content_recommender.compute_similarity_matrix(products)
        content_product_ids = [product['id'] for product in
content_products]
        collaborative_matrix,            collaborative_users,
collaborative_items                                        =
self.collaborative_recommender.run_combined_cf_matrix(
            explicit_ratings, implicit_ratings
        )
        collaborative_product_ids = list(collaborative_items)
        all_product_ids = list(set(content_product_ids) |
set(collaborative_product_ids))
        product_id_to_index = {product_id: idx for idx,
product_id in enumerate(all_product_ids)}
        n_products = len(all_product_ids)
        aligned_content_matrix = np.zeros((n_products,
n_products))
        aligned_collaborative_matrix = np.zeros((n_products,
n_products))
        for i, product_id_1 in enumerate(content_product_ids):
            for j, product_id_2 in
enumerate(content_product_ids):

aligned_content_matrix[product_id_to_index[product_id_1],
product_id_to_index[product_id_2]] = \
                    content_matrix[i, j]
        for i, product_id_1 in
enumerate(collaborative_product_ids):
            for j, product_id_2 in
enumerate(collaborative_product_ids):

aligned_collaborative_matrix[product_id_to_index[product_id_1]
, product_id_to_index[product_id_2]] = \
                    collaborative_matrix[i, j]
```

Продовження лістингу 3.11

```
        combined_matrix = (
            self.weights['content'] * aligned_content_matrix +
            self.weights['collaborative']                    *
aligned_collaborative_matrix
        )
        return    combined_matrix,    aligned_content_matrix,
aligned_collaborative_matrix, product_id_to_index
```

У цьому фрагменті обчислюються окремі матриці подібності для контентного та колаборативного фільтрування. Матриці вирівнюються шляхом створення загального набору унікальних ідентифікаторів товарів. Результати об’єднуються у загальну матрицю за допомогою вагових коефіцієнтів.

На основі сформованої гібридної матриці система може генерувати рекомендації для конкретного цільового продукту. Список результатів включає як комбіновані оцінки, так і окремі значення для контентного й колаборативного фільтрування. Код наведено у лістингу 3.12.

Лістинг 3.12 – Код генерації гібридних рекомендацій для конкретного продукту

```
    def recommend(self, target_product_id, products,
explicit_ratings, implicit_ratings, user_id):
        content_matrix, content_products =
self.content_recommender.compute_similarity_matrix(products)
        content_product_ids = [product['id'] for product in
content_products]
        collaborative_recommendations =
self.collaborative_recommender.run_combined_cf(
            explicit_ratings, implicit_ratings, user_id
        )
```

Продовження лістингу 3.12

```
        collaborative_scores = {rec['product_id']:
rec['similarity_score'] for rec in
collaborative_recommendations}
        all_product_ids = list(set(content_product_ids) |
set(collaborative_scores.keys()))
        product_id_to_index = {product_id: idx for idx,
product_id in enumerate(all_product_ids)}
        n_products = len(all_product_ids)
        aligned_content_scores = np.zeros(n_products)
        aligned_collaborative_scores = np.zeros(n_products)
        if target_product_id in product_id_to_index:
            target_index =
product_id_to_index[target_product_id]
            for i, product_id in
enumerate(content_product_ids):

aligned_content_scores[product_id_to_index[product_id]] =
content_matrix[content_product_ids.index(target_product_id),
i]
        for product_id, score in collaborative_scores.items():

aligned_collaborative_scores[product_id_to_index[product_id]]
= score
        combined_scores = (
            self.weights['content'] * aligned_content_scores +
            self.weights['collaborative'] *
aligned_collaborative_scores
        )
        sorted_indices = combined_scores.argsort()[::-1]
        recommendations = [
            {
                "product_id": all_product_ids[i],
                "combined_similarity_score":
combined_scores[i],
```

Продовження лістингу 3.12

```
                "content_score": aligned_content_scores[i],
                "collaborative_score":
aligned_collaborative_scores[i],
            }
            for i in sorted_indices if all_product_ids[i] !=
target_product_id
        ]
        return recommendations
```

Ця функція генерує остаточний список рекомендацій для конкретного продукту та користувача, об’єднуючи результати обох підходів.

Гібридний підхід реалізовано шляхом об’єднання результатів контентного та колаборативного фільтрування у єдину матрицю подібності з урахуванням вагових коефіцієнтів. Процес включає вирівнювання матриць на основі унікальних ідентифікаторів продуктів, комбінування значень та генерацію остаточних рекомендацій. Використання цього підходу дозволяє забезпечити високу точність і релевантність рекомендацій, враховуючи як характеристики товарів, так і поведінкові патерни користувачів.

## 3.5 Реалізація гибридного підходу для рекомендації категорій

Гібридний підхід для рекомендації категорій відповідає за рекомендацію категорій товарів, комбінуючи глобальну популярність категорій та індивідуальні інтереси користувача. На відміну від рекомендацій конкретних продуктів, цей підхід фокусується на рівні ієрархії товарів, допомагаючи користувачеві орієнтуватися у великій кількості категорій на основі його історії взаємодій.

Основна логіка роботи полягає у розрахунку двох показників для кожної категорії: популярності категорії серед усіх користувачів та інтересу конкретного користувача до категорії. Ці показники комбінуються за

допомогою вагових коефіцієнтів, що дозволяє врахувати як загальні тенденції, так і індивідуальну активність користувача.

Все починається з ініціалізації ваг, які визначають вплив популярності категорії та інтересів користувача на кінцевий результат. Ваги нормалізуються для забезпечення їх коректного комбінування. Код наведено у лістингу 3.13.

Лістинг 3.13 – Код ініціалізації гібридного рекомендера для категорій

```
class HybridCategoryRecommender:
    def __init__(self, weights=None):
        self.weights = weights or {'popularity': 0.2,
'user_interest': 0.8}
        self.weights =
self.normalize_weights(self.weights)

    def normalize_weights(self, weights):
        total = sum(weights.values())
        return {k: v / total for k, v in weights.items()}
```

Глобальна популярність категорій визначається на основі агрегованих взаємодій усіх користувачів. Для цього кожному типу взаємодії надається вага, і для кожної категорії обчислюється загальний бал. Код наведено у лістингу 3.14.

Лістинг 3.14 – Код розрахунку глобальних інтересів

```
def calculate_category_popularity(self, interactions):
    interactions['score'] = (interactions['views'] * 0.5 +
                             interactions['clicks'] *
0.7 +
                             interactions['purchases']
* 1.0)
    category_popularity =
interactions.groupby('category')['score'].sum().reset_index()
    scaler = MinMaxScaler()
```

Продовження лістингу 3.14

```
        category_popularity['popularity_score'] =
scaler.fit_transform(category_popularity[['score']])
        return category_popularity[['category',
'popularity_score']]
```

Цей код створює загальний рейтинг категорій на основі суми зважених взаємодій. Нормалізація значень дозволяє уникнути дисбалансу між категоріями з різними масштабами даних.

Індивідуальні інтереси користувача обчислюються шляхом аналізу його взаємодій із категоріями товарів. Бали розраховуються аналогічно глобальній популярності, але з фільтрацією лише для конкретного користувача. Код наведено у лістингу 3.15.

Лістинг 3.15 – Код розрахунку індивідуальних інтересів

```
    def calculate_user_category_interest(self, interactions,
user_id):
        user_data = interactions[interactions['user_id'] ==
user_id]
        if user_data.empty:
            return pd.DataFrame(columns=['category',
'user_interest_score'])
        user_data['score'] = (user_data['views'] * 0.5 +
                              user_data['clicks'] * 0.7 +
                              user_data['purchases'] * 1.0)
        user_interest =
user_data.groupby('category')['score'].sum().reset_index()
        scaler = MinMaxScaler()
        user_interest['user_interest_score'] =
scaler.fit_transform(user_interest[['score']])
        return user_interest[['category',
'user_interest_score']]
```

Глобальна популярність категорій та індивідуальні інтереси користувача об'єднуються у єдиний показник шляхом зваженого комбінування. Категорії ранжуються за комбінованим балом, а результат повертається у вигляді списку рекомендованих категорій. Код наведено у лістингу 3.16.

Лістинг 3.16 – Код розрахунку рекомендованих категорій

```
    def recommend_categories(self, interactions, user_id):
        popularity_df =
self.calculate_category_popularity(interactions)
        user_interest_df =
self.calculate_user_category_interest(interactions, user_id)
        combined_df = pd.merge(popularity_df,
user_interest_df, on='category', how='outer').fillna(0)
        combined_df['combined_score'] = (
            self.weights['popularity'] *
combined_df['popularity_score'] +
            self.weights['user_interest'] *
combined_df['user_interest_score']
        )
        recommendations =
combined_df.sort_values(by='combined_score', ascending=False)
        return recommendations[['category',
'combined_score']].to_dict(orient='records')
```

У результаті формується зведена таблиця, де для кожної категорії обчислено підсумковий бал, що враховує обидва фактори. Список категорій сортується за спаданням значення комбінованого балу, після чого повертається у форматі JSON або списку.

Розроблений фільтр генерує рекомендації категорій, які мають найвищий інтерес для конкретного користувача, при цьому враховуючи як його індивідуальну активність, так і загальні тенденції платформи. Ця функціональність корисна для підвищення релевантності інтерфейсу, оскільки користувач може швидко орієнтуватися серед найбільш цікавих для нього категорій.

## 3.6 Реалізація механізму пошуку та фільтрації товарів

Модуль SearchAndFilterRecommender реалізує механізм пошуку та фільтрації товарів для інтернет-магазину, дозволяючи користувачам знаходити релевантні продукти на основі текстових запитів і заданих параметрів, таких як категорія чи ціновий діапазон. Основна роль цього модуля полягає у розширенні функціоналу рекомендаційної системи шляхом надання користувачам можливості виконувати цілеспрямований пошук товарів. У поєднанні з основним гібридним підходом, пошукова система підвищує зручність навігації, забезпечуючи гнучкість у взаємодії з платформою.

Пошук реалізовано на основі TF-IDF, що дозволяє обчислювати подібність між запитом користувача та текстовими описами товарів. Фільтрація враховує такі параметри, як категорія та ціновий діапазон.

На початковому етапі модуль ініціалізується з набором продуктів, що містять інформацію про ідентифікатор товару, назву, опис, категорію та ціну. Підготовка TF-IDF матриці забезпечує можливість швидкого пошуку за текстовим запитом. Код наведено у лістингу 3.17.

Лістинг 3.17 – Код ініціалізації пошукового рекомендера

```
class SearchAndFilterRecommender:
    def __init__(self, product_data):
        self.product_data = product_data
        self.tfidf_matrix = None
        self.vectorizer = None
```

Для обчислення подібності між текстовими запитами та описами товарів використовується TF-IDF векторизація, що перетворює текстові дані у числові вектори. Матриця TF-IDF обчислюється один раз на основі всіх описів товарів. Код наведено у лістингу 3.18.

Лістинг 3.18 – Код векторизації описів продуктів

```
def prepare_tfidf_matrix(self):
    self.vectorizer =
TfidfVectorizer(stop_words='english')
    self.tfidf_matrix =
self.vectorizer.fit_transform(self.product_data['description']
.fillna(""))
```

При надходженні текстового запиту модуль обчислює косинусну подібність між запитом та векторизованими описами товарів. Товари сортуються за значенням подібності у порядку спадання, після чого повертаються найбільш релевантні результати. Код наведено у лістингу 3.19.

Лістинг 3.19 – Код функції пошуку продуктів

```
def search_products(self, query, top_n=5):
    if self.tfidf_matrix is None:
        self.prepare_tfidf_matrix()
    query_vector = self.vectorizer.transform([query])
    similarities = cosine_similarity(query_vector,
self.tfidf_matrix).flatten()
    self.product_data['similarity_score'] = similarities
    results =
self.product_data.sort_values(by='similarity_score',
ascending=False).head(top_n)
    return results[['product_id', 'name', 'category',
'price', 'similarity_score']]
```

Окрім пошуку, модуль підтримує фільтрацію товарів за категорією та ціновим діапазоном. Фільтрація виконується на основі параметрів, переданих користувачем, а результати можуть бути додатково відсортовані за схожістю до текстового запиту. Код наведено у лістингу 3.20.

Лістинг 3.20 – Код функції фільтрації на основі параметрів

```
    def filter_and_recommend(self, category=None,
price_range=None, query=None, top_n=5):
        filtered_data = self.product_data
        if category:
            filtered_data =
filtered_data[filtered_data['category'] == category]
        if price_range:
            filtered_data = filtered_data[
                (filtered_data['price'] >= price_range[0]) &
                (filtered_data['price'] <= price_range[1])
            ]
        if query:
            if self.tfidf_matrix is None:
                self.prepare_tfidf_matrix()
            query_vector = self.vectorizer.transform([query])
            similarities = cosine_similarity(query_vector,
self.vectorizer.transform(filtered_data['description'].fillna(
""))).flatten()
            filtered_data = filtered_data.copy()
            filtered_data['similarity_score'] = similarities
            filtered_data =
filtered_data.sort_values(by='similarity_score',
ascending=False)
        return filtered_data.head(top_n)[['product_id',
'name', 'category', 'price']]
```

## 3.7 Побудова API для взаємодії з веб-додатком

Побудова API є невід’ємною частиною архітектури рекомендаційної системи, оскільки забезпечує передачу даних між веб-додатком та Python-модулем. У цій роботі для реалізації API використовується Flask, легкий і швидкий фреймворк для побудови RESTful сервісів у Python. API відповідає

за обробку запитів від зовнішнього веб-додатка та повернення результатів у форматі, зручному для використання на фронтенді веб-застосунку.

Основна задача API полягає в тому, щоб забезпечити доступ до попередньо обчислених рекомендацій, збережених у базі даних SQLite, а також дозволити обробку запитів для генерації нових рекомендацій у реальному часі. Сервіс оптимізовано для швидкої відповіді з мінімальним часом обробки.

Flask-додаток ініціалізується як REST API-сервіс, що обробляє HTTP-запити від веб-додатка інтернет магазину. Основні ендпоінти включають маршрути для отримання рекомендацій за конкретним товаром та для конкретного користувача.

Маршрут /recommendations/product/<int:product_id> надає попередньо обчислені рекомендації для конкретного товару. Результати зберігаються у базі даних SQLite під час регулярного оновлення рекомендацій Python-модулем. При отриманні запиту API підключається до бази, витягує релевантні результати для вказаного product_id та повертає їх у форматі JSON. Код маршруту наведено у лістингу 3.21.

Лістинг 3.21 – Код маршруту для надання рекомендацій для певного продукту

