МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАДИМА ГЕТЬМАНА

Навчально-науковий інститут
«Інститут інформаційних технологій в економіці»

Кафедра математичного моделювання та статистики

Освітньо-професійна програма «Економічна кібернетика і Дата Сайнс»

Галузь знань 05 «Соціальні та поведінкові науки»

Спеціальність 051 «Економіка»

Форма навчання: очна (денна)

КВАЛІФІКАЦІЙНА МАГІСТРЕСЬКА РОБОТА

на тему **«Розробка персоналізованої рекомендаційної системи для стримінгових сервісів»**

здобувача Миколюка Владислава Руслановича

_____
(підпис)

Науковий керівник: доцент, к.е.н. Ольга ОСИПОВА

_____
(підпис)

Робота допущена до захисту перед екзаменаційною комісією з атестації здобувачів вищої освіти (ЕК)

Завідувач кафедри ММС: кандидат фізико-математичних наук, професор Галина ВЕЛИКОІВАНЕНКО

_____
(підпис)

Київ 2024

ЗМІСТ

ВСТУП

Рекомендаційні системи сьогодні є невід'ємною частиною багатьох цифрових сервісів, зокрема стримінгових платформ, які надають користувачам доступ до великого обсягу контенту, включаючи фільми, серіали, музичні треки та інші медіаматеріали. В умовах зростання кількості даних і різноманіття доступного контенту ключовим завданням стає надання індивідуальних рекомендацій, які відповідають уподобанням кожного окремого користувача. Персоналізація не лише покращує користувацький досвід, але й сприяє зростанню лояльності аудиторії, ефективності сервісу та його комерційного успіху.

У даній кваліфікаційній магістерській роботі досліджується проблема оптимізації роботи електронних комерційних платформ та розглядаються різні методи й моделі для покращення персоналізації користувацького досвіду. Дослідження включає аналіз та порівняння різних підходів до персоналізації, таких як колаборативна фільтрація, контентна фільтрація, методи гібридних систем, а також їх ефективність у збільшенні релевантності рекомендацій і конверсії на основі реальних даних.

Актуальність теми зумовлена стрімким зростанням індустрії стримінгових платформ та підвищенням вимог до якості обслуговування користувачів. Запропоновані в роботі підходи можуть бути корисними для покращення функціоналу існуючих сервісів, а також для розробки нових платформ з врахуванням сучасних вимог до персоналізації контенту.

Персоналізація рекомендацій на стримінгових платформах має велике значення для користувачів, які прагнуть отримати релевантний контент без зайвих зусиль, а також для платформ, які хочуть підвищити залученість аудиторії та рівень її задоволеності. Відомо, що великий обсяг доступного контенту може призводити до так званого "паралічу вибору", коли користувачам важко знайти щось підходяще. Тому наявність ефективних методів персоналізації є ключовою для

покращення користувацького досвіду та підвищення конкурентоспроможності сервісу.

Різноманіття підходів до персоналізації контенту, таких як колаборативна фільтрація, контентна фільтрація та гібридні методи, дозволяє стримінговим платформам враховувати різні аспекти поведінки користувачів і надавати більш точні рекомендації. Швидкий розвиток технологій машинного навчання, зокрема нейронних мереж і методів аналізу великих даних, сприяє постійному вдосконаленню алгоритмів персоналізації та забезпечує їхню адаптацію до динамічних змін у вподобаннях користувачів.

Аналіз останніх досліджень і публікацій. Проблема персоналізації рекомендацій на стримінгових платформах є актуальною та вивчається багатьма науковцями та фахівцями в усьому світі, серед яких зазначимо таких як Y. Koren, R. Bell, C. Volinsky [1], P. Resnick та H. Varian [2], J. Bennett і S. Lanning [3], X. Liu та X. Zhang [4], A. Alslaity [5] та інші.

Загалом, ці дослідження зробили значний внесок у розвиток рекомендаційних систем, що використовуються на стримінгових платформах. Вони впровадили базові методи, як-от співфільтрація, та розробили ефективні алгоритми, зокрема матричну факторизацію і SVD, для точнішої персоналізації рекомендацій. Також було створено гібридні моделі, які поєднують різні підходи, забезпечуючи високу якість рекомендацій. Ці досягнення лягли в основу сучасних систем, які аналізують вподобання користувачів для пропонування релевантного контенту.

Мета і завдання дослідження. Метою даного дослідження є розробка персоналізованої рекомендаційної системи для стримінгових сервісів, яка покращить процес рекомендацій контенту для користувачів на основі їхніх інтересів та поведінки.

Поставлена мета передбачає вирішення таких завдань у даній кваліфікаційній роботі:

- проаналізувати існуючі методи та алгоритми рекомендаційних систем, зокрема в контексті стримінгових платформ;

- оцінити технології та інструменти, які використовуються в стримінгових сервісах для персоналізації контенту;
- розробити архітектуру та дизайн персоналізованої рекомендаційної системи;
- обрати найбільш підходящі алгоритми та методи для рекомендацій з урахуванням специфіки стримінгових сервісів;
- реалізувати систему та протестувати її ефективність через серію експериментів;
- сформулювати висновки щодо отриманих результатів.

Об'єктом дослідження є персоналізовані рекомендаційні системи стримінгових сервісів, що забезпечують індивідуалізовану пропозицію контенту на основі вподобань і поведінки користувачів.

Предметом дослідження є технології та методи створення персоналізованих рекомендаційних систем для стримінгових сервісів, спрямованих на аналіз поведінкових даних для покращення користувацького досвіду.

Методи дослідження. Основою для кваліфікаційної роботи стали дослідження науковців та фахівців щодо різноманітних аспектів розробки та оцінки ефективності рекомендаційних систем, зокрема в контексті стримінгових сервісів. Під час виконання роботи було використано відомі підходи та алгоритми, такі як контентне фільтрування, а також методи персоналізації контенту для аналізу вподобань користувачів. Для реалізації та тестування розробленої системи було застосовано середовище програмування Python.

Теоретична, методична та практична значущість отриманих результатів. Теоретична значущість отриманих результатів полягає в розвитку наукових підходів до створення персоналізованих рекомендаційних систем, зокрема в контексті стримінгових сервісів. Розроблені моделі та алгоритми персоналізації контенту розширюють існуючі теоретичні знання в галузі машинного навчання, обробки даних і поведінкової аналітики. Окрім цього, дослідження сприяє глибшому розумінню процесів взаємодії користувачів з

цифровими платформами та важливості індивідуального підходу до контенту для підвищення залученості користувачів.

Методична значущість полягає в розробці методології створення та тестування персоналізованих рекомендаційних систем. Це включає вибір ефективних алгоритмів, побудову архітектури системи, а також методи оцінки точності та релевантності рекомендацій. Отримані методи можуть бути застосовані для оптимізації інших систем персоналізації в різних сферах, де важлива адаптація контенту до індивідуальних переваг користувачів.

Практична значущість отриманих результатів полягає в створенні робочої моделі персоналізованої рекомендаційної системи для стримінгових сервісів, що може бути безпосередньо впроваджена на платформах для покращення досвіду користувачів. Вона дозволяє підвищити точність та релевантність рекомендацій, що в свою чергу може привести до збільшення користувацької лояльності, більш ефективної взаємодії з контентом та покращення бізнес-показників компаній, що надають такі послуги.

Інформаційною базою в даній кваліфікаційній роботі є публікації різних авторів, наукові статті та технічні звіти, що стосуються розробки рекомендаційних систем та методів персоналізації контенту. Крім того, для аналізу користувацьких даних та побудови моделі рекомендаційної системи використовувалися відкриті набори даних про поведінку користувачів стримінгових платформ, а також ресурси для роботи з великими даними та машинним навчанням, зокрема бібліотеки Python.

Структура кваліфікаційної роботи. Кваліфікаційна робота складається із вступу, трьох розділів, висновків та списку використаних джерел.

РОЗДІЛ 1
ОГЛЯД ТА АНАЛІЗ ІСНУЮЧИХ РЕКОМЕНДАЦІЙНИХ СИСТЕМ

1.1 Сучасні підходи та алгоритми в розробці рекомендаційних систем

Рекомендаційні системи використовуються в багатьох галузях сучасного світу, допомагаючи покращити досвід користувачів, підвищити ефективність і персоналізувати взаємодію з продуктами чи послугами. Ось кілька основних сфер застосування:

1. Електронна комерція (E-commerce):

- Amazon, eBay, Aliexpress та інші онлайн-магазини використовують рекомендаційні системи для того, щоб пропонувати користувачам продукти на основі їх попередніх покупок або переглядів, а також на основі поведінки інших користувачів із схожими уподобаннями.

Приклад: на головній сторінці Amazon ви можете побачити безліч рекомендацій, персоналізованих під ваші вподобання. Система аналізує вашу історію покупок, перегляди та взаємодію з іншими користувачами, щоб запропонувати релевантні товари. Наприклад, якщо ви недавно купували книгу про кулінарію, вам можуть запропонувати новий кухонний комбайн або збірник рецептів.

2. Мультимедійні платформи (стримінгові сервіси):

- Netflix, Spotify, YouTube використовують рекомендаційні алгоритми для персоналізації контенту. Наприклад, Netflix рекомендує фільми і серіали на основі того, що ви вже дивилися, а Spotify — музику відповідно до ваших уподобань.

Netflix використовує складні алгоритми для персоналізації контенту. На основі ваших переглядів, оцінок та поведінки інших користувачів зі схожими смаками, вам пропонуються фільми та серіали, які вам можуть сподобатися.

Spotify створює персоналізовані плейлисти, такі як "Discover Weekly", які містять нові пісні, які вам можуть сподобатися. Алгоритми Spotify аналізують вашу музичну бібліотеку, а також дані про те, яку музику слухають ваші друзі.

3. Соціальні мережі:

- Facebook, Instagram, TikTok використовують рекомендаційні системи для того, щоб пропонувати користувачам пости, відео, рекламу та інший контент, що може їх зацікавити, на основі їхньої активності та інтересів.

Вкладка "Explore" в Instagram показує вам пости, які можуть вас зацікавити, на основі ваших підписок, лайків та коментарів. Система також враховує вашу геолокацію та час дня, щоб пропонувати більш релевантний контент.

4. Інтернет-реклама:

- Рекомендаційні системи використовуються для таргетингу реклами, щоб показувати користувачам продукти та послуги, які вони з більшою ймовірністю придбають або зацікавляться, зважаючи на їхні попередні пошукові запити, покупки або поведінку в Інтернеті.

5. Освітні платформи та онлайн-курси:

- Платформи, такі як Coursera, Udemy, Khan Academy, використовують рекомендаційні системи для того, щоб пропонувати курси на основі інтересів та попередніх навчальних досягнень користувачів.

6. Туризм та готельний бізнес:

- Платформи для бронювання, такі як Airbnb чи Booking.com, рекомендують варіанти житла та напрямки на основі попередніх пошуків користувачів та їхніх уподобань.

7. Онлайн-новини та медіа:

- Рекомендаційні системи також використовуються на новинних платформах, таких як Google News чи Apple News, щоб персоналізувати потік новин відповідно до інтересів читача.

8. Здоров'я та медицина:

У галузі медицини рекомендаційні системи можуть допомогти у виборі ліків, діагностичних тестів або лікарів на основі медичних записів і історії пацієнта.

9. Фінансові послуги:

У фінансових технологіях, наприклад, банки чи платформи для інвестицій використовують рекомендаційні системи для надання персоналізованих порад щодо інвестицій, кредитів або заощаджень на основі фінансового стану користувача.

10. Ігри та розваги:

Відеоігри можуть застосовувати рекомендаційні системи для пропонування нових ігор, рівнів або віртуальних товарів, що можуть зацікавити гравця, виходячи з його вподобань чи попереднього досвіду.

Завдяки використанню таких систем, компанії можуть значно покращити персоналізацію послуг і товарів, що підвищує задоволеність користувачів та їх лояльність.

Рекомендаційні системи (РС) [6] – це технології, що використовуються для персоналізації взаємодії користувача з різними сервісами, зокрема в інтернет-торгівлі, стримінгових платформах, соціальних мережах, тощо. Основною метою РС є допомогти користувачам знаходити релевантний контент серед великої кількості доступних варіантів, підвищуючи задоволення від взаємодії та ефективність використання сервісу (див рис 1.1).

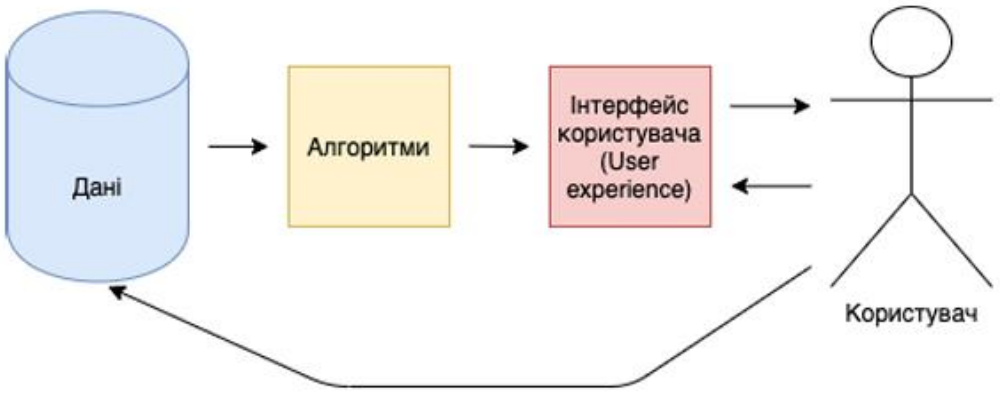


Рисунок 1.1 - Принцип роботи РС

Джерело: розроблено автором

Основні підходи до розробки рекомендаційних систем:

1. Колаборативна фільтрація [7]. Є одним з найпоширеніших підходів до створення рекомендаційних систем. Вона ґрунтується на припущенні, що користувачі, які мали схожі вподобання в минулому, будуть мати схожі уподобання й у майбутньому. Існує два основних типи колаборативної фільтрації:

Фільтрація на основі користувачів (User-based Collaborative Filtering): Рекомендації формуються на основі схожості між користувачами.

Фільтрація на основі предметів (Item-based Collaborative Filtering): Рекомендації формуються на основі схожості між предметами або контентом, яким користувач вже цікавився (див.рис.1.2).

- Переваги:

Простота реалізації.

Ефективність при наявності великих наборів даних від користувачів.

- Недоліки:

Проблема холодного старту: важко рекомендувати нові об'єкти або користувачів без історії.

Скалірованість: з великою кількістю користувачів та об'єктів, алгоритми можуть бути надто повільними.

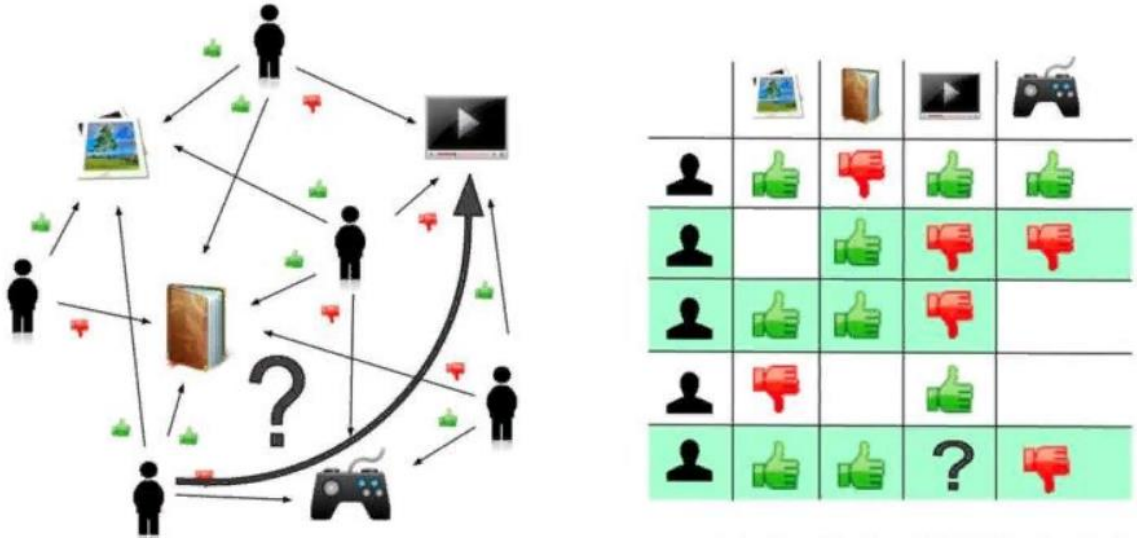


Рисунок 1.2 – Робота колоборативної фільтрації

Джерело: розроблено автором

Даний підхід ефективний, але має обмеження, зокрема, потребу у великій кількості даних для створення надійних рекомендацій, а також проблему "холодного старту", коли система не має достатньо інформації про нових користувачів або нові предмети.

