ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Оптимізація алгоритмів персоналізованих рекомендацій у стрімінгових сервісах на основі гібридних моделей машинного навчання»

на здобуття освітнього ступеня магістра
зі спеціальності 121 Інженерія програмного забезпечення
освітньо-професійної програми «Інженерія програмного забезпечення»

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Артем СТЕПАНЧЕНКО
(підпис)

Виконав: здобувач вищої освіти групи ПДМ-61
Артем СТЕПАНЧЕНКО

Керівник: Наталія ТРІНТІНА
канд. техн. наук.,доц.

Рецензент: _____

Київ 2026

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти Магістр

Спеціальність 121 Інженерія програмного забезпечення

Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
_____ Ірина ЗАМРІЙ
«_____» _____ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Степанченко Артему Олеговичу

1. Тема кваліфікаційної роботи: «Оптимізація алгоритмів персоналізованих рекомендацій у стрімінгових сервісах на основі гібридних моделей машинного навчання»

керівник кваліфікаційної роботи Наталія ТРІНТІНА, канд. техн. наук, доц.,

затверджені наказом Державного університету інформаційно-комунікаційних технологій від «30» жовтня 2025 р. № 467.

2. Строк подання кваліфікаційної роботи «19» грудня 2025 р.

3. Вихідні дані до кваліфікаційної роботи: науково-технічна література з рекомендаційних систем та стрімінгових сервісів, відкриті набори даних для задач рекомендацій, параметри роботи стрімінгових платформ, вимоги до якості рекомендацій, характеристика користувацької активності.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1.Аналіз предметної галузі персоналізованих рекомендацій у стрімінгових сервісах.

2. Дослідження методів колаборативної, контентної та гібридної фільтрації у рекомендаційних системах.
3. Розробка математичної моделі та методу динамічного зважування у гібридній рекомендаційній системі.
4. Програмна реалізація та дослідження запропонованого методу

5. Перелік ілюстративного матеріалу: *презентація*
   1. Порівняльна характеристика.
   2. Математична модель гібридних рекомендацій.
   3. Метод динамічного визначення ваг.
   4. Алгоритм роботи запропонованого методу.
   5. Діаграма діяльності.
   6. Практична реалізація та структура моделювання.
   7. Результати моделювання та порівняльний аналіз.

6. Дата видачі завдання «31» жовтня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз наявної науково-технічної літератури	31.10 - 05.11.2025	
2	Вивчення методів колаборативної та контентної фільтрації	06.11 - 09.11.2025	
3	Аналіз існуючих гібридних моделей рекомендаційних систем	14.11 - 17.11.2025	
4	Розробка математичної моделі гібридної рекомендаційної системи з динамічними вагами	18.11 - 25.11.2025	
5	Розробка алгоритму динамічного коригування вагових коефіцієнтів	26.11 - 01.12.2025	
6	Програмна реалізація методу та проведення експериментальних досліджень	02.12 - 12.12.2025	
7	Оформлення роботи: вступ, висновки, реферат	13.12 - 14.12.2025	
8	Розробка демонстраційних матеріалів	15.12 - 16.12.2025	
9	Попередній захист роботи	17.12 - 19.12.2025	

Здобувач вищої освіти _____ Артем СТЕПАНЧЕНКО
(підпис)

Керівник
кваліфікаційної роботи _____ Наталія ТРІНТІНА
(підпис)

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня магістра: 72 стор., 4 табл., 4 рис., 31 джерел.

Мета роботи – розробити та дослідити метод динамічного коригування вагових коефіцієнтів у гібридній рекомендаційній моделі для стрімінгових сервісів, спрямований на підвищення точності рекомендацій (зокрема, показників Accuracy та F1-Score) та покращення їх адаптивності, особливо в умовах проблеми «холодного старту».

Об'єкт дослідження – процес формування персоналізованих рекомендацій у стрімінгових платформах із застосуванням алгоритмів машинного навчання.

Предмет дослідження – моделі та алгоритми оптимізації процесу гібридизації рекомендаційних систем (зокрема, метод динамічного зважування) на основі факторів користувацької поведінки та сесійного контексту.

Короткий зміст роботи: у роботі розглянуто підходи до побудови персоналізованих рекомендацій у стрімінгових сервісах на основі контентно-орієнтованої, колаборативної та гібридної фільтрації. Запропоновано математичну модель гібридної рекомендаційної системи з динамічним визначенням ваг між базовими моделями. Реалізовано експериментальну модель на основі датасету MovieLens 25M та проведено порівняльну оцінку якості рекомендацій за метриками MAE, RMSE, Accuracy та F1-Score, що підтвердило ефективність запропонованого підходу.

КЛЮЧОВІ СЛОВА: ПЕРСОНАЛІЗОВАНІ РЕКОМЕНДАЦІЇ, СТРІМІНГОВІ СЕРВІСИ, ГІБРИДНІ МОДЕЛІ, МАШИННЕ НАВЧАННЯ, COLLABORATIVE FILTERING, CONTENT-BASED FILTERING, ДИНАМІЧНЕ ЗВАЖУВАННЯ.

ABSTRACT

Text part of the master's qualification work: 72 pages, 4 tables, 4 figures, 31 references.

Purpose of the work – to develop and investigate a method for dynamic adjustment of weighting coefficients in a hybrid recommendation model for streaming services, aimed at improving recommendation accuracy (in particular, the Accuracy and F1-Score metrics) and enhancing adaptability, especially under cold-start conditions.

Object of research – the process of forming personalized recommendations in streaming platforms using machine learning algorithms.

Subject of research – models and algorithms for optimizing the hybridization process of recommendation systems (in particular, the dynamic weighting method) based on user behavior factors and session context.

Summary of the work: the paper considers approaches to building personalized recommendations in streaming services based on content-oriented, collaborative, and hybrid filtering. A mathematical model of a hybrid recommendation system with dynamic weighting between basic models is proposed. An experimental model based on the MovieLens 25M dataset was implemented, and a comparative assessment of the quality of recommendations was carried out using MAE, RMSE, Accuracy, and F1-Score metrics, which confirmed the effectiveness of the proposed approach.

KEYWORDS: PERSONALIZED RECOMMENDATIONS, STREAMING SERVICES, HYBRID MODELS, MACHINE LEARNING, COLLABORATIVE FILTERING, CONTENT-BASED FILTERING, DYNAMIC WEIGHTING

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

CF – Collaborative Filtering, колаборативна фільтрація.

CBF – Content-Based Filtering, фільтрація на основі контенту.

MF – Matrix Factorization, матрична факторизація.

SVD – Singular Value Decomposition, сингулярний розклад.

ML – Machine Learning, машинне навчання.

MAE – Mean Absolute Error, середня абсолютна похибка.

RMSE – Root Mean Squared Error, корінь із середньоквадратичної похибки.

Accuracy – метрика класифікаційної точності.

F1-Score – гармонійне середнє між precision та recall.

XGBoost – eXtreme Gradient Boosting, бібліотека градієнтного бустингу.

MovieLens 25M – відкритий набір даних користувацьких оцінок фільмів.

Hybrid-XGB – запропонована динамічна гібридна модель на основі XGBoost.

ВСТУП

Обгрунтування вибору теми та її актуальність: Завдяки широкому спектру матеріалів, доступних споживачам, включаючи музику, фільми та аудіокниги, сучасний бізнес стрімінгових сервісів стрімко розвивається. Впровадження рішень, які дають змогу користувачам отримувати персоналізовані пропозиції, є необхідним через збільшення кількості даних, які вже доступні. Це покращує користувацький досвід, а також збільшує час, проведений у сервісі, утримує користувачів і підвищує загальну продуктивність бізнесу.

Об'єкт дослідження - процес формування персоналізованих рекомендацій у стрімінгових платформах із застосуванням алгоритмів машинного навчання.

Предмет дослідження - моделі та алгоритми оптимізації процесу гібридизації рекомендаційних систем (зокрема, метод динамічного зважування) на основі факторів користувацької поведінки та сесійного контексту.

Мета роботи - розробити та дослідити метод динамічного корегування вагових коефіцієнтів у гібридній рекомендаційній моделі для стрімінгових сервісів, спрямований на підвищення точності рекомендацій (зокрема, показників Accuracy та F1-Score) та покращення їх адаптивності, особливо в умовах проблеми «холодного старту».

Методи дослідження - аналіз і узагальнення наукових джерел з рекомендаційних систем, математичне моделювання гібридних алгоритмів, комп'ютерне моделювання та експериментальна перевірка на основі відкритих датасетів з використанням алгоритмів машинного навчання (CF, CBF, градієнтний бустинг, метамоделі)

Для реалізації поставленої мети потрібно вирішити наступні завдання:

1. Проаналізувати предметну область стрімінгових сервісів та роль персоналізованих рекомендацій у підвищенні якості користувацького досвіду.
2. Дослідити класичні підходи до побудови рекомендаційних систем (Content-Based Filtering, Collaborative Filtering, гібридні моделі), виявити їхні переваги та обмеження в контексті динамічних сценаріїв використання.
3. Проаналізувати існуючі гібридні рекомендаційні моделі, включно з моделями на основі глибинного навчання, та обґрунтувати необхідність динамічного визначення ваг між окремими компонентами (CF, CBF, ML).
4. Розробити математичну модель гібридних рекомендацій, що поєднує колаборативну та контентно-орієнтовану фільтрацію, та формально ввести метод динамічного визначення ваг між ними.
5. Розробити алгоритм роботи запропонованого методу, включно з логікою обчислення рекомендацій базових моделей, формуванням ознак для метамоделі та оновленням ваг у часі.
6. Реалізувати експериментальну програмну модель гібридної рекомендаційної системи з динамічним зважуванням на основі відкритого датасету (MovieLens 25M) із використанням сучасних бібліотек машинного навчання.
7. Провести експериментальні дослідження, порівняти якість базових моделей (CF, CBF), простого статичного гібриду та запропонованого динамічного методу за метриками MAE, RMSE, Accuracy, F1-Score.
8. Оцінити вплив динамічного зважування на точність, стійкість до проблеми «холодного старту» та здатність моделі адаптуватися до різних сценаріїв використання стрімінгового сервісу.

Практичне значення результатів: запропонований метод динамічного корегування вагових коефіцієнтів у гібридній рекомендаційній моделі дозволяє підвищити точність та адаптивність персоналізованих рекомендацій

у стрімінгових сервісах, зменшити вплив проблеми «холодного старту» та покращити утримання й залученість користувачів.

Для реалізації експериментальної частини роботи використовувалися - мова програмування Python, середовище Jupyter Notebook, бібліотеки pandas та NumPy для обробки даних, бібліотека Surprise для реалізації колаборативної фільтрації (SVD), бібліотека scikit-learn для розрахунку метрик якості, бібліотека XGBoost для побудови метамоделі динамічного зважування, а також відкритий датасет MovieLens 25M як вихідна база користувацьких оцінок.

Практична значущість результатів: отримані підходи та експериментальні результати можуть бути використані як основа для впровадження або модернізації модулів рекомендацій у реальних стрімінгових платформах та мобільних застосунках, де потрібна гнучка, масштабована та адаптивна гібридна рекомендаційна система.

Галузь використання - стрімінгові платформи, що працюють із медіа-контентом (музика, відео, подкасти тощо), рекомендаційні підсистеми мобільних і веб-застосунків, а також аналітичні системи, що потребують побудови персоналізованих пропозицій на основі поведінкових даних користувачів.

1. ТЕОРЕТИЧНА ЧАСТИНА

1.1 Аналіз предметної області

Використання стрімінгових платформ стало невід'ємною частиною повсякденного життя. Щодня мільйони користувачів у всьому світі звертаються до музичних, відеосервісів та сервісів подкастів для прослуховування музики, перегляду фільмів, серіалів, прямих трансляцій та іншого медіаконтенту. Обсяг доступних матеріалів постійно зростає, а саме: щодня з'являються тисячі нових пісень, відео та повнометражних фільмів. Така різноманітність, з одного боку, створює широкий вибір і покращує користувацький досвід, але з іншого це призводить до інформаційного перевантаження[15,17].