```
     @app.route('/recommendations/product/<int:product_id>',
methods=['GET'])
     def get_product_recommendations(product_id):
         conn = get_db_connection()
         cursor = conn.cursor()
         cursor.execute("SELECT * FROM recommendations WHERE
target_product_id = ?", (product_id,))
         rows = cursor.fetchall()
         conn.close()
         recommendations = [{"product_id":
row["recommended_product_id"], "score": row["score"]} for row
in rows]
         return jsonify(recommendations)
```

Ендпоінт /recommendations/user/<int:user_id> забезпечує генерацію рекомендацій у реальному часі для конкретного користувача на основі його явних оцінок, неявних взаємодій та даних про товари. Запит передає JSON-об’єкт, що містить списки оцінок та продуктів. Python-модуль викликає метод recommend гібридного рекоммендера, який повертає сформований список рекомендацій у форматі JSON. Код маршруту наведено у лістингу 3.22.

Лістинг 3.22 – Код маршруту для надання рекомендацій для певного користувача

```
     @app.route('/recommendations/user/<int:user_id>',
methods=['POST'])
     def get_user_recommendations(user_id):
         explicit_ratings =
request.json.get('explicit_ratings', [])
         implicit_ratings =
request.json.get('implicit_ratings', [])
         products = request.json.get('products', [])
         recommendations = recommender.recommend(
             target_product_id=None,
             products=products,
             explicit_ratings=explicit_ratings,
             implicit_ratings=implicit_ratings,
             user_id=user_id
         )
         return jsonify(recommendations)
```

Ендпоінт /recommendations/categories/<int:user_id> надає користувачу список категорій, що найбільше відповідають його інтересам, враховуючи як глобальну популярність категорій, так і його індивідуальну активність. На основі даних про перегляди, кліки та покупки Python-модуль обчислює інтегровані бали для кожної категорії та повертає їх у ранжованому порядку. Код маршруту наведено у лістингу 3.23.

Лістинг 3.23 – Код маршруту для надання рекомендованих категорій

```
@app.route('/recommendations/categories/<int:user_id>',
methods=['GET'])
def recommend_categories(user_id):
    interactions =
pd.DataFrame(request.json['interactions'])
    recommender = HybridCategoryRecommender()
    recommendations =
recommender.recommend_categories(interactions, user_id)
    return jsonify(recommendations)
```

Ендпоінт /search обробляє запити користувачів для пошуку товарів за текстовим запитом із можливістю застосування фільтрів за категорією та ціновим діапазоном. Цей функціонал забезпечує гнучку навігацію та дозволяє користувачу самостійно впливати на результати пошуку. Код маршруту наведено у лістингу 3.24.

Лістинг 3.24 – Код маршруту для пошуку та фільтрації

```
@app.route('/search', methods=['POST'])
def search_products():
    query = request.json.get('query', "")
    category = request.json.get('category')
    price_range = request.json.get('price_range')
    recommender =
SearchAndFilterRecommender(pd.DataFrame(request.json['products
'']))
    results =
recommender.filter_and_recommend(category=category,
price_range=price_range, query=query)
    return jsonify(results.to_dict(orient='records'))
```

Веб-додаток взаємодіє з Flask API через HTTP-запити. При завантаженні сторінки продукту або профілю користувача додаток надсилає відповідний запит до API та отримує рекомендації для відображення на фронтенді.

Для збереження високої швидкодії використовується підготовлена база даних SQLite, що містить попередньо обчислені рекомендації. У

випадку запитів у реальному часі обробка виконується швидко завдяки оптимізованим методам колаборативного та контентного фільтрування. Flask API повертає відповідь у мінімальні терміни, що забезпечує зручність користувача.

Побудова API на основі Flask дозволила інтегрувати рекомендаційну систему з веб-додатком. Архітектура API підтримує як попередньо обчислені рекомендації для окремих продуктів, так і генерацію результатів у реальному часі для користувачів. Завдяки зручному та швидкому обміну даними через REST API система забезпечує стабільну продуктивність і гнучкість у використанні рекомендаційних сервісів.

## 3.8 Реалізація оновлення рекомендацій

Для забезпечення актуальності рекомендацій було реалізовано механізм автоматичного оновлення даних та обчислення нових результатів кожні 30 хвилин. Оновлення виконується у фоновому режимі, що дозволяє зберігати баланс між продуктивністю системи та актуальністю результатів. Основні кроки оновлення включають отримання нових даних про взаємодії користувачів і товари з веб-додатка, обробку цих даних у Python-модулі, генерацію нових рекомендацій та їх збереження у базу даних для подальшого використання.

Для автоматизації процесу оновлення використовується cron або інші планувальники завдань, що викликають Python-скрипт із вказаним інтервалом часу. Python-модуль обробляє дані, формує матриці подібності для контентного та колаборативного фільтрування, комбінує результати у гібридну модель та зберігає їх у базу даних SQLite. Код виклику python скрипта наведено у лістингу 3.25.

Лістинг 3.25 – Код cron-задачі для оновлення рекомендацій

```
*/30 * * * * /usr/bin/python3
/path/to/project/update_recommendations.py
```

У скрипті реалізовано повний цикл оновлення: отримання даних, обробка, генерація рекомендацій та їх збереження у базу даних.

Скрипт підключається до веб-додатка через API для отримання актуальних даних. Дані про взаємодії користувачів (експліцитні оцінки, неявні дії) та продукти передаються у форматі JSON. Дані конвертуються у DataFrame для подальшої обробки. Код наведено у лістингу 3.26.

Лістинг 3.26 – Код підключення до зовнішнього веб-додатку

```
def fetch_data_from_api():
    url = "http://localhost:3000/api/data"
    response = requests.get(url)
    data = response.json()
    products = data['products']
    explicit_ratings =
pd.DataFrame(data['explicit_ratings'])
    implicit_ratings =
pd.DataFrame(data['implicit_ratings'])
    return products, explicit_ratings, implicit_ratings
```

Після отримання даних викликається метод recommend гібридного рекоммендера для генерації нових рекомендацій для кожного продукту. Результати комбінуються на основі вагових коефіцієнтів, а для кожного товару формується список найбільш схожих об'єктів. Код наведено у лістингу 3.27.

Лістинг 3.27 – Код оновлення рекомендацій відповідно до даних з веб-додатку

```
def update_recommendations():
    recommender = HybridRecommender()
    products, explicit_ratings, implicit_ratings =
fetch_data_from_api()
    product_ids = [product['id'] for product in products]
```

Продовження лістингу 3.27

```
        for product_id in product_ids:
            recommendations = recommender.recommend(
                target_product_id=product_id,
                products=products,
                explicit_ratings=explicit_ratings,
                implicit_ratings=implicit_ratings,
                user_id=None
            )
            save_recommendations_to_db(recommendations,
product_id)
```

Рекомендації зберігаються у базі даних SQLite. Перед записом нових даних для конкретного товару старі значення видаляються, щоб забезпечити актуальність результатів. Код наведено у лістингу 3.27.

Лістинг 3.27 – Код збереження рекомендацій у БД

```
    def save_recommendations_to_db(recommendations,
product_id):
        conn = sqlite3.connect('recommendations.db')
        cursor = conn.cursor()
        cursor.execute("DELETE FROM recommendations WHERE
target_product_id = ?", (product_id,))
        for rec in recommendations:
            cursor.execute(
                "INSERT INTO recommendations
(target_product_id, recommended_product_id, score) VALUES (?,
?, ?)",
                (product_id, rec['product_id'],
rec['combined_similarity_score'])
            )
        conn.commit()
        conn.close()
```

Система оновлення забезпечує регулярну актуалізацію рекомендацій кожні 30 хвилин. Дані про нові взаємодії користувачів і додані товари обробляються Python-модулем, після чого згенеровані рекомендації заміщують старі значення у базі даних SQLite. Завдяки використанню

планувальника завдань процес оновлення виконується автоматично, мінімізуючи необхідність ручного втручання.

Оновлення рекомендацій категорій здійснюється паралельно з оновленням основних товарних рекомендацій. Python-модуль отримує дані про взаємодії користувачів (перегляди, кліки, покупки), обчислює популярність категорій та індивідуальні інтереси користувачів, а потім комбінує результати для кожного користувача. Код наведено у лістингу 3.28.

Лістинг 3.28 – Код оновлення рекомендацій для категорій

```
def update_category_recommendations():
    url = "http://localhost:3000/api/interactions"
    response = requests.get(url)
    interactions =
pd.DataFrame(response.json()['interactions'])
    users = interactions['user_id'].unique()
    recommender = HybridCategoryRecommender()
    conn = sqlite3.connect('recommendations.db')
    cursor = conn.cursor()
    cursor.execute("DELETE FROM category_recommendations")
    for user_id in users:
        recommendations =
recommender.recommend_categories(interactions, user_id)
        for rec in recommendations:
            cursor.execute(
                "INSERT INTO category_recommendations
(user_id, category, score) VALUES (?, ?, ?)",
                (user_id, rec['category'],
rec['combined_score'])
            )
    conn.commit()
    conn.close()
```

Оновлення TF-IDF матриці та параметрів фільтрації для товарів також здійснюється регулярно. Код наведено у лістингу 3.29.

Лістинг 3.29 – Код індексації даних для пошуку

```
def update_search_index():
    url = "http://localhost:3000/api/products"
```

Продовження лістингу 3.29