2. Контентна фільтрація (Content-Based Filtering) [8]. Цей підхід використовує характеристику предметів для створення рекомендацій. Для кожного елемента контенту (наприклад, фільму, товару) система будує профіль на основі його властивостей, а потім рекомендує інші предмети, схожі на ті, що користувач уже обрав або переглядав. Наприклад, на основі жанру, автора чи іншої інформації про фільм система може рекомендувати інші фільми схожого жанру.

Цей метод зручний для випадків, коли колаборативна фільтрація може бути малоефективною, проте він обмежений у здатності вловлювати складні взаємозв'язки між елементами.

Приклад: якщо користувач часто дивиться фільми певного жанру (наприклад, наукова фантастика), система буде рекомендувати йому нові фільми цього ж жанру.

- Переваги:

Не залежить від інших користувачів, тому не виникає проблеми холодного старту для нових користувачів.

Легко налаштувати під специфічні потреби користувача.

- Недоліки:

Обмеження на основі конкретних характеристик об'єктів (наприклад, важко рекомендувати нові об'єкти без достатньої інформації).

Може виникнути проблема "замкнутого кола" — система рекомендує лише схожі на вже переглянуті об'єкти, що обмежує різноманіття.

3. Гібридні методи рекомендаційних систем поєднують в собі переваги колаборативної фільтрації (Collaborative Filtering, CF) та контентної фільтрації (Content-Based Filtering, CBF), що дозволяє компенсувати обмеження кожного з цих підходів і створювати більш точні та ефективні рекомендації.

Основні підходи до гібридних методів:

Комбінування результатів різних моделей: система може комбінувати результати, отримані від колаборативної та контентної фільтрацій, щоб надавати рекомендації. Наприклад, можна зважити результати з обох підходів та інтегрувати їх для отримання кращих рекомендацій.

Метод, що використовує один підхід для визначення важливості іншого: один метод може допомагати визначити значимість або параметри для іншого методу. Наприклад, колаборативна фільтрація може виявити схожість між користувачами, що допомагає підвищити точність рекомендацій за допомогою контентної фільтрації.

Паралельне застосування методів: контентна фільтрація та колаборативна фільтрація можуть працювати паралельно, створюючи рекомендації для різних груп користувачів. Наприклад, система може використовувати колаборативну фільтрацію для активних користувачів з багатим профілем, а для нових користувачів застосовувати контентну фільтрацію, оскільки у них ще немає достатньо даних для колаборативних рекомендацій.

Моделі на основі мета-даних: для створення рекомендацій система може поєднувати мета-дані контенту (наприклад, категорії товарів, жанри фільмів) з даними поведінки користувачів (наприклад, рейтинги або перегляди), щоб знайти найбільш релевантні рекомендації.

Переваги гібридних методів:

Покращення точності рекомендацій: завдяки поєднанню різних джерел інформації, гібридні системи можуть створювати більш точні рекомендації, що враховують різні аспекти користувацьких уподобань.

Уникнення проблеми холодного старту: холодний старт — це проблема, коли система не має достатньо інформації про нових користувачів або нові елементи. Гібридні методи дозволяють компенсувати цю проблему, застосовуючи контентну фільтрацію для нових користувачів або нових товарів, поки не збереться достатньо даних для ефективної колаборативної фільтрації.

Зменшення обмежень окремих методів: кожен з класичних підходів має свої недоліки (наприклад, колаборативна фільтрація не працює для нових користувачів або товарів, контентна фільтрація може бути обмежена лише тим, що відомо про товар). Гібридні методи дозволяють знижувати ці обмеження.

Недоліки гібридних методів:

Складність реалізації: комбінування різних підходів потребує додаткових зусиль при розробці системи, як у плані алгоритмів, так і у плані обробки даних. Це може бути технічно складним і вимагати більше часу та ресурсів.

Потреба в більшій кількості обчислювальних ресурсів: оскільки гібридні методи часто поєднують кілька алгоритмів, їх реалізація може вимагати більших обчислювальних потужностей, що збільшує вимоги до апаратного забезпечення.

Ризик надмірної складності моделей: у разі неправильного застосування або комбінування методів система може стати занадто складною для користувачів або важкою для управління.

Приклад використання гібридних методів:

Amazon: Amazon використовує гібридний підхід для рекомендацій товарів, поєднуючи колаборативну фільтрацію (аналізуючи покупки інших користувачів, які мають подібні інтереси) та контентну фільтрацію (рекомендації на основі характеристик товарів, таких як категорія, бренд, ціна). Такий підхід дозволяє знижувати ефективність "холодного старту" і створювати персоналізовані рекомендації для нових та існуючих користувачів.

4. Методи машинного навчання, зокрема алгоритми класифікації та кластеризації для передбачення уподобань користувачів. Глибоке навчання, яке використовує нейронні мережі, застосовується для побудови складних моделей, здатних враховувати величезні обсяги даних і знайти приховані патерни в інформації про користувачів і контент. Одним із прикладів є використання рекурентних нейронних мереж (RNN) для прогнозування вподобань користувачів на основі їхньої історії взаємодій.

Глибоке навчання: використання нейронних мереж для побудови складних моделей рекомендацій, здатних виявляти приховані патерни у великих даних.

- Переваги:

Може ефективно працювати з великими і складними наборами даних.

Можна моделювати більш складні патерни в даних.

- Недоліки:

Потрібно більше даних для навчання.

Високі вимоги до обчислювальних ресурсів.

5. Алгоритми факторизації матриць використовуються для зменшення розмірності великих матриць і є потужним інструментом для виділення латентних (невидимих) факторів, що лежать в основі взаємодій між користувачами та предметами (наприклад, товарами, фільмами, музикою тощо). Вони застосовуються, зокрема, в рекомендаційних системах, де необхідно передбачити вподобання користувачів на основі їх попередніх взаємодій з предметами.

Основні поняття

У класичній задачі факторизації матриць ми маємо матрицю взаємодій R розміру m×n, де:

m — кількість користувачів,

n — кількість предметів (наприклад, фільмів),

елементи матриці Rij — це оцінки, які користувач iii поставив предмету j.

Матриця R часто буває дуже розрідженою, тобто більшість елементів відсутні. Метою є побудова моделі, яка дозволить передбачити відсутні оцінки, виявити приховані фактори і покращити рекомендації.

Алгоритми факторизації матриць

SVD (Singular Value Decomposition)

SVD — це один з найбільш відомих методів для факторизації матриць. Цей метод розкладає матрицю на три складові:

R=UΣVTR

де:

U — матриця розміру m×k, де k — це кількість латентних факторів,

Σ\SigmaΣ — діагональна матриця розміру k×k, яка містить сингулярні числа (які визначають важливість кожного латентного фактора),

VTV^TVT — матриця розміру k×n, що містить інформацію про предмети в контексті латентних факторів.

SVD дозволяє зменшити розмірність і виявляти латентні фактори, такі як жанри фільмів або переваги в покупках.

Застосування SVD у рекомендаційних системах включає:

Виявлення основних латентних факторів (наприклад, теми фільмів, які можуть впливати на оцінки),

Створення прогнозів відсутніх оцінок через відновлення матриці R.

ALS (Alternating Least Squares)

ALS є популярним методом для факторизації розріджених матриць. У цьому методі ми мінімізуємо помилку відновлення матриці за допомогою чергування двох етапів:

Фіксуємо матрицю V і знаходимо U,

Фіксуємо матрицю U і знаходимо V.

Це дозволяє знаходити латентні фактори в розріджених матрицях, що робить його корисним для задач, де матриця взаємодій користувачів і предметів є великою та розрідженою.

Non-negative Matrix Factorization (NMF)

NMF є варіантом факторизації, де обидві матриці U і V обмежуються невід'ємними значеннями. Цей підхід корисний, коли значення в матриці (наприклад, оцінки або кількість покупок) не можуть бути від'ємними.

Ідея полягає в тому, щоб знайти таку факторизацію матриці, яка б наближала її з урахуванням обмеження на невід'ємність елементів.

Застосування факторизації матриць

Рекомендаційні системи: Один з основних напрямків використання алгоритмів факторизації матриць — це рекомендаційні системи. Алгоритми, як SVD або ALS, допомагають прогнозувати вподобання користувачів на основі їх попередніх взаємодій (оцінок, переглядів, покупок).

Обробка природної мови: Факторизація матриць може бути використана для аналізу текстових даних. Наприклад, для пошуку латентних тем в колекціях документів (тема може бути представлена як вектор, подібно до латентних факторів у рекомендаційних системах).

Аналіз зображень та відео: Застосовується для зменшення розмірності зображень або відео, виділення суттєвих ознак (наприклад, при стисканні зображень).

6. Генерація рекомендацій за допомогою випадкових лісів і ансамблів. Дозволяють об'єднувати результати декількох моделей для покращення точності рекомендацій. Випадкові ліси є ансамблевим методом, що включає в себе

комбінацію декількох дерев рішень, що забезпечує стабільність та підвищену точність моделей (див.рис.1.3).

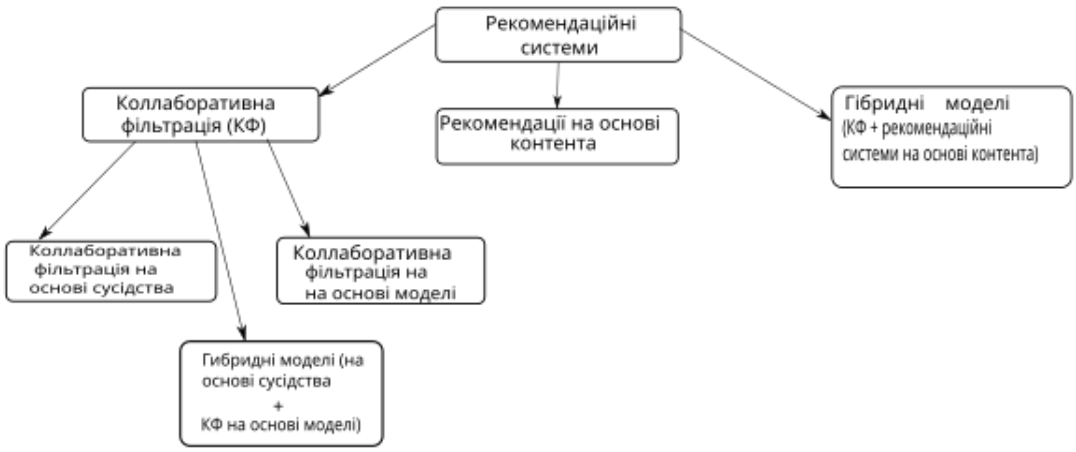


Рисунок 1.3 – Види рекомендаційних систем

Джерело: розроблено автором

Технології та інструменти для розробки рекомендаційних систем:

Сучасні алгоритми для розробки рекомендаційних систем часто використовують такі технології як Python, R, а також спеціалізовані бібліотеки, такі як Surprise, TensorFlow, PyTorch, що дозволяють розробляти як традиційні, так і глибокі моделі. Для роботи з великими даними використовуються платформи, як Apache Hadoop, Spark.

У підсумку, сучасні підходи до розробки рекомендаційних систем використовують різноманітні техніки, що дозволяють вирішувати складні задачі персоналізації та покращувати користувацький досвід в умовах обробки великих даних.

Також варто наголосити що роботи вчених у галузі розробки персоналізованих рекомендаційних систем має величезне значення для розвитку сучасних стримінгових сервісів. Завдяки їхнім дослідженням ми можемо насолоджуватися більш персоналізованим контентом, який відповідає нашим індивідуальним смакам і потребам.

Стаття Koren, Bell та Volinsky [1] є фундаментальною роботою, яка показала, як матрична факторизація може бути ефективно використана для побудови рекомендаційних систем.

Ідея: Велика матриця, що відображає оцінки користувачів для різних предметів, розкладається на два менших матриці. Це дозволяє виявити приховані фактори, такі як вподобання користувачів або характеристики предметів.

Переваги:

Точність: Дає точні прогнози навіть за відсутності частини даних.

Ефективність: Знижує обчислювальну складність для великих даних.

Гнучкість: Може бути поєднана з іншими методами.

Застосовність: Працює з різними типами даних взаємодії користувачів.

Недоліки:

Холодний старт: Проблеми з новими користувачами та предметами.

Обчислювальна складність: Високі вимоги до ресурсів для великих даних.

Перенавчання: Модель може бути надмірно налаштована на тренувальні дані. Робота Koren та співавторів є важливим внеском у розвиток рекомендаційних систем. Однак, методи матричної факторизації мають свої обмеження, які потребують подальших досліджень.

Робота "Recommender Systems" П. Ресніка і Г. Веріана [2], опублікована в Communications of the ACM у 1997 році, є одним із основоположних матеріалів у галузі систем рекомендацій. У статті автори досліджують різні методи створення рекомендаційних систем, які допомагають користувачам знаходити інформацію чи продукти, що їх цікавлять, серед великої кількості можливих варіантів. Вони детально описують механізми, на яких базуються ці системи, зокрема методи, такі як фільтрація на основі контенту та колаборативна фільтрація, а також поєднання цих підходів.

Плюси роботи:

Інноваційність: Стаття була однією з перших, що привернула увагу до важливості ідеї рекомендаційних систем.

Доступність: Робота пропонує чітке пояснення основних принципів створення таких систем, що робить її корисною як для дослідників, так і для практиків.

Погляд на майбутнє: Автори бачать потенціал для значного розвитку цієї області, зокрема завдяки розвитку Інтернету та нових технологій.

Мінуси роботи:

Обмеженість технологій того часу: Оскільки стаття була написана в 1997 році, в ній обговорюються методи, які на той час були передовими, але на сьогоднішній день вони вже значно застаріли, зокрема в контексті великих даних та сучасних підходів до машинного навчання.

Простота аналізу: Стаття більше зосереджена на загальних принципах і не містить глибоких технічних або математичних деталей, що може бути обмеженням для тих, хто шукає більш детальну інформацію.

У цілому, ця робота стала основою для подальших досліджень в області рекомендаційних систем і має велике значення для розвитку цієї технології.

Робота "The Netflix Prize" [3] авторів J. Bennett і S. Lanning описує змагання, яке компанія Netflix організувала для поліпшення своєї системи рекомендацій. Метою конкурсу було створити алгоритм, який зможе на 10% точніше передбачити вподобання користувачів на основі їхньої історії переглядів.

Плюси:

Інноваційність: Робота продемонструвала новий підхід до вирішення задачі прогнозування вподобань користувачів, використовуючи спільні зусилля команди з різних областей.

Практичний вплив: Змагання стало основою для подальших розробок у сфері рекомендаційних систем.

Розвиток наукових підходів: Конкурс стимулював розвиток нових методів і алгоритмів у машинному навчанні та статистиці.

Мінуси:

Обмеження даних: Доступ до даних був обмежений, що могло вплинути на точність результатів. Окрім того, дані були історичні, і їх використання не завжди відповідало реальним умовам.

Використання простих методів: Деякі алгоритми, хоча й демонстрували високі результати, не були найбільш інноваційними в довгостроковій перспективі.

Відсутність врахування контексту: Рекомендаційні системи не враховували контекстуальних аспектів (наприклад, часу доби або емоційний стан користувача), що могло б значно підвищити точність.

Робота "Deep Learning Based Recommender System: A Survey and New Perspectives" (2017) авторів X. Liu та X. Zhang [4] є оглядовою статтею, яка досліджує застосування глибокого навчання (deep learning) в рекомендаційних системах. У роботі описуються основні підходи, алгоритми та техніки, що використовуються для покращення точності рекомендацій з допомогою методів глибокого навчання. Окрім того, автори пропонують нові перспективи для розвитку цієї технології, акцентуючи увагу на перевагах і викликах, що виникають при використанні таких методів.

Плюси:

Комплексний огляд: Робота пропонує всебічний огляд сучасних підходів до використання глибокого навчання для рекомендаційних систем, що дає зрозуміле уявлення про поточний стан галузі.

Аналіз нових підходів: Автори не лише описують існуючі методи, але й пропонують нові перспективи для розвитку, що може стимулювати подальші дослідження.

Практичне застосування: Висвітлені ідеї та методи можуть бути корисними для розробників рекомендаційних систем, оскільки вони допомагають покращити точність і персоналізацію рекомендацій.

Мінуси:

Висока вимога до ресурсів: Одним з головних недоліків глибокого навчання є потреба в значних обчислювальних ресурсах та великих наборах даних, що може обмежувати використання таких систем на менш потужних платформах.

Обмеження щодо інтерпретованості: Моделі глибокого навчання часто є "чорними ящиками", що ускладнює розуміння того, як вони приймають рішення. Це може бути проблемою для розробників, які хочуть зрозуміти, чому система зробила певне передбачення.

Залежність від даних: Якість рекомендацій сильно залежить від наявних даних. Якщо дані неповні або неякісні, то й точність рекомендацій може значно погіршитися.

Загалом, робота є корисним ресурсом для дослідників та розробників, що працюють у сфері рекомендаційних систем, оскільки дає чітке уявлення про сучасні досягнення та напрямки розвитку цієї технології, а також вказує на можливі труднощі і обмеження, з якими можна зіткнутися при впровадженні глибокого навчання.