У таких умовах користувачеві стає дедалі складніше швидко знаходити релевантний контент, який відповідає його інтересам і настрою. Одночасно стрімінгові платформи стикаються з потребою втримувати увагу користувачів, адже конкуренція між сервісами постійно зростає. Саме тому персоналізовані рекомендації стали ключовим елементом у розвитку сучасних стрімінгових систем[1,2,15].

Персоналізація зменшує вибірковий шум, забезпечує релевантність та суттєво покращує взаємодію користувача з платформою. Користувачеві більше не потрібно витрачати час на перегляд довгих списків фільмів або плейлистів, оскільки система автоматично пропонує найбільш відповідний контент. Тому важливо дослідити алгоритми та методи машинного навчання, які лежать в основі сучасних рекомендаційних систем, та знайти способи їх оптимізації.

Проблеми та потреби предметної області

Аналізуючи сферу стрімінгових сервісів, можна виділити низку характерних проблем та потреб, які формують запит на персоналізовані рекомендації:

1) Велика кількість контенту

Як вже згадували, щодня створюються тисячі нових музичних треків, відео та фільмів. Через це стрімінгові платформи мають величезні бібліотеки контенту, в яких користувачу складно орієнтуватися без допомоги алгоритмів рекомендацій.

2) Зростаюча конкуренція між платформами

Користувач очікує максимально швидкого доступу до релевантного контенту і якщо сервіс не забезпечує якісні рекомендації, користувач без вагань переходить до альтернативної платформи. Саме тому рекомендації стали інструментом утримання користувачів.

3) Індивідуалізація досвіду

У кожного користувача власні вподобання, поведінка та патерни споживання. Вони часто не вписуються у прості жанрові категорії. Тому системи рекомендацій повинні враховувати десятки сигналів: історію переглядів, контекст, час доби, тип пристрою, поведінкові патерни та інші фактори[15,17].

4) Точність та адаптивність рекомендацій

Інтереси користувачів динамічні, наприклад: сьогодні слухають поп, завтра це може бути інді, а післязавтра вже починає слухати подкасти про технології. Алгоритми повинні враховувати такі зміни та адаптувати рекомендації в режимі реального часу.

5) Утримання і комерційна ефективність

Персоналізація впливає на тривалість сесії, кількість переглядів, частоту підписок, взаємодій та навіть на монетизацію, оскільки релевантний контент збільшує цінність платформи для користувачів[2].

Особливості поведінки користувачів у стрімінгових сервісах

Дослідження показують, що користувач приймає рішення про вибір контенту в середньому за 10-20 секунд. Якщо сервіс не пропонує релевантних варіантів, користувач або покидає застосунок, або переходить до іншої платформи. Це формує певні закономірності поведінки[15,17]:

- короткі сесії пошуку, але тривалі сесії споживання;
- висока залежність від рекомендацій (на Netflix ≈80% переглядів припадає саме на рекомендований контент);
- мінливість інтересів;
- велика роль візуальних і аудіосигналів (трейлери, прев'ю, обкладинки);
- безперервність споживання, тобто автоперехід між епізодами, автоматичне відтворення наступного відео чи треку.

Такі особливості формують вимоги до алгоритмів персоналізації: вони повинні бути точними, швидкими, масштабованими та здатними адаптуватися в реальному часі.

Приклади існуючих рішень на ринку

Найбільші світові стрімінгові сервіси вже впровадили потужні системи персоналізації:

- **Spotify** використовує методи машинного навчання, зокрема колаборативну фільтрацію та аналіз аудіофіч, для формування добірок та рекомендацій.
- **Apple Music** аналізує слухацькі вподобання та активність користувача для створення персональних плейлистів.
- **YouTube** застосовує глибокі нейронні мережі для ранжування відео та прогнозування ймовірності перегляду[19].
- **Netflix** використовує складні гібридні алгоритми, поєднуючи контентний аналіз, поведінкові дані, графові зв'язки та моделі глибокого навчання[15,17].

Ці приклади демонструють, що ефективні рекомендаційні системи зазвичай ґрунтуються не на одному методі, а на їх комбінації.

Цільова аудиторія дослідження

Цільовою аудиторією є користувачі стрімінгових платформ -від слухачів музики до глядачів відео та подкастів. Їхні інтереси й поведінка значною мірою визначають вимоги до рекомендаційних алгоритмів, тому персоналізація має вирішальне значення для покращення користувацького досвіду[15,17].

Технологічна база дослідження

У межах подальших досліджень планується розглянути та оптимізувати алгоритми персоналізованих рекомендацій на основі гібридних моделей машинного навчання. Для цього будуть використані такі технології:

- **Сервіси: Firebase** - для збору, зберігання та обробки даних користувачів.
- **Алгоритми**: колаборативна фільтрація, контентний аналіз, гібридні моделі, методи глибокого навчання[15,17].
- **Фреймворки ML**: TensorFlow Lite або PyTorch Mobile - для інтеграції моделей у мобільні застосунки та роботи в реальному часі[10,28-30].

1.2 Персоналізований контент

Персоналізованим контентом називають матеріали, які адаптовані під унікальні інтереси, поведінку, вподобання та інші характеристики кожного окремого користувача. На відміну від універсальних пропозицій, що надаються всім однаково, персоналізований контент формується з урахуванням історії взаємодій, демографічних даних, контексту використання та інших факторів. Його основна мета підвищити якість користувацького досвіду, забезпечивши доступ до найбільш релевантних і цікавих матеріалів[1,15,17].

У сучасних цифрових екосистемах персоналізація є одним з ключових інструментів взаємодії між брендом і користувачем. За даними Adobe

Experience Cloud, персоналізація розглядається як стратегія, що використовує дані користувачів для формування індивідуального досвіду в кожній точці контакту з платформою [1,15]. Завдяки цим даним алгоритми можуть визначати, які категорії контенту цікавлять користувача, які матеріали привертають його увагу, як він реагує на рекомендації чи сповіщення, та відповідно адаптувати інтерфейс або контентну стрічку.

Прикладом реалізації такої персоналізації може бути індивідуально сформована головна сторінка застосунку, де користувач бачить плейлисти, відео чи добірки, відібрані саме під його інтереси. Це дозволяє зменшити час пошуку необхідного матеріалу й покращує взаємодію з платформою [1].

Навіщо потрібен персоналізований контент

Персоналізація відіграє критично важливу роль у стрімінгових сервісах. Основні її переваги полягають у такому:

- **Утримання користувачів.** Релевантний контент значно підвищує ймовірність того, що користувач повернеться до застосунку. Це знижує рівень відтоку та підвищує лояльність аудиторії [2,15,17].
- **Підвищення взаємодії.** Користувачі активніше взаємодіють із платформою, коли отримують персоналізовані рекомендації. Вони частіше переглядають матеріали, додають їх у списки відтворення, лайкають або коментують[2].
- **Ефективність рекламних кампаній.** Персоналізація не лише підбирає релевантний контент, а й дає змогу показувати цільову рекламу, яка краще відповідає інтересам користувача. Це підвищує конверсії та зменшує витрати на маркетинг[2].
- **Покращення монетизації.** Користувачі, що отримують персоналізовані рекомендації, частіше оформлюють підписки, купують додаткові матеріали або здійснюють інші транзакції. Це напряму впливає на прибутковість платформи [2].
- **Оптимізація пошуку.** Персоналізовані добірки та рекомендації скорочують час, який користувач витрачає на пошук відповідного

контенту. Система може пропонувати найкращі варіанти автоматично, спираючись на поведінкові дані[1,2,15,17].

Завдяки цьому персоналізація формує більш тісний зв’язок між платформою та користувачем: підвищує задоволеність, покращує взаємодію, а також створює цінність як для сервісу, так і для кінцевого користувача [1-2,15,17].

1.3 Методи персоналізації контенту

1.3.1 Класифікація методів персоналізації

Хоча методи персоналізації дуже різняться між собою, вони завжди спрямовані на те, щоб надати користувачеві саме той матеріал, який зацікавить його з огляду на наявну в нього інформацію. Залежно від того, як аналізуються дані та формуються пропозиції, всі підходи можна розділити на три основні групи[15,17]:

1) **Колаборативна фільтрація (Collaborative Filtering).** Працює на основі поведінки користувачів цей підхід групує їх у різні групи. Використовуючи загальні групові характеристики, а потім він повертає конкретні товари всій групі за принципом, що подібні користувачам (з точки зору поведінки) зацікавлених у подібних товарах. Можем сказати, що базується на аналізі поведінки інших користувачів. Він розглядає як підходи, засновані на пам’яті, так і підходи, що засновані на моделі [3-4,17].

Переваги:

- **Добре працює на практиці.** У практичних ситуаціях спільна фільтрація постійно доводить свою ефективність. Якість його рекомендацій тісно пов'язана з його залежністю від оцінок користувачів. Надаючи пов'язані товари

покупцям зі схожими смаками, багато підприємств електронної комерції використовують цю стратегію, щоб збільшити потоки доходів і підвищити рівень задоволеності та залученості клієнтів [5].

- **Метадані користувача не є обов'язковими.** Колаборативна фільтрація ефективно працює лише з оцінками користувачів, усуваючи потребу у вичерпній інформації про користувача, на відміну від деяких алгоритмів рекомендацій. Така простота знижує обчислювальні витрати і прискорює процес впровадження [5].
- **Різноманітні міждоменні пропозиції.** Колаборативна фільтрація пропонує різноманітні рекомендації з різних галузей, незалежно від доменних кордонів. Незалежно від того, чи рекомендуєте ви фільми, книги або товари, цей підхід легко підлаштовується під смаки та пристрасті користувача.
- **Часто буває достатньо неявного вкладу користувача.** Ефективне використання неявних відгуків користувачів стає можливим завдяки спільній фільтрації. Унарні рейтинги, такі як кліки на посилання або взаємодія з контентом, є корисними предикторами поведінки користувачів. Навіть за відсутності явних оцінок система може використовувати ці дані, щоб визначити вподобання користувачів і надати відповідні рекомендації.

Недоліки:

- **Проблема «Холодний старт» -** одна з найбільших перешкод для спільної фільтрації. Ця проблема виникає, коли алгоритму потрібна допомога, щоб надати точні пропозиції для товарів з невеликою історією взаємодії або

для нових користувачів. Якщо даних достатньо, система може надавати релевантні рекомендації, що знизить задоволення та залученість клієнтів.

- **Обмеженість даних.** Дані про взаємодію користувача з елементом є ключовим компонентом спільної фільтрації. Однак у реальності цих даних може не вистачати, особливо для спеціалізованих або менш популярних продуктів. Особливо для продуктів з невеликою кількістю користувацького вкладу, розріджені дані можуть призвести до більш точних пропозицій і кращого користувацького досвіду.
- **Проблема «Grey sheep».** У компанії виникають проблеми з користувачами, які мають незвичні або відмінні смаки. У таких ситуаціях стає важко визначити людей зі схожими вподобаннями, що призводить до неідеальних пропозицій. Таке обмеження ускладнює роботу системи з різноманітними вподобаннями користувачів.
- **Проблема «Black sheep».** Система вразлива до маніпуляцій та зловмисних дій користувачів. У деяких випадках користувачі можуть цілеспрямовано маніпулювати рейтингами, ставлячи під загрозу точність і надійність системи рекомендацій. Через ці серйозні проблеми з надійністю та довірою система є вразливою до маніпуляцій та атак.

2) **Фільтрація на основі контенту (Content-Based Filtering).** Даний алгоритм пошуку інформації використовує функції елементів вибору та повернення елементів, що відповідають запитам користувача. Цей метод часто враховує характеристики інших предметів, до яких користувач проявляє інтерес, наприклад створює поради за допомогою

характеристик контенту, таких як теги та жанри. Ефективно використовувати цей алгоритм, коли дані користувача обмежені [6-7].