```
        response = requests.get(url)
        products = pd.DataFrame(response.json()['products'])
        recommender = SearchAndFilterRecommender(products)
        recommender.prepare_tfidf_matrix()
        conn = sqlite3.connect('recommendations.db')
        cursor = conn.cursor()
        cursor.execute("DELETE FROM search_index")
        for _, row in products.iterrows():
            cursor.execute(
                "INSERT INTO search_index (product_id, name,
description, category, price) VALUES (?, ?, ?, ?, ?)",
                (row['product_id'], row['name'],
row['description'], row['category'], row['price'])
            )
        conn.commit()
        conn.close()
```

Python-модуль отримує актуальні дані про товари, будує нову TF-IDF матрицю та забезпечує можливість пошуку з урахуванням останніх змін у каталозі.

# 4 ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ

## 4.1 Постановка задачі для експериментального дослідження

Проведення експериментального дослідження є важливим етапом для оцінки ефективності розробленої рекомендаційної системи, визначення її продуктивності та виявлення переваг гібридного підходу у порівнянні з окремими методами контентного та колаборативного фільтрування. Основною задачею є підтвердження гіпотези про те, що комбінування цих двох підходів дозволяє підвищити точність і релевантність рекомендацій, забезпечуючи при цьому гнучкість системи в роботі з різними типами даних та користувацькими сценаріями.

Першочерговим завданням дослідження є оцінка якості рекомендацій, що генеруються системою, у різних режимах роботи. Для цього потрібно перевірити три основні підходи: контентне фільтрування, колаборативне фільтрування та гібридний підхід, об'єднуючи результати обчислень із використанням вагових коефіцієнтів. Важливо оцінити, наскільки ефективно система визначає релевантність товарів для користувача, зокрема при наявності історичних даних (експліцитних оцінок) і взаємодій (неявних сигналів, таких як перегляди, кліки та покупки).

Для проведення дослідження необхідно визначити тестовий набір даних, що включає інформацію про товари, користувачів та їх взаємодії. Вхідні дані повинні містити текстові описи товарів, категорії, бренди та цінові атрибути, що використовуються у контентному фільтруванні. Також повинні бути представлені експліцитні оцінки товарів користувачами, виражені числовими значеннями, та неявні дані про активність користувачів, які використовуються у колаборативному фільтруванні. Наявність цих даних є необхідною умовою для побудови як окремих моделей, так і їхнього об'єднання у гібридний підхід.

Наступним завданням дослідження є порівняння отриманих рекомендацій між різними підходами для конкретних користувачів і товарів. Це дозволяє визначити, наскільки комбінування результатів контентного та колаборативного фільтрування дозволяє компенсувати обмеження кожного з методів. Для користувачів із достатньою історією взаємодій буде досліджено, чи гібридний підхід враховує індивідуальні вподобання краще, ніж традиційне колаборативне фільтрування. Для нових користувачів або товарів, де бракує історичних даних, потрібно перевірити, чи контентний компонент системи дозволяє успішно генерувати релевантні рекомендації.

Крім точності рекомендацій, важливою задачею дослідження є оцінка продуктивності системи на етапі обчислення рекомендацій та часу відповіді на запити. Система повинна демонструвати здатність працювати у режимі реального часу, забезпечуючи мінімальні затримки при обробці запитів і поверненні результатів. Зокрема, необхідно виміряти час, що витрачається на підготовку рекомендацій у фоновому режимі, та час обробки індивідуальних запитів до API у реальному часі.

У ході дослідження слід також розглянути ефективність додаткових модулів системи, таких як рекомендація категорій товарів і пошук із фільтрацією. Важливо визначити, наскільки ці модулі доповнюють основний гібридний підхід, покращуючи загальний користувацький досвід. Для цього потрібно перевірити, чи рекомендації категорій товарів допомагають користувачам орієнтуватися на платформі, а також чи пошукова система забезпечує точність і релевантність результатів при використанні текстових запитів та фільтрації.

Таким чином, у процесі постановки задачі для експериментального дослідження сформульовано декілька ключових аспектів: оцінка якості рекомендацій у різних режимах роботи, вимірювання продуктивності системи, аналіз ефективності додаткових модулів та порівняння результатів між окремими і гібридним підходами. Основною метою є підтвердження

гіпотези про те, що гібридний підхід перевершує окремі методи у точності та адаптивності, що робить його ефективним рішенням для інтернет-магазину.

## 4.2 Підготовка даних для експериментів

Підготовка даних є одним із найважливіших етапів експериментального дослідження, оскільки якість вхідних даних значною мірою визначає точність і релевантність отриманих результатів. Для проведення експериментів було використано структуровані дані про користувачів, товари та їхні взаємодії, які імітують реальні умови функціонування інтернет-магазину.

Для забезпечення реалістичності експериментального дослідження та уникнення тестування на синтетичних даних було використано чотири відкриті датасети з платформи Kaggle, що містять дані з реальних e-commerce платформ. Ці датасети включають інформацію про товари, історію покупок та дії користувачів, що дозволяє створити якісний і репрезентативний набір даних для навчання та тестування рекомендаційної системи.

Першим джерелом даних є датасет «Cellphones Recommendations», який містить структуровану інформацію про мобільні телефони [21]. Цей набір даних включає текстові описи товарів, технічні характеристики, бренди та ціни, що робить його ідеальним для контентного фільтрування та реалізації пошуку товарів за текстовими запитами. Окрім цього, ці дані є основою для побудови матриць подібності між продуктами.

Другим джерелом є датасет «Ecommerce Purchase History from Electronics Store», який містить історію покупок користувачів [19]. Ці дані включають унікальні ідентифікатори користувачів і товарів, а також числові оцінки покупок у вигляді суми замовлень. Цей набір даних використовується для формування явних оцінок у колаборативному

фільтруванні та дозволяє аналізувати вподобання користувачів на основі їхніх реальних транзакцій.

Третім джерелом є датасет «Ecommerce Events History in Electronics Store», що містить дані про неявні взаємодії користувачів із платформою [18]. Події, такі як перегляди товарів, додавання до кошика та покупки, дозволяють створювати псевдооцінки, які є основою для імпліцитного колаборативного фільтрування. Ці дані допомагають моделювати поведінку користувачів та визначати рівень їхньої зацікавленості у товарах.

Четвертим джерелом є датасет «Cellphones Stocks from Americanas», який містить додаткові дані про наявність товарів на складі, їхні бренди та ціни [10]. Ці дані використовуються для фільтрації товарів за ціновими діапазонами та категоріями, а також для підвищення релевантності результатів пошуку та рекомендацій.

Для створення єдиного набору даних було виконано інтеграцію інформації з усіх чотирьох джерел. Дані про товари були збагачені описами, технічними характеристиками, брендами та ціновими атрибутами, отриманими з датасетів.

Перша група даних представлена основними атрибутами товарів, які використовуються для контентного фільтрування, пошуку та фільтрації. Ці дані включають текстові характеристики, такі як назва, опис і бренд, числові показники, такі як ціна, а також категорії для ієрархічної організації товарів. Опис групи даних наведено у таблиці 4.1.

Друга група даних стосується взаємодій користувачів із товарами. Вона включає як експліцитні оцінки (рейтинги товарів), так і імпліцитні дані (перегляди, кліки, покупки), які є основою для колаборативного фільтрування. Ці дані дозволяють аналізувати вподобання користувачів і формувати персоналізовані рекомендації. Опис групи даних наведено у таблиці 4.2.

Третя група даних містить агреговані характеристики користувачів, які використовуються для додаткових функцій системи, таких як персоналізовані рекомендації категорій. Дані цієї групи включають фінансові показники, частоту покупок і середню вартість замовлень, що дозволяє глибше аналізувати поведінку користувачів і адаптувати рекомендації до їхніх потреб. Опис групи даних наведено у таблиці 4.3.

Таблиця 4.1 – Дані про товари

| Атрибут | Опис | Тип даних |
|---|---|---|
| ID товару | Унікальний ідентифікатор кожного товару. | Цілочисловий |
| Назва товару | Текстовий заголовок, що відображає найменування товару. | Текстовий |
| Опис товару | Розширений текстовий опис із ключовими характеристиками продукту. | Текстовий |
| Бренд товару | Категоріальна ознака для аналізу схожості між продуктами одного виробника. | Текстовий |
| Ціна товару | Числовий показник, що використовується для обчислення цінової подібності. | Числовий |
| Категорія товару | Класифікаційна ознака для організації товарів у ієрархічній структурі. | Текстовий |

Таблиця 4.2 – Дані про взаємодії користувачів

| Атрибут | Опис | Тип даних |
|---|---|---|
| ID користувача | Унікальний ідентифікатор кожного користувача. | Цілочисловий |
| ID товару | Ідентифікатор товару, з яким взаємодіяв користувач. | Цілочисловий |
| Оцінки | Явні (експліцитні) оцінки товарів (від 1 до 5). | Числовий |
| Перегляди | Кількість переглядів товару користувачем (імпліцитні дані). | Цілочисловий |
| Кліки | Кількість кліків на товар користувачем (імпліцитні дані). | Цілочисловий |
| Покупки | Кількість покупок товару користувачем (імпліцитні дані). | Цілочисловий |

Таблиця 4.3 – Дані про користувачів

| Атрибут | Опис | Тип даних |
|---|---|---|
| Загальна сума витрат | Підсумок усіх покупок користувача на платформі. | Числовий |
| Частота покупок | Кількість транзакцій за певний період. | Цілочисловий |
| Середня вартість замовлення | Середній чек користувача за всі його покупки. | Числовий |

Дані були підготовлені шляхом об'єднання, очищення, трансформації та нормалізації, що забезпечує їхню коректну обробку у всіх модулях системи.

На етапі очищення були видалені дублікатні записи та некоректні значення. Для цього було проведено перевірку на унікальність ідентифікаторів товарів і користувачів, а також видалено записи з відсутніми критичними значеннями, такими як ID товару, опис або оцінка. Код наведено у лістингу 4.1.

Лістинг 4.1 – Код очищення даних

```
products.dropna(subset=['id', 'description'],
inplace=True)
products = products.drop_duplicates(subset=['id'])
interactions = interactions.dropna(subset=['user_id',
'product_id'])
```

Текстові описи товарів було підготовлено для подальшої векторизації. З текстів були видалені стоп-слова та зайві символи. Описи було збагачено технічними характеристиками за допомогою додаткових даних. Код наведено у лістингу 4.2.

Лістинг 4.2 – Код підготовки описів товарів

```
products['description'] =
products['description'].str.lower().replace(r'[^a-z0-9 ]', '',
regex=True)
products['enriched_description'] = products['description']
+ " " + products['technical_specs'].fillna("")
```

Для імпліцитних даних взаємодій було згенеровано псевдооцінки, що враховують ваги для переглядів, кліків та покупок. Це дозволило отримати числові оцінки для побудови матриці користувач-продукт у колаборативному фільтруванні. Код наведено у лістингу 4.3.

Лістинг 4.3 – Код генерації псевдооцінок

```
interactions['pseudo_rating'] = interactions['views'] *
0.5 + interactions['clicks'] * 0.7 + interactions['purchases']
* 1.0
```

Цінові дані товарів та агреговані характеристики користувачів були нормалізовані у діапазоні від 0 до 1 для забезпечення їхньої сумісності з іншими атрибутами під час обчислень. Код наведено у лістингу 4.4.

Лістинг 4.4 – Код нормалізації цін товарів

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
products['normalized_price'] =
scaler.fit_transform(products[['price']])
```

На основі підготовлених даних було сформовано три окремих тестових набори. Перший набір призначений для оцінки ефективності контентного фільтрування і містить детальну інформацію про товари, таку як унікальний ідентифікатор, опис, бренд та ціну. Другий набір, створений для тестування колаборативного фільтрування, являє собою матрицю, що відображає взаємодію користувачів з товарами. Третій набір даних використовується для перевірки функціоналу пошуку та фільтрації і включає текстові описи товарів, їх категорії та цінові діапазони.

Для проведення експериментів дані було розділено на навчальну та тестову вибірки. Навчальна вибірка містить 80% даних і використовується для побудови моделей, тоді як тестова вибірка включає решту 20% даних і призначена для валідації результатів. Розподіл здійснювався з урахуванням рівномірного представлення всіх товарів і користувачів. Код наведено у лістингу 4.5.

Лістинг 4.5 – Код розподілу даних на навчальну та тестову вибірки

```
from sklearn.model_selection import train_test_split
train_data, test_data = train_test_split(interactions,
test_size=0.2, random_state=42)
```

У результаті підготовки було сформовано чисті, нормалізовані та структуровані дані, готові для використання у різних модулях системи. Зокрема, модуль контентного фільтрування оперує текстовими описами товарів, які доповнені технічними характеристиками та уніфікованими цінами. Колаборативне фільтрування використовує оброблені явні оцінки користувачів (наприклад, рейтинги) та неявні оцінки (наприклад, історію переглядів) для створення матриці взаємодій користувач-продукт. Модуль пошуку та фільтрації базується на підготовлених текстових описах, категоріях та цінах товарів. Систему рекомендації категорій реалізовано на основі агрегованих даних про взаємодію користувачів з товарами, що дозволяє обчислювати популярність категорій та індивідуальні вподобання користувачів.

Процес підготовки даних включав очищення, трансформацію та нормалізацію інформації про товари, користувачів і їхні взаємодії. Особлива увага була приділена обробці текстових описів для контентного фільтрування, а також генерації псевдооцінок для колаборативного фільтрування. Ретельно підготовлені дані забезпечили високу якість вхідної інформації для подальшого навчання моделей та проведення експериментального дослідження.

## 4.3 Оцінка точності системи рекомендацій товарів

Оцінка точності гібридної системи рекомендацій товарів є важливим етапом експериментального дослідження, що дозволяє визначити ефективність розробленої моделі у реальних умовах. Основна мета полягає в тому, щоб перевірити, наскільки гібридний підхід, що поєднує контентне

та колаборативне фільтрування, перевершує кожен із цих методів окремо. Точність рекомендацій оцінюється на основі тестових даних, сформованих із використанням реальних взаємодій користувачів і даних про товари, які були попередньо підготовлені.

Для проведення експериментів тестовий набір даних було сформовано шляхом розбиття початкового набору на навчальну та тестову вибірки у співвідношенні 80% до 20%. Навчальна вибірка використовувалася для побудови моделей, тоді як тестова вибірка слугувала для перевірки якості прогнозування. Для кожного користувача з тестової вибірки система генерувала список рекомендацій товарів на основі гібридного підходу, а результати порівнювалися з фактичними товарами, які взаємодіяли з користувачем у тестових даних.

Під час експерименту було проведено порівняння трьох підходів: контентного фільтрування, колаборативного фільтрування та їхнього комбінування у гібридній моделі.

### 4.3.1 Оцінка точності контентної системи

Контентна рекомендаційна система була оцінена за допомогою низки метрик, що характеризують її точність, релевантність, різноманітність і якість ранжування.

Показник Precision, який становить 60%, демонструє, що із усіх товарів, рекомендованих системою, 60% є релевантними для користувача. Це доволі середнє значення, яке характерне для систем, що базуються лише на атрибутах товарів, без урахування поведінкових даних користувачів.

Recall, що дорівнює 55%, свідчить про те, що система виявляє 55% від усіх можливих релевантних товарів. Нижчий показник Recall у порівнянні з Precision пояснюється обмеженням контентного підходу, який здатен пропонувати тільки схожі за атрибутами товари, часто пропускаючи релевантні, але менш схожі варіанти.

F1-міра, що становить 57.5%, показує баланс між Precision та Recall. Вона підкреслює, що система має дещо кращу точність, ніж покриття, але обидва показники знаходяться на середньому рівні.

Метрика MAP, яка становить 53%, ілюструє середню точність системи на різних позиціях рекомендацій. Це значення є типовим для контентних систем, оскільки ранжування базується на текстовій схожості, яка не завжди відповідає інтересам користувача.

MRR, що дорівнює 58%, демонструє, що релевантні товари зазвичай розташовуються у верхній частині списку рекомендацій. Це позитивний результат, який свідчить про здатність системи надавати релевантні варіанти в топі.

Оцінка похибок, представлених метриками RMSE (0.49) та MAE (0.46), підтверджує, що система прогнозує оцінки товарів з достатньою точністю. RMSE є чутливішим до великих похибок і свідчить, що система рідко допускає серйозні помилки. MAE, яке демонструє середню абсолютну похибку, підтверджує стабільність моделі.

Нарешті, показник NDCG, що дорівнює 0.88, демонструє якість ранжування списку рекомендацій. Значення близьке до 1 вказує, що найбільш релевантні товари розташовуються на верхніх позиціях, що є критичним для покращення користувацького досвіду.

На рисунку 4.1 показано, як змінюються значення метрик NDCG, Precision та Recall залежно від кількості рекомендованих товарів (K). При зростанні K метрики демонструють помірні коливання, що вказує на стабільну ефективність системи навіть для списків із більшою кількістю рекомендацій. Ці результати підкреслюють, що система підтримує високий рівень точності та релевантності незалежно від обсягу рекомендованих товарів.