Робота А. Алслаїті "Towards a Comprehensive Evaluation of Recommenders: A Cognition-Based Approach", опублікована в 2018 році [5], пропонує новий підхід до оцінки ефективності систем рекомендацій, зосереджуючи увагу на когнітивних аспектах взаємодії користувачів з такими системами. Автор розглядає існуючі методи оцінки систем рекомендацій і пропонує розширити їх за рахунок врахування когнітивних процесів, які відбуваються під час взаємодії користувачів з рекомендаційними системами.

Плюси роботи:

Інноваційний підхід: Стаття вводить нову перспективу для оцінки ефективності рекомендаційних систем, що дозволяє краще розуміти, як саме користувачі взаємодіють з такими системами.

Теоретична глибина: Автор пропонує розширене бачення оцінки систем, яке включає когнітивні, психологічні та поведінкові аспекти, що може призвести до покращення якості рекомендацій.

Відповідність сучасним тенденціям: Пропозиція щодо використання когнітивних процесів для оцінки систем рекомендацій відповідає тенденціям, що з'являються в дослідженнях штучного інтелекту та взаємодії людини з технологіями.

Мінуси роботи:

Теоретична складність: Когнітивний підхід є складним і може бути важким для реалізації на практиці. Для його впровадження потрібно більше досліджень і тестувань, що може потребувати значних ресурсів.

Обмежене застосування: Пропоновані критерії оцінки можуть бути важкими для масштабного впровадження, особливо в комерційних і практичних додатках, де швидкість і простота оцінки є важливими.

Відсутність практичних прикладів: Хоча стаття пропонує теоретичні основи для нового підходу, вона не містить достатньо реальних прикладів або кейсів, що могли б продемонструвати ефективність когнітивного підходу на практиці.

Загалом, робота пропонує цікаву та новаторську ідею для оцінки рекомендаційних систем, зосереджуючи увагу на важливих психологічних і когнітивних аспектах, але її реалізація на практиці може бути складною.

Пропоновані нові методи, такі як використання когнітивних аспектів в оцінці ефективності систем, або застосування глибокого навчання, відкривають нові горизонти для поліпшення точності рекомендацій, але й створюють нові виклики, зокрема, в обчислювальних ресурсах та інтерпретованості моделей.

В цілому, РС значно покращують користувацький досвід, однак, через свої складнощі та обмеження, потребують подальших досліджень і вдосконалення для досягнення максимального потенціалу.

1.2 Методики персоналізації контенту в рекомендаційних системах

Персоналізація контенту є ключовим аспектом рекомендаційних систем (РС), оскільки вона дозволяє створювати індивідуальні рекомендації для кожного користувача на основі його уподобань, інтересів та поведінки. Різноманітні методики персоналізації застосовуються для того, щоб забезпечити найкращий досвід користувачів і підвищити ефективність сервісів.

Відповідно до цього, персоналізація контенту може охоплювати різні стратегії та технології, серед яких виділяються такі основні підходи:

1. Колаборативна фільтрація. Один із найпоширеніших підходів до персоналізації контенту. За допомогою колаборативної фільтрації система робить персоналізовані рекомендації на основі уподобань інших користувачів, схожих до поточного. Технологія передбачає створення профілю користувача, що містить історію його взаємодії з різними елементами контенту. Для нових користувачів та предметів, коли даних недостатньо, можуть застосовуватись гібридні підходи, які поєднують колаборативну фільтрацію з іншими методами.

1.1. Метод на основі користувачів (User-based) [9]: Система намагається знайти користувачів із схожими вподобаннями і пропонує контент, який вони оцінили високо.

1.2. Метод на основі елементів (Item-based): Система аналізує, які предмети були схожі на ті, що користувач вже переглядав чи оцінював, і на основі цього генерує рекомендації.

1.3. . Функція recommend — це часто використовувана функція в системах рекомендацій, яка дозволяє знаходити схожі об'єкти (в даному випадку, фільми) на основі певних характеристик або параметрів. Вона є важливим елементом у більшості рекомендальних систем, зокрема в системах для медіаконтенту, таких як фільми, музика, книги тощо.

Зазвичай такі функції застосовуються в алгоритмах Content-Based Filtering (фільтрація на основі контенту) або Collaborative Filtering (фільтрація на основі співпраці). Вони допомагають зібрати список фільмів, схожих на той, який користувач вже переглянув чи оцінив, на основі різних характеристик: жанру, акторів, режисера, теми, сюжету, або ж відгуків інших користувачів.

2. Контентна фільтрація [10]. Фокусується на характеристиках самого контенту для генерації персоналізованих рекомендацій. Замість того щоб шукати схожих користувачів, система звертає увагу на характеристики предметів, які користувач вже переглядав або оцінив. Наприклад, для стримінгових сервісів це можуть бути жанри, актори, режисери чи ключові слова, що описують фільми чи серіали.

2.1. Профіль користувача: Створюється на основі контенту, який він споживав, щоб передбачити інші предмети, які можуть бути йому цікаві.

2.2. Аналіз тексту та метаданих[11]: Часто використовуються методи обробки природної мови (NLP), щоб автоматично класифікувати і порівнювати контент на основі його змісту.

3. Гібридні методи. Поєднують різні методи, зокрема колаборативну та контентну фільтрацію, для досягнення кращих результатів. Такий підхід дозволяє покращити точність рекомендацій, зменшити проблеми, пов'язані з "холодним стартом", та забезпечити більш персоналізований досвід для користувачів.

Комбінація моделей: Використовуються як методи колаборативної фільтрації, так і контентні методи, а потім їх результати комбінуються для досягнення найкращих рекомендацій.

Рейтингова система: Наприклад, система може спочатку використовувати контентні рекомендації для нових користувачів, а потім переключатися на колаборативні методи, коли з'являється достатньо даних [12].

4. Персоналізація через контекст. Сучасні системи часто враховують не лише вподобання користувача, але й контекст, у якому відбувається взаємодія з контентом. Це може включати:

4.1. Час і місце: Рекомендації можуть варіюватися залежно від часу дня або місця перебування користувача. Наприклад, для користувачів, які слухають музику вранці, система може порекомендувати музику для ранкових пробуджень, а вночі – спокійніші композиції.

4.2. Пристрій: Рекомендації можуть бути адаптовані під тип пристрою, яким користується людина (смартфон, планшет, комп'ютер).

5. Алгоритми на основі глибокого навчання. У останні роки все більшу популярність здобувають методи глибокого навчання для персоналізації контенту. Глибокі нейронні мережі можуть аналізувати складні патерни в даних, що дозволяє виявляти приховані зв'язки між користувачами та контентом. Ці алгоритми здатні навчатися на великих обсягах даних, що дозволяє зробити рекомендації ще точнішими та персоналізованими [13]:

5.1. Рекурентні нейронні мережі (RNN): Використовуються для аналізу послідовних даних, таких як історія переглядів чи прослуховувань, для побудови прогнозів щодо майбутніх уподобань.

5.2. Кодувальники: Застосовуються для зменшення розмірності даних і виділення важливих характеристик контенту для створення персоналізованих рекомендацій.

6. Адаптивне навчання та онлайн-методи [14]. Іншим важливим аспектом персоналізації є адаптація рекомендацій до змінюваних вподобань користувача. Це може включати використання алгоритмів, що постійно адаптуються до нових даних і зворотного зв'язку від користувача:

6.1. Методи онлайн-навчання: Системи, що постійно оновлюють свої моделі в реальному часі, зважаючи на нові взаємодії користувачів, дозволяють підтримувати актуальність рекомендацій.

6.2. Рекомендації на основі зворотного зв'язку: Враховують не лише перегляди та покупки, але й явний відгук користувача, наприклад, оцінки або коментарі.

7. Соціальна персоналізація. Цей підхід бере до уваги соціальні мережі та відгуки інших користувачів. Рекомендації можуть бути сформовані на основі активності друзів або впливових осіб (наприклад, блогерів чи знаменитостей), а також на основі того, що інші користувачі з подібними інтересами або соціальними зв'язками оцінювали високо.

1.3 Аналіз технологій та інструментів, що використовуються у стримінгових сервісах

Стримінгові сервіси, такі як Netflix, Spotify, YouTube, Amazon Prime Video та інші [15], здобули велику популярність завдяки своїй здатності надавати користувачам безперешкодний доступ до мультимедійного контенту (відео, музика, подкасти тощо). Для ефективної роботи таких платформ, а також для персоналізації контенту та забезпечення безперебійного сервісу, використовуються різні технології та інструменти. Ось кілька основних категорій технологій, які застосовуються в стримінгових сервісах:

1. Технології обробки та доставки мультимедійного контенту

1.1. CDN (Content Delivery Network). Стримінгові сервіси активно використовують мережі доставки контенту (CDN) [16], які дозволяють забезпечити високу швидкість завантаження та відтворення мультимедійного контенту з мінімальними затримками. CDN складаються з численних серверів, розташованих в різних частинах світу, що забезпечує швидку доставку контенту користувачам, незалежно від їхнього місця знаходження.

1.2. Кодування та стиснення відео та аудіо. Для ефективного зберігання і передачі великих обсягів мультимедійного контенту використовуються алгоритми стиснення, такі як H.264, HEVC (H.265) для відео та AAC для аудіо. Вони дозволяють зменшити розмір файлів без суттєвої втрати якості, що критично

важливо для стримінгових сервісів з високими вимогами до пропускної здатності мереж [17].

1.3. Адаптивний бітрейт (ABR). Технологія адаптивного бітрейту дає можливість динамічно налаштовувати якість відео або аудіо в залежності від швидкості інтернет-з'єднання користувача. Це дозволяє зменшити буферизацію та підтримувати плавне відтворення контенту навіть за умов поганого з'єднання [18].

2. Алгоритми рекомендацій та персоналізація контенту

2.1 Машинне навчання та глибоке навчання. Для покращення рекомендаційних систем стримінгові сервіси використовують різноманітні алгоритми машинного навчання, зокрема колаборативну фільтрацію, контентну фільтрацію та гібридні методи. Застосовуються також глибокі нейронні мережі для побудови складних моделей, які можуть адаптуватися до поведінки користувача в реальному часі. Це дозволяє рекомендувати персоналізований контент на основі вподобань, історії переглядів або оцінок.

2.2 Аналіз великих даних (Big Data). Стримінгові сервіси обробляють великі обсяги даних, які збираються про кожного користувача: історію переглядів, взаємодію з контентом, оцінки, час перегляду, демографічні дані і т.д. Використовуючи технології для обробки великих даних, такі як Apache Hadoop, Spark та інші, сервіси можуть створювати більш точні моделі для прогнозування уподобань і поведінки користувачів.

2.3 Персоналізація через соціальні мережі та зворотний зв'язок. Багато стримінгових платформ також інтегруються з соціальними мережами, дозволяючи користувачам обмінюватися рекомендаціями, створювати плейлисти або ділитися контентом [19]. Соціальні сигнали (наприклад, лайки, коментарі, активність друзів) використовуються для вдосконалення рекомендацій.

3. Інструменти для роботи з медіа-контентом

3.1. Мультимедійні плеєри та програвачі. Веб та мобільні додатки стримінгових сервісів зазвичай включають вбудовані медіаплеєри для відтворення контенту. Ці плеєри підтримують різні формати відео та аудіо, вміють працювати

з субтитрами, інтерактивними елементами та іншими функціями, необхідними для комфортного перегляду.

3.2. Інтерактивні елементи. Деякі сервіси додають інтерактивні функції до контенту, наприклад, вбудовані вікторини, вибір варіантів розвитку подій у відео або можливість вибору сценаріїв (як у форматі "перегляд по вимогу"). Це дозволяє створювати більш персоналізований та захоплюючий досвід для користувачів.

4. Обробка та моніторинг даних в реальному часі

4.1. Обробка в реальному часі (Real-time Data Processing). Стримінгові сервіси використовують технології для обробки даних у реальному часі, що дає можливість адаптувати рекомендації та вміст на основі поведінки користувачів. Наприклад, якщо користувач додає новий фільм у список бажаного або оцінює серіал, система може миттєво відобразити ці зміни в рекомендаціях.

4.2. Аналіз поведінки користувача. Для покращення взаємодії з платформою та збільшення часу перегляду контенту, стримінгові сервіси здійснюють постійний моніторинг поведінки користувачів (наприклад, які серії фільмів вони дивляться, на якому контенті зупиняються, як часто ставлять паузу). Ці дані використовуються для оптимізації рекомендацій та покращення UX.

5. Інфраструктура та хмарні сервіси

5.1. Хмарні платформи. Для забезпечення масштабованості та високої доступності стримінгові сервіси часто використовують хмарні платформи, такі як Amazon Web Services (AWS), Google Cloud, Microsoft Azure. Це дозволяє ефективно управляти великими обсягами даних та масштабувати інфраструктуру в залежності від попиту.

Системи зберігання даних. Стримінгові сервіси використовують різноманітні системи для зберігання великої кількості контенту (фільмів, серіалів, музичних треків тощо). Вони можуть використовувати як традиційні реляційні бази даних (для зберігання метаданих), так і NoSQL системи (для зберігання неструктурованих даних, таких як відео- та аудіофайли) [20].

РОЗДІЛ 2
ТЕХНОЛОГІЇ ТА ІНСТРУМЕНТИ РОЗРОБКИ ПЕРСОНАЛІЗОВАНОЇ РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ

2.1 Постановка задачі розробки системи

Розробка персоналізованої рекомендаційної системи для стримінгових сервісів є складною і багатогранною задачею, що передбачає вирішення низки технічних та практичних питань. Головною метою цієї системи є створення механізму, який би ефективно і точно надавав рекомендації користувачам щодо контенту (фільми, серіали, музика, подкасти, відео тощо) в залежності від їхніх індивідуальних уподобань, історії переглядів, рейтингу, а також інших параметрів взаємодії з платформою. Врахування цих факторів є критично важливим для підвищення задоволеності користувачів і утримання їх на сервісі (див.рис. 2.1).

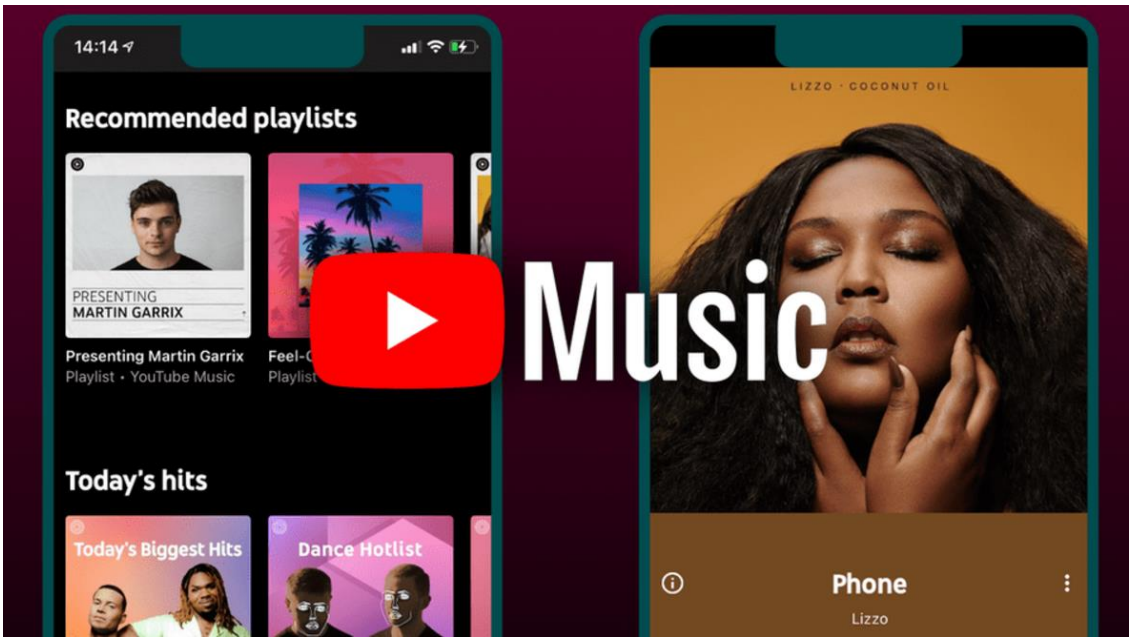


Рисунок 2.1- Приклад персоналізованої рекомендації [53]

Персоналізація рекомендацій. Основною задачею персоналізованої рекомендаційної системи є створення індивідуальних рекомендацій для кожного користувача. Це включає в себе аналіз попередніх переглядів і виборів користувача, його оцінок контенту, поведінкових патернів та взаємодії з платформою (наприклад, час доби, пристрій доступу тощо) [21]. Персоналізовані рекомендації повинні враховувати такі чинники:

Історія переглядів: Дослідження контенту, який користувач переглядав раніше, дозволяє зробити висновки про його уподобання щодо жанрів, акторів, режисерів тощо.