Переваги:

- **Незалежність від даних інших користувачів.** Можна зазначити, що це основаня перевага цього методу на відміну від колоборативної фільтрації тому, що вона незалежна від даних інших користувачів та не потребує великої кількості взаємодій користувача, вона може надавати персоналізований контент й з мінімальною активністю користувача [8].
- **Адаптовані до вподобань користувача.** Фільтрація на базі контенту найкраще поєднує рекомендації з інтересами та вподобаннями користувача. Зіставляючи атрибути об'єктів бази даних із профілем користувача, система пропонує максимально персоналізовані рекомендації. Наприклад, якщо користувач надає перевагу нішевому контенту, такому як незалежні документальні фільми про культуру Шотландії або маловідомі техаські музичні гурти, система запропонує схожий контент, який відповідає його інтересам.
- **Прозорість рекомендацій.** Фільтрація на основі вмісту підвищує довіру користувачів і робить рекомендації більш прозорими.
- Системи фільтрації на основі контенту є більш простими для створення та аналізу даних, ніж колаборативні фільтраційні системи. Використовуючи такі методи, як частотний аналіз термінів і моделі векторного простору, системи на основі контенту зосереджені на класифікації об’єктів на основі їхніх атрибутів.

- **Подолання «холодного старту».** Проблема «холодного старту», яка часто виникає під час спільної фільтрації, легко вирішується фільтрацією на основі вмісту.

Недоліки:

- **Обмежена новизна та різноманітність.** Для систем рекомендацій на основі контенту баланс між новизною та релевантністю є однією з головних проблем. Ці системи є чудовим інструментом для класифікації вподобань користувачів, але вони також можуть запропонувати надто звичні варіанти, обмежуючи різноманітність вибору користувача.
- **Масштабованість і призначення атрибутів** є важливою проблемою у фільтрації на основі контенту. Кожен додаток нового продукту, послуги або елемента контенту потребує опису та опису його характеристик. Можна зробити масштабування системи складним і трудомістким через цей постійний і складний процес призначення атрибутів. Такі методи, як матрична факторизація (яка розкладає матрицю взаємодії користувача з елементом на менші матриці для виявлення прихованих особливостей) можуть допомогти в управлінні великими наборами даних, але основне завдання оновлення та підтримки інформації про атрибути залишається великою перешкодою.
- **Точність і коректне використання атрибутів** є ключовими чинниками успіху рекомендаційних систем, що базуються на контенті. Цей тип систем значною мірою залежить від присвоєння тегів, яке проводять спеціалісти відповідної галузі. Однак можуть виникати помилки або невідповідності у визначенні атрибутів через велику кількість об’єктів, які потребують ручного опрацювання.

Така суб’єктивність та потенційна неточність у тегуванні значно вплинути на ефективність системи. Подібні системи потребують процесу, який гарантує послідовне та точне призначення функцій, щоб вони могли працювати якнайкраще.

3) **Гібридні методи (Hybrid methods).** Даний метод використовує переваги обох алгоритмів для усунення недоліків, таких як проблема «холодного старту», що виникає у Collaborative Filtering за допомогою Content-Based Filtering[18-19].

Переваги:

- **Компенсує недоліки інших методів.** Дозволяє уникати проблеми «холодного старту», яка є поширеною для колоборативної фільтрації за допомогою використання атрибутів контенту, тобто за допомогою фільтрації на основі контенту. Забезпечує кращу обробку складних або нішевих інтересів користувача, враховуючи його поведінку та властивості контенту.
- **Підвищена точність.** Завдяки тому, що поєднується декілька підходів, даний метод надає змогу створювати рекомендації, які точніше відповідають інтересам користувача, а також зменшуючи ризик неактуальних пропозицій користувачу.
- **Гнучкість.** Гібридні методи дозволяють адаптувати систему до змін у поведінці користувачів і оновлення контенту, що гарантує постійну актуальність рекомендацій.
- **Різноманітність рекомендацій.** Поєднання різних підходів, дає перевагу в тому, що уникається одноманітність, оскільки система пропонує контент не тільки, що базується на вподобаннях схожих користувачів, а й задопомогою характеристик контенту.

Недоліки:

- **Складність реалізації.** Складність реалізації полягає в тому, що розробка гібридної системи вимагає значних зусиль, оскільки нам необхідно інтегрувати кілька методів та забезпечити їх взаємодію.
- **Високі обчислювальні ресурси.** Оскільки, йде поєднання кількох методів, збільшується і аналіз великих обсягів даних, як-от атрибитів контенту й поведінки користувачів, що потребує складних алгоритмів та потужних серверів.
- **Синхронізація даних.** Розбіжності у форматах, швидкості оновлення та неповнота даних можуть зробити інтеграцію різних джерел інформації складною.
- **Витрати на підтримку.** Обробка даних, регулярне оновлення алгоритмів і нагляд за ефективністю гібридних систем збільшують загальні витрати.

1.3.2 Алгоритми персоналізації

Розглянемо основні алгоритми персоналізації в цьому розділі. Вони включають колаборативну фільтрацію, кластеризацію, дерева рішень, нейронні мережі та графові алгоритми. Залежно від типу платформи, доступних даних і цілей системи рекомендацій кожен із цих методів може бути використаний. Далі ми розглянемо кожен із цих методів більш детально, розглядаючи їхню роботу, місце застосування[15,17].

1) **Метод колаборативної фільтрації на основі матриць (Matrix Factorization).** В основі цього підходу лежить аналіз вподобань користувачів за допомогою оцінок контенту, наданих самими користувачами. **Сингулярна декомпозиція** (Singular Value Decomposition, SVD) - це один із способів спільної матричної фільтрації, який використовує методологію матричної декомпозиції

для виявлення латентних (неявних) елементів, що визначають схожість між користувачами та елементами контенту. Цей метод дозволяє нам пропонувати нові продукти, які, ймовірно, зацікавлять користувачів, і прогнозувати рейтинги контенту, які ще не були присвоєні. Можливість адаптувати рекомендації відповідно до смаків інших користувачів, схожих на поточного користувача, є однією з головних переваг методу [9,17].

2) **Аналіз на основі кластеризації.** Алгоритм кластеризації групує людей або інформацію на основі спільних атрибутів за допомогою статистичних методів. K-середні є популярною технікою кластеризації, яка розділяє дані на групи, або кластери, де елементи з одного кластера схожі між собою, а елементи з інших кластерів відрізняються один від одного. Розпізнаючи групи користувачів зі схожими смаками або групи контенту, які часто споживаються разом, кластеризація може допомогти у створенні персоналізованих пропозицій в контексті персоналізації контенту. Це може підвищити точність пропозицій без необхідності ретельного вивчення особистої інформації користувача[15,17].

3) **Дерева рішень.** Дерева рішень використовуються для аналізу та прогнозування залежно від характеристик людини або матеріалу. Вони будують дерево рішень з обмеженнями в кожному вузлі, які визначають, як потрібно поводитися з певними параметрами (наприклад, вікова група користувачів або жанр контенту). Це дає змогу чітко зрозуміти, який матеріал, виходячи з унікальних особливостей кожного користувача, слід йому запропонувати. Перевагою цих моделей є їхня прозорість і зрозумілість, що полегшує адаптацію системи до мінливих обставин.

4) **Нейронна мережа.** Важливою тенденцією в розробці персоналізованих пропозицій є застосування глибокого навчання. За допомогою нейронних мереж, таких як **згорткові нейронні мережі**

(CNN) та **рекурентні нейронні мережі (RNN)**, можна виявити складні закономірності в даних. Щоб допомогти в розробці пропозицій, які враховують попередню поведінку користувача, RNN використовуються для аналізу послідовностей, таких як історія переглядів або відгуків. CNN використовуються для обробки мультимедійного контенту, наприклад, зображень або фільмів, що дозволяє створювати адаптовані пропозиції, засновані на слухових або візуальних особливостях контенту. Перевага нейронних мереж полягає в тому, що вони здатні навчатися на великих обсягах даних, що дає змогу створювати дуже точні пропозиції[18,19].

5) **Графові алгоритми.** Зв'язки між людьми і контентом моделюються графовими алгоритмами за допомогою графів, в яких користувачі і компоненти контенту представлені вершинами графа, а їхня взаємодія або схожість - ребрами графа. Вивчаючи зв'язки користувача з іншими користувачами та матеріалами, графові алгоритми, такі як **PageRank** і **Nearest Neighbour Search**, можуть прогнозувати його вподобання. Графові алгоритми дозволяють створювати більш адаптивні системи рекомендацій, які можуть включати складні взаємозв'язки між елементами, наприклад, соціальні мережі або спільні інтереси[15,17].

Ці алгоритми лежать в основі сучасних систем персоналізації контенту, а тип матеріалу, обсяг доступних даних і необхідний рівень точності впливають на вибір методу. Надаючи кожному споживачеві персоналізований підхід, вони можуть значно покращити користувацький досвід.

1.3.3 Дані для персоналізації

Ефективна кастомізація контенту на потокових платформах вимагає збору та обробки різноманітних типів даних, які допомагають краще прогнозувати вподобання користувачів і надавати їм релевантні

матеріали[15,17]. Основні категорії даних, що використовуються для кастомізації, такі:

1) **Історія дій користувача.** Історія взаємодії користувача з платформою є основним джерелом даних для побудови персоналізованих рекомендацій. Ці дані включають в себе:

- **Перегляд контенту**, що користувач дивився або слухав, скільку часу провів на певному контенті та чи завершив він перегляд чи прослуховування до кінця.
- **Оцінки та лайки,** а саме контент, що був оцінений за допомогою зірочок або лайків, може вказувати на його переваги.
- **Списки відтворення.** Контент, який користувач зберігає або додає до своїх списків відтворення, також є важливим фактором.
- **Перегляд і повторні перегляди.** Кількість разів, коли користувач дивиться певний контент, допомагає краще зрозуміти, чи є він актуальним для користувача.

2) **Демографічні дані користувача.** Демографічні характеристики користувача також грають важливу роль у персоналізації контенту. Зазвичай, це такі параметри:

- **Вік** користувача є важливим чинником для пропозиції контенту. Наприклад, для дітей будуть пропонуватися мультфільми та фільми відповідно до їх вікової категорії, для більш молодіжної категорії будуть запроновані новинки, що зацікавлять їх
- **Стать,** цей параметр не завжди є визначальним чинником, проте деякі види контенту можуть бути привабливими для певної статі, наприклад спортивні чи музикальні жанри.
- **Географічне розташування,** данний параметр є дуже важливим для рекомендацій локалізованого контенту, наприклад пісні локальних виконавців.

3) **Контекст.** Конкретні дані містять інформацію, яка допомагає змінювати рекомендації відповідно до умов використання платформи:

- **Час доби.** Рекомендації можуть змінюватися залежно від часу, коли користувач використовує платформу. Наприклад, під час обідньої перерви може бути корисно переглянути короткий ролик або музичну композицію, а ввечері переглянути фільм або серіал.
- **Використання.** Контент може бути адаптований до різних умов, якщо користувач знаходиться вдома або в транспорті. Наприклад, короткі ролики або подкасти можуть бути корисними для використання на ходу.
- **Тип пристрою.** Оскільки, платформу часто використовують на різних пристроях, таких як смартфони, планшети та комп'ютери, це впливає на тип контенту, який можна переглядати. На комп'ютерах зазвичай доступний більш тривалий контент, тоді як на мобільних пристроях зазвичай доступні короткі відео або легка музика.

5) **Активність у соціальних мережах.** На персоналізацію контенту також може впливати активність користувачів в соціальних мережах і взаємодія з іншими. Це може бути:

- **Лайки та коментарі.** Контент, який користувачі лайкають або коментують на платформі, може бути використаний для визначення їхніх уподобань.
- **Рекомендації друзів.** Якщо користувач взаємодіє з контентом, який рекомендують йому друзі чи інші користувачі, це може бути індикатором контенту, який є популярним або актуальним.
- **Фоловінг.** Інформація про те, кого користувач приймає в друзі, за ким слідкує та чиї профілі переглядає, також може допомогти визначити його інтереси.