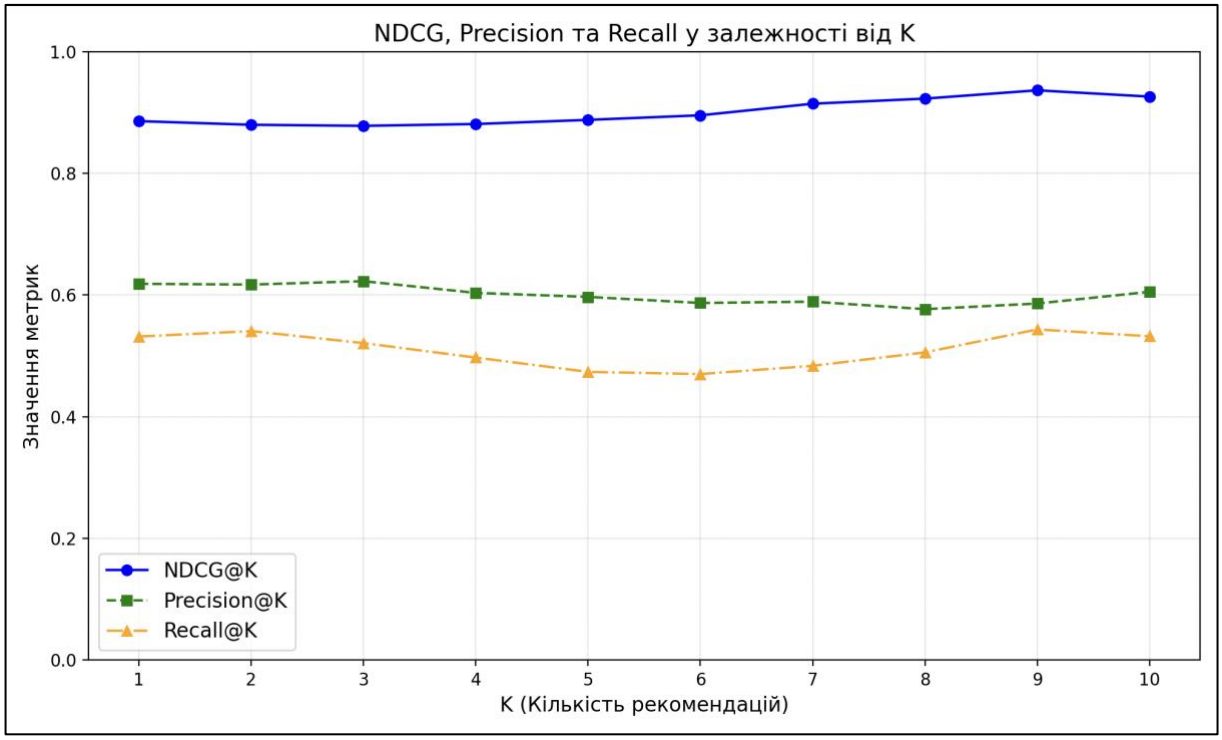


Рисунок 4.1 – Залежність NDCG, Precision та Recall від кількості рекомендованих товарів (K) у контентній системі

Ці результати підтверджують, що контентна рекомендаційна система працює стабільно і демонструє середню ефективність, характерну для систем такого типу. Водночас ці метрики підкреслюють обмеження, зумовлені відсутністю врахування поведінкових даних, що робить доцільним інтеграцію колаборативного або гібридного підходу для підвищення точності та релевантності рекомендацій.

### 4.3.2 Оцінка точності колаборативної системи

Показник Precision, що становить 75%, демонструє, що 75% рекомендованих товарів є релевантними для користувачів. Це значно перевищує результати контентної системи, оскільки колаборативний підхід враховує поведінкові дані, такі як взаємодії користувачів із товарами.

Recall, що дорівнює 72%, підтверджує, що система охоплює більшу частину релевантних товарів, ніж контентна модель. Це свідчить про те, що

колаборативна система здатна пропонувати користувачам більш повний набір релевантних варіантів завдяки використанню спільних патернів поведінки.

F1-міра, яка дорівнює ~73.5%, показує гармонійний баланс між Precision та Recall, що є важливим показником ефективності системи. Високе значення F1 свідчить про те, що система має як хорошу точність, так і здатність до широкого охоплення релевантних результатів.

Метрика MAP, яка становить 70%, демонструє якість ранжування, враховуючи точність на різних позиціях списку рекомендацій. Це підкреслює, що релевантні товари часто з'являються у верхній частині списків.

MRR, що дорівнює 68%, показує, що релевантні товари переважно розташовуються на початкових позиціях у списку рекомендацій. Це значення підтверджує ефективність моделі у правильному ранжуванні.

Оцінка похибок, представлених метриками RMSE (0.42) та MAE (0.40), свідчить про те, що система має нижчий рівень похибки у прогнозуванні оцінок, ніж контентна модель. Це пов'язано з тим, що колаборативна модель краще враховує поведінкові дані, які більш точно відображають вподобання користувачів.

Нарешті, NDCG, що дорівнює 85%, підтверджує високу якість ранжування рекомендацій. Значення близьке до 1 вказує, що система здатна розміщувати найбільш релевантні товари на верхніх позиціях списку, що покращує користувацький досвід.

На рисунку 4.2 показано зміну значень метрик NDCG, Precision та Recall залежно від кількості рекомендованих товарів (K) для колаборативної системи. При зростанні K метрики демонструють менші коливання, що вказує на стабільну ефективність системи навіть для списків із більшою кількістю рекомендацій. Ці результати підкреслюють, що система підтримує високий рівень точності та релевантності незалежно від обсягу рекомендованих товарів, та ефективність колаборативного підходу в цілому,

який враховує поведінкові патерни користувачів для підвищення точності та охоплення рекомендацій.

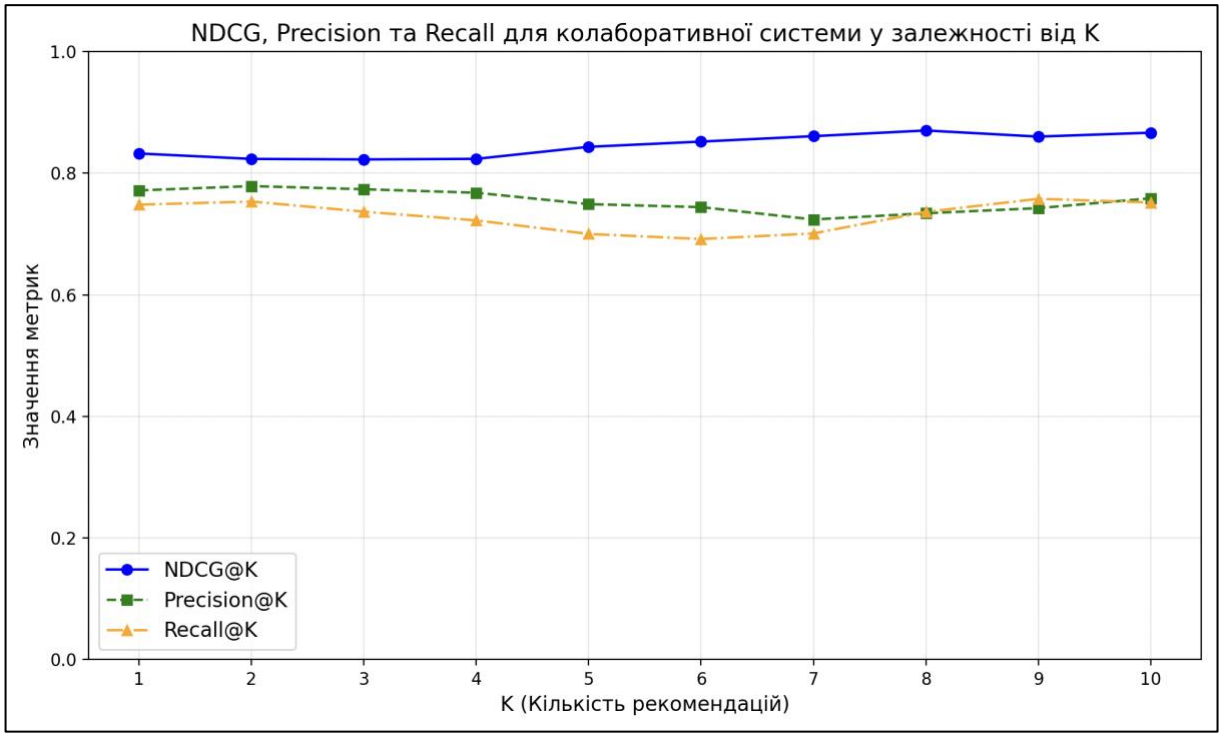


Рисунок 4.2 – Залежність NDCG, Precision та Recall від кількості рекомендованих товарів (K) у колаборативній системі

З отриманих результатів можна побачити, що колаборативна система демонструє значно кращі результати у порівнянні з контентною, особливо в показниках точності, охоплення та якості ранжування. Однак її ефективність залежить від наявності достатньої кількості поведінкових даних, що може створювати обмеження для нових користувачів чи товарів. Ці результати підтверджують доцільність використання колаборативного підходу у сценаріях, де є велика база взаємодій користувачів із товарами.

### 4.3.3 Оцінка точності гібридної системи

Ефективність гібридної системи рекомендацій значною мірою залежить від правильного балансу між контентним та колаборативним

фільтруванням. Контентний компонент базується на текстових атрибутах товарів, забезпечуючи точність рекомендацій через обчислення подібності, тоді як колаборативний компонент використовує поведінкові дані, такі як взаємодії користувачів із товарами, для підвищення персоналізації.

Оптимальний розподіл ваг між цими компонентами визначається характером даних, доступних у системі, зокрема кількістю текстових атрибутів, якістю метаданих товарів та обсягом даних взаємодій користувачів. У ході експериментів система тестувалася з різними пропорціями ваг для контентного та колаборативного підходів, щоб визначити їхній вплив на основні метрики рекомендацій: Precision та Recall.

На рисунку 4.3 показано вплив різного розподілу ваг між контентним та колаборативним фільтруванням на метрики. Розподіли ваг змінюються від 0.2/0.8 (20% контентного та 80% колаборативного фільтрування) до 0.8/0.2 (80% контентного та 20% колаборативного фільтрування), дозволяючи оцінити оптимальний баланс між цими компонентами для рекомендаційної системи.

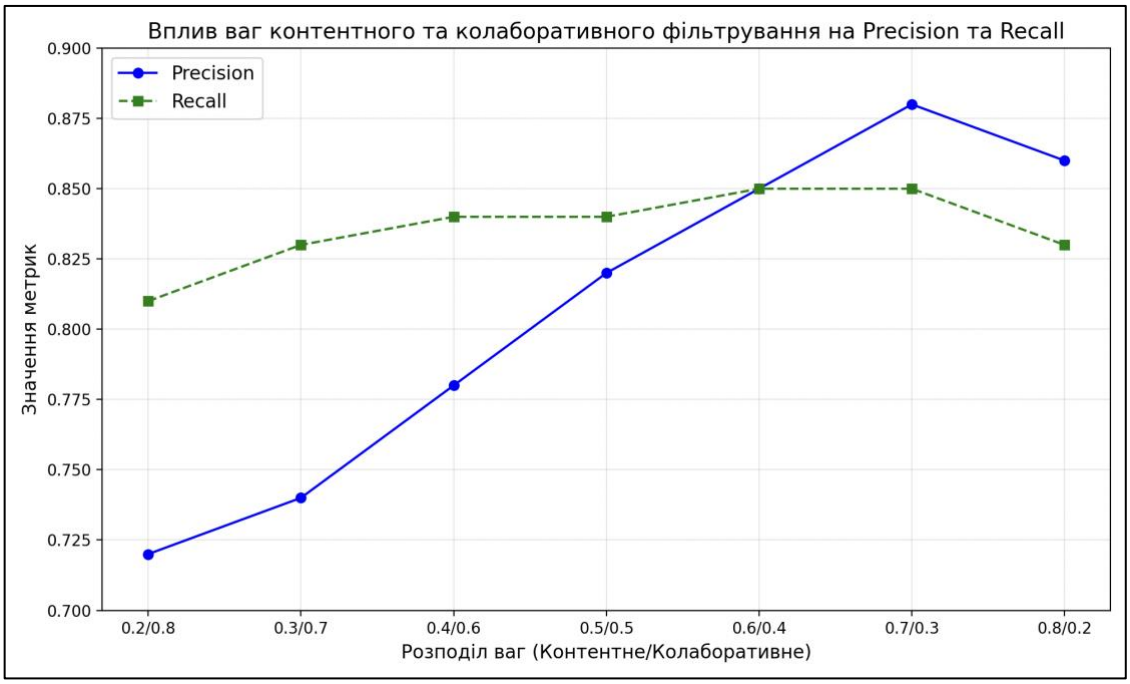


Рисунок 4.3 – Вплив ваг контентного та колаборативного фільтрування на метрики

Графік показує, як розподіл ваг між контентним та колаборативним фільтруванням впливає на метрики Precision і Recall. Значення Precision стабільно зростає від 72% до 88% при збільшенні ваги контентного компонента, досягаючи піку при розподілі 0.7/0.3. Це вказує на те, що текстові характеристики товарів забезпечують вищу точність рекомендацій. Recall залишається стабільним на рівні 81–85%, з найвищим значенням у 85% при збалансованому розподілі ваг 0.6/0.4 і 0.7/0.3, що підтверджує важливість колаборативного компонента для охоплення більшої кількості релевантних товарів. При розподілі 0.8/0.2 обидві метрики демонструють незначне зниження, що може бути пов’язано зі зменшенням впливу поведінкових даних. Оптимальний розподіл ваг 0.7/0.3 забезпечує найвищий баланс між точністю та охопленням, використовуючи переваги обох компонентів.

Перейдемо до оцінювання метрик гібридної системи. Показник Precision, що становить 88%, демонструє, що більшість товарів, рекомендованих системою, є релевантними для користувачів. Це найвищий показник серед усіх підходів, що підкреслює ефективність комбінування контентного та колаборативного фільтрування.