Оцінки та відгуки: Врахування оцінок, які користувач ставить контенту, дозволяє ще точніше прогнозувати його майбутні вподобання (див рис 2.2).

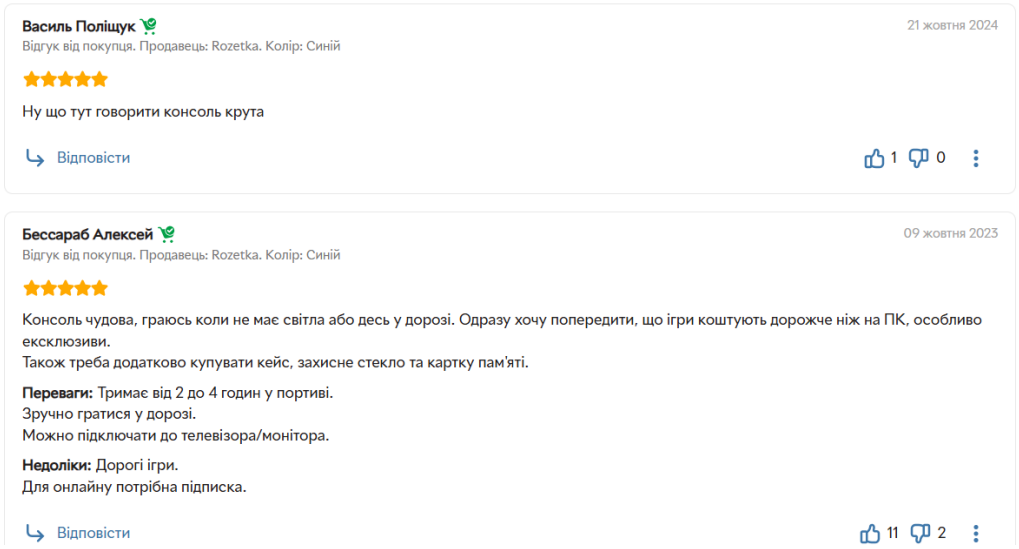


Рисунок 2.2 - Приклад відгуків [54]

Швидкість перегляду та завершеність: Система повинна враховувати не лише те, що було переглянуте, але і наскільки користувач завершив перегляд, що може вказувати на більш глибоке зацікавлення певним типом контенту.

Адаптивність до змін у поведінці користувача. Персоналізовані рекомендації повинні бути динамічними і здатними адаптуватися до змін у перевагах користувача. Наприклад, якщо користувач починає цікавитися новими жанрами або зміною контексту (наприклад, під час святкових періодів або зміни сезону), система повинна швидко змінити рекомендації відповідно до нових

патернів. Важливо, щоб система могла відслідковувати зміни у поведінці користувача і відповідно оновлювати профіль його вподобань [22].

Забезпечення масштабованості. Стримінгові сервіси обслуговують велику кількість користувачів одночасно. Тому рекомендована система повинна бути масштабованою, щоб обробляти велику кількість запитів без зниження ефективності. Це включає в себе:

Обробка великих обсягів даних: Система повинна бути здатна працювати з великими базами даних, включаючи численні параметри, що стосуються кожного користувача, та величезні каталоги контенту.

Швидкість обробки запитів: Рекомендації повинні генеруватися в реальному часі, забезпечуючи миттєвий відгук на запити користувачів. Час затримки між запитом і відповіддю повинен бути мінімальним, щоб забезпечити безперебійну роботу сервісу.

Інтеграція з іншими системами стримінгового сервісу. Система має бути гнучкою і інтегруватися з іншими компонентами стримінгової платформи, такими як:

Механізм пошуку контенту: Рекомендаційна система повинна доповнювати функціонал пошуку контенту, пропонуючи відповідні варіанти навіть тоді, коли користувач не знає точно, що шукає [23].

Категоризація контенту: Алгоритми персоналізації повинні враховувати категорії, жанри та підкатегорії контенту для точнішого прогнозування уподобань (див рис 2.3).

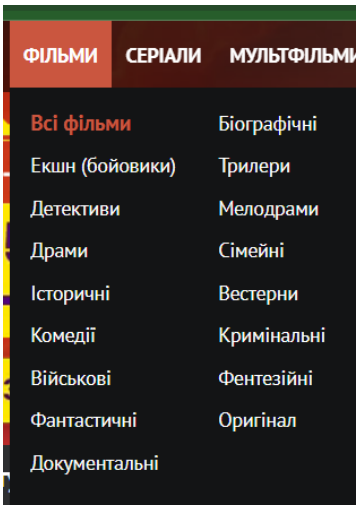


Рисунок 2.3 - Приклад категоризації по вподобанням [55]

Системи управління користувачами: Рекомендаційна система повинна мати можливість інтегрувати профілі користувачів, зберігати історію їхніх дій на платформі та здійснювати персоналізацію на основі цих даних.

Забезпечення різноманітності рекомендацій. Для уникнення одноманітності в рекомендаціях та збереження інтересу користувачів, система повинна пропонувати не тільки найбільш популярні варіанти, але й цікаві та менш очевидні варіанти контенту. Це включає:

Рекомендації на основі контенту, схожого на улюблений: Використання алгоритмів, які аналізують схожість між контентом (наприклад, на основі жанрів, акторів, тематики) і пропонують користувачеві нові варіанти [23].

Різноманітність у запропонованому контенті: Для підтримки зацікавленості користувачів можна використовувати стратегії, які заохочують користувачів дивитися нові або менш популярні матеріали.

Безпека та конфіденційність даних. Персоналізовані рекомендації вимагають обробки особистих даних користувачів, що ставить питання захисту конфіденційності цих даних. Система повинна забезпечувати:

Захист даних користувачів: Всі персональні дані, такі як історія переглядів, оцінки контенту, мають бути захищені від несанкціонованого доступу та зловживання[24].

Відповідність нормам безпеки: Система повинна відповідати стандартам безпеки, таким як GDPR (Загальний регламент захисту даних ЄС), та іншим вимогам, що регулюють обробку персональних даних.

Оцінка ефективності рекомендаційної системи. Необхідно оцінити ефективність системи після її впровадження. Це включає:

Аналіз задоволення користувачів: Використання зворотного зв'язку від користувачів для вдосконалення алгоритмів рекомендацій.

Використання метрик якості: Наприклад, такі показники, як точність рекомендацій, рівень утримання користувачів і конверсії (як часто рекомендації призводять до дій, таких як перегляд контенту)[25].

Таким чином, розробка персоналізованої рекомендаційної системи для стримінгових сервісів вимагає вирішення цілої низки завдань, які спрямовані на забезпечення високої якості рекомендацій, масштабованості системи та безпеки обробки даних, а також інтеграції з іншими компонентами платформи для досягнення найкращих результатів.

2.2 Обґрунтування вибору алгоритмів та методів

Проведення процесу обґрунтованого вибору алгоритмів і методів для розробки рекомендаційної системи для стрімінгових сервісів — це складний етап, який потребує уважного аналізу даних, потреб користувачів, а також вибору відповідних підходів для досягнення найкращих результатів у персоналізації рекомендацій [26]. Ось кроки та фактори, які слід врахувати при виборі алгоритмів для стрімінгових платформ:

1. Аналіз вимог та цілей

Перед тим, як вибирати методи, потрібно чітко зрозуміти цілі вашої рекомендаційної системи:

Покращення користувацького досвіду — наприклад, рекомендація контенту (фільмів, музики тощо), що найбільше відповідає вподобанням користувача.

Монетизація — збільшення продажів, переглядів або підписок, що досягається за допомогою персоналізованих рекомендацій.

Динамічність контенту — в стрімінгових сервісах контент постійно оновлюється, тому рекомендаційна система має адаптуватися до нових даних.

2. Аналіз даних

Стрімінгові сервіси мають багатий набір даних про взаємодію користувачів з контентом:

Історія переглядів або прослуховувань (для фільмів, серіалів, музики).

Рейтинги та відгуки від користувачів (якщо вони є).

Демографічні дані (вік, місцезнаходження, уподобання).

Контентні метадані — жанр, актори, режисери, ключові слова.

Типи даних визначають, який алгоритм буде найбільш ефективним.

3. Вибір відповідного методу

Для вибору оптимального алгоритму для рекомендаційної системи стрімінгового сервісу слід врахувати такі фактори:

Тип даних: Якщо є багато метаданих про контент (жанри, актори, режисери тощо), може бути корисним контентно-орієнтоване фільтрування. Якщо ж основний акцент на взаємодії користувачів, то краще вибирати колаборативне фільтрування.

Розмір і динаміка даних: Для великих наборів даних і часто оновлюваного контенту гібридні або глибокі алгоритми можуть бути кращими, оскільки вони можуть адаптуватися до змін.

Проблема "холодного старту": Гібридні методи або алгоритми на основі факторизації матриць можуть допомогти вирішити проблему, коли новий контент або нові користувачі не мають достатньо даних для точних рекомендацій [27].

Обчислювальні ресурси: Алгоритми на основі глибокого навчання або гібридні методи можуть бути дуже ресурсозатратними, тому їх використання слід обґрунтовувати з урахуванням можливостей інфраструктури

4. Оцінка ефективності

Після вибору алгоритму важливо протестувати його ефективність, оцінюючи такі метрики:

- Точність рекомендацій (precision, recall).
- Різноманітність рекомендацій (diversity).
- Користувацька задоволеність (через опитування або аналіз поведінки).

Вибір і налаштування рекомендаційної системи для стрімінгового сервісу є ключовим етапом для забезпечення високої якості персоналізованих рекомендацій, що підвищують задоволеність користувачів і залучення до платформи.

Вибір алгоритму для рекомендаційної системи (РС) безпосередньо залежить від кількох ключових факторів, серед яких можна виділити такі:

1. Характер даних

Різні типи даних впливають на вибір алгоритму рекомендаційної системи:

Тип даних: Стрімінгові сервіси, як правило, мають великий обсяг даних про взаємодію користувачів з контентом (перегляди, прослуховування, рейтинги), а також метадані про сам контент (жанри, актори, виконавці). Якщо метадані про контент є багатими (наприклад, жанр, виконавці, ключові слова для фільмів, музики або подкастів), то контентно-орієнтоване фільтрування може бути доцільним. Водночас, якщо дані про взаємодії користувачів мають велику вагу, кращими будуть колаборативні методи (на основі користувачів чи елементів).

Наявність метаданих: У випадку, коли метаданих про контент обмаль, можна застосовувати колаборативне фільтрування, оскільки цей метод не потребує додаткової інформації про сам контент [28].

Динамічність даних: Стрімінгові платформи швидко оновлюють контент, і тому система повинна бути здатною швидко адаптуватися до нових даних. У цьому випадку, методи, що не потребують повного перерахунку всіх рекомендацій після кожної зміни контенту (наприклад, факторизація матриць чи методи глибокого навчання), можуть бути більш ефективними.

2. Технічні обмеження

Обчислювальні ресурси: Алгоритми на основі глибокого навчання (наприклад, нейронні мережі, рекурентні мережі або трансформери) зазвичай потребують великих обчислювальних ресурсів і можуть бути важкими для реалізації на обмежених інфраструктурах [29]. Якщо платформа не має достатньо потужних серверів або ресурсів для тренування складних моделей, кращим вибором буде застосування легших методів, таких як SVD (сингуларне розкладання матриць) або методи на основі контенту.

Масштабованість: Стрімінгові сервіси часто мають величезні обсяги користувачів та контенту. Вибір алгоритму повинен враховувати здатність системи

ефективно працювати з великими даними. Методи, такі як колаборативне фільтрування на основі елементів, або гібридні підходи, можуть бути більш масштабованими завдяки можливості зберігати і обробляти рекомендації в зручному вигляді для великих даних.

Інтеграція з іншими системами: Якщо система повинна інтегруватися з іншими платформами або використовувати зовнішні API (наприклад, для отримання додаткових даних про контент), вибір алгоритму повинен передбачати можливість легкої інтеграції, зокрема у випадку використання гібридних методів або глибоких нейронних мереж, які можуть обробляти різноманітні джерела інформації.

3. Час навчання моделі

Вибір алгоритму також залежить від того, скільки часу ви готові витратити на навчання моделі:

Швидкість навчання: Для деяких алгоритмів, як-от SVD або KNN (метод найближчих сусідів), процес навчання відносно швидкий і дозволяє швидко тестувати і модифікувати модель, особливо коли дані постійно оновлюються. Однак, складніші методи, як нейронні мережі або глибоке навчання, можуть вимагати значних ресурсів та часу для тренування, що може бути обмеженням при необхідності швидко реагувати на нові дані або змінювати параметри.

Час обробки нових даних: Стрімінгові сервіси мають постійно змінювані дані — нові користувачі, новий контент, нові взаємодії. Алгоритм повинен бути здатний швидко адаптуватися до цих змін. Наприклад, методи на основі глибокого навчання можуть вимагати тривалого часу на перенавчання моделей, тоді як гібридні методи можуть бути більш гнучкими і швидкими у адаптації до нових даних.

4. Основна мета розробки PC

Мета розробки рекомендаційної системи визначає, який підхід буде найбільш ефективним:

Персоналізація користувацького досвіду: Якщо головною метою є покращення персоналізації, важливо вибирати методи, які надають точні рекомендації. Для цього підходять колаборативне фільтрування (як на основі користувачів, так і на основі елементів), а також гібридні методи та глибокі нейронні мережі, які можуть знаходити складні патерни у взаємодії користувачів з контентом [31].

Управління доходами: Якщо основною метою є збільшення монетизації (наприклад, продажі підписок або рекомендації преміум-контенту), вибір алгоритму може зосереджуватися на максимізації конверсій (наприклад, рекомендація контенту, що найбільше відповідає попиту). У такому випадку гібридні моделі або методи, які оптимізують рекомендації на основі економічних метрик (зниження часу перегляду, максимізація користувальницької участі), можуть бути ефективними.

Рішення проблеми "холодного старту": Якщо системи необхідно працювати з новими користувачами або контентом (наприклад, нові користувачі не мають достатньо даних для традиційного колаборативного фільтрування), слід обрати гібридні методи або контентно-орієнтовані алгоритми, які не вимагають великих обсягів історії взаємодій.

2.3 Архітектура та дизайн системи

Архітектура та дизайн персоналізованої рекомендаційної системи для стримінгових сервісів є важливим етапом розробки, оскільки він визначає, як система буде взаємодіяти з користувачем, обробляти великі обсяги даних та генерувати персоналізовані рекомендації. Основна мета архітектури полягає в

тому, щоб забезпечити ефективність, масштабованість та зручність для кінцевого користувача, при цьому оптимізуючи витрати ресурсів і час обробки запитів.

Основні компоненти системи включають інтерфейс користувача, серверну частину, базу даних і механізм алгоритмів рекомендацій. Інтерфейс користувача (UI) є тим елементом, з яким безпосередньо взаємодіє кінцевий користувач. Він повинен бути інтуїтивно зрозумілим, дозволяти швидко знаходити потрібний контент та надавати користувачеві персоналізовані рекомендації. Це можуть бути списки, надані алгоритмами системи, підготовлені на основі попередньої активності користувача, його вподобань та переглянутих матеріалів. Важливо, щоб інтерфейс надавав зручні інструменти для взаємодії, наприклад, фільтри для вибору контенту, рейтинг матеріалів, можливість подивитись або прослухати контент, а також інші функціональні можливості для покращення взаємодії [31].

Серверна частина системи займається обробкою запитів, які надходять від користувачів, генеруванням рекомендацій та виконанням всіх основних алгоритмічних операцій. Вона відповідає за прийом та відправку даних між клієнтським інтерфейсом і базою даних, а також взаємодіє з механізмами персоналізації [32]. Серверна частина має бути розроблена таким чином, щоб мати змогу обробляти тисячі, а можливо й мільйони запитів одночасно, забезпечуючи високий рівень продуктивності та мінімальні затримки при наданні рекомендацій.

База даних є ще одним важливим компонентом архітектури, оскільки вона зберігає всю необхідну інформацію про користувачів, їхню історію переглядів, уподобання, а також про сам контент. Вибір типу бази даних залежить від характеру даних, які потрібно зберігати. Якщо більша частина даних структурована, може бути використана реляційна база даних, наприклад, PostgreSQL. Якщо ж система повинна працювати з неструктурованими даними або великими обсягами інформації, доцільно застосувати NoSQL бази даних, такі як MongoDB або Cassandra [33]. У разі необхідності швидкого доступу до даних, можна використовувати системи кешування для зберігання найбільш затребуваних даних, що дозволяє значно зменшити час на обробку запитів.

Основною частиною системи є механізм рекомендацій, який генерує персоналізовані пропозиції на основі даних про користувача і контент. Це досягається через використання різноманітних алгоритмів, таких як колаборативна фільтрація, контентна фільтрація або гібридні моделі, що комбінують різні підходи. Колаборативна фільтрація зосереджена на аналізі поведінки користувачів та надання рекомендацій на основі схожих вподобань інших людей. Контентна фільтрація, у свою чергу, орієнтована на аналіз характеристик самого контенту, наприклад, жанрів фільмів або категорій музики, і надає рекомендації, що базуються на схожості між предметами контенту. Гібридні моделі поєднують обидва підходи, що дозволяє підвищити точність рекомендацій.