Комбінація цих типів даних надає можливість будувати більш точні й адаптивні моделі, які відображають як довгострокові інтереси, так і короткостроковий контекст користувача[15,17].

1.4 Впровадження машинного навчання для покращення персоналізації у мобільних застосунках

Машинне навчання в мобільних додатках створює нові можливості для автоматизації процесів та покращення користувацького досвіду. TensorFlow Lite та Firebase ML Kit є основними технологіями для розробників, які прагнуть впровадити машинне навчання в додатки[10-14,28-30].

TensorFlow Lite (TFLite) - це фреймворк, який дозволяє високопродуктивно виконувати моделі машинного навчання на мобільних та вбудованих пристроях. Він має такі можливості, як:

- Підтримка кастомних моделей, наприклад для ефективної роботи на мобільних пристроях TensorFlow Lite дозволяє конвертувати ваші кастомні моделі, створені за допомогою TensorFlow, у формат TFLite. Що добре підходить для складних завдань, таких як розпізнавання слів і обробка зображень [10,28-30].
- Відмінна продуктивність, а саме апаратна оптимізація, яка включає підтримку процесорів TPU і GPU, підтримка Quantization, що прискорює обчислення і мінімізує розмір моделі [11,28-30].
- Робота в автономному режимі, оскільки моделі TFLite зберігаються локально, програми можуть працювати без підключення до інтернету, гарантуючи швидкі результати та безпеку даних [12,28-30].

Недоліки:

- Складність у налаштуванні, а саме вимагає глибокого розуміння процедур машинного навчання, включаючи розробку, навчання та оптимізацію моделей. Для попереднього навчання необхідно використовувати TensorFlow [13].
- Потреба в технічних знаннях, таких як перетворення моделей, оптимізація та інтеграція додатків - це навички, якими повинні володіти розробники [10].

Firebase ML Kit - це хмарна платформа машинного навчання, що надає простий доступ до попередньо навчених моделей і функцій ML без необхідності створення власних моделей.

Переваги:

- Проста інтеграція. Для таких додатків, як переклад, аналіз тексту та розпізнавання облич, ML Kit надає API-інтерфейси. Інтеграція з хмарним сховищем Firestore та іншими сервісами Firebase [12].
- Наявність навчених моделей. Розробники можуть використовувати готові моделі Google для таких завдань, як розпізнавання об'єктів і розпізнавання тексту, без необхідності їхнього навчання [13].
- Простота для новачків. Не обов'язково потрібно мати глибокі знання в галузі машинного навчання, достатньо мати базові знання про налаштування SDK та API [4].

Недоліки:

- Обмежені можливості кастомізації: Більшість API не передбачають вдосконалення або модифікацію моделі, що обмежує можливість адаптації моделей до певних завдань [13].
- Залежність від інтернет-з'єднання: Певні функції (наприклад, хмарні моделі) вимагають підключення до Інтернету для роботи [14].

TensorFlow Lite підходить для досвідчених розробників, які прагнуть створювати високопродуктивні рішення, тоді як Firebase ML Kit є оптимальним вибором для швидкої інтеграції машинного навчання в додатки з мінімальними витратами ресурсів. Інтеграція двох платформ може бути оптимальною стратегією для створення адаптивних рішень.

2. АНАЛІЗ ТЕХНОЛОГІЙ ТА МЕТОДІВ ПОБУДОВИ ПЕРСОНАЛІЗОВАНИХ РЕКОМЕНДАЦІЙ У СТРІМІНГОВИХ СЕРВІСАХ

2.1 Аналіз класичних моделей фільтрації: обґрунтування необхідності гібридизації

2.1.1 Фільтрація на основі Контенту (Content-Based Filtering, CBF)

Фільтрації на основі контенту ґрунтується на використанні характеристики об’єкта (наприклад, фільму, відео чи музичного треку) для генерації рекомендацій. Система починає аналізувати властивості елементів, які раніше сподобалися користувачу, на основі його минулих дій або явних оцінок, і вибирає інші елементи, що мають подібні характеристики[6-8,15,17]. Таким чином, рекомендації генеруються на основі індивідуальних уподобань конкретного користувача.

Щоб продемонструвати роботу фільтрації на основі контенту у стрімінгових сервісах, розглянемо спрощений приклад із платформою Netflix. Припустимо, що ми вручну формуємо набір ознак для кожного медіа-елемента (фільму, серіалу). На наступному рисунку 1.1 уявимо матрицю ознак, де кожний рядок відповідає окремому елементу контенту, а кожен стовпець - певній характеристиці. Наприклад, для відеосервісу це можуть бути жанр (драма, екшн, бойовик), країна виробництва, режисер, основні актори або тематичні теги (супергерої, подорожі, реальні події).

Для спрощення вважатимемо, що ця матриця є двійковою, тобто значення 1 означає, що елемент має відповідну характеристику.

Так само і користувач може бути представлений у тому ж просторі ознак. Частина характеристик користувача може задаватися явно, наприклад, він вказує у профілі, що віддає перевагу жанрам драма та режисеру Кристоферу Нолану. Інші ознаки можуть бути визначені системою автоматично: наприклад, користувач переглянув декілька фільмів з актором Райаном Рейнальдсом.

Тоді завдання моделі стає рекомендувати нові елементи, що найбільше відповідають профілю цього користувача. Для цього спочатку обирається метрика схожості (наприклад, косинусна подібність або скалярний добуток)[6-8,17], після чого система обчислює ступінь схожості кожного потенційного елемента контенту з користувацьким профілем. Елементи з найбільшим значенням подібності потрапляють у рекомендації.

Варто підкреслити, що така модель формує рекомендації виключно на основі індивідуальних вподобань цього користувача та не використовує дані інших користувачів платформи.

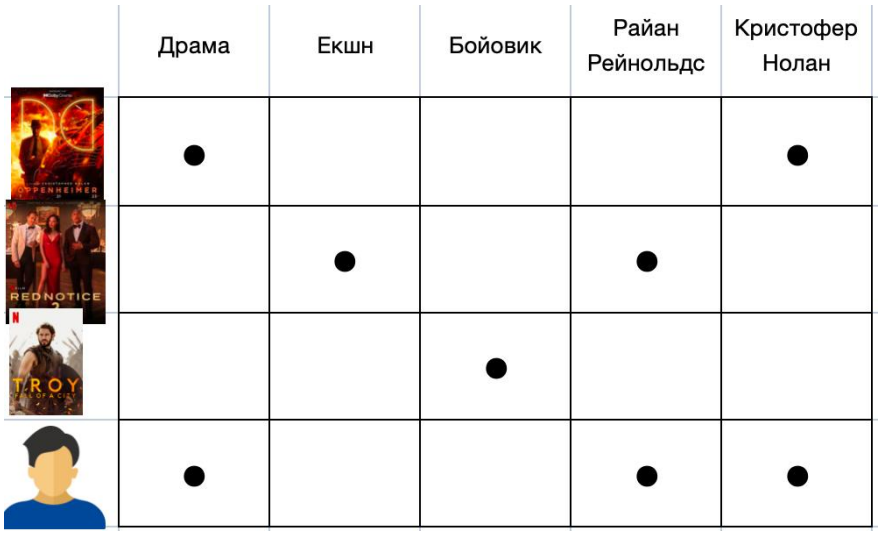


Рис. 1.1 Спрощена матриця ознак фільмів та профілю користувача для CBF-моделі

Проведемо розрахунки схожості за допомогою скалярного добутку. Оскільки всі вектори є двійковими, скалярний добуток показує кількість спільних активних ознак.

Використовуватимо таку формулу, для обчислення:

$$U \times A_i = \sum_{j=1}^{n} U_j \times A_{ij}, \qquad (2.1)$$

де:

U - вектор характеристик користувача;

A_i - вектор характеристик і-го елемента контенту (фільму, серіалу тощо);

n - загальна кількість ознак у векторному просторі;

U_j - значення j-ої ознаки користувача (0 або 1 у двійковій моделі);

A_{ij}- значення j-ої ознаки для і-го елемента контенту (також 0 або 1);

$U \times A_i$ - кількість спільних активних ознак користувача та елемента, тобто міра їхньої схожості.

Дізнаємося схожість користувача з фільмом 1:

$$\mathrm{U} = [1,0,0,1,1]$$

$$\mathrm{A}_1 = [1,0,0,0,1]$$

$$\mathrm{U} \times \mathrm{A}_1 = 1 \times 1 + 0 \times 0 + 0 \times 0 + 1 \times 0 + 1 \times 1 = 2$$

Дізнаємося схожість користувача з фільмом 2:

$$\mathrm{A}_2 = [0,1,0,1,0]$$

$$\mathrm{U} \times \mathrm{A}_1 = 1 \times 0 + 0 \times 1 + 0 \times 0 + 1 \times 1 + 1 \times 0 = 1$$

Дізнаємося схожість користувача з фільмом 3:

$$\mathrm{A}_2 = [0,0,1,0,0]$$

$$\mathrm{U} \times \mathrm{A}_1 = 1 \times 0 + 0 \times 0 + 0 \times 1 + 1 \times 0 + 1 \times 0 = 0$$

Підсумкові результати відображені у таблиці 2.1:

Таблиця 2.1

Результати розрахунку схожості між користувачем та фільмами

Фільм	Скалярний добуток	Рівень схожості
Фільм 1 (“Oppenheimer”)	2	Найвища схожість
Фільм 2 (“Red Notice”)	1	Помірна схожість
Фільм 3 (“Troy”)	0	Немає схожості

У наведеному прикладі найбільш релевантним для користувача є Фільм 1, оскільки він має дві спільні характеристики з профілем користувача - жанр драма та режисера Крістофера Нолана. Таким чином, система рекомендуватиме саме цей фільм у першу чергу.

Детальний огляд переваг та обмежень методів фільтрації на основі контенту було наведено у підрозділі 1.3.1. У межах цього розділу вони враховуються як підґрунтя для подальшого обґрунтування доцільності гібридизації рекомендаційних систем[6-8,15,17].

2.1.2 Колаборативна фільтрація (Collaborative Filtering, CF)

На відміну від фільтрації на основі контенту, яка орієнтується на властивості самих елементів, колаборативна фільтрація ґрунтується на аналізі поведінкових патернів великої кількості користувачів. Основна ідея колаборативної фільтрації CF полягає в тому, що користувачам зі схожими інтересами можна рекомендувати ті елементи, які вже сподобалися іншим представникам цієї "поведінкової спільноти"[3-5,15,17,20].

Тобто CF не розглядає самі характеристики об’єктів, а працює виключно з матрицею взаємодій користувач-елемент, що може включати такі дані, як:

- рейтинги (явний фідбек),
- перегляди, кліки, лайки (неявний фідбек),
- додавання до списків,
- тривалість перегляду або прослуховування,
- інші маркери зацікавленості.

Таким чином, система встановлює розуміння прихованих уподобань користувачів, порівнюючи їхню поведінку з поведінкою інших користувачів у системі[15,17].

Якщо описувати структуру даних для CF (Collaborative Filtering), то у класичному колаборативному підході центральною є матриця зворотного

зв’язку (user-item matrix). А саме: рядки - це наші користувачі, а стовпці - це елементи (фільми, треки, відео), де значення відображає інтенсивність взаємодії[15,17,20].

Типи даних в матриці є двох типів:

- **Явні оцінки (Explicit Feedback)**, а саме числові значення, такі як 1-5, лайки/дизлайки тощо.
- **Неявні ознаки (Implicit Feedback)**, як ознаки поведінки, які не є прямою оцінкою, але свідчать про інтерес користувача до контенту, наприклад: перегляд епізоду, додавання до списку “Моє/обранне/збережені”, прослуховування треку більш ніж N секунд[15,17,20].

Для багатьох стрімінгових систем ,наприклад таких як YouTube, Netflix - неявний фідбек є переважаючим, оскільки користувачі рідко залишають явні оцінки[20].