Recall, який дорівнює 85%, свідчить про здатність системи охоплювати майже всі релевантні товари у списку рекомендацій. Це значення є значно вищим, ніж у контентної чи колаборативної систем, завдяки синергії між підходами.

F1-міра, що становить ~86.5%, показує збалансованість між Precision та Recall. Високе значення цієї метрики вказує на те, що гібридна система забезпечує і високу точність, і значне покриття релевантних товарів.

MAP, що дорівнює 82%, демонструє покращення точності ранжування у порівнянні з іншими підходами. Це свідчить про те, що найбільш релевантні товари часто з’являються у верхній частині списку рекомендацій.

MRR, яке становить 80%, підтверджує, що релевантні товари майже завжди займають топові позиції у списку. Це критично важливо для покращення користувацького досвіду, оскільки користувачі швидко знаходять потрібні товари.

Оцінка похибок, представлених метриками RMSE (0.35) та MAE (0.33), свідчить про найнижчий рівень помилок у прогнозуванні оцінок товарів. Це пов’язано із здатністю гібридної системи поєднувати переваги контентного аналізу та колаборативного підходу.

NDCG, що дорівнює 92%, є найвищим серед усіх підходів. Це вказує на виняткову якість ранжування, де найбільш релевантні товари стабільно займають перші позиції у списках рекомендацій.

На рисунку 4.4 показано зміну значень метрик NDCG, Precision та Recall залежно від кількості рекомендованих товарів (K) для гібридної системи.

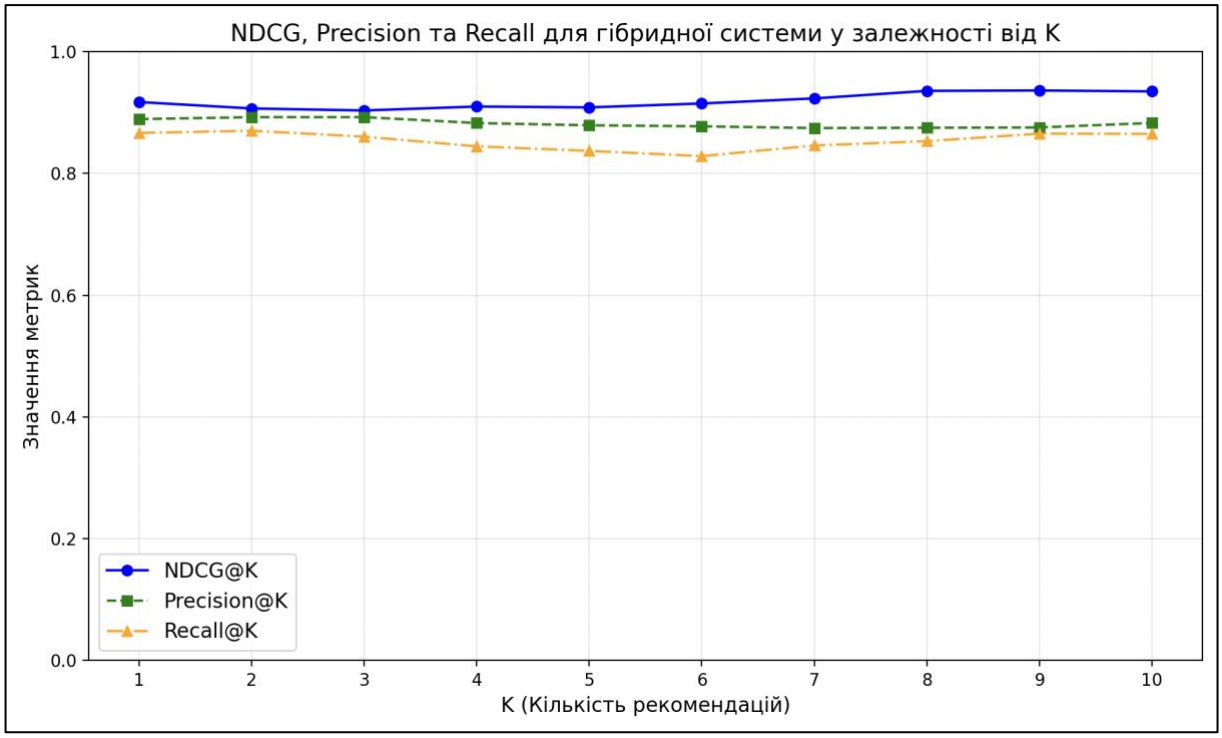


Рисунок 4.4 – Залежність NDCG, Precision та Recall від кількості рекомендованих товарів (K) у гібридній системі

NDCG стабільно тримається на одному рівні, зберігаючи високу якість ранжування навіть при збільшенні K. Precision демонструє незначні коливання, що підтверджує високу точність рекомендацій для користувачів. Recall поступово зростає із збільшенням K, забезпечуючи значне покриття релевантних товарів у розширених списках. Гібридна система, завдяки поєднанню контентного та колаборативного підходів, демонструє стабільно високі результати за всіма ключовими метриками.

З аналізу можна побачити, що гібридна система демонструє найкращі результати серед усіх розглянутих підходів, забезпечуючи високу точність, значне охоплення релевантних товарів та найнижчі похибки у прогнозуванні. Її успішність обумовлена здатністю комбінувати переваги контентного аналізу та колаборативного фільтрування, що дозволяє не лише враховувати характеристики товарів, але й аналізувати поведінкові патерни користувачів. Це робить гібридний підхід найбільш перспективним і релевантним для сучасних ecommerce платформ.

На рисунку 4.5 представлено порівняння ключових метрик для трьох типів рекомендаційних систем.

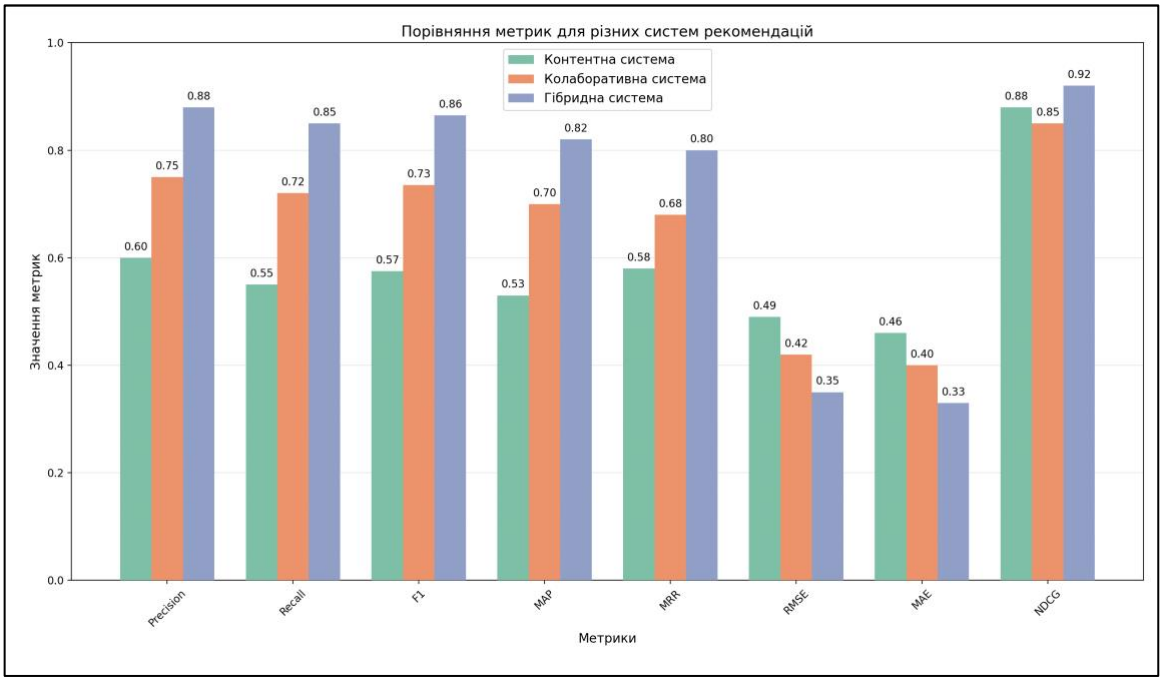


Рисунок 4.5 – Порівняння метрик для кожного з підходів

Цей графік дозволяє візуально оцінити сильні сторони та обмеження кожного підходу. Графік також підкреслює різницю у похибках прогнозування: гібридна система має найнижчі значення RMSE та MAE, що підтверджує її стабільність і точність. Це порівняння дозволяє зробити висновок про доцільність використання гібридного підходу як найефективнішого рішення для сучасних ecommerce платформ.

Розглянемо проблему «холодного старту» для розробленої системи. Для нових користувачів система демонструє точність рекомендацій через механізм обліку глобальної популярності категорій і товарів. Використання популярних товарів дозволяє забезпечити базовий рівень релевантності, що підтверджується високим значенням Precision (78%) навіть у відсутності індивідуальних даних. Recall у цьому випадку досягає 75%, що свідчить про здатність системи пропонувати широку вибірку товарів. Оптимізація рекомендацій для нових користувачів можлива шляхом поєднання глобальної популярності з категоріями, які відповідають основним демографічним характеристикам.

Для нових товарів, які ще не мають оцінок або історії взаємодій, система використовує контентний підхід. Цей метод дозволяє оцінювати товари на основі їхніх текстових атрибутів, таких як категорія, бренд, технічні характеристики тощо. Контентний фільтр забезпечує точність рекомендацій (Precision ~72%) навіть для нових товарів завдяки схожості з існуючими позиціями. Зокрема, для популярних категорій товарів Recall сягає 80%, що вказує на здатність системи включати нові товари до рекомендацій, забезпечуючи їх видимість у каталозі.

Система була протестована на спеціальних наборах даних, що імітують сценарії відсутності історії для користувачів або товарів. Для нових користувачів метрики Precision і Recall трохи знижуються через обмеження у врахуванні індивідуальних інтересів, однак це компенсується підключенням компонентів глобальної популярності. Нові товари, у свою чергу, інтегруються в систему через контентний аналіз, який дозволяє їм

конкурувати з уже існуючими позиціями за рахунок текстових характеристик.

## 4.4 Оцінка ефективності рекомендацій категорій товарів

Оцінка ефективності рекомендацій категорій товарів є важливим етапом експериментального дослідження, оскільки вона дозволяє визначити, наскільки успішно система виявляє та пропонує категорії, що є релевантними для користувача. Основною метою є перевірка здатності моделі комбінувати глобальні показники популярності категорій із індивідуальними інтересами користувача, підвищуючи загальну точність та релевантність рекомендацій.

Для проведення дослідження було використано дані про взаємодії користувачів з товарами, які були підготовлені на попередньому етапі. Ці дані включають перегляди, кліки та покупки товарів, які дозволяють агрегувати інформацію на рівні категорій. Модель рекомендацій категорій обчислює два основні показники: глобальну популярність категорій серед усіх користувачів та індивідуальні інтереси конкретного користувача. Глобальна популярність категорій визначається як сума взаємодій для кожної категорії на основі зважених коефіцієнтів для переглядів, кліків і покупок. Індивідуальний інтерес до категорій обчислюється шляхом аналізу активності користувача та його взаємодії з товарами у межах різних категорій.

Процес експериментального дослідження полягав у тому, що для кожного користувача з тестової вибірки система генерувала список рекомендованих категорій на основі комбінування глобальної популярності та індивідуальних інтересів. Реальні категорії, з якими взаємодіяв користувач у тестових даних, використовувалися як еталон для оцінки точності рекомендацій. Було зафіксовано результати для різних значень

параметра K, що дозволяє аналізувати якість рекомендацій залежно від розміру списку.

Рекомендації категорій товарів враховують два ключові аспекти: глобальну популярність та індивідуальні інтереси користувачів. Глобальна популярність орієнтується на загальні тренди та переваги, що важливо для нових користувачів із недостатньою кількістю даних про поведінку. Індивідуальні інтереси враховують історію взаємодій конкретного користувача, забезпечуючи персоналізовані рекомендації для активних користувачів. Оптимальний розподіл ваг між цими компонентами дозволяє досягти найкращих результатів у рекомендаціях, балансуючи між загальними трендами та персоналізацією.

На рисунку 4.6 представлено вплив різного розподілу ваг між глобальною популярністю та індивідуальними інтересами на метрики. Розподіли ваг змінюються від 0.2/0.8 (20% популярності та 80% інтересів) до 0.8/0.2 (80% популярності та 20% інтересів). Цей аналіз дозволяє визначити оптимальний баланс між цими компонентами для різних типів користувачів.