Для розробки архітектури вибираються відповідні технології та інструменти, які допоможуть реалізувати всі вимоги до продуктивності та ефективності системи [34]. Одним з важливих аспектів є вибір мов програмування та фреймворків. Python є популярною мовою для реалізації алгоритмів машинного навчання, завдяки численним бібліотекам, таким як TensorFlow, PyTorch, Scikit-learn, які значно спрощують процес розробки моделей. Для серверної частини можуть бути використані такі фреймворки, як Flask або Django, які дозволяють швидко створювати веб-сервіси. Для масштабування та обробки великих обсягів даних часто застосовують технології, такі як Apache Spark, що дозволяють обробляти дані в реальному часі, та Apache Kafka, яка забезпечує обробку потокових даних.

Важливою частиною архітектури є забезпечення масштабованості, оскільки рекомендаційні системи для стримінгових сервісів повинні бути здатними обробляти великі обсяги даних [35]. Масштабованість можна досягти через використання мікросервісної архітектури, коли кожен компонент системи працює незалежно, і його можна масштабувати окремо в залежності від навантаження. Розподілені системи, такі як Hadoop або Spark, дозволяють здійснювати обробку великих обсягів даних на кількох серверах одночасно, що значно підвищує продуктивність і знижує час обробки запитів.

Безпека та конфіденційність даних — це ще одна важлива складова дизайну системи. Для того, щоб гарантувати захист персональних даних користувачів, система повинна використовувати шифрування даних під час зберігання та передачі, а також забезпечувати надійну аутентифікацію та авторизацію користувачів. Регулярні перевірки на вразливості та моніторинг безпеки системи дозволяють своєчасно виявляти та усувати можливі загрози[36].

Ітераційний підхід до розробки є важливим для постійного вдосконалення рекомендаційної системи. Це включає регулярний аналіз ефективності рекомендацій за допомогою метрик, таких як точність (precision) або відгук (recall), а також збір зворотного зв’язку від користувачів. Це дозволяє адаптувати алгоритми та поліпшити точність рекомендацій, що в свою чергу сприяє кращому задоволенню потреб користувачів та підвищує ефективність роботи системи в цілому.

Загалом, архітектура персоналізованої рекомендаційної системи має бути комплексною та гнучкою, здатною обробляти великі обсяги даних, масштабуватися в умовах зростаючого навантаження і при цьому забезпечувати високу продуктивність і безпеку для кінцевих користувачів.

У рамках цієї кваліфікаційної роботи детально буде описано та розроблено лише один ключовий компонент системи — алгоритм рекомендаційних систем (РС). Всі аспекти, пов’язані з його розробкою, описом та реалізацією, будуть детально проаналізовані, починаючи від теоретичних основ, закінчуючи практичним застосуванням та тестуванням цього алгоритму в контексті обраної задачі.

2.4 Особливості адаптації рекомендаційної системи до стримінгових сервісів

Реалізація персоналізованої рекомендаційної системи для стримінгових сервісів вимагає врахування специфічних особливостей цих платформ, таких як велика кількість користувачів, постійно оновлюваний контент, високі вимоги до швидкості обробки запитів і взаємодія з багатьма джерелами даних. У цьому підрозділі описується, як ці фактори мають бути враховані при реалізації системи, зокрема в аспектах обробки даних, алгоритмів рекомендацій, масштабування та інтеграції з існуючими платформами [37].

Однією з основних характеристик стримінгових сервісів є величезна кількість контенту, яку потрібно ефективно організувати та обробляти. Це можуть бути фільми, серіали, музика, подкасти, відео або інший мультимедійний контент, і кожен тип контенту має свої унікальні атрибути, такі як жанр, тривалість, виконавець, дата випуску тощо. Таким чином, при розробці рекомендаційної системи важливо враховувати ці атрибути для точного формування рекомендацій. Окрім цього, необхідно постійно оновлювати бази даних з контентом, щоб користувачі отримували актуальні рекомендації, що відповідають останнім додаванням на платформі.

Для обробки великих обсягів даних, що генеруються стримінговими сервісами, система повинна використовувати відповідні інструменти та технології, які забезпечать високу продуктивність і низький час відгуку. Це можуть бути розподілені системи для обробки даних, такі як Apache Kafka та Apache Spark, які дозволяють обробляти великі потоки даних у реальному часі. Використання таких технологій дозволяє швидко реагувати на нові запити користувачів і актуалізувати рекомендації без затримок. Крім того, для зберігання даних про контент, історії переглядів та вподобань користувачів можна використовувати як реляційні, так і NoSQL бази даних. Реляційні бази даних, такі як PostgreSQL, можуть бути

використані для зберігання структурованих даних, тоді як для зберігання неструктурованих даних, таких як метадані контенту чи перегляди, більш підходять NoSQL рішення, як MongoDB або Cassandra [38].

Ще однією важливою складовою є адаптивність системи до змін у поведінці користувачів та контенті. Стримінгові сервіси зазвичай мають величезний і динамічний обсяг контенту, тому система повинна бути здатною до постійного оновлення рекомендацій. Реалізація системи персоналізації має враховувати не лише поточні вподобання користувача, але й зміну цих вподобань з часом, що дозволяє зберігати релевантність рекомендацій. Для цього можна застосовувати алгоритми, що постійно аналізують нові дані та коригують рекомендації відповідно до нових вподобань.

Масштабованість системи є ключовою вимогою для стримінгових сервісів, оскільки вони повинні одночасно обслуговувати мільйони користувачів по всьому світу. Для досягнення цього ефекту потрібно використовувати мікросервісну архітектуру, яка дозволяє кожному компоненту системи працювати незалежно і масштабуватися за необхідністю. Наприклад, окремі мікросервіси можуть бути відповідальними за обробку даних користувачів, генерацію рекомендацій, зберігання контенту та взаємодію з іншими частинами системи. Це дозволяє ефективно розподіляти навантаження між серверами та зберігати високу продуктивність навіть за високих навантажень. Крім того, система повинна бути здатною швидко адаптуватися до зростаючих обсягів даних, що зберігаються на платформі [39].

Особливістю стримінгових сервісів є також те, що рекомендації мають бути надані в реальному часі, тобто система повинна забезпечити низьку латентність при відправці пропозицій користувачеві. Для цього використовуються технології кешування, такі як Redis або Memcached, які зберігають часто запитувану інформацію, що дозволяє скоротити час доступу до даних. Кешування може бути корисним для рекомендацій, які не змінюються дуже часто, наприклад, для популярних фільмів або музичних треків.

Крім того, важливо інтегрувати рекомендаційну систему з іншими модулями стримінгового сервісу, такими як платіжні системи, рекламні мережі, соціальні функції або аналітика. Рекомендації можуть бути пов'язані з комерційними цілями, такими як монетизація контенту або показ реклами. Соціальна інтеграція також може покращити рекомендації, дозволяючи користувачам отримувати пропозиції на основі вподобань їхніх друзів чи колег.

Безпека є ще однією важливою складовою при реалізації такої системи, оскільки платформи, що працюють з персональними даними, повинні дотримуватися високих стандартів захисту інформації. Реалізація надійних механізмів аутентифікації та авторизації користувачів, використання шифрування даних при передачі та зберіганні інформації, а також регулярний моніторинг безпеки мають бути інтегровані в систему.

Тобто, реалізація персоналізованої рекомендаційної системи для стримінгових сервісів передбачає врахування специфічних вимог до обробки великих обсягів даних, забезпечення високої продуктивності та масштабованості, інтеграцію з іншими системами та гарантування безпеки і конфіденційності. Ці фактори забезпечують ефективну роботу системи, покращують досвід користувачів та дозволяють стримінговим сервісам надавати персоналізовані рекомендації в реальному часі [40].

РОЗДІЛ 3
ПРАКТИЧНА РЕАЛІЗАЦІЯ ПЕРСОНАЛІЗОВАНОЇ РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ

3.1. Огляд та аналіз даних для побудови рекомендаційної системи

Для розробки механізму рекомендаційної системи для стримінгових платформ буде використана мова програмування PYTHON.

Початкові дані для розробки механізму рекомендаційно системи отримано з відкритої платформи Kaggle [52].

База даних складається з двох датасетів:

1. tmdb_5000_credits.csv:

Містить інформацію про знімальну групу та акторський склад фільмів.

Колонки:

- movie_id: Унікальний ідентифікатор фільму.
- title: Назва фільму.
- cast: Інформація про акторів.
- crew: Інформація про знімальну групу.

Загалом записів: 4803.

2. mdb_5000_movies.csv:

Включає загальну інформацію про фільми: бюджет, популярність, рейтинги тощо.

Колонки:

- budget: Бюджет фільму.
- genres: Жанри фільму.
- homepage: Посилання на офіційний сайт фільму.

- id: Унікальний ідентифікатор фільму (збігається з movie_id з першого файлу).
- keywords: Ключові слова .
- original_language: Оригінальна мова.
- original_title: Оригінальна назва.
- overview: Опис фільму.
- popularity: Популярність фільму.
- production_companies: Виробничі компанії.
- production_countries: Країни виробництва.
- release_date: Дата виходу.
- revenue: Доходи.
- runtime: Тривалість фільму.
- spoken_languages: Мови, якими говорять у фільмі.
- status: Статус (наприклад, Released — випущено).
- tagline: Слоган фільму.
- title: Назва фільму.
- vote_average: Середній рейтинг.
- vote_count: Кількість голосів.

Після отримання загальної інформації про набір даних виконуємо візуалізацію для кращого розуміння набору даних[41]. Створимо графік для візуалізації залежності revenue від runtime (див.рис. 3.1)

revenue vs runtime

```
# @title revenue vs runtime

from matplotlib import pyplot as plt
movies.plot(kind='scatter', x='revenue', y='runtime', s=32, alpha=.8)
plt.gca().spines[['top', 'right',]].set_visible(False)
```

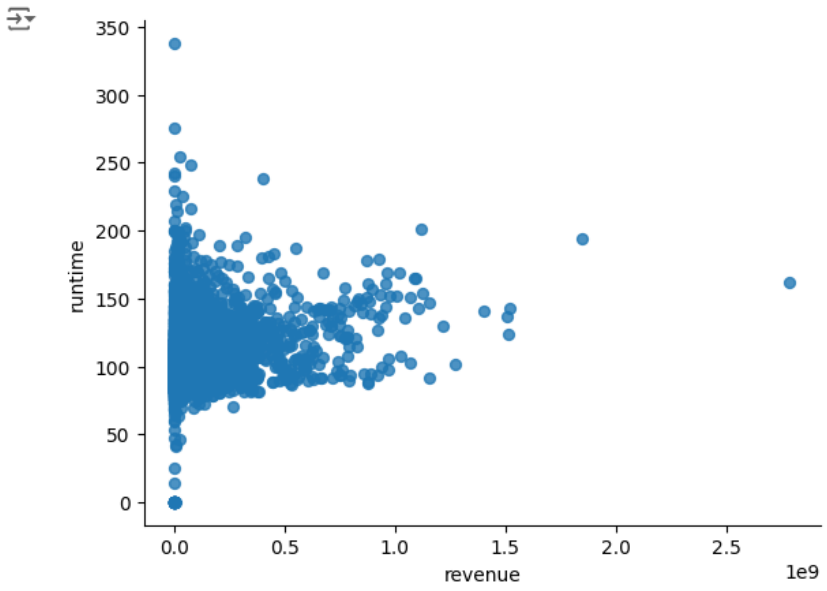


Рисунок 3.1. – Графік залежності revenue від runtime

Джерело: розроблено автором за даними [52]

На графіку видно розкид значень між тривалістю фільмів і їх касовими зборами. Більшість фільмів мають тривалість від 80 до 150 хвилин. Збори таких фільмів варіюються від кількох мільйонів до мільярдів доларів. Можна побачити що фільми з тривалістю близько 120–150 хвилин частіше мають високі касові збори (наприклад, понад $1 млрд). Із аномалій є кілька коротких фільмів із високими доходами, але вони є винятками.

Надто довгі фільми (понад 180 хвилин) рідко показують надзвичайно високі збори.

Далі проведем порівняння між популярністю та касовими зборами фільмів (див.рис. 3.2).

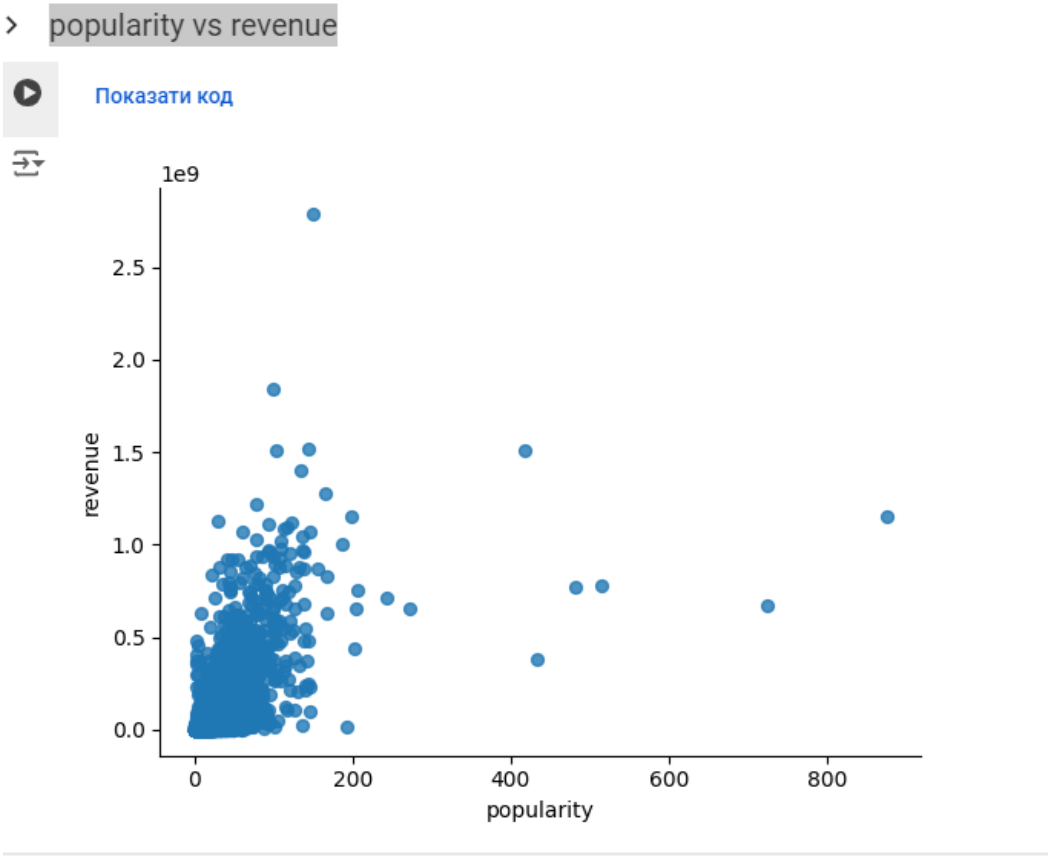


Рисунок 3.2. – Графік порівняння між популярністю та касовими зборами фільмів

Джерело: розроблено автором за даними [52]

Одразу можна побачити такі загальні тенденції:

- Спостерігається велика кількість фільмів з низькою популярністю та низькими касовими зборами (вони зібрані в лівому нижньому куті).
- Є кілька фільмів з високою популярністю, але дуже низькими зборами (що може свідчити про неефективний маркетинг чи інші проблеми).
- Кілька фільмів з дуже високими касовими зборами знаходяться з правого боку графіка (найімовірніше, це великі голлівудські хіти).
- Є точки з дуже високими зборами (більше $1 млрд), але це поодинокі випадки, ймовірно, це блокбастери, що мають високий рівень популярності.

Можна зробити висновок, що висока популярність не завжди гарантує високі касові збори, хоча є деякі виключення для великих хітів [42].

Зробимо аналіз залежності між бюджетом (*budget*) фільмів та їх касовими зборами (*revenue*) (див. рис. 3.3). Це допоможе зрозуміти:

1. Чи завжди високий бюджет означає високі збори?
2. Наскільки ефективно витрачаються кошти для отримання великих доходів?