Колаборативна фільтрація (CF) поділяється на два великі класи, які відрізняються своїм підходом до обробки матриці взаємодій "користувач-елемент"[15,17].

Перший клас, який розглянемо називається **Memory-based CF (Пам’яттєві методи)**. Це методи, які напряму працюють з великою матрицею взаємодій і використовують в обчисленняї схожості між користувачами або елементами для генерації нових рекомендацій. Вони не створюють окремої моделі[3-5,17,20].

1) User-Based Collaborative Filtering (UB-CF)

Отже принцип такий, що система знаходить певну групу користувачів, схожих до нашого цільового користувача (так званих "сусідів"), і рекомендує елементи, які ці сусіди споживали, але які цільовий користувач ще не бачив[3-5,17]. Схожість між двома користувачами (u та v) обчислюється за допомогою таких метрик, як:

- Cosine Similarity (Косинусна схожість),
- Pearson Correlation (Кореляція Пірсона),

- Jaccard Similarity (для бінарних даних)[3-5,17,20].

Якщо продеменструвати матиматичною формулою, то User-based CF шукає користувачів зі схожим профілем переглядів, саме так:

$$\mathrm{sim}(u, v) = \frac{\vec{r}_u \times \vec{r}_v}{\|\vec{r}_u\| \times \|\vec{r}_v\|}, \qquad (2.2)$$

де $\vec{r}_u$ - вектор оцінок (або бінарних взаємодій) користувача u.

Прогноз оцінки для пари (u, i):

$$\hat{r}_{u,i} = \frac{\sum_{v \in N_k(u)} \mathrm{sim}(u,v) \times r_{v,i}}{\sum_{v \in N_k(u)} |\mathrm{sim}(u,v)|}, \qquad (2.3)$$

де $N_k(u)$- множина k найбільш схожих користувачів[3-5,17,20].

Перевагами даного методу є інтуїтивність та простота реалізації, вона має гарна якість для малих наборів даних та легко адаптується до змінних уподобань користувача. Проте можемо виділити і такі недоліки, як погане масштабування (складність $O(n^2)$, де n - це кількість користувачів), нестійкість до розрідженості даних (Sparsity) через це важко знайти достатню кількість спільних елементів для обчислення надійної схожості[15,17,20].

2) Item-Based Collaborative Filtering (IB-CF)

Даний метод, популяризований Amazon у 2000-х роках. Система шукає елементи, що є подібними до тих, з якими вже взаємодіяв користувач. Схожість обчислюється між парами елементів (наприклад, фільм A і фільм B)[3-5,15,17].

Математично представлена так:

$$\mathrm{sim}(i, j) = \cos(\vec{r}_i, \vec{r}_j), \qquad (2.4)$$

$$\hat{r}_{u,i} = \frac{\sum_{j \in I_u} \mathrm{sim}(i,j) \times r_{u,j}}{\sum_{j \in I_u} |\mathrm{sim}(i,j)|}, \qquad (2.5)$$

де I_u- набір елементів, з якими взаємодіяв корситувач u.

Серед переваг є краща масштабованість порівняно з UB-CF, оскільки кількість елементів зазвичай повільніше зростає, ніж кількість користувачів та рекомендації стабільніші, оскільки схожість між елементами змінюється рідше, ніж уподобання користувачів, а також добре працює у великих

каталогах. Недоліками є вимагання регулярного оновлення матриці схожості елементів при додаванні нових елементів та погане виявляння "сусідів"-користувачів[3-5,17].

Другий клас це **Model-based CF (Модельні методи)**. На відміну від пам’яттєвих методів, ці підходи будують математичну модель, що узагальнює патерни взаємодій у даних[9,15,17]. Дана модель використовується для передбачення невідомих оцінок.

Матрична факторизація (Matrix Factorization - MF). Основний підхід цього методу полягає у великій та розрідженій матриці взаємодії (R) ,яка розкладається (факторизується) на дві компактні матриці з низьким рангом, а саме: матрицю прихованих характеристик користувачів (P) та матрицю прихованих характеристик елементів (Q).

$$R \approx P \times Q^T, \qquad (2.6)$$

де:

P - це матриця прихованих характеристик користувачів (User Latent Factors) в якій рядки - це користувачі, а стовпці - це приховані ознаки (наприклад, "любов до фантастики", "схильність до драм").

Q - це матриця прихованих характеристик елементів (Item Latent Factors) в якій рядки - це елементи,а стовпці - це ті самі приховані ознаки.

І звідси випливає мета даного методу - це знайти такі матриці P та Q, щоб їх добуток максимально точно апроксимував початкову матрицю R.

Матрична факторизаці має декілька алгоритмів:

- **SVD (Singular Value Decomposition)** у варіантах для рекомендаційних систем.
- **ALS (Alternating Least Squares)** часто використовується для великих розріджених даних.
- **BPR (Bayesian Personalized Ranking**) оптимізований для роботи з неявним фідбеком.

Переваги матричної факторизації - це висока точність передбачень, стійкість до розрідженості, тобто здатність заповнювати прогалини в даних,

можливість знаходити приховані патерни (наприклад жанрові, стильові уподобання), які не є очевидними, добра масштабованість та ефективна робота з великими даними. І саме цей клас методів був основою алгоритмів-переможців змагання Netflix Prize.

Детальний огляд переваг та обмежень методів колоборативної фільтрації було наведено у підрозділі 1.3.1. У межах цього розділу вони враховуються як підґрунтя для подальшого обґрунтування доцільності гібридизації рекомендаційних систем.

2.1.3 Порівняльний аналіз CF та CBF в табличному та схемному форматі

Для зручності порівняння узагальнимо властивості підходів у таблиці 2.2[15-17].

Таблиця 2.2

Порівняння контентно-орієнтованої та колаборативної фільтрації

Критерій	Content-Based Filtering (CBF)	Collaborative Filtering (CF)
Джерело даних	Атрибути контенту (жанр, теги, опис тощо)	Поведінка користувачів (рейтинги, перегляди, кліки)
Врахування думки інших	Ні	Так
Cold-start для нового контенту	Відсутній (якщо є метадані)	Виражений (потрібні перші взаємодії)
Cold-start для нового юзера	Частково (можна будувати профіль з анкетних даних)	Виражений (немає історії → немає рекомендацій)

Продовження таблиці 2.2

Порівняння контентно-орієнтованої та колаборативної фільтрації

Критерій	Content-Based Filtering (CBF)	Collaborative Filtering (CF)
Залежність від метаданих	Висока	Низька
Масштабованість	Залежить від кількості ознак	Модельні CF (MF, NCF) - добре масштабуються
Пояснюваність	Висока («рекомендується через жанр/актора»)	Середня («схоже на те, що дивилися подібні користувачі»)

Наступним див. рис. 2.2 є візуальне відображення порівняння принципу роботи контентно-орієнтованої та колаборативної фільтрації

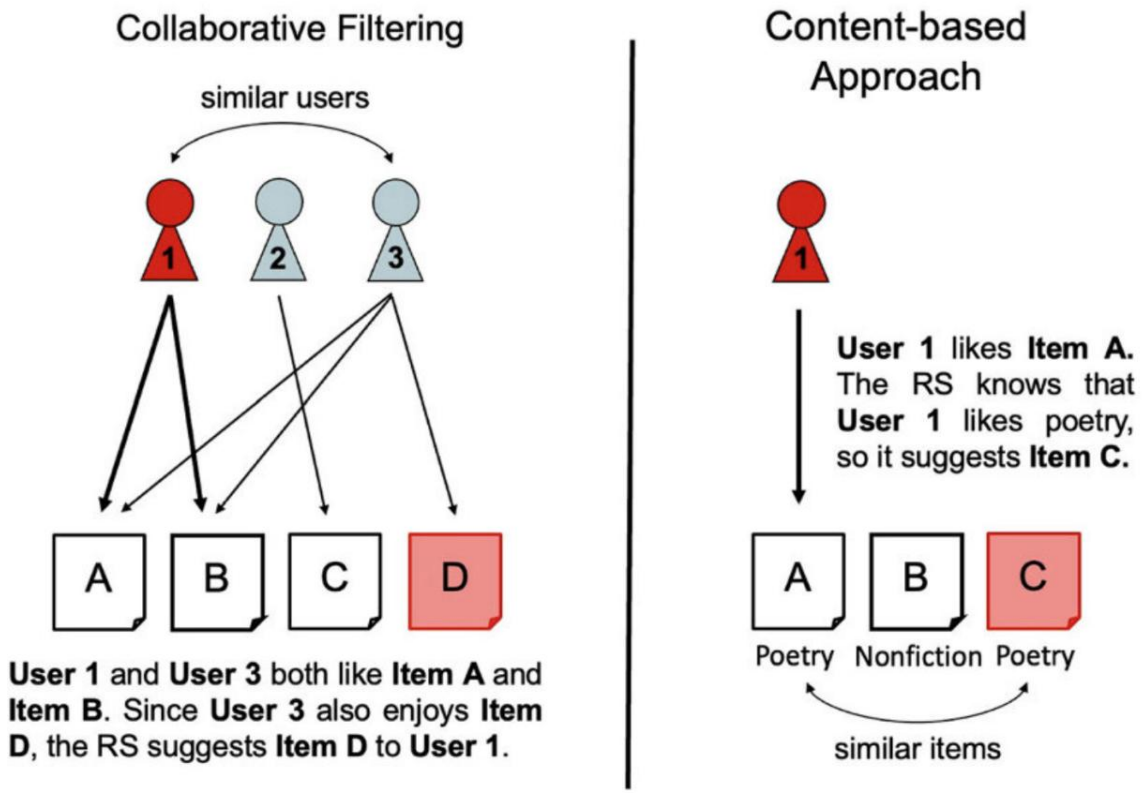


рис. 2.2 Принцип роботи контентно-орієнтованої та колаборативної фільтрації

2.1.4 Недоліки класичних підходів і потреба в гібридизації

Незважаючи на успішне застосування CF і CBF в реальних системах, обидва підходи мають загальні та специфічні обмеження[15-17,20]:

CF:

- не працює без достатньої кількості взаємодій, оскільки має проблему холодного запуску;
- чутливий до розрідженості матриці взаємодій;
- в чистому вигляді не враховує контентні характеристики (наприклад жанр, тривалість, стиль).

CBF:

- генерує дуже схожі рекомендації, обмежуючись контентом, який вже був спожитий;
- сильно залежить від якості та повноти метаданих;
- не використовує інформацію про інших користувачів, що в свою чергу призводить до втрати «колективної мудрісті»[6-8,15,17].

Загалом обидва підходи:

- слабко враховують часові зміни вподобань (concept drift);
- не оптимізують довготривалі метрики (удержання користувача, LTV);
- погано працюють у сценаріях з дуже динамічним контентом (короткі відео, стрімінг музики, новини).

Отже, випливає логічний висновок: для реальних стрімінгових сервісів розумно інтегрувати різні моделі, використовуючи їхні відповідні сильні сторони та пом'якшуючи їхні недоліки. Такий підхід породжує клас методологій, які називаються гібридними системами рекомендацій[15,16,23].

2.2 Аналіз існуючих Гібридних Моделей та обґрунтування динамічного підходу

2.2.1 Типи гібридних рекомендаційних систем

Гібридні рекомендаційні системи поєднують кілька підходів, таких як CF, CBF, демографічні моделі, ML-моделі, графові алгоритми тощо, в єдину архітектуру. Метою гібридизації є підвищення точності прогнозів, стійкість до cold-start, збільшення різноманітності та адаптивності[15,16,22,23].

У літературі виділяють кілька основних типів гібридів (за Burke та подальшими оглядами)[16,22,23]:

1) Weighted Hybrid (зважене поєднання)

Кожна модель (CF, CBF тощо) повертає свій прогноз $\hat{r}_{u,i}^{k}$, а підсумковий рейтинг - це зважена сума:

$$\hat{r}_{u,i} = \sum_k \alpha_k \hat{r}_{u,i}^{k}, \ \sum_k \alpha_k = 1 . \qquad (2.7)$$

Ваги α_k зазвичай фіксовані або налаштовуються разово на валідаційних даних[16,23].