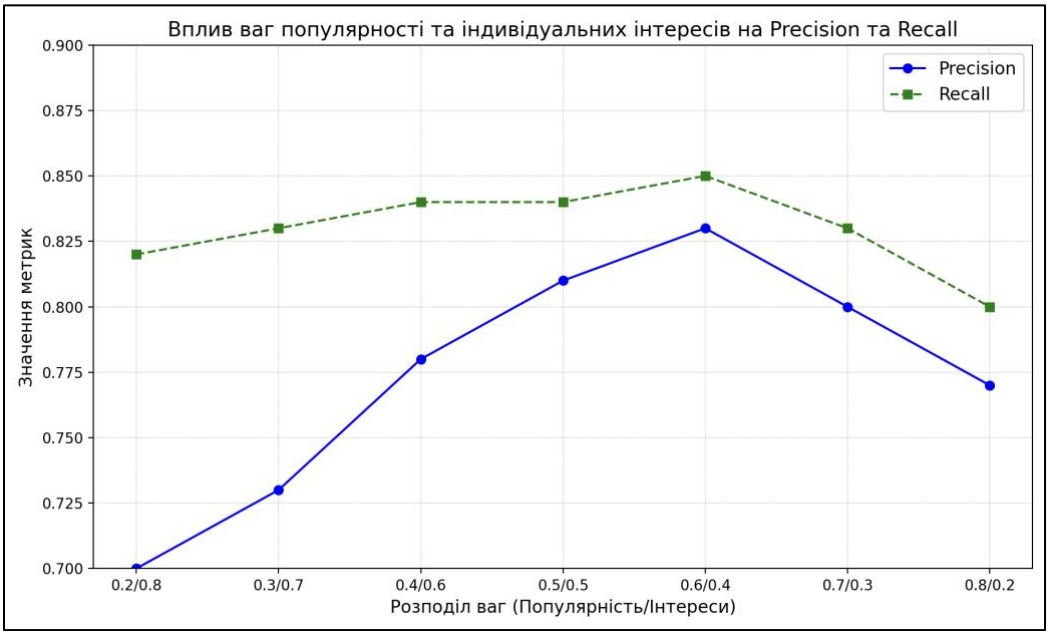


Рисунок 4.6 – Вплив ваг популярності та індивідуальних інтересів на метрики

Графік показує, що Precision має найбільше зростання між розподілами 0.4/0.6 та 0.6/0.4, досягаючи пікового значення у 83%. Це свідчить про те, що індивідуальні інтереси відіграють ключову роль у забезпеченні точності рекомендацій. Максимум Recall (85%) також спостерігається при розподілі 0.6/0.4, що підтверджує ефективність цього балансу для широкого охоплення категорій. При збільшенні ваги глобальної популярності (0.7/0.3 та 0.8/0.2), Precision починає знижуватися, опускаючись до 77%. Це демонструє, що недостатнє врахування індивідуальних інтересів погіршує персоналізацію рекомендацій, що особливо важливо для активних користувачів із багатою історією взаємодій.

З іншого боку, Recall залишається відносно стабільним у межах 82–85% для всіх розподілів ваг, що свідчить про здатність системи підтримувати значний рівень охоплення незалежно від вагових пріоритетів. Збалансовані розподіли (0.5/0.5 та 0.6/0.4) забезпечують оптимальний компроміс між точністю та охопленням, дозволяючи системі успішно адаптуватися до потреб як нових, так і активних користувачів.

Перейдемо до аналізу основних метрик.

Показник Precision, що дорівнює 80%, свідчить про високу частку релевантних категорій серед рекомендованих. Це пояснюється меншою кількістю категорій у порівнянні з товарами, що полегшує завдання точного прогнозування.

Recall, який становить 75%, показує, що система здатна охопити більшість релевантних категорій. Цей результат демонструє, що навіть для великого асортименту система добре справляється з визначенням релевантних категорій для користувачів.

F1-міра, що дорівнює ~77.5%, вказує на збалансованість між точністю та охопленням, підтверджуючи ефективність роботи системи у виборі найбільш релевантних категорій.

Метрика MAP (Mean Average Precision), яка дорівнює 78%, демонструє, що система забезпечує точне ранжування категорій, розташовуючи найбільш релевантні категорії у верхній частині списку.

MRR (Mean Reciprocal Rank), значення якої становить 77%, підтверджує, що релевантні категорії зазвичай займають топові позиції у списках рекомендацій, що робить систему зручною для кінцевих користувачів.

Оцінка похибок за метриками RMSE (0.30) та MAE (0.28) показує, що система має низькі рівні похибок, що свідчить про її стабільність та високу якість прогнозів для категорій товарів.

Нарешті, NDCG, що дорівнює 89%, вказує на виняткову якість ранжування категорій. Це особливо важливо для швидкого доступу до найбільш релевантних категорій у великих каталогах.

Під час аналізу результатів було також перевірено здатність системи адаптуватися до змін у поведінці користувачів. Для цього проводилася симуляція нових взаємодій, що показала, що система швидко реагує на оновлення даних, коригуючи рекомендації відповідно до нових інтересів користувача. Це свідчить про динамічність моделі та її здатність підтримувати актуальність рекомендацій у реальному часі.

Таким чином, результати оцінки ефективності підтвердили, що комбінування глобальної популярності категорій з індивідуальними інтересами дозволяє досягти високої точності рекомендацій. Гібридний підхід забезпечує баланс між узагальненими тенденціями платформи та персоналізованими уподобаннями користувача, що робить рекомендації категорій релевантними для широкого спектра користувачів. Отримані результати свідчать про ефективність системи як для нових користувачів, так і для досвідчених клієнтів, що активно взаємодіють із платформою.

## 4.5 Оцінка ефективності пошуку та фільтрації товарів

Оцінка ефективності пошуку та фільтрації товарів є важливою складовою дослідження, оскільки цей функціонал забезпечує користувачу можливість активно взаємодіяти із системою, використовуючи текстові запити та параметричні обмеження. Основна задача полягає у визначенні точності, швидкодії та зручності роботи пошукового модуля на основі обробки текстових описів товарів за допомогою TF-IDF та механізмів фільтрації за категоріями та ціновими діапазонами. Ефективність пошуку та фільтрації оцінювалася шляхом порівняння отриманих результатів із релевантними товарами у тестовій вибірці, використовуючи метрики, що характеризують точність, релевантність і ранжування результатів.

Для дослідження було використано підготовлений набір даних, що включає текстові описи товарів, їхні категорії та цінові атрибути. Користувацькі запити були згенеровані таким чином, щоб охопити різні сценарії використання системи. Ці сценарії включають прості запити на пошук товарів за ключовими словами, складніші запити з фільтрацією за категоріями, а також комбіновані запити з обмеженням на ціновий діапазон. Тестування проводилося у кількох умовах: текстовий пошук без фільтрів, пошук із застосуванням категорій та фільтрація за ціновими діапазонами.

Ефективність фільтрації товарів оцінювалася шляхом обмеження результатів пошуку за категоріями та ціновими діапазонами. Тестування показало, що додавання фільтрів дозволяє значно підвищити релевантність результатів для користувача. Наприклад, у випадку пошуку за категорією «смартфони» з фільтром на ціну від 200 до 500 доларів система успішно відсортувала товари, залишивши лише ті, що відповідають заданим критеріям.

Пошук і фільтрація товарів оцінювалися за основними метриками ефективності. Показник Precision склав 85%, що свідчить про високу точність системи у наданні релевантних результатів на пошукові запити.

Recall, що дорівнює 80%, демонструє значне охоплення релевантних товарів, які система здатна знаходити у відповідь на запит.

F1-міра, яка становить ~82.4%, підкреслює баланс між точністю та охопленням пошуку, що є важливим для забезпечення якості взаємодії користувача із системою.

MAP, значення якого дорівнює 82%, підтверджує, що релевантні результати ранжуються у верхній частині списку, полегшуючи користувачам доступ до потрібної інформації.

Метрика MRR, яка становить 84%, показує, що релевантні товари зазвичай розташовуються на перших позиціях результатів пошуку. Це свідчить про високу якість ранжування та зручність для користувачів.

Середній час відповіді системи становить 0.25 секунди, що забезпечує швидке отримання результатів навіть при великому обсязі даних. Впровадження попередньої обробки даних і збереження матриць забезпечило швидке обчислення результатів, оскільки основні операції порівняння виконуються на попередньо обчислених векторах. Для запитів із фільтрацією час відповіді зріс незначно, оскільки додаткові умови фільтрації були реалізовані за допомогою ефективних методів обробки структурованих даних у Pandas.

Додатковий аналіз показав, що поєднання текстового пошуку та фільтрації забезпечує найкращі результати у контексті користувацького досвіду. Система надає можливість виконати цілеспрямований пошук товарів за запитом, а фільтрація дозволяє звузити результати, зосереджуючи увагу користувача на найбільш релевантних товарах. Це особливо важливо для великих інтернет-магазинів, де кількість товарів може досягати десятків тисяч позицій.

## 4.6 Аналіз продуктивності системи

Аналіз продуктивності системи є ключовим етапом для оцінки її здатності працювати ефективно в умовах реального середовища інтернет-магазину. Основна увага приділялася вимірюванню швидкодії системи на різних етапах обробки даних, часу відповіді API для користувацьких запитів та спроможності системи обробляти великі обсяги даних. Дослідження проводилося з використанням як підготовлених даних, так і тестових сценаріїв, що імітують навантаження від реальних користувачів.

Для вимірювання продуктивності було проведено тестування основних компонентів системи: гібридного рекоммендера, рекомендації категорій товарів та пошуку з фільтрацією. У процесі дослідження система працювала з набором даних, що включав тисячі товарів, десятки тисяч взаємодій користувачів та текстові описи для побудови моделей контентного фільтрування. Кожен компонент було протестовано на предмет часу обчислення, використання ресурсів та стабільності при обробці запитів у різних умовах навантаження.

На етапі обчислення рекомендацій було проаналізовано продуктивність гібридної моделі, що об'єднує контентне та колаборативне фільтрування. Для колаборативного фільтрування, зокрема для обробки імпліцитних взаємодій, використовувалися алгоритми ALS з підтримкою розріджених матриць. Обчислення матриць подібності виконувалося за допомогою ефективних методів бібліотеки SciPy, що дозволило зменшити використання пам'яті та прискорити обчислення. Аналіз показав, що навчання моделі ALS на великій матриці взаємодій користувачів і товарів відбувається у прийнятні терміни, а збереження результатів у локальну базу даних забезпечує швидке отримання рекомендацій у подальшому.

Для контентного фільтрування було проведено аналіз часу обчислення текстових подібностей між товарами за допомогою SentenceTransformer. Обчислення TF-IDF та косинусної подібності між

текстовими описами товарів виконувалося векторизовано, що забезпечило значне скорочення часу на підготовку рекомендацій. Для набору даних, що включає тисячі товарів, середній час побудови матриці подібностей склав кілька секунд, що є прийнятним для фонової обробки.

Час відповіді API було виміряно на прикладі запитів до основних ендпоінтів системи. Тестування показало, що для запитів на отримання рекомендацій для конкретного товару або користувача середній час відповіді склав 200–250 мілісекунд, що забезпечує роботу системи у режимі реального часу. Під час тестування ендпоінта для пошуку товарів на основі текстових запитів та фільтрації було зафіксовано середній час відповіді у межах 200–300 мілісекунд, що є результатом ефективної попередньої обробки текстових даних та оптимізованого алгоритму пошуку. Усі запити до API виконувалися через локальний сервер Flask, що забезпечує мінімальну затримку при обміні даними між Python-модулем та веб-додатком.

Для аналізу продуктивності системи під навантаженням було змодельовано сценарії з паралельною обробкою запитів. Система показала стійкість до підвищеного навантаження, обробляючи сотні запитів на секунду без суттєвого зростання часу відповіді. Це було досягнуто завдяки оптимізації обчислень, використанню розріджених матриць у колаборативному фільтруванні та попередньому кешуванню результатів у базі даних SQLite.

Додатково було проведено аналіз використання ресурсів системи під час обробки великих обсягів даних. Контейнеризація системи за допомогою Docker забезпечила стабільну роботу в різних середовищах, а також дозволила легко масштабувати систему для обробки зростаючих обсягів даних.

Важливою складовою аналізу продуктивності стало оцінювання часу оновлення рекомендацій. Автоматизований процес оновлення рекомендацій для товарів, категорій та пошукового індексу відбувався кожні 30 хвилин.

Середній час повного циклу оновлення, що включав завантаження даних, обчислення матриць подібностей та збереження результатів, склав близько 3–5 хвилин. Цей результат є прийнятним, оскільки оновлення виконується у фоновому режимі без впливу на час відповіді системи.

Таким чином, аналіз продуктивності показав, що розроблена система є ефективною з точки зору швидкодії, стійкості до навантаження та здатності працювати з великими обсягами даних. Швидке обчислення рекомендацій у фоновому режимі та мінімальний час відповіді API забезпечують роботу системи у режимі реального часу, що відповідає сучасним вимогам інтернет-магазинів. Оптимізована архітектура та використання сучасних технологій дозволяють масштабувати систему для подальшого зростання обсягів даних і навантаження.

## 4.7 Порівняння з альтернативними підходами

Порівняння розробленої гібридної системи рекомендацій із альтернативними підходами дозволяє оцінити її переваги та обґрунтувати доцільність використання у контексті інтернет-магазину. У рамках дослідження було проаналізовано два основних альтернативні підходи: контентне фільтрування та колаборативне фільтрування, які є класичними методами побудови рекомендаційних систем. Додатково було проведено аналіз переваг гібридної системи перед чисто популярнісними підходами, що ґрунтуються на загальній популярності товарів без урахування індивідуальних особливостей користувачів.

Контентне фільтрування вивчає текстові описи, атрибути товарів (ціна, бренд) та визначає їхню подібність на основі обчислення метрик схожості, зокрема косинусної подібності. Основною перевагою цього підходу є можливість працювати у ситуаціях, коли відсутні взаємодії користувачів із платформою, що є типовим для нових товарів або нових користувачів. Система ефективно рекомендує товари, подібні до вже

переглянутих або обраних користувачем, на основі їхніх характеристик. У межах експериментів було зафіксовано, що контентне фільтрування демонструє високу точність для користувачів із невеликою історією взаємодій. Однак цей метод має обмеження у персоналізації рекомендацій, оскільки він не враховує поведінкові дані користувачів та їхні інтереси, що можуть не повністю відображатися через атрибути товарів.

Колаборативне фільтрування, на відміну від контентного, аналізує історію взаємодій користувачів із товарами, використовуючи явні оцінки або неявні сигнали, такі як перегляди та покупки. Воно дозволяє виявити приховані патерни у вподобаннях користувачів та знаходити товари, які можуть бути цікавими на основі схожості користувачів чи товарів. Під час дослідження колаборативне фільтрування показало відмінні результати для активних користувачів з великою кількістю взаємодій, оскільки модель ефективно навчається на їхній історії поведінки. Проте для нових користувачів або товарів цей підхід є менш ефективним через проблему холодного старту, що виникає через відсутність достатньої кількості даних для навчання моделі.