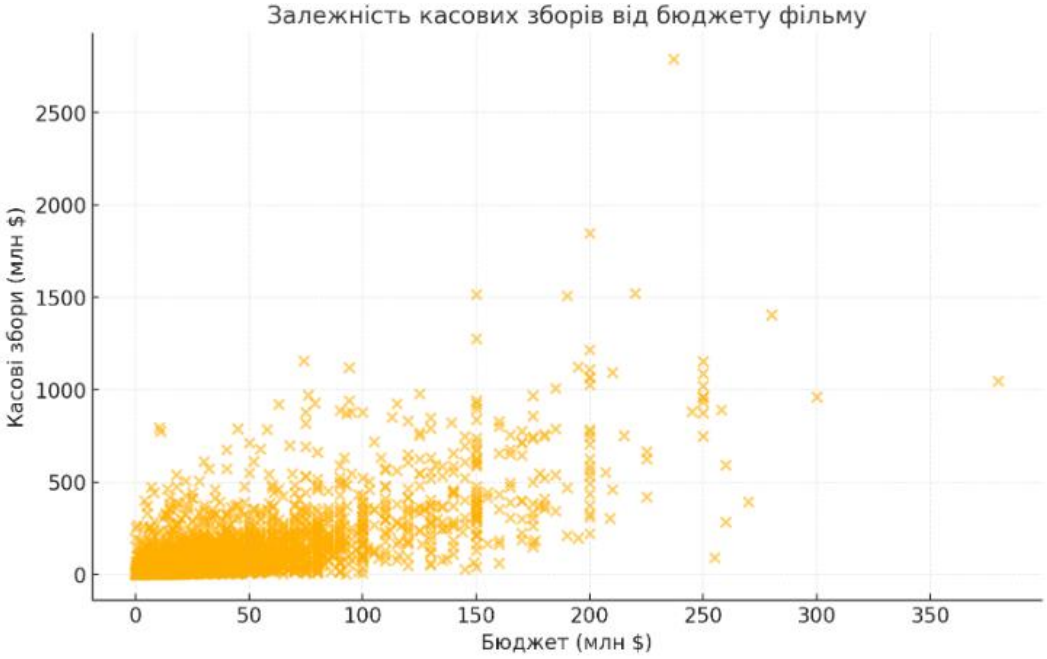


Рисунок 3.3. – Аналіз залежності між бюджетом (*budget*) фільмів та їх касовими зборами (*revenue*)

Джерело: розроблено автором за даними [52]

Із наведеного графіку можна побачити залежність між бюджетом і зборами:

- Є певний тренд: фільми з більшим бюджетом часто досягають вищих касових зборів.
- Однак, високий бюджет не гарантує надзвичайного успіху — деякі дорогі фільми мають відносно низькі збори.
- Деякі фільми з низьким бюджетом демонструють високі доходи. Це може свідчити про успіх незалежних чи малобюджетних хітів.
- Існують кілька фільмів із дуже високими зборами (понад $1 млрд), серед яких фільми з бюджетом як середнього, так і високого рівня.

Цей аналіз дозволяє глибше зрозуміти складну природу кінематографічних успіхів і провалів, які не завжди піддаються простим закономірностям.

3.2. Розробка механізму рекомендацій

Спочатку зчитаємо файл з початковими даними, який був створений на основі даних про світові фільми.

Перший крок - код зчитує дані з файлу tmdb_5000_credits.csv у форматі DataFrame за допомогою бібліотеки pandas.

Роглянемо перші 2 рядки з DataFrame під назвою movies. Даний фрагмент даних з таблиці, яка описує фільми. Кожен рядок містить інформацію про окремий фільм (див.рис 3.4).

Рисунок 3.4. – Дані датасету «movies»

Джерело: розроблено автором за даними [52]

Так як для аналізу використовується 2 датасети, для коректної роботи їх потрібно об’єднати. Код об'єднує два набори даних — movies і credits — за допомогою методу merge на основі спільного стовпця title (назва фільму).

Ось що робить кожна частина коду:

print("before merge") — виводить текст "before merge", щоб показати стан даних до об'єднання.

print(movies.shape) — виводить розміри (кількість рядків і стовпців) набору даних movies.

print(credits.shape) — виводить розміри набору даних credits.

movies=movies.merge(credits, on='title') — об'єднує обидва набори даних за стовпцем title. Після об'єднання результат зберігається в movies.

print("After merge:", movies.shape) — виводить розміри movies після об'єднання [43].

Якщо кількість однакових назв фільмів в обох наборах даних менша за загальну кількість фільмів, то кількість рядків у результаті об'єднання зменшиться (див.рис 3.5).

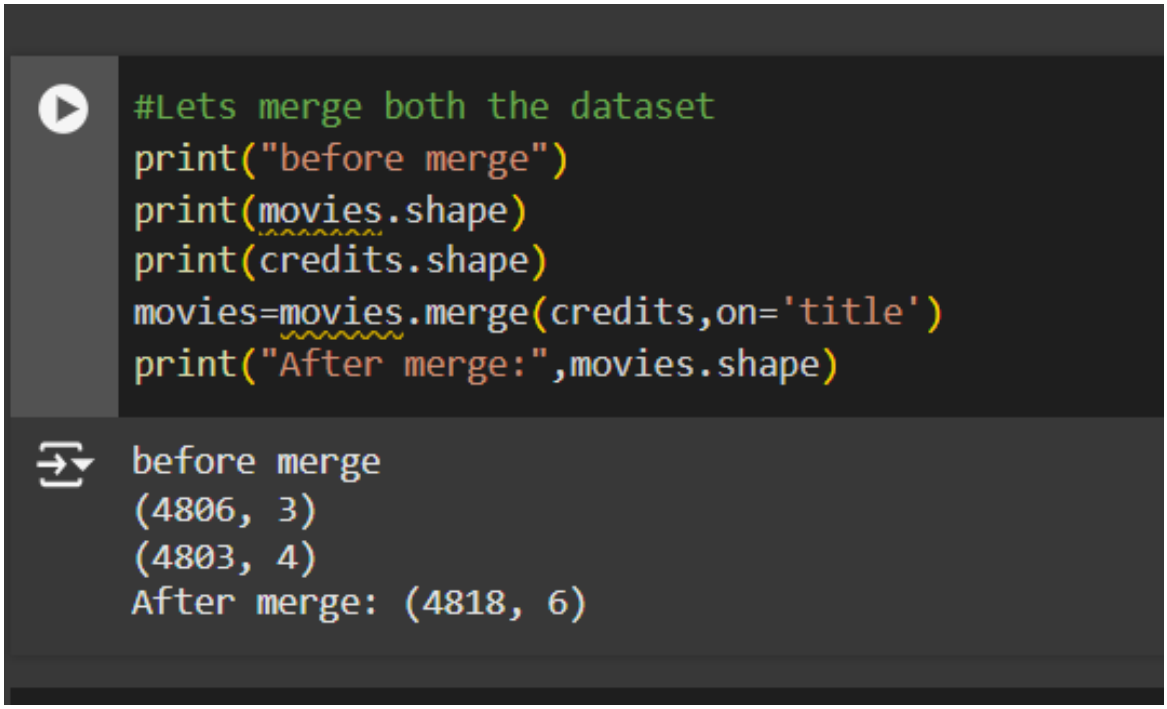


Рисунок 3.5. – Об’єднання 2 датасетів

Джерело: розроблено автором за даними [52]

Перед об'єднанням:

Movies.shape = (4806, 3): Це означає, що датафрейм movies містить 4806 рядків і 3 стовпці. Кожен рядок, ймовірно, представляє окремий фільм, а стовпці містять дані, як-от назва фільму, рік випуску та, можливо, іншу інформацію.

credits.shape = (4803, 4): Датафрейм credits має 4803 рядків і 4 стовпці. Кожен рядок, ймовірно, містить інформацію про кредити для фільму (актори, режисери тощо).

Після об'єднання:

After merge: (4818, 6): Після мержу датафрейм movies збільшився до 4818 рядків і 6 стовпців. Це означає, що:

Додано кілька нових фільмів (кількість рядків збільшилася на 12 — з 4806 до 4818). Це може вказувати на те, що деякі фільми з credits не мали відповідності в movies і були включені в результаті об’єднання.

Додано 3 нові стовпці (з 3 до 6) з інформацією з credits (наприклад, список акторів, режисерів, чи інші дані).

Далі виконуємо перевірку наявності відсутніх або пустих значень у DataFrame, використовуючи бібліотеку pandas (Python). І видаляємо пусті значення (див.рис 3.6) .

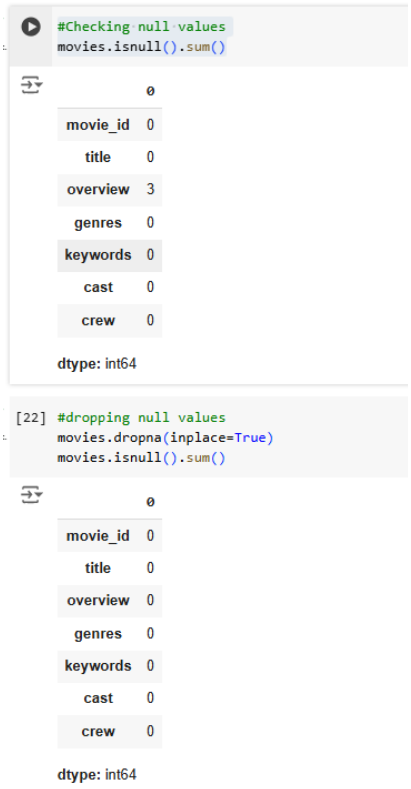


Рисунок 3.6. – Видалення пустих та нульових значень

Джерело: розроблено автором за даними [52]

Виконанаємо перетворення стовпця genres у DataFrame, це необхідно для того, щоб зручніше працювати з даними, що містять інформацію про жанри фільмів.

Далі продовжуємо коригування даних для побудови більш точної системи рекомендаційних систем. Для цього потрібно виконати такі кроки:

Витягування жанрів:

Функція convert застосовується до стовпця genres в DataFrame movies. Вона перетворює рядкове подання списку словників на список жанрів (вибираються лише значення за ключем 'name') [44].

Витягування ключових слів:

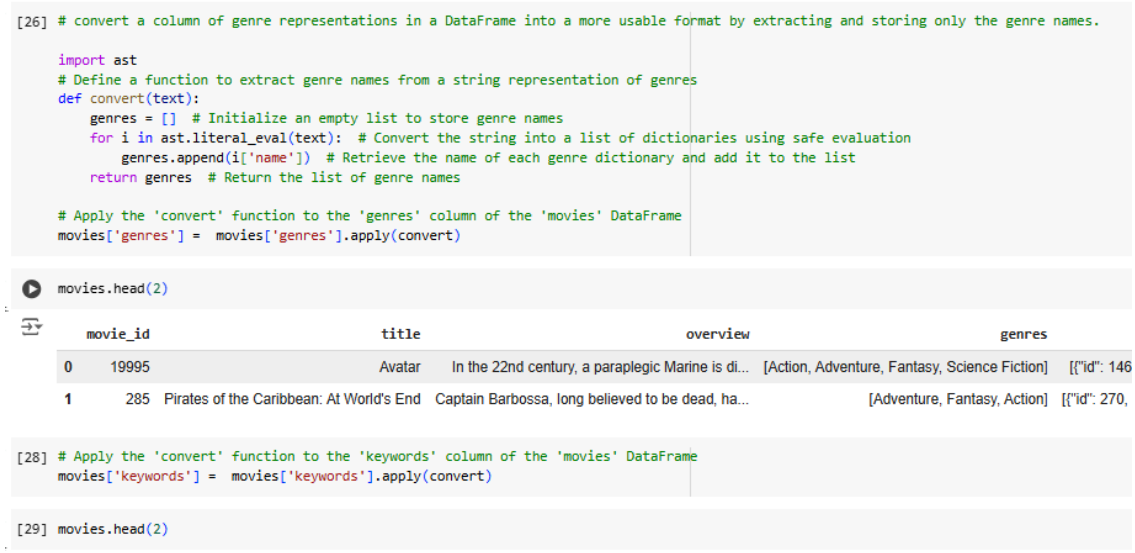
Аналогічно, функція convert застосовується до стовпця keywords, перетворюючи його в список ключових слів.

Витягування акторів:

Функція convert_actors вибирає імена перших трьох акторів з рядкового подання списку акторів (потрібні лише перші три імена).

Витягування імені режисера:

Функція fetch_director перебирає список членів знімальної групи з стовпця crew і вибирає лише ті елементи, де значення ключа 'job' дорівнює 'Director', зберігаючи їхні імена (див.рис 3.7).



```
[26] # convert a column of genre representations in a DataFrame into a more usable format by extracting and storing only the genre names.

     import ast
     # Define a function to extract genre names from a string representation of genres
     def convert(text):
         genres = []  # Initialize an empty list to store genre names
         for i in ast.literal_eval(text):  # Convert the string into a list of dictionaries using safe evaluation
             genres.append(i['name'])  # Retrieve the name of each genre dictionary and add it to the list
         return genres  # Return the list of genre names

     # Apply the 'convert' function to the 'genres' column of the 'movies' DataFrame
     movies['genres'] =  movies['genres'].apply(convert)
```

```
movies.head(2)
```

	movie_id	title	overview	genres	
0	19995	Avatar	In the 22nd century, a paraplegic Marine is di...	[Action, Adventure, Fantasy, Science Fiction]	[{"id": 146
1	285	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	[Adventure, Fantasy, Action]	[{"id": 270,

```
[28] # Apply the 'convert' function to the 'keywords' column of the 'movies' DataFrame
     movies['keywords'] =  movies['keywords'].apply(convert)
```

```
[29] movies.head(2)
```

Рисунок 3.7. – Коригування даних 1 частина

Джерело: розроблено автором за даними [52]

Застосуємо функцію fetch_director, яка призначена для вилучення імені режисера з колонки crew в DataFrame movies. Застосовуємо дану функцію до відповідних стовпців (genres, keywords, cast, crew) для отримання більш зручних для роботи списків жанрів, ключових слів, акторів та режисерів (див.рис 3.8).

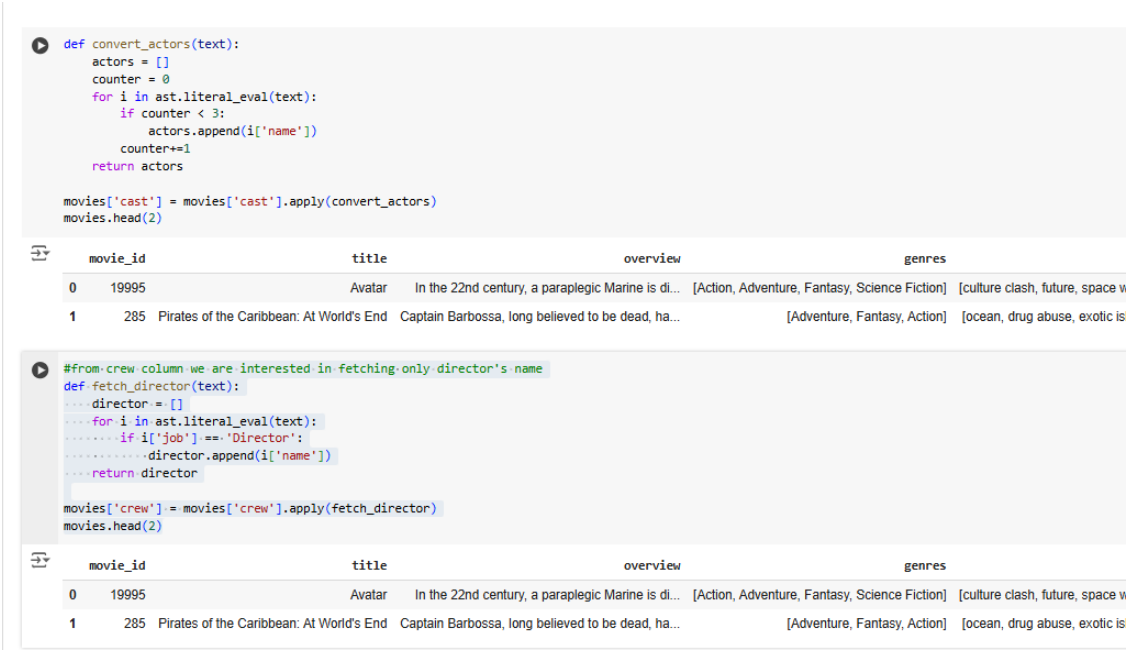
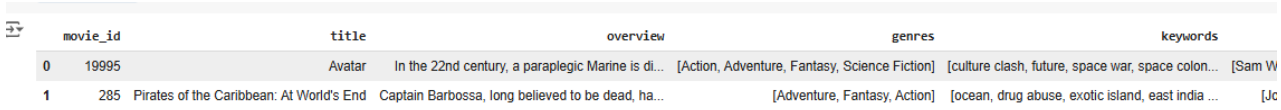


Рисунок 3.8. – Коригування даних 2 частина

Джерело: розроблено автором за даними [52]

Перевірка результатів:

За допомогою movies.head(2) виводяться перші два рядки DataFrame для перевірки результату перетворень (див.рис 3.9).



	movie_id	title	overview	genres	keywords	
0	19995	Avatar	In the 22nd century, a paraplegic Marine is di...	[Action, Adventure, Fantasy, Science Fiction]	[culture clash, future, space war, space colon...	[Sam W
1	285	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	[Adventure, Fantasy, Action]	[ocean, drug abuse, exotic island, east india ...	[Jc

Рисунок 3.9. – Вигляд датасету після коригуваннь

Джерело: розроблено автором за даними [52]

Наступний крок конвертація стовпця overview у список:

У стовпці overview, який містить текст (рядок), застосовується метод split() [45]. Це розділяє кожен рядок на список слів, використовуючи пробіли як роздільник. Після цього стовпець overview перетворюється на список слів.