2) Switching Hybrid (перемикаючий гібрид)

Система не комбінує результати, а обирає одну з моделей залежно від стану:

- мало даних про користувача → використовуємо CBF;
- достатньо даних → CF;
- відсутні метадані → тільки CF тощо[16,22].

3) Mixed Hybrid (змішаний гібрид)

Користувачу одночасно показуються рекомендації з кількох джерел, наприклад:

- «рекомендовано для вас» (CF),
- «схоже на переглянуте» (CBF),
- «популярне зараз» (статистика чи тренди)[16,22].

4) Feature Combination (комбінація ознак)

Ознаки з різних джерел поєднуються в єдиний вектор, який подається на вхід однієї ML-моделі (логістична регресія, градієнтний бустинг, нейронна мережа). Наприклад:

$$x_{u,i} = concat(emb_{CF}(u,i), emb_{CBF}(i), behaviour\_features(u), context(t)),$$

а модель $f(x_{u,i})$ повертає ймовірність взаємодії[16,18,23].

5) Feature Augmentation (збагачення ознак)

Одна система генерує ознаки, наприклад контентний embedding, які потім використовуються як вхід для іншої (CF, графова модель тощо).

6) Cascade Hybrid (каскадний гібрид)

Робота відбувається в кілька етапів:

- 1-й етап - швидка модель (часто CF) відбирає десятки або сотні кандидатів;
- 2-й етап - більш складна модель (ML/DNN) ранжує цих кандидатів[16,18,23].

7) Meta-level Hybrid (метарівневий гібрид)

Модель, навчена одним методом (наприклад, CBF-профіль користувача), стає повноцінним входом для іншої моделі (CF або ML).

У Таблиці 2.3 представлено загальну характеристику ключових типів гібридних підходів, висвітлюючи їхню суть, основні переваги та потенційні недоліки.

Таблиця 2.3

Типи гібридних рекомендаційних систем

Тип гібриду	Суть	Переваги	Недоліки
Weighted	Лінійне поєднання кількох моделей	Простота, стабільність	Ваги часто статичні, не адаптивні
Switching	Вибір однієї моделі	Логічна адаптація до сценарію	Правила перемикання жорсткі, «ручні»
Mixed	Паралельне відображення кількох списків	Різноманітність	Не оптимізує загальну цільову функцію
Feature Combination	Єдина ML-модель над усіма ознаками	Висока гнучкість і точність	Складність фічерінгу, потреба в даних
Feature Augmentation	Одна модель генерує ознаки іншій	Покращує базові моделі	Зростання складності системи
Cascade	Послідовний пайплайн	Добре масштабується	Чутливість до якості першого етапу
Meta-level	Одна модель → повний вхід для іншої	Сильне поєднання знань	Важко інтерпретувати, складно обслуговувати

2.2.2 Гібридні моделі на основі глибинного навчання

Сучасні промислові рекомендаційні системи, зокрема ті, що використовуються в Google, YouTube, TikTok та Meta, активно використовують глибокі нейронні мережі як основу для гібридних моделей[18,19,21]. Можна виділити кілька основних архітектурних підходів.

Two-Tower (dual encoder) архітектура

Two-Tower-моделі складаються з двох нейронних веж:

1) User tower - нейромережа, що будує embedding користувача на основі:

- його історії взаємодій;
- демографічних ознак;
- пристрою, часу, контексту.

2) Item tower - нейромережа, що будує embedding елемента:

- жанри, теги, тривалість;
- стилі, актори, ембеддинги аудіо/відео.

Схожість між користувачем і елементом визначається як:

$$score(u, i) = \langle f_{user}(u), f_{item}(i) \rangle. \quad (2.8)$$

Ці моделі добре масштабується, а саме embeddings можна індексувати і швидко шукати схожі елементи[18,19].

DNN-классифікатор з softmax

Ще один підхід - розглядати рекомендацію як задачу багатокласової класифікації: для пари «користувач - контекст» модель прогнозує розподіл ймовірності по всіх елементах каталогу за допомогою softmax. Через обчислювальну складність використовуються різні варіанти, як:

- sampled softmax,
- негативний семплінг,
- ієрархічний softmax тощо[18,19,21].

DeepFM, Wide & Deep та інші гібридні ML-архітектури

Архітектури на кшталт DeepFM або Wide & Deep поєднують:

- «широку частину» (лінійні моделі, факторизаційні машини для взаємодій ознак),
- «глибоку частину» (нейронна мережа для складних нелінійних взаємодій).

Це по суті також гібрид: поєднання лінійних/факторизаційних моделей з глибинним навчанням.

Переваги ML-гібридів[18,19,21]:

- Можуть одночасно враховувати контентні, поведінкові, демографічні та контекстні ознаки.
- Підходять для двоступеневої архітектури «candidate generation → ranking».
- Дають високу якість рекомендацій у промислових масштабах.

Обмеження ML-гібридів[18]:

- Потребують дуже великих обсягів даних і потужних обчислювальних ресурсів.
- Чутливі до якості фічерінгу та налаштування гіперпараметрів.
- Часто оптимізують статичну цільову функцію (CTR, watch time) без явного урахування довготривалого утримання, новизни або різноманітності.
- Не завжди адаптуються до швидких змін смаків конкретного користувача без складних онлайнових механізмів перенавчання[18].

2.2.3 Обмеження існуючих гібридних моделей

Попри значний прогрес, більшість гібридних систем досі мають спільні слабкі місця, особливо якщо розглядати динамічні стрімінгові сервіси та мобільні застосунки[16,18,23,24].

1) Статичність комбінування

У зважених гібридах ваги a_k зазвичай фіксовані або змінюються дуже рідко. І саме це не дозволяє системі:

- адаптуватися до змін настрою користувача;
- враховувати різні сценарії використання, наприклад : вечірні перегляди, короткі сесії у дорозі тощо;
- підлаштовуватися під етап «знайомства» з новим контентом чи платформою[16,18,23,24].

2) Грубі правила перемикання

Switching-гібриди часто покладаються на прості правила («якщо історія < N → CBF»). Такий підхід:

- не враховує індивідуальні особливості користувачів;
- не враховує якість доступних моделей у конкретний момент;
- не оптимізує глобальну метрику, наприклад загальний watch time[16,23].

3) Неповне врахування контексту та сесійної динаміки

Навіть складні ML-гібриди часто працюють із узагальненими ознаками:

- статичний профіль користувача;
- усереднені фічі історії.

При цьому втрачається:

- сесійна структура (серії кліків/переглядів),
- короткострокові патерни («сьогодні слухає тільки подкасти», «зараз цікавить навчальний контент»),
- контекст (час доби, тип мережі, мобільний/настільний пристрій)[18,23].

4) Висока складність та ресурсомісткість

Великі DNN-гібриди:

- важко розгорнути на мобільних пристроях;
- потребують складної серверної інфраструктури для онлайн-навчання;
- у мобільному сценарії часто зводяться до спрощених евристик.

5) Неявне або відсутнє управління балансом CF/CBF/ML в реальному часі

Більшість гібридів не мають явної моделі, яка динамічно для кожного користувача і кожного моменту часу вираховує, наскільки важливими мають бути:

- колаборативний компонент (CF),
- контентний компонент (CBF),

- додатковий ML-компонент (наприклад, сесійна або контекстна модель).

Ці обмеження відкривають нішу для нових гібридних методів, у яких ваги між підходами адаптивно змінюються в залежності від стану користувача, історії його взаємодій, якості доступних сигналів та обмежень платформи (наприклад, мобільний пристрій)[18,23,24].

2.3 Висновки до розділу 2

У цьому розділі було проведено аналіз основних методів побудови персоналізованих рекомендацій у стрімінгових сервісах. Отже, підіб'ємо загальні підсумки.

1) Класичні підходи (CF та CBF):

- CBF ефективна для нових елементів і добре пояснюється, але створює вузьке інформаційне поле й залежить від якості метаданих[6-8,15,17].
- CF дозволяє виявляти приховані вподобання та формувати «сюрпризні» рекомендації, але страждає від cold-start і розрідженості даних[3-5,15,17,20].
- Жоден із підходів у «чистому» вигляді не задовольняє вимоги сучасних стрімінгових платформ щодо масштабованості, адаптивності та роботи в реальному часі[15-17].

2) Гібридні моделі:

- Класифікація гібридів (weighted, switching, mixed, cascade, meta-level тощо) демонструє широкий спектр способів комбінування CF, CBF і додаткових моделей[16,22,23].
- Сучасні ML-гібриди (Two-Tower, DNN-ranker, DeepFM) значно покращують точність та дають змогу враховувати комплексні ознаки користувача, контенту та контексту[18,19,21].

- Однак більшість цих підходів залишаються статичними з точки зору відношення між різними моделями та не дають механізму гнучкого управління внеском кожного компонента[16,18,23,24].

3) Актуальні обмеження:

- відсутність динамічної адаптації ваг між окремими рекомендаційними підходами для конкретного користувача;
- недостатнє врахування сесійної та часової динаміки інтересів;
- обмежена придатність частини методів до мобільного середовища з жорсткими вимогами до затримки й ресурсів;
- складність інтеграції таких рішень у реальні стрімінгові сервіси середнього масштабу.

4) Обґрунтування розробки нового методу:

На основі проведеного аналізу можна зробити висновок, що перспективним напрямом є розробка динамічного гібридного методу, який:

- поєднує сильні сторони CF, CBF та ML-моделей;
- використовує адаптивні вагові коефіцієнти, що залежать від стану користувача, якості даних та контексту;
- може бути реалізований з урахуванням обмежень мобільних пристроїв та сучасних ML-платформ (наприклад, TensorFlow Lite)[15,16,18,23,24].

У наступному розділі буде представлено методологію розробки такого динамічного гібридного підходу, побудованого на гібридних моделях машинного навчання, а також математичну модель, алгоритмічну реалізацію.

3. РОЗРОБКА ТА РЕАЛІЗАЦІЯ ДИНАМІЧНОГО ГІБРИДНОГО МЕТОДУ ПЕРСОНАЛІЗОВАНИХ РЕКОМЕНДАЦІЙ

3.1. Постановка задачі оптимізації гібридної моделі

У цьому розділі розроблено та формально описано гібридний метод персоналізованих рекомендацій, який поєднує класичні підходи (колаборативна та контентно-орієнтована фільтрація) з моделями машинного навчання для динамічного визначення ваг між ними. Мета методу - це оптимізація якості рекомендацій у стрімінгових сервісах за рахунок адаптивного керування внеском кожної складової моделі залежно від поведінки та контексту користувача.

Нехай:

$U = \{u_1, \ldots, u_M\}$ - множина користувачів;

$I = \{i_1, \ldots, i_N\}$ - множина елементів контенту (фільми, серіали, треки тощо);

t - момент часу або крок взаємодії користувача із системою;

$\mathcal{D}$ - множина доступних даних (історія переглядів, рейтинги, лайки, метадані контенту, контекстні ознаки).

Потрібно побудувати рекомендаційну функцію:

$$s(u, i, t) = F(u, i, t; \Theta), \qquad (3.1)$$

яка для кожного користувача u та моменту часу t формує впорядкований список елементів $i \in I$, максимізуючи певну цільову функцію якості (наприклад, CTR, Precision@K, NDCG@K, середній час перегляду)[17].

Особливістю пропонованого підходу є те, що функція F не є монолітною моделлю, а є гібридом кількох базових алгоритмів рекомендацій, зокрема:

- колаборативної фільтрації (CF),
- контентно-орієнтованої фільтрації (CBF),
- за потреби - додаткових ML-моделей (наприклад, сесійної або контекстної).

Завдання оптимізації полягає в тому, щоб:

1. Навчити базові моделі (CF, CBF) так, щоб вони давали адекватні часткові прогнози.

2. Розробити механізм **динамічного визначення ваг** цих моделей для кожного користувача й кожного моменту часу.