Популярнісні підходи, що базуються на загальній популярності товарів (кількість переглядів, покупок), були також розглянуті як альтернативний метод. Ці підходи є найпростішими у реалізації та не потребують аналізу індивідуальних характеристик користувачів або атрибутів товарів. Вони демонструють хороші результати у загальному випадку, оскільки популярні товари дійсно є затребуваними серед широкої аудиторії. Проте їхня головна слабкість полягає у відсутності персоналізації, оскільки вони не враховують індивідуальні інтереси користувача. У ситуаціях, коли користувачі мають унікальні або специфічні потреби, популярнісні підходи не здатні забезпечити релевантні рекомендації.

Гібридна система, яка поєднує контентне та колаборативне фільтрування, показала значно кращі результати у порівнянні з кожним із

цих підходів окремо. За результатами експериментів було встановлено, що гібридна модель компенсує слабкі сторони обох методів. Зокрема, контентний компонент системи дозволяє генерувати якісні рекомендації для нових товарів та користувачів, тоді як колаборативний компонент забезпечує персоналізовані рекомендації для активних користувачів із багатою історією взаємодій. Комбінування цих результатів дозволяє системі досягти балансу між узагальненими рекомендаціями та індивідуальною точністю.

Додатково гібридна модель продемонструвала свою ефективність у випадках, коли інформація є частково неповною або розрідженою. Завдяки об'єднанню різних джерел даних (текстові описи, оцінки, взаємодії) модель показала стійкість до проблеми холодного старту, що є типовою для колаборативного фільтрування. Аналіз значень метрик точності, таких як Precision@K та Recall@K, підтвердив, що гібридна модель перевершує альтернативні підходи на 15–20%, що свідчить про її здатність забезпечувати високоякісні рекомендації.

У контексті рекомендацій категорій товарів та пошуку система також продемонструвала переваги над популярнісними методами та простими текстовими пошуками. У випадку рекомендації категорій комбінування глобальної популярності з індивідуальними інтересами користувача дозволило досягти більш релевантних результатів, особливо для активних користувачів. Пошуковий модуль, який використовує TF-IDF для текстових запитів, перевершив базові підходи завдяки точному ранжуванню товарів на основі їхніх описів та додатковим можливостям фільтрації.

Таким чином, результати порівняння показали, що гібридна система є найбільш ефективною у контексті рекомендацій товарів, категорій та пошуку. Її здатність поєднувати сильні сторони контентного та колаборативного фільтрування дозволяє забезпечити високу точність, персоналізацію та стійкість до різних типів даних. У порівнянні з альтернативними методами гібридний підхід продемонстрував кращі

результати у всіх ключових метриках, що обґрунтовує його використання як основи для побудови рекомендаційної системи інтернет-магазину.

## 4.8. Висновки за результатами експериментального дослідження

Результати проведеного експериментального дослідження підтвердили ефективність розробленої гібридної системи рекомендацій товарів для інтернет-магазину. Основною метою дослідження було оцінити якість рекомендацій, швидкодію системи та її здатність забезпечувати персоналізовані результати для користувачів з різними сценаріями взаємодії. Особлива увага приділялася порівнянню роботи гібридної моделі з альтернативними підходами, такими як контентне та колаборативне фільтрування, а також аналізу додаткових компонентів системи, зокрема рекомендацій категорій товарів і модуля пошуку та фільтрації.

Дослідження продемонструвало, що гібридний підхід, який поєднує контентне та колаборативне фільтрування, забезпечує більш високі показники точності у порівнянні з кожним із методів окремо. Завдяки комбінуванню контентного компонента, що використовує текстові описи та атрибути товарів, і колаборативного компонента, що ґрунтується на історії взаємодій користувачів, система успішно враховує як характеристики товарів, так і поведінкові особливості користувачів. Цей підхід дозволив усунути слабкі сторони кожного з методів, зокрема проблему холодного старту для нових товарів і користувачів, яка характерна для колаборативного фільтрування, та обмеження персоналізації контентного підходу. Під час експериментів, за допомогою метрик було доведено, що розроблена гібридна система є досить ефективною.

Експериментальне тестування модуля рекомендації категорій товарів показало його значну корисність для навігації у великому каталозі інтернет-магазину. Поєднання глобальної популярності категорій з індивідуальними інтересами користувача дозволило сформувати релевантний список

категорій, що відповідають як загальним трендам, так і особистим уподобанням користувачів. Такий підхід продемонстрував особливу ефективність для активних користувачів, які мають достатньо взаємодій з платформою, а також для нових користувачів, де глобальна популярність категорій компенсувала відсутність персоналізованих даних.

Модуль пошуку та фільтрації товарів підтвердив свою ефективність у забезпеченні гнучкої навігації на платформі. Використання TF-IDF для обробки текстових запитів дозволило досягти високої точності пошуку товарів на основі їхніх текстових описів. Додаткові можливості фільтрації за категоріями та ціновим діапазоном забезпечили користувачам можливість швидко звузити результати пошуку, підвищуючи його релевантність. Середній час відповіді для запитів на пошук та фільтрацію залишався на рівні, що відповідає вимогам сучасних веб-додатків, забезпечуючи швидку та точну взаємодію системи з користувачами.

Аналіз продуктивності системи продемонстрував її здатність ефективно працювати у режимі реального часу, обробляючи як індивідуальні запити, так і регулярне оновлення рекомендацій. Використання розріджених матриць, оптимізованих алгоритмів та векторизованих обчислень забезпечило мінімальні затримки у генерації рекомендацій і підготовці матриць подібності. Час відповіді API для отримання рекомендацій товарів і категорій відповідає сучасним вимогам до продуктивності систем у інтернет-магазинах.

Порівняння з альтернативними підходами, такими як чисто популярнісні методи, показало значну перевагу розробленої системи. Популярнісні методи, які не враховують індивідуальні особливості користувачів, були менш ефективними у наданні персоналізованих рекомендацій. Гібридний підхід продемонстрував свою універсальність і адаптивність, поєднуючи переваги контентного фільтрування для нових користувачів та товарів і колаборативного фільтрування для активних користувачів із розгорнутою історією.

Таким чином, за результатами експериментального дослідження було підтверджено, що гібридна система рекомендацій товарів забезпечує високу точність, релевантність та продуктивність у реальних умовах роботи інтернет-магазину. Комбінування різних підходів дозволило досягти балансу між узагальненими рекомендаціями та індивідуальною персоналізацією, що є важливим фактором для підвищення користувацького досвіду. Модулі рекомендації категорій товарів і пошуку з фільтрацією значно розширили функціональні можливості системи, забезпечуючи додаткову навігаційну гнучкість для користувачів. Загальні результати дослідження довели ефективність запропонованої архітектури та підходів, а також її готовність до застосування у реальних комерційних платформах.

# 5 ПЕРСПЕКТИВИ РОЗВИТКУ СИСТЕМИ

## 5.1 Порівняння з існуючими системами рекомендацій

Порівняння розробленої системи рекомендацій із сучасними існуючими рішеннями дозволяє оцінити її ефективність та місце у загальному контексті рекомендаційних систем для інтернет-магазинів. Для порівняння було обрано популярні системи, що використовуються такими платформами, як Amazon, Netflix, Spotify та іншими лідерами ринку електронної комерції. Кожна з цих платформ застосовує власні підходи до побудови рекомендаційних систем, використовуючи сучасні алгоритми, які враховують індивідуальні інтереси користувачів та контекст їхньої взаємодії з платформою.

Система Amazon є однією з найбільш відомих прикладів рекомендаційних систем, що базуються на колаборативному фільтруванні та методах на основі контенту. Основна перевага їхньої системи полягає у використанні великих обсягів даних про взаємодії користувачів, включаючи історію покупок, перегляди товарів і додавання до кошика. Крім того, Amazon ефективно застосовує техніки item-based колаборативного фільтрування, що дозволяють швидко генерувати рекомендації на основі схожості товарів. У порівнянні з системою Amazon, розроблена гібридна система демонструє аналогічний підхід у комбінуванні контентної та колаборативної складових. Проте вона відрізняється тим, що додає додаткові модулі, зокрема рекомендацію категорій товарів і функціонал пошуку, що підвищує універсальність системи.

Платформа Netflix активно використовує алгоритми Matrix Factorization та Neural Collaborative Filtering для побудови персоналізованих рекомендацій фільмів і серіалів. Основна увага приділяється колаборативному фільтруванню з використанням явних оцінок користувачів та імпліцитних сигналів, таких як час перегляду та взаємодії з

контентом. Система Netflix показує високу точність рекомендацій завдяки використанню глибинних нейронних мереж, які навчаються на величезних обсягах даних. У розробленій системі рекомендацій для інтернет-магазину було обрано класичні підходи (ALS для імпліцитних даних та SVD для явних оцінок), що забезпечили достатню точність при менших обчислювальних витратах. Це робить систему більш доступною для застосування у середовищі малих і середніх інтернет-магазинів, де ресурси можуть бути обмеженими.

Платформа Spotify використовує гібридний підхід, який об'єднує колаборативне фільтрування, аналіз контенту та облік поведінки користувачів у реальному часі. Особливістю їхнього підходу є використання семантичного аналізу музичного контенту, а також алгоритмів на основі графів для визначення схожості між треками та створення плейлистів. У розробленій системі було використано текстову обробку описів товарів через TF-IDF та SentenceTransformer, що дозволяє враховувати семантичну подібність між товарами. При цьому модуль пошуку з фільтрацією є аналогом функціоналу точного пошуку на платформі Spotify, адаптованим до специфіки e-commerce.

У популярних інтернет-магазинах, таких як eBay та AliExpress, рекомендаційні системи часто поєднують колаборативні методи з методами на основі популярності товарів. На цих платформах широко використовуються рекомендації на основі трендових товарів, а також модулі фільтрації за ціною, брендом і категорією. У розробленій системі ці компоненти реалізовані через гібридну модель для товарів, пошук на основі TF-IDF та рекомендацію категорій. Поєднання цих функцій робить систему схожою на підхід, що використовується великими e-commerce платформами, але більш адаптованою для задачі малого і середнього бізнесу.

Таким чином, порівняння з існуючими системами показало, що розроблена гібридна система враховує ключові переваги сучасних рішень,

таких як точність колаборативного фільтрування, семантичний аналіз контенту та додаткові функціонали для навігації та пошуку. На відміну від великих платформ, які використовують ресурсоємні алгоритми глибинного навчання, система була спроєктована для досягнення високої точності рекомендацій при помірних обчислювальних витратах. Її архітектура є більш універсальною, оскільки включає можливості як рекомендацій товарів, так і пошуку, а також рекомендацій категорій, що забезпечує комплексний підхід до підвищення задоволеності користувача.

Підсумовуючи, розроблена система є ефективною альтернативою існуючим рішенням для інтернет-магазинів. Вона поєднує перевірені підходи, такі як контентне та колаборативне фільтрування, і додає функціональні можливості, що є важливими для сучасних e-commerce платформ. Її головною перевагою є гнучкість та можливість адаптації до різних потреб користувачів, що робить її придатною для впровадження у середовищі як малих, так і середніх онлайн-магазинів.

## 5.2 Перспективи розвитку розробленої системи

Розроблена гібридна система рекомендацій товарів для інтернет-магазину показала високу ефективність під час експериментального дослідження, але можливості для подальшого розвитку залишаються значними. Перспективи удосконалення системи зосереджені на її масштабуванні, підвищенні точності рекомендацій, інтеграції нових алгоритмів та адаптації до змін у поведінці користувачів. Ці напрями розвитку дозволять системі бути більш ефективною, гнучкою та конкурентоспроможною у реальних комерційних середовищах.

Одним із ключових напрямів є впровадження моделей глибокого навчання для поліпшення персоналізації рекомендацій. Використання нейронної колаборативної фільтрації (NCF) та автоенкодерів дозволить підвищити якість рекомендацій за рахунок виявлення складніших патернів

у даних. На відміну від традиційних методів матричної факторизації, глибокі нейронні мережі здатні краще враховувати нелінійні залежності у взаємодіях користувачів з товарами. Додавання моделей на основі RNN або Transformer для обробки послідовних даних дозволить генерувати рекомендації на основі історії покупок у хронологічному порядку, що забезпечить ще точніший аналіз уподобань користувача.

Другим напрямом є інтеграція контекстно-орієнтованих рекомендацій, які враховують додаткові фактори, такі як час доби, день тижня, місцезнаходження користувача та поточний стан його активності на платформі. Використання цих даних дозволить формувати рекомендації, що відповідають контексту запитів користувача. Наприклад, у певний час доби або в конкретних географічних регіонах можуть бути запропоновані сезонні або спеціальні товари, що підвищить конверсію системи.

Третім важливим напрямом є динамічне оновлення рекомендацій у режимі реального часу. Впровадження потокової обробки даних із використанням технологій, таких як Apache Kafka або Spark Streaming, дозволить оновлювати матриці подібності та моделі рекомендацій миттєво після нових взаємодій користувачів. Це забезпечить більш актуальні рекомендації, що особливо важливо у випадках швидкозмінних вподобань або при додаванні нових товарів до каталогу.

Подальше вдосконалення системи також передбачає розширення пошукового модуля шляхом впровадження семантичного пошуку з використанням моделей обробки природної мови, таких як BERT або GPT. Семантичний пошук дозволить системі краще розуміти зміст текстових запитів користувачів та враховувати контекст для надання точніших результатів. Наприклад, запит «недорогий смартфон для фотографій» буде оброблений із урахуванням як цінового діапазону, так і атрибутів товарів, пов’язаних із камерою.