Видалення пробілів у словах для інших стовпців:

Для стовпців cast, crew, genres та keywords, де зберігаються списки слів, застосовується лямбда-функція, яка перебирає кожен елемент списку і видаляє всі пробіли з кожного слова за допомогою replace(" ", "").

Таким чином, усі пробіли в середині слів видаляються (див.рис 3.10).

```
[33] #Need to convert overview column into list for concatenation as it is in string and list+string concatenation not possibl
     movies['overview'] = movies['overview'].apply(lambda x:x.split())

     #Remove spaces between words in specific columns containing lists of words
     movies['cast'] = movies['cast'].apply(lambda x: [word.replace(" ", "") for word in x])
     movies['crew'] = movies['crew'].apply(lambda x: [word.replace(" ", "") for word in x])
     movies['genres'] = movies['genres'].apply(lambda x: [word.replace(" ", "") for word in x])
     movies['keywords'] = movies['keywords'].apply(lambda x: [word.replace(" ", "") for word in x])

     movies.head(2)
```

	movie_id	title	overview	genres	keywords	cast	
0	19995	Avatar	[In, the, 22nd, century,, a, paraplegic, Marin...	[Action, Adventure, Fantasy, ScienceFiction]	[cultureclash, future, spacewar, spacecolony, ...	[SamWorthington, ZoeSaldana, SigourneyWeaver]	[Jam
1	285	Pirates of the Caribbean: At World's End	[Captain, Barbossa,, long, believed, to, be, d...	[Adventure, Fantasy, Action]	[ocean, drugabuse, exoticisland, eastindiatrad...	[JohnnyDepp, OrlandoBloom, KeiraKnightley]	[Go

Рисунок 3.10. – Видалення пробілів в датасеті

Джерело: розроблено автором за даними [52]

Після повної обробки даних можна почати обчислення косинусної подібності – це метрика, яка враховує кут між векторами запиту та документа.

Використовуючи функцію cosine_similarity, обчислюємо матрицю подібності між всіма фільмами на основі їхніх векторних представлень [46].

Косинусна подібність вимірює, наскільки схожі два вектори, порівнюючи косинус кута між ними. Вона повертає значення від 0 до 1, де:

1 означає, що вектори ідентичні (повна подібність).

0 означає, що вектори зовсім не схожі (повна відсутність подібності).

Кожен елемент матриці similarity представляє собою подібність між парою фільмів.

3. Приклад використання:

Після обчислення матриці косинусної подібності, можна виконати кілька дій:

Пошук найбільш подібних фільмів до конкретного:

вибрати певний фільм, подивитися на рядок відповідної матриці і знайти інші фільми, які мають високу подібність [47].

Рекомендація фільмів:

На основі подібності можна порекомендувати фільми, які найбільш схожі на вибраний (див.рис 3.11).

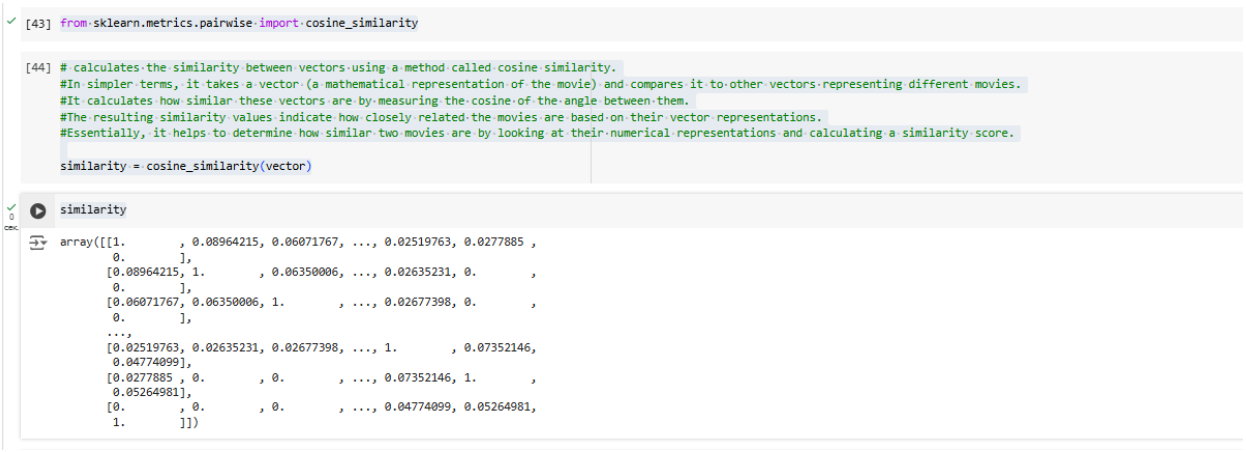


Рисунок 3.11. – Косинусова подібність

Джерело: розроблено автором за даними [52]

Проситими словами функція **recommend** допомагає знайти фільми, які найбільше схожі на той, який ви ввели. Якщо , наприклад, введете "John Carter", вона знайде п'ять інших фільмів, які найбільше нагадують цей, за певними характеристиками [48].

Кроки роботи:

1. **Знаходимо фільм**: Функція шукає фільм у списку фільмів (тобто в базі даних, яка називається movies). Вона знаходить фільм за його назвою, яку ви ввели.
2. **Обчислюємо схожість**: Потім програма порівнює цей фільм з усіма іншими фільмами в списку, щоб визначити, наскільки вони схожі один на одного. Для цього використовуються спеціальні алгоритми, які можуть порівнювати фільми за різними критеріями (наприклад, за жанром, сюжетом, акторським складом).
3. **Сортуємо фільми за схожістю**: Функція сортує всі фільми за ступенем схожості, щоб найсхожіші фільми були на початку списку [49].

4. **Виводимо п'ять найбільш схожих фільмів (див.рис 3.12)**: В результаті функція виведе назви п’яти фільмів, які найбільше схожі на введений вами фільм, за умови, що це не буде сам фільм, який ви ввели [50].

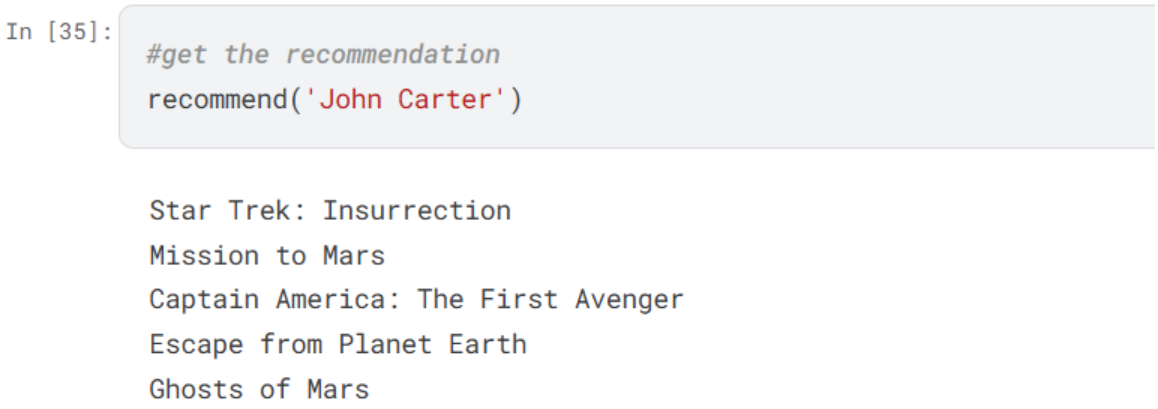


Рисунок 3.12. – Результати дослідження

Джерело: розроблено автором за даними [52]

3.3. Аналіз результатів та порівняння з існуючими рішеннями

Порівнюючи результати дослідження з уже існуючими рекомендаційними системами на платформі Google, можна побачити, що рекомендації, такі як:

Mission to Mars

Captain America: The First Avenger

Escape from Planet Earth

Ghosts of Mars

Є схожими по категорії фільмів, як у сфері наукової фантастики та пригодницьких фільмів, оскільки вони включають в себе елементи міжпланетних подорожей, боротьби між героями та космічні пригоди. Далі можна зазначити кілька важливих спільних рис та відмінностей цих фільмів у контексті рекомендацій:

Спільні риси:

Жанр: Наукова фантастика та пригоди

Усі ці фільми належать до жанрів наукової фантастики та пригод, що означає, що вони зосереджені на фантастичних елементах, таких як космос, інопланетні цивілізації, технології майбутнього та героїчні пригоди.

Інопланетні теми і космічні подорожі:

Фільми, такі як Mission to Mars та Ghosts of Mars, безпосередньо пов'язані з дослідженням інших планет, що є спільною темою в багатьох науково-фантастичних фільмах, у тому числі й у John Carter.

Героїзм та боротьба з ворогами:

У таких фільмах, як Captain America: The First Avenger та Escape from Planet Earth, головні герої борються з численними ворогами або надзвичайними ситуаціями, що є загальною рисою багатьох пригодницьких фільмів.

Технічний прогрес і фантастичні технології:

У фільмах присутні елементи, що відображають високий рівень технологій, інопланетні технології чи бойові пристрої майбутнього, що часто зустрічається в науково-фантастичних стрічках.

Після порівняння можна зробити висновок що, функція recommend працює досить ефективно для пошуку фільмів, подібних до того, який був введений користувачем. Ось детальний аналіз її результатів та порівняння з іншими існуючими методами рекомендацій фільмів:

1. Алгоритм на основі схожості:

Функція використовує метод подібності, щоб знайти найбільш схожі фільми в базі даних. Це може бути корисним, коли користувач хоче отримати фільми, що схожі за сюжетом, жанром чи іншими характеристиками.

Позитивний аспект: Подібні алгоритми дозволяють отримати результати, які базуються на вивченні великої кількості фільмів, їхніх характеристик і взаємозв'язків, що створює ефективні рекомендації для користувачів.

2. Метод сортування за подібністю:

Після обчислення подібності, фільми сортуються за рівнем схожості, що дозволяє отримати п'ять найбільш відповідних фільмів (див.рис 3.13).

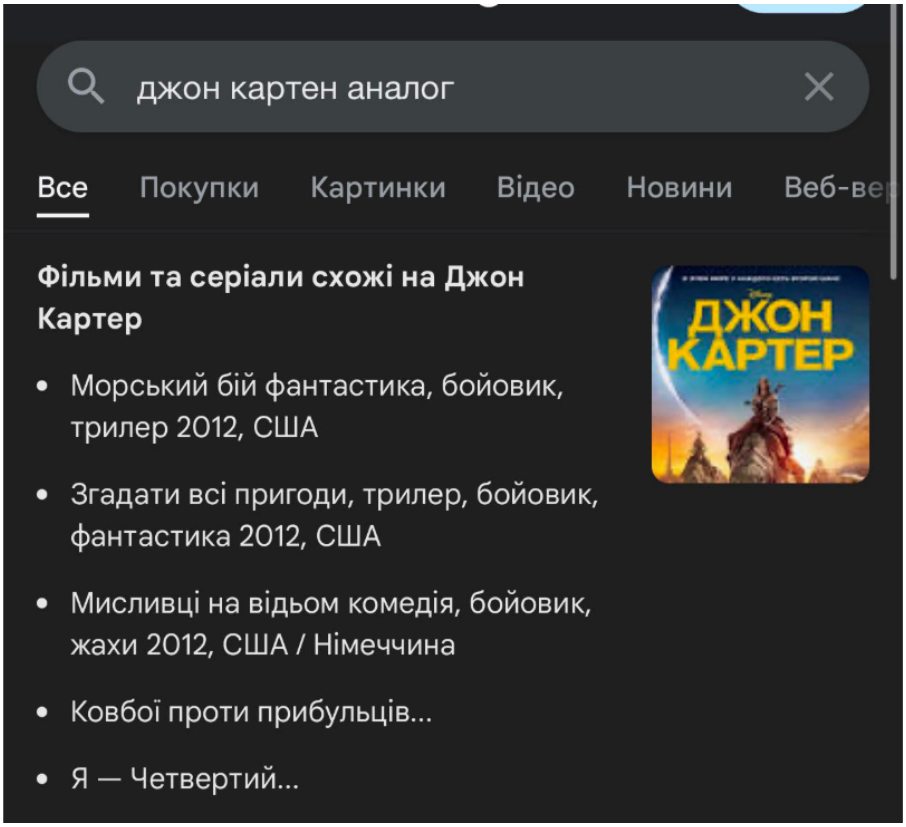


Рисунок 3.13 Рекомендації Google [56]

Позитивний аспект: Це дозволяє користувачеві швидко знайти найбільш релевантні фільми.

3. Виключення введеного фільму:

Програма виключає сам фільм, який було введено, з результатів, що є важливим для того, щоб уникнути ситуації, коли користувач побачить сам фільм в результатах пошуку.

Позитивний аспект: Це дозволяє фокусуватися на нових фільмах, схожих на обраний.

Підводячи підсумки рекомендаційної системи провести її оцінку можна за кількома ключовими аспектами, які розкривають його переваги, недоліки та можливості застосування в різних сценаріях. Цей алгоритм базується на матриці подібності, що дозволяє ефективно підбирати фільми, схожі за жанром, сюжетом чи іншими характеристиками, але має певні обмеження, які впливають на його універсальність.

По-перше, варто відзначити точність роботи алгоритму, яка проявляється в здатності правильно визначати фільми, схожі за ключовими параметрами. Це

особливо важливо для користувачів, які шукають фільми із подібною тематикою чи жанровою спрямованістю. Наприклад, якщо ввести фільм у жанрі наукової фантастики з елементами космічних подорожей, система рекомендує подібні стрічки, що відповідають заданим критеріям, такі як *Mission to Mars* чи *Ghosts of Mars*. Таким чином, алгоритм демонструє високу релевантність результатів. Водночас існують обмеження, які можуть знижувати точність у більш складних випадках. Алгоритм не враховує специфічні вподобання користувачів, такі як улюблені актори, режисери чи особливі аспекти сюжету. Наприклад, користувач, якому важливі емоційні аспекти фільму, може не отримати рекомендацію, яка враховує цей фактор. Крім того, алгоритм менш ефективний для нових чи маловідомих фільмів, оскільки вони можуть не мати достатньо метаданих у базі даних для коректного обчислення подібності.

По-друге, ефективність роботи алгоритму проявляється у швидкості обчислень і зручності для користувача. Завдяки використанню матриці подібності, результати генеруються швидко, що особливо важливо для користувачів, які цінують оперативність. Сортування рекомендованих фільмів за ступенем подібності полегшує навігацію і дозволяє зосередитися на найбільш релевантних варіантах. Наприклад, після введення фільму *John Carter*, система рекомендує топ-5 фільмів, які найбільше відповідають введеному запиту. Однак якість рекомендацій напряму залежить від розміру та наповнення бази даних. Якщо база обмежена або нерівномірно представлена, це може вплинути на точність і різноманітність результатів. Наприклад, якщо у базі переважають фільми певного жанру, це може знизити різноманітність рекомендацій для інших жанрів.

По-третє, алгоритм забезпечує високий рівень зручності використання. Простота функції робить її інтуїтивно зрозумілою навіть для нових користувачів. Важливою перевагою є автоматичне виключення введеного фільму зі списку результатів, що підвищує корисність рекомендацій і запобігає повторенню контенту. Наприклад, якщо користувач шукає фільми, схожі на *Captain America: The First Avenger*, цей фільм не з’явиться у результатах, дозволяючи зосередитися на нових рекомендаціях. Водночас відсутність можливості налаштовувати пошук

за додатковими параметрами, такими як актори, рік випуску чи тривалість, обмежує можливості користувача у деталізації запиту.

Ефективність алгоритму у різних сценаріях також варто розглянути окремо. Алгоритм чудово працює для популярних фільмів із багатим контекстом і метаданими, адже такі стрічки легко піддаються класифікації та порівнянню. Наприклад, система добре рекомендує схожі фільми для блокбастерів чи широко обговорюваних кінострічок. Однак для нішевих жанрів чи маловідомих фільмів рекомендації можуть бути менш точними через недостатність даних у базі. Це створює обмеження для користувачів із унікальними смаками або тих, хто шукає менш популярний контент.

Для глибшого аналізу ефективності алгоритму можна використовувати спеціальні метрики. Точність (precision) відображає частку рекомендованих фільмів, які дійсно цікаві для користувача. Повнота (recall) визначає, наскільки велика частина релевантних фільмів була включена до рекомендацій. Баланс між точністю і повнотою можна оцінити за допомогою F1-міри. Середня точність (MAP) дозволяє оцінити ранжування результатів, а метрика різноманітності (diversity) визначає, чи охоплює алгоритм різні типи фільмів, а не лише вузько тематичні рекомендації.