3. Забезпечити можливість реалізації цього механізму в умовах стрімінгового сервісу, зокрема у мобільному середовищі[10,28-30].

3.2. Математична модель гібридних рекомендацій

Розглянемо два базові алгоритми:

- $f_{CF}(u,i,t)$ - модель колаборативної фільтрації;
- $f_{CB}(u,i,t)$- контентно-орієнтована модель (content-based).

Нехай вони повертають **нормовані оцінки релевантності** в інтервалі [0,1]:

$$s_{CF}(u,i,t) = f_{CF}(u,i,t) \in [0,1], \quad (3.2)$$

$$s_{CB}(u,i,t) = f_{CB}(u,i,t) \in [0,1]. \quad (3.3)$$

Гібридний скор (підсумкова оцінка релевантності) визначимо як зважену комбінацію:

$$s_{hyb}(u,i,t) = w_{CF}(u,t) \times s_{CF}(u,i,t) + w_{CB}(u,t) \times s_{CB}(u,i,t), \quad (3.4)$$

де

$$w_{CF}(u,t) \geq 0,\ w_{CB}(u,t) \geq 0,\ w_{CF}(u,t) + w_{CB}(u,t) = 1. \quad (3.5)$$

Ключова відмінність запропонованого методу від класичних weighted hybrid-підходів полягає в тому, що ваги

$w_{CF}(u,t)$ та $w_{CB}(u,t)$ **не є сталими, а динамічно обчислюються** на основі поведінкових та контекстних факторів[16,23,24].

Позначимо через $z_{u,t}$ вектор ознак, що описує стан користувача u у момент часу t. До нього можуть входити:

- кількість взаємодій користувача ($|I_u|$);
- свіжість останніх переглядів;
- частка контенту, який краще передбачає CF або CBF;
- тип пристрою, час доби, локальний контекст;
- показники якості попередніх рекомендацій (клік, перегляд до кінця тощо).

Тоді ваги задаються як результат роботи **моделі ваг:**

$$w_{CF}(u,t),\ w_{CB}(u,t) = g(z_{u,t}; \Theta_w), \quad (3.6)$$

де $g(\cdot)$ - ML-модель (логістична регресія, градієнтний бустинг, невелика нейромережа), θ_w - її параметри[31].

Для зручності можна використовувати softmax-параметризацію:

$$\widetilde{w}_{CF} = h_{CF}(z_{u,t}),\ \widetilde{w}_{CB} = h_{CB}(z_{u,t}), \quad (3.7)$$

$$w_{CF}(u,t) = \frac{exp(\widetilde{w}_{CF})}{exp(\widetilde{w}_{CF}) + exp(\widetilde{w}_{CB})}, \quad (3.8)$$

$$w_{CB}(u,t) = \frac{exp(\widetilde{w}_{CB})}{exp(\widetilde{w}_{CF}) + exp(\widetilde{w}_{CB})}. \quad (3.9)$$

У такий спосіб завжди гарантується умова $w_{CF} + w_{CB} = 1$ та невід'ємність ваг.

Ціль навчання всієї гібридної моделі - мінімізувати втрати на історичних даних:

$$L(\Theta_{CF}, \Theta_{CB}, \Theta_w) = \sum_{(u,i,t)\in D} l(y_{u,i,t}, s_{hyb}(u,i,t)) + \Omega(\Theta_{CF}, \Theta_{CB}, \Theta_w), \quad (3.10)$$

де

$y_{u,i,t}$ - фактична реакція (клік, рейтинг, перегляд),

ℓ - функція втрат (наприклад, log-loss або MSE),

Ω - регуляризація.

3.3. Метод динамічного визначення ваг

Запропонований **метод динамічного визначення ваг** полягає в тому, що для кожного користувача і кожного сеансу роботи з системою:

1) Визначається **набір ознак** $z_{u,t}$, що описують поточний стан.

2) Оцінюється **локальна "релевантність"** кожної базової моделі (CF та CBF) з урахуванням:

- наявності/якості даних для CF (кількість сусідів, розрідженість);
- інформативності контентних ознак для CBF;
- успішності попередніх рекомендацій кожної моделі.

3) На основі цих факторів модель ваг $g(z_{u,t})$ обчислює $w_{CF}(u,t)$ та $w_{CB}(u,t)$.

3.3.1. Формування вектора ознак $z_{u,t}$

Приклад складу $z_{u,t}$:

- $n_{int}(u)$ - кількість взаємодій користувача (розмір історії);
- $d_{last}(u)$ - час від останньої активності;
- $p_{cold}(u)$ - індикатор «холодності» користувача (наприклад, $p_{cold}(u) = 1$, якщо $n_{int}(u) < N_{min}$);

- $q_{CF}(u,t)$ - якість попередніх рекомендацій CF (наприклад, середній hit-rate за останні K рекомендацій);
- $q_{CB}(u,t)$ - аналогічний показник для CBF;
- контекстні ознаки: час доби (ранок/день/вечір/ніч), тип пристрою (мобільний/ПК), тип мережі (Wi-Fi/мобільні дані).

Таким чином:

$$z_{u,t} = \left[n_{int}(u), d_{last}(u), q_{CF}(u,t), q_{CB}(u,t), context(t)\right]. \quad (3.11)$$

3.3.2. Модель ваг $g(\cdot)$ та навчання

У практичній реалізації в якості $g(\cdot)$ може використовуватися[31]:

- логістична регресія;
- дерево рішень/градієнтний бустинг (наприклад, XGBoost);
- невелика повнозв'язна нейромережа (MLP), яку надалі можна конвертувати в TensorFlow Lite для мобільного пристрою[10,28-30].

Навчання здійснюється на логах взаємодій:

1) Для кожної рекомендації з минулого логуються:

- s_{CF}, s_{CB};
- $z_{u,t}$;
- фактичний результат $y_{u,i,t}$ (клік / неклік, рейтинг тощо).

2) Оптимізується Θ_w так, щоб гібридний скор s_{hyb} максимально узгоджувався з $y_{u,i,t}$ (максимізація likelihood або мінімізація втрат).

Таким чином, система **вчиться сама**, у яких ситуаціях довіряти більше CF, а в яких - CBF.

3.3.3. Оновлення ваг у режимі онлайн

Окрім «офлайнового» навчання Θ_w, можливе **онлайнове оновлення ваг** на основі спостережень за останніми взаємодіями:

$$w_{CF}(u, t+1) = (1-\eta) w_{CF}(u, t) + \eta \times r_{CF}(u, t), \quad (3.12)$$

$$w_{CB}(u, t+1) = (1-\eta) w_{CB}(u, t) + \eta \times r_{CB}(u, t), \quad (3.13)$$

де

η - швидкість оновлення (learning rate),

$r_{CF}(u, t)$, $r_{CB}(u, t)$ - показники «успішності» рекомендацій CF та CBF у поточній взаємодії (наприклад, 1 - якщо рекомендація CF спрацювала краще, 0 - інакше).

Після оновлення ваги нормуються так, щоб $w_{CF} + w_{CB} = 1$.

Такий механізм дозволяє алгоритму **адаптуватися на льоту** до конкретного користувача.

3.4. Алгоритм роботи запропонованого методу

3.4.1. Послідовність кроків роботи динамічного гібридного методу

Роботу запропонованого методу можна описати за послідовністю таких кроків (узагальнений вигляд подано на рисунку 3.1).

1) Отримання даних користувача

Система ідентифікує користувача u, потім отримує його історію взаємодій, дані про попередні рейтинги/перегляди, а також поточний контекст, наприклад: час, тип пристрою тощо.

2) Обчислення рекомендацій CF та CBF

- Модель колоборативної фільтрації CF (наприклад, матрична факторизація або SVD[9,17]) обчислює скор $s_{CF}(u,i,t)$ для кандидатів i.
- Модель контентно-орієнтованої фільтрації CBF обчислює для кожного кандидата скор $s_{CB}(u,i,t)$ беручи за основу профіль користувача та ознак контенту,як жанр, теги тощо.

3) Збір поведінкових та контекстних факторів

На основі історії та поточного контексту формується вектор ознак $z_{u,t}$:

- кількість взаємодій,
- показники якості попередніх рекомендацій(CF/CBF),
- контекстні параметри.

4) Обчислення релевантності базових моделей

Модель $g(z_{u,t})$ обчислює «сирі» вагові логіти $\widetilde{w}_{CF}$, $\widetilde{w}_{CB}$, які потім перетворюються в нормовані ваги w_{CF}, w_{CB}.

5) Нормалізація ваг

Переконуємося, що $w_{CF} + w_{CB} = 1$ за допомогою softmax або нормування вручну.

6) Формування гібридної рекомендації

Для кожного кандидата i обчислюється гібридна оцінка релевантності:

$$s_{hyb}(u,i,t) = w_{CF} \times s_{CF}(u,i,t) + w_{CB} \times s_{CB}(u,i,t), \quad (3.14)$$

після чого елементи сортуються за спаданнямs_{hyb}, і формується топ-N рекомендацій, який повертається користувачу.

7) Оновлення ваг при кожній взаємодії

Після того як користувач взаємодіє з рекомендованими елементами, тобто подивився, пропустив, поставив рейтинг, оновлюються:

- показники q_{CF}, q_{CB},
- онлайн-ваги $w_{CF}(u, t+1)$,$w_{CB}(u, t+1)$,
- за необхідності - «офлайново» донавчається модель $g(\cdot)$.

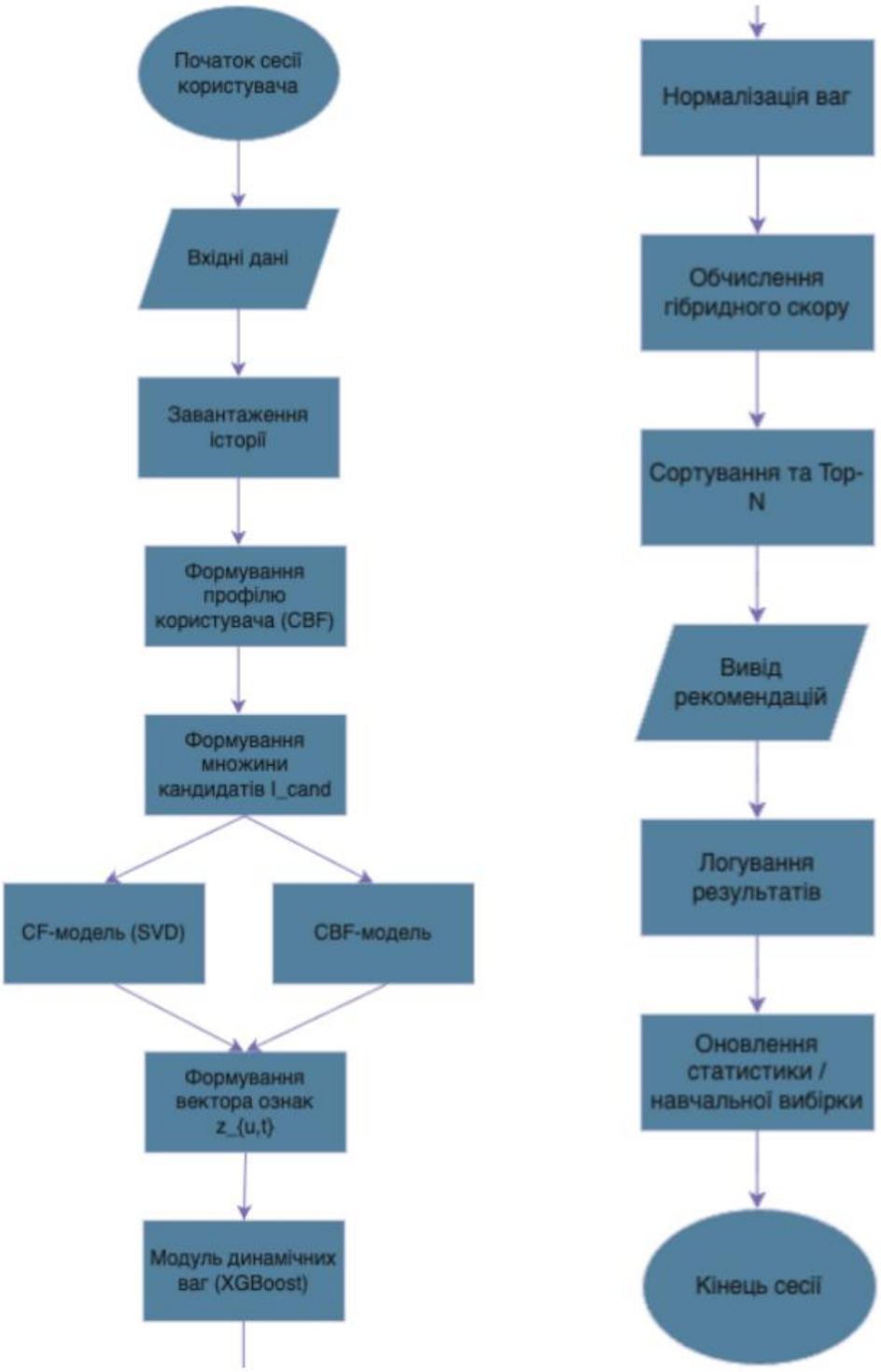


рис. 3.1 Блок-схема роботи запропонованого гібридного методу рекомендацій

3.4.2. Діаграма діяльності та потоків даних із плавальними доріжками

На рисунку 3.2 продемонстровано діаграму діяльності з «плавальними доріжками» (swimlanes), яка деталізує обмін даними між основними компонентами системи:

- клієнтським застосунком (Client App),
- сервісом рекомендацій (Recommender Service),
- модулем динамічних ваг (Dynamic Weight Module).

Кожна доріжка відповідає окремому учаснику процесу, а стрілки показують послідовність викликів та напрямок передачі даних.

Основні етапи, що відображені на діаграмі, такі.

1) Ініціація запиту клієнтським застосунком.

Користувач відкриває стрічку рекомендацій. Клієнтський застосунок формує запит до Recommender Service, передаючи ідентифікатор користувача та контекст (userId, context). Це стартова подія на діаграмі.

2) Внутрішня обробка запиту сервісом рекомендацій.

У доріжці Recommender Service послідовно виконуються дії:

- завантаження історії користувача;
- обчислення колаборативних скорів CF для набору кандидатів;
- обчислення контентно-орієнтованих скорів CBF;
- формування множини кандидатів для подальшого гібридного ранжування.

3) Запит до модуля динамічних ваг.

Далі Recommender Service синхронно звертається до Dynamic Weight Module, передаючи мета-ознаки (активність користувача, популярність фільмів, попередні вподобання, час доби, тип пристрою тощо).

Мета запиту - отримати актуальні значення ваг w_{CF} та w_{CB} для конкретного користувача і поточної сесії.

4) Нормальний сценарій: модуль ваг працює коректно.

У гілці «Dynamic Weight Module healthy» модуль:

- на своїй доріжці обчислює ваги за допомогою навченої ML-моделі (наприклад, XGBoost);
- повертає Recommender Service пару (w_{CF}, w_{CB}).

5) Резервний сценарій (fallback): помилка або тайм-аут модуля ваг.

Якщо модуль ваг недоступний або не відповідає вчасно, активується альтернативна гілка «Module error or timeout»:

- Dynamic Weight Module повертає помилку або не повертає відповідь;
- у доріжці Recommender Service запускається блок «Fallback handling», який:
  - використовує заздалегідь визначені статичні ваги (наприклад $w_{CF} = 0{,}7, w_{CB} = 0{,}3$) для того, щоб все одно сформувати список рекомендацій;
  - логує інформацію про помилку для подальшого аналізу;
  - оновлює статус «здоров'я» (health status) Dynamic Weight Module у системі моніторингу, що дозволяє оперативно виявляти проблеми.

Таким чином, сервіс рекомендацій залишається працездатним, навіть якщо модуль динамічних ваг тимчасово не працює.

6) Обчислення гібридного скору і повернення результату клієнту.

Отримавши або динамічні, або резервні статичні ваги, Recommender Service:

- нормалізує їх так, щоб $w_{CF} + w_{CB} = 1$;
- обчислює гібридні скори для всіх кандидатів;
- відбирає топ-N елементів і повертає їх до клієнтського застосунку у вигляді готового списку рекомендацій.

7) Фіксація взаємодій користувача.

Після відображення рекомендацій користувач взаємодіє з ними (клік, перегляд, рейтинг). Клієнтський застосунок надсилає ці дії назад на Recommender Service.

Сервіс:

- логує нові взаємодії (click, rating);
- формує записи з новими ознаками і відправляє їх до Dynamic Weight Module як дані для офлайнового донавчання.

8) Асинхронне офлайн-донавчання моделі ваг.

В окремій опціональній гілці «Offline retraining» показано, що Dynamic Weight Module періодично оновлює навчальний набір і перенавчає модель ваг в асинхронному режимі. Це не блокує онлайн-запити, але дозволяє поступово підвищувати якість динамічного зважування на основі все нових логів.

Таким чином, діаграма діяльності (рисунок 3.2) наочно показує, як саме модуль динамічних ваг інтегрується у загальний пайплайн рекомендацій, яким чином обробляються помилки та як відбувається замкнене коло «рекомендації → фідбек → донавчання», що забезпечує адаптивність запропонованого методу.

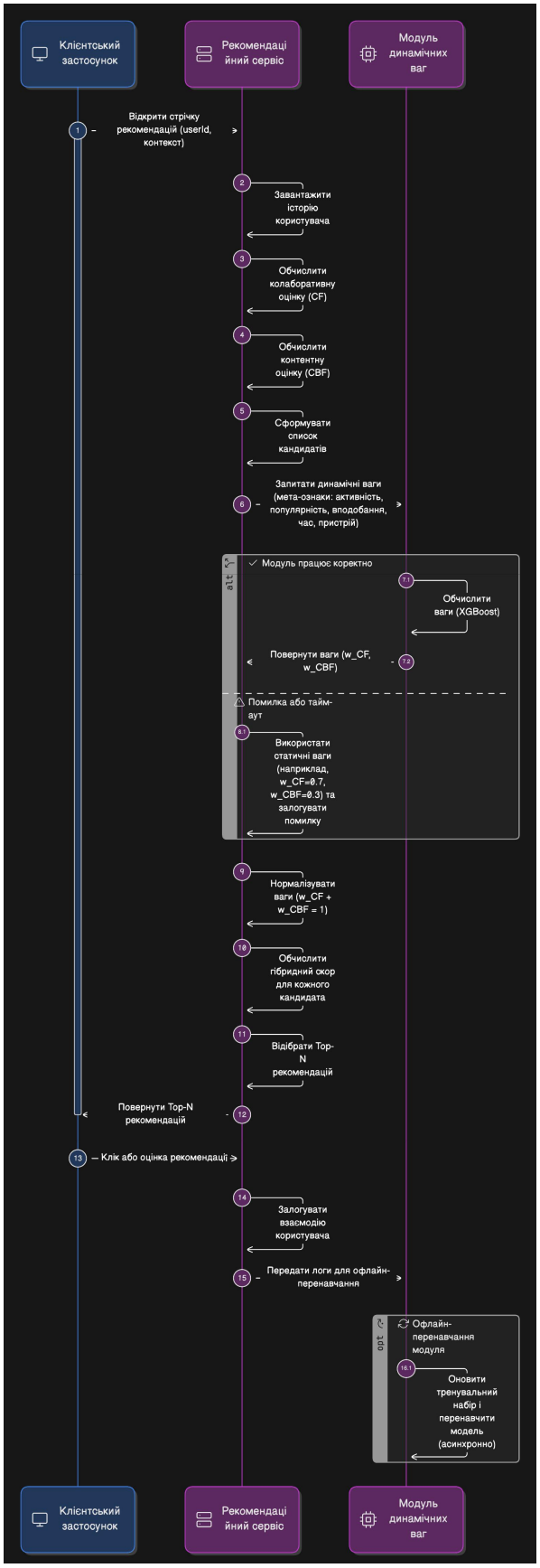


рис. 3.2 Діаграма діяльності взаємодії клієнтського застосунку, сервісу рекомендацій та модуля динамічних ваг

3.5. Відповідність запропонованого методу темі роботи

Запропонований підхід безпосередньо відповідає темі «Оптимізація алгоритмів персоналізованих рекомендацій у стрімінгових сервісах на основі гібридних моделей машинного навчання»:

- Використовуються **гібридні моделі ML:**
  - CF (наприклад, матрична факторизація, нейронні CF-моделі[9,17]),
  - CBF (моделі з ознаками контенту),
  - метамодель ваг (градієнтний бустинг або нейронна мережа[31]).
- **Оптимізація** досягається не лише покращенням кожної окремої моделі, а **саме через динамічне комбінування** їхніх результатів у залежності від конкретного користувача, його поведінки та контексту використання.

Метод може бути реалізований у **мобільному середовищі,** де:

- CF/CBF-моделі можуть працювати на сервері;
- метамодель ваг у компактній формі (наприклад, невелика нейромережа) може бути розгорнута через TensorFlow Lite на мобільному пристрої для швидкого inference[10, 28-30].

4. ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

4.1. Методика експериментального дослідження

4.1.1. Вихідні дані та попередня обробка

Для оцінки якості запропонованого гібридного методу було використано відкритий датасет **MovieLens 25M**, що містить близько 25 млн оцінок фільмів користувачами[25-27]. У рамках роботи датасет було попередньо очищено[25,26]:

- видалено користувачів з кількістю оцінок менше 5;
- видалено фільми з кількістю оцінок менше 5;
- видалено технічне поле timestamp.

Після фільтрації залишилося **24 945 870** оцінок і **32 720** фільмів, для яких були доступні текстові жанрові мітки[27].

Для подальшого використання в контентно-орієнтованій моделі було виконано **one-hot кодування жанрів**: для кожного жанру (усього 20 унікальних жанрів) створювалася окрема бінарна ознака. На основі цього сформовано:

- **матрицю ознак елементів** (item features matrix) розмірності 32 720 × 20, де кожний рядок відповідає фільму, а стовпці - жанрам;
- **профілі користувачів** (user profiles), що обчислювалися як зважене середнє жанрових векторів переглянутих фільмів з вагами, пропорційними рейтингам користувача.

Таким чином, контентна частина моделі працює лише з інформацією про жанри фільмів та їхніми рейтингами.

4.1.2. Базові моделі рекомендацій

У дослідженні порівнювалися чотири моделі:

1) **CF (Collaborative Filtering)** - модель колаборативної фільтрації на основі **матричної факторизації SVD** з бібліотеки surprise[9,17].

Параметри SVD:

- n_factors = 100 (кількість явних факторів),
- n_epochs = 12,
- lr_all = 0.007,
- reg_all = 0.02,
- random_state = 42.

Модель навчалась на повному тренувальному наборі (усі наявні оцінки після фільтрації). Для пари (u, i) вона повертає прогнозований рейтинг $\hat{r}_{CF}(u, i)$ в діапазоні $[0.5; 5.0]$.

2) **CBF (Content-Based Filtering)** - контентно-орієнтована модель, заснована на косинусній подібності між профілем користувача та вектором жанрів фільму[6-8,17].

Для кожного користувача u будується вектор профілю $\vec{p}_u$, отриманий як зважене середнє one-hot жанрових векторів фільмів, які він оцінив, з вагами, пропорційними рейтингам. Для фільму i використовується вектор жанрів $\vec{x}_i$. Подібність:

$$\mathrm{sim}(u, i) = \frac{\vec{p}_u \times \vec{x}_i}{\|\vec{p}_u\| \times \|\vec{x}_i\|}. \qquad (4.1)$$

Щоб привести результат до шкали рейтингу, використано лінійне перетворення:

$$\hat{r}_{CB}(u, i) = clip(1 + 4 \times sim(u, i), 0.5, 5.0). \qquad (4.2)$$

3) **Hybrid-Simple** - простий