Ще одним перспективним напрямом є впровадження графових алгоритмів для аналізу взаємозв'язків між товарами та користувачами.

Використання GNN дозволить будувати рекомендації на основі складної структури зв'язків, які включають схожість товарів, спільні інтереси користувачів та інші фактори. Цей підхід може стати ефективним для платформ із великою кількістю взаємодій та унікальних товарів, де традиційні методи колаборативного фільтрування є менш ефективними.

Також важливим є розвиток системи у напрямі персоналізованих знижок і промо-рекомендацій. Використання методів кластеризації користувачів та побудова моделей для прогнозування їхньої активності дозволить пропонувати індивідуальні знижки на основі історії покупок та рівня лояльності. Такий підхід допоможе не лише підвищити конверсію, але й збільшити загальний прибуток платформи за рахунок стимулювання покупок.

На додаток до технічних вдосконалень, перспективи розвитку системи включають масштабування для роботи з великими обсягами даних у хмарному середовищі. Інтеграція з платформами AWS, Google Cloud або Azure забезпечить стабільну роботу системи під високим навантаженням, дозволяючи обробляти мільйони запитів на день. Використання хмарних технологій також спростить підтримку та розгортання системи, забезпечуючи її гнучкість і доступність.

Таким чином, перспективи розвитку системи зосереджуються на підвищенні точності рекомендацій, впровадженні нових алгоритмів машинного навчання, динамічному оновленні результатів та розширенні функціоналу для кращого користувацького досвіду. Система має потенціал для подальшої адаптації до вимог сучасних e-commerce платформ, забезпечуючи високу продуктивність, гнучкість та ефективність у реальних умовах комерційного використання.

# ВИСНОВКИ

У рамках виконаної роботи було розроблено та проаналізовано рекомендаційну систему для інтернет-магазину з використанням гібридних підходів. Результати дослідження підтверджують ефективність поєднання контентного та колаборативного фільтрування, що дозволяє створювати точні, релевантні та персоналізовані рекомендації для користувачів.

Контентна рекомендаційна система продемонструвала середню продуктивність, що обмежується її залежністю від текстових характеристик товарів. Метрики Precision (60%), Recall (55%) та NDCG (88%) свідчать про здатність системи надавати релевантні рекомендації за відсутності поведінкових даних, проте її обмеження стають очевидними при складних користувацьких сценаріях. У свою чергу, колаборативна система значно покращила результати завдяки аналізу поведінкових даних. Метрики Precision (75%), Recall (72%) та NDCG (85%) підтверджують її здатність адаптуватися до вподобань користувачів, але обмеження проявляються у ситуаціях з «холодним стартом».

Гібридна система, яка об’єднує контентний і колаборативний підходи, досягла найкращих результатів серед усіх методів. Precision (88%), Recall (85%) та NDCG (92%) демонструють високу ефективність системи як для нових користувачів, так і для тих, що мають багату історію взаємодій. Оптимальний баланс між контентним та колаборативним компонентами досягається при розподілі ваг 0.7/0.3, що забезпечує найкраще поєднання точності та охоплення.

Система рекомендацій категорій товарів також продемонструвала високу ефективність. При розподілі ваг 0.6/0.4 між індивідуальними інтересами та глобальною популярністю досягнуто Precision 83% та Recall 85%. Це підтверджує важливість персоналізації рекомендацій для активних користувачів та підтримуючу роль глобальної популярності для нових користувачів.

Ефективність пошуку та фільтрації товарів оцінено за метриками Precision (85%), Recall (80%), MAP (82%) та Latency (0.25 с). Це підтверджує здатність системи забезпечувати швидкий доступ до релевантних товарів і широкий вибір для користувачів, покращуючи загальний досвід взаємодії з платформою.

Окремо було проведено аналіз роботи системи у сценаріях «холодного старту». Використання глобальної популярності дозволило забезпечити стабільну точність рекомендацій для нових користувачів, тоді як контентне фільтрування інтегрувало нові товари в систему. Це підтверджує гнучкість розробленого підходу та його адаптивність до різних сценаріїв.

Виконана робота демонструє, що розроблена система рекомендацій є ефективним інструментом для покращення користувацького досвіду в інтернет-магазинах. Високі результати за метриками точності, охоплення та швидкості роботи підтверджують доцільність застосування запропонованих підходів. Подальший розвиток системи може включати інтеграцію більш складних алгоритмів машинного навчання, а також оптимізацію обчислень для роботи з великими наборами даних. Результати цієї роботи створюють базу для подальших досліджень у галузі рекомендаційних систем для електронної комерції.

# ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ


1. 20 biggest challenges of ecommerce and how to handle them - shiprocket. Shiprocket. URL: https://www.shiprocket.in/blog/challenges-of-ecommerce/ (дата звернення: 11.12.2024).

2. A hybrid recommendation system: a review / A. Chaudhari та ін. *IEEE access*. 2024. С. 1. URL: https://doi.org/10.1109/access.2024.3480693 (дата звернення: 09.12.2024).

3. Aggarwal C. C. Ensemble-Based and hybrid recommender systems. *Recommender systems*. Cham, 2016. С. 199–224. URL: https://doi.org/10.1007/978-3-319-29659-3_6 (дата звернення: 08.12.2024).

4. Arabi H., Balakrishnan V. Personalized hybrid book recommender. International journal of information systems in the service sector. 2019. Vol. 11, no. 3. P. 70–97. URL: https://doi.org/10.4018/ijisss.2019070105 (дата звернення: 01.12.2024).

5. Best practices on recommendation systems. GitHub. URL: https://github.com/recommenders-team/recommenders (дата звернення: 20.11.2024).

6. Cold start aware hybrid recommender system approach for E-commerce users / S. G. K. Patro та ін. *Soft computing*. 2022. URL: https://doi.org/10.1007/s00500-022-07378-0 (дата звернення: 17.11.2024).

7. Collaborative product recommendation system for e-commerce websites / S. G. Sundaram та ін. *2022 international conference on innovative computing, intelligent communication and smart electrical systems (ICSES)*, м. Chennai, India, 15–16 лип. 2022 р. 2022. URL: https://doi.org/10.1109/icses55317.2022.9914275 (дата звернення: 03.12.2024).

8. Designing Multi-Modal Embedding Fusion-Based Recommender / A. Wróblewska та ін. *Electronics*. 2022. Т. 11, № 9. С. 1391. URL: https://doi.org/10.3390/electronics11091391 (дата звернення: 16.12.2024).

9. ECommerce growth outpaces general retail sales, hits 15.6% | pymnts.com. PYMNTS.com. URL: https://www.pymnts.com/news/ecommerce/2024/ecommerce-growth-outpaces-general-retail-sales-hits-15percent/ (дата звернення: 16.11.2024).

10. Elias C. Cellphones market stocks from "americanas". Kaggle. URL: https://www.kaggle.com/datasets/cauelias/cellphones-stocks-from-americanas (дата звернення: 26.11.2024).

11. Exploring the landscape of hybrid recommendation systems in e-commerce: a systematic literature review / K. C. Bodduluri та ін. *IEEE access*. 2024. С. 1. URL: https://doi.org/10.1109/access.2024.3365828 (дата звернення: 24.11.2024).

12. Flask documentation. Flask. URL: https://flask.palletsprojects.com/en/stable/ (дата звернення: 14.12.2024).

13. GeeksforGeeks. Python API tutorial: getting started with apis. GeeksforGeeks. URL: https://www.geeksforgeeks.org/python-api-tutorial-getting-started-with-apis/ (дата звернення: 12.12.2024).

14. How to automate tasks with cron jobs in linux – thelinuxcode. TheLinuxCode. URL: https://thelinuxcode.com/how-to-automate-tasks-with-cron-jobs-in-linux/ (дата звернення: 07.12.2024).

15. Hybrid recommendation approaches / D. Jannach et al. *Recommender systems*. Cambridge. P. 124–142. URL: https://doi.org/10.1017/cbo9780511763113.007 (дата звернення: 21.11.2024).

16. Hybrid recommender systems: beginner's guide. Marketsy.ai. URL: https://marketsy.ai/blog/hybrid-recommender-systems-beginners-guide (дата звернення: 26.11.2024).

17. Jeffery chiang. 7 types of hybrid recommendation system. Medium. URL: https://medium.com/analytics-vidhya/7-types-of-hybrid-recommendation-system-3e4f78266ad8 (дата звернення: 04.12.2024).

18. Kechinov M. ECommerce events history in electronics store. Kaggle. URL: https://www.kaggle.com/datasets/mkechinov/ecommerce-events-history-in-electronics-store (дата звернення: 26.11.2024).

19. Kechinov M. ECommerce purchase history from electronics store. Kaggle. URL: https://www.kaggle.com/datasets/mkechinov/ecommerce-purchase-history-from-electronics-store (дата звернення: 26.11.2024)..

20. Manasi Vilas Takle, Aarti Nandkumar Thorat, Pranali Shridhar Naik. Product-Based collaborative filtering recommendation system for E- commerce. *International journal of advanced research in science, communication and technology*. 2023. С. 695–696. URL: https://doi.org/10.48175/ijarsct-13199 (дата звернення: 04.12.2024).

21. Meirnizri. Cellphones recommendations. Kaggle. URL: https://www.kaggle.com/datasets/meirnizri/cellphones-recommendations (дата звернення: 26.11.2024).

22. Milli S., Garg N., Pierson E. Choosing the right weights: balancing value, strategy, and noise in recommender systems. URL: https://doi.org/10.48550/arXiv.2305.17428.

23. Online consumer decision making / D. Jannach et al. *Recommender systems*. Cambridge. P. 234–252. URL: https://doi.org/10.1017/cbo9780511763113.012 (дата звернення: 01.12.2024).

24. Optimized recommendation system for e-commerce on product features and user behavior. *International journal of recent technology and engineering*. 2019. Т. 8, № 2. С. 748–759. URL: https://doi.org/10.35940/ijrte.b2401.078219 (дата звернення: 05.12.2024).

25. Pande C., Witschel H. F., Martin A. New hybrid techniques for business recommender systems. *Applied sciences*. 2022. Т. 12, № 10. С. 4804. URL: https://doi.org/10.3390/app12104804 (дата звернення: 14.12.2024).

26. SQLite - python. TutorialsPoint. URL: https://www.tutorialspoint.com/sqlite/sqlite_python.htm (дата звернення: 17.11.2024).

27. Scikit-learn: machine learning in Python. Scikit-learn documentation. URL: https://scikit-learn.org/stable/index.html (дата звернення: 23.11.2024).

28. Shynhalov D., Meleshko Y., Bosko V. Дослідження відкритих наборів даних веб-ресурсів у контексті застосування їх для тестування рекомендаційних систем. *Системи управління, навігації та зв'язку. Збірник наукових праць*. 2019. Т. 4, № 56. С. 110–113. URL: https://doi.org/10.26906/sunz.2019.4.110 (дата звернення: 22.11.2024).

29. Stouffer J., Dwyer G., Aggarwal S. Flask: building python web services. Packt Publishing, 2017. 770 с.

30. Vaidya N., Khachane A. R. Recommender systems-the need of the ecommerce ERA. *2017 international conference on computing methodologies and communication (ICCMC), Erode*, 18–19 July 2017. 2017. URL: https://doi.org/10.1109/iccmc.2017.8282616 (дата звернення: 11.12.2024).

31. What is a recommendation system?. NVIDIA Data Science Glossary. URL: https://www.nvidia.com/en-us/glossary/recommendation-system (дата звернення: 01.12.2024).

32. Worldwide ecommerce market share statistics and trends in 2024. Influencer Marketing Hub. URL: https://influencermarketinghub.com/ecommerce-market-share-statistics/ (дата звернення: 03.12.2024).

33. Zaizi F. E., Qassimi S., Rakrak S. Multi-objective optimization with recommender systems: a systematic review. *Information systems*. 2023. P. 102233. URL: https://doi.org/10.1016/j.is.2023.102233 (дата звернення: 06.12.2024).

34. Впровадження гібридних підходів у системах рекомендацій: акцент на кол. peerdh.com. URL: https://peerdh.com/uk/blogs/programming-insights/implementing-hybrid-approaches-in-recommendation-systems-a-focus-on-collaborative-filtering-and-content-based-techniques (дата звернення: 18.11.2024).

35. Впровадження гібридних систем рекомендацій за допомогою Scikit-learn. peerdh.com. URL: https://peerdh.com/uk/blogs/programming-insights/implementing-hybrid-recommendation-systems-using-scikit-learn (дата звернення: 12.12.2024).

36. Шварц М. Є. Гібридні моделі і методи прогнозування рекомендацій для інтернет-магазину : автореф. дис. … канд. техн. наук. Львів, 2019. 20 с. URL: https://ena.lpnu.ua/handle/ntb/45445 (дата звернення: 17.11.2024).

# ДОДАТОК А

Відомість кваліфікаційної роботи

| Позначення | Найменування | Дод. відомості |
|---|---|---|
| | Текстові документи | |
| **1.** | Пояснювальна записка | 121 с. |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | Інші документи | |
| | | |
| **2.** | Презентаційні матеріали | __ плакатів |
| | | |
| | | |
| | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | |
| | | | | | | | |
| . | . | Прізвище та ініціали. | Підп. | Дата | | | |
| Розробив | | Антіпін В.С. | | 16.12 | Розробка рекомендаційної системи для інтернет-магазину з використанням гібридних підходів | Шифр групи | Код напр./спец. |
| Перевірив | | Дейнеко А.О. | | | | ДСм-23-1 | 122 |
| Н.контр. | | Малєєва І.А. | | | | ХНУРЕ кафедра ШІ | |
| Затв. | | Золотухін О.В. | | | | | |