Загалом, функція **recommend** демонструє високу ефективність у вузькому колі завдань, таких як пошук фільмів, схожих за жанром чи сюжетом. Вона є простою у використанні та забезпечує швидке отримання результатів. Однак її обмеження, зокрема відсутність персоналізації, врахування контексту користувача та можливість деталізації запиту, знижують точність рекомендацій у складніших випадках. Для покращення роботи алгоритму можна інтегрувати гібридні методи, які поєднують контент-орієнтований підхід із колаборативною фільтрацією, забезпечуючи врахування вподобань користувачів, історії переглядів і рекомендацій інших користувачів із подібними інтересами. Це дозволить значно підвищити точність і універсальність рекомендаційної системи.

ВИСНОВКИ

У рамках проведеного дослідження було детально розглянуто ключові підходи та алгоритми, які застосовуються в рекомендаційних системах для стримінгових платформ та акцентовано увагу на важливості розробки персоналізованих систем рекомендацій для суттєвого покращення користувацького досвіду. Така персоналізація не лише дозволяє оптимізувати досвід користувача, але й сприяє підвищенню рівня задоволеності та лояльності до платформи. Одним із важливих напрямків дослідження стало вивчення сучасних методів персоналізації, зокрема таких, як колаборативне фільтрування, контентне фільтрування та гібридні моделі, які поєднують переваги кількох підходів одночасно.

Для досягнення мети кваліфікаційної роботи було проведено збір та аналіз даних, що включав кілька етапів обробки, очищення та вивчення різноманітних джерел інформації. Основним завданням цього етапу було виявлення ключових закономірностей, структур та характеристик даних, що дозволило створити міцну основу для подальшої розробки рекомендаційної системи. Під час аналізу було застосовано техніки препроцисингу та візуалізації даних, які дозволили не лише виявити важливі взаємозв'язки і патерни, але й зробити висновки щодо того, як ці закономірності можуть бути використані для покращення роботи системи.

Особлива увага була приділена розробці простого, але ефективного алгоритму для побудови рекомендаційної системи. Для цього використовувалися класичні методи фільтрації за подібністю та популярністю. Завдяки цьому алгоритму, система здатна пропонувати користувачам релевантні варіанти на основі їхніх вподобань, попередньої історії взаємодії з платформою, а також на підставі даних, отриманих від інших користувачів з подібними інтересами. Це дозволяє значно покращити персоналізацію досвіду користувачів та підвищити рівень їх задоволеності. Користувачі отримують більш точні та відповідні

пропозиції, що забезпечує їм комфортний та інтуїтивно зрозумілий досвід взаємодії з системою.

Процес розробки включав також оцінку ефективності рекомендаційної системи, що передбачало перевірку різних підходів та оптимізацію їхньої роботи на основі отриманих результатів. В рамках цієї роботи був реалізований прототип, що здатен адаптуватися до різних умов використання, в тому числі для застосування на платформах з великою кількістю користувачів. Це забезпечує можливість масштабування системи в майбутньому, що робить її актуальною не лише для невеликих проектів, але й для більших бізнес-рішень.

Розроблений механізм рекомендаційної системи має великий потенціал для подальшого удосконалення та розширення. Він може бути застосований у різних сферах, зокрема в електронній комерції, на контентних платформах, у соціальних мережах, а також у різноманітних бізнес-програмах для покращення взаємодії з користувачами. Завдяки гнучкості і адаптивності розробленої системи, вона може стати ефективним інструментом для оптимізації процесу прийняття рішень та покращення досвіду користувачів у різних галузях.

Таким чином, виконана робота дозволила досягти значних результатів у створенні простого, але ефективного механізму рекомендаційних систем. Це дає можливість зробити подальші кроки в розвитку технологій, пов'язаних з персоналізацією та покращенням взаємодії між користувачем і платформою. Розроблений прототип є надійною основою для майбутніх досліджень та може бути використаний для практичних цілей у різних сферах, що забезпечить значні покращення в організації та управлінні інформацією.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Y. Koren, R. Bell, C. Volinsky. Matrix Factorization Techniques for Recommender Systems. 2009. – №42. DOI: 10.1109/MC.2009.263 (дата звернення:01.11.2024).

2. P. Resnick and H. Varian, “Recommender Systems,” Communications of the ACM, Vol. 40, No. 3, 1997, pp. 56-58. DOI: http://dx.doi.org/10.1145/245108.245121 (дата звернення: 02.11.2024).

3. J. Bennett і S. Lanning. The Netflix Prize. 2009. URL: https://www.bibsonomy.org/bibtex/257ef9d0119acf19856b408297e5a2e5f/jaeschke (дата звернення: 02.11.2024).

4. X. Liu та X. Zhang. Deep Learning Based Recommender System: A Survey and New Perspectives. 2017. DOI:10.1145/3285029 (дата звернення: 02.11.2024).

5. A. Alslaity. Towards a Comprehensive Evaluation of Recommenders: A Cognition-Based Approach. 2018. DOI:10.1007/978-3-319-89656-4_32 (дата звернення: 02.11.2024).

6. What Is a Recommendation System? 2024 URL: https://www.nvidia.com/en-us/glossary/recommendation-system/ .(дата звернення:05.11.2024).

7. H. Al-bashiri, M. A. Abdulgabber Abdulhak, A. Romli, F. Hujainah. Collaborative Filtering Recommender System: Overview and Challenges. 2017. DOI: 10.1166/asl.2017.10020 (дата звернення: 05.11.2024).

8. What Is Content-Based Filtering? Benefits and Examples in 2024? 2024 URL: https://www.upwork.com/resources/what-is-content-based-filtering. (дата звернення:08.11.2024).

9. User-Based Collaborative Filtering. 2023. URL: https://www.geeksforgeeks.org/user-based-collaborative-filtering/. (дата звернення:08.11.2024)

10. I. Coppens, T. De Pessemier, L. Martens. Exploring the added effect of three recommender system techniques in mobile health interventions for physical activity: a

longitudinal randomized controlled trial.2014. DOI:10.1007/s11257-024-09407-z. (дата звернення:08.11.2024)

11. Recommender Systems: Behind the Scenes of Machine Learning-Based Personalization. 2021. URL: https://www.altexsoft.com/blog/recommender-system-personalization/ (дата звернення:10.11.2024).

12. G. Andersen. Recommender Systems in Data Science: Personalized Recommendations and Algorithms 2024. URL: https://moldstud.com/articles/p-recommender-systems-in-data-science-personalized-recommendations-and-algorithms (дата звернення:10.11.2024).

13. G. Uchyigit and M. Y. Ma. Personalization Techniques and Recommender Systems. 2008p. 336 DOI: doi.org/10.1142/6788 (дата звернення:15.11.2024).

14. Flitto DataLab. AI Personalization: Techniques and Applications. 2024. URL: https://www.dacast.com/blog/video-streaming-technology/ (дата звернення:16.11.2024).

15. G. Uchyigit, M. Y. Ma. The Definitive Guide to Video Streaming Technology in 2024. 2024. URL: https://ied.eu/blog/technology-blog/what-are-the-factors-influencing-the-crypto-market-today/ (дата звернення:16.11.2024)

16. V. Mysholovskyi and Y. Panasenko. Instant analysis of time-critical data: Real-time data streaming use cases, examples, and benefits. 2022. URL: https://yalantis.com/blog/real-time-data-streaming-use-cases/ (дата звернення:18.11.2024).

17. Memgraph. Top 10 Streaming Analytics Tools. 2023. URL: https://memgraph.com/blog/streaming-analytics-tools (дата звернення:18.11.2024).

18. J. Richman. 15 Best Data Streaming Technologies & Tools For 2023. 2023. URL: https://estuary.dev/data-streaming-technologies/ (дата звернення:19.11.2024)

19. C. Papagianni, N. D. Tselikas, E. Kosmatos, A. Papadakis. A complete content production and delivery system in a controlled multimedia network 2008. DOI:10.1109/ISCC.2008.4625773 (дата звернення: 19.11.2024)

20. I. Md Siddique. Digital Satellite Technology: Innovations and Applications in the Modern Era. 2024. DOI:10.5281/zenodo.12748154 (дата звернення:21.04.2023).

21. Developing a personalized recommendation system in a smart product service system based on unsupervised learning model / M.-C. Chiu та ін. Computers in Industry. 2021. Т. 128. С. 103421. URL: https://doi.org/10.1016/j.compind.2021.103421 (дата звернення: 26.11.2024).

22. Personalized Recommendation Systems (PRES): A Comprehensive Study and Research Issues. International Journal of Modern Education and Computer Science. 2018. Т. 10, № 10. С. 11–21. URL: https://doi.org/10.5815/ijmecs.2018.10.02 (дата звернення: 26.11.2024).

23. Intelligent classification and personalized recommendation of E-commerce products based on machine learning / K. Xu та ін. Applied and Computational Engineering. 2024. Т. 64, № 1. С. 143–149. URL: https://doi.org/10.54254/2755-2721/64/20241365 (дата звернення: 26.11.2024).

24. Zhao W. Enhancing user engagement and satisfaction through personalized news recommendation systems. Applied and Computational Engineering. 2024. Т. 69, № 1. С. 13–18. URL: https://doi.org/10.54254/2755-2721/69/20241454 (дата звернення: 26.11.2024).

25. Yan X., Qi S., Chen C. Recommender Systems: Collaborative Filtering and Content-based Recommender System. Applied and Computational Engineering. 2023. Vol. 2, no. 1. P. 346–351. URL: https://doi.org/10.54254/2755-2721/2/20220658 (date of access: 26.11.2024).

26. Bouza A. Hypothesis-Based Collaborative Filtering. Lulu Press, Inc., 2012.

27. Recommender system using item based collaborative filtering (CF) and K-means / M. Garanayak та ін. International Journal of Knowledge-based and Intelligent Engineering Systems. 2019. Т. 23, № 2. С. 93–101. URL: https://doi.org/10.3233/kes-190402 (дата звернення: 26.11.2024).

28. Luo S. Research on recommended model of personalized interior environment. Infrastructure Asset Management. 2024. С. 1–32. URL: https://doi.org/10.1680/jinam.23.00055 (дата звернення: 26.11.2024).

29. Perzonalization In Streaming Services With Middleware | MwareTV. MwareTV. URL: https://www.mwaretv.com/newsroom/personalization-in-streaming-services/ (дата звернення: 26.11.2024).

30. Sharma D. D., Aggarwal D. D., Saxena D. A. B. Content Based Recommendation System on Netflix Data. Feb-Mar 2024. 2024. № 42. С. 19–26. URL: https://doi.org/10.55529/ijrise.42.19.26 (дата звернення: 26.11.2024).

31. Recommendation System: Techniques and Issues. International Journal of Recent Technology and Engineering. 2019. Т. 8, № 3. С. 2821–2824. URL: https://doi.org/10.35940/ijrte.c5211.098319 (дата звернення: 26.11.2024).

32. Mylavarapu B. K. Collaborative Filtering and Artificial Neural Network Based Recommendation System for Advanced Applications. Journal of Computer and Communications. 2018. Т. 06, № 12. С. 1–14. URL: https://doi.org/10.4236/jcc.2018.612001 (дата звернення: 26.11.2024).

33. B S. Movie Lens – Movie Recommendation System Using Deep Learning. Interantional journal of scientific research in engineering and management. 2024. Т. 08, № 05. С. 1–5. URL: https://doi.org/10.55041/ijsrem33379 (дата звернення: 26.11.2024).

34. Tilak Garg, Sahil Shekhar, Prof. Renu Narwal. An Evaluation of Machine Learning Algorithms Used for Recommender Systems in Streaming Services. International Journal of Advanced Research in Science, Communication and Technology. 2024. С. 340–346. URL: https://doi.org/10.48175/ijarsct-17652 (дата звернення: 26.11.2024).

35. Mei T. Recommender system design. Medium. URL: https://ted-mei.medium.com/recommender-system-design-5986922c9cc2 (дата звернення: 26.11.2024).

36. Serbia L. How to implement the recommendation system. Medium. URL: https://medium.com/levi-niners-crafts/how-to-implement-the-recommendation-system-e06d9e11f0ad (дата звернення: 26.11.2024).

37. A Web-based personalized recommendation system for mobile phone selection: Design, implementation, and evaluation / D.-N. Chen та ін. Expert Systems with Applications. 2010. Т. 37, № 12. С. 8201–8210. URL: https://doi.org/10.1016/j.eswa.2010.05.066 (дата звернення: 26.11.2024).

38. Zeng Q. W., Jiang J. Research and Implementation on Personalized Search Engine. Key Engineering Materials. 2011. Т. 467-469. С. 129–133. URL: https://doi.org/10.4028/www.scientific.net/kem.467-469.129 (дата звернення: 26.11.2024).

39. How to create personalized product recommendations for your website. Abmatic AI | Transforming Account-Based Marketing. URL: https://abmatic.ai/blog/how-to-create-personalized-product-recommendations-for-website (date of access: 26.11.2024).

40. Design and Implementation of a Personalized Text-Based Recommendation System / A. Aditya та ін. International Journal for Research in Applied Science and Engineering Technology. 2024. Т. 12, № 10. С. 1486–1496. URL: https://doi.org/10.22214/ijraset.2024.64907 (дата звернення: 26.11.2024).

41. Hernández del Olmo F., Gaudioso E. Evaluation of recommender systems: A new approach. Expert Systems with Applications. 2008. Т. 35, № 3. С. 790–804. URL: https://doi.org/10.1016/j.eswa.2007.07.047 (дата звернення: 30.11.2024).

42. Lonappan J., Aithal P. S., Jacob M. E-Professionalism as a Professional Identity in the Digital Era of Medical Education. International Journal of Health Sciences and Pharmacy. 2023. С. 35–48. URL: https://doi.org/10.47992/10.5281/zenodo.8329407 (дата звернення: 30.11.2024).

43. Saputra S. B., Pamungkas E. W. Development of scheduling system with genetic algorithm in website-based smk negeri 1 sine. Jurnal Teknik Informatika (Jutif). 2023. Т. 4, № 4. С. 797–806. URL: https://doi.org/10.52436/1.jutif.2023.4.4.784 (дата звернення: 30.11.2024).

44. R.A. D. B. Development of Movie Recommendation System Using Machine Learning. Interantional journal of scientific research in engineering and management. 2024. Т. 08, № 04. С. 1–5. URL: https://doi.org/10.55041/ijsrem29879 (дата звернення: 30.11.2024).

45. Verma S., Harit S., Munjal K. SRRS: Design and Development of a Scholarly Reciprocal Recommendation System. Scientometrics. 2024. URL: https://doi.org/10.1007/s11192-024-05143-8 (дата звернення: 30.11.2024).

46. Vroomans R. Review: “Computational evolution of neural and morphological development”, Yaochu Jin, ISBN 978-981-99-1853-9, Springer, 2023. Genetic Programming and Evolvable Machines. 2024. Т. 26, № 1. URL: https://doi.org/10.1007/s10710-024-09499-x (дата звернення: 30.11.2024).

47. How to Build a Recommendation System: Explained Step by Step - Stratoflow. *Stratoflow*. URL: https://stratoflow.com/how-to-build-recommendation-system/ (дата звернення: 30.11.2024).

48. Le J. Recommendation System Series Part 1: An Executive Guide to Building Recommendation System. Medium. URL: https://towardsdatascience.com/recommendation-system-series-part-1-an-executive-guide-to-building-recommendation-system-608f83e2630a (дата звернення: 30.11.2024).

49. ritesh ratti, Ph.D. Evolution of Recommender Systems. Medium. URL: https://medium.com/@ritesh.ratti/evolution-of-recommender-systems-fb7ab34bdcbe (дата звернення: 28.11.2024).

50. Akinbohun F. Development of Models by Energy Expended and Age Classifications for Diet Recommendation System. European Journal of Computer Science and Information Technology. 2024. Т. 12, № 6. С. 24–34. URL: https://doi.org/10.37745/ejcsit.2013/vol12n62434 (дата звернення: 28.11.2024).

51. Ghadami A., Tran T. TriDeepRec: a hybrid deep learning approach to content- and behavior-based recommendation systems. User Modeling and User-Adapted Interaction. 2024. URL: https://doi.org/10.1007/s11257-024-09418-w (дата звернення: 28.11.2024).

52. TMDB 5000 Movie Dataset. URL: https://www.kaggle.com/datasets/tmdb/tmdb-movie-metadata. (дата звернення: 28.11.2024).

53. Про систему рекомендацій YouTube. Офіційний Блог - Google Україна. URL: https://ukraine.googleblog.com/2021/09/youtube.html (дата звернення: 28.11.2024).

54. URL: https://rozetka.com.ua/ua/sony-playstation-5-slim-digital-edition/p410219112/ (дата звернення: 28.11.2024).

55. Джон Картер: між двох світів дивитися онлайн українською. UAKino - дивитися фільми та серіали онлайн в HD якості. URL: https://uakino.me/filmy/genre-action/548-dzhon-karter-mzh-dvoh-svtv.html (дата звернення: 28.11.2024).

56. Bevor Sie zur Google Suche weitergehen. Google. URL: https://www.google.com/search (дата звернення: 28.11.2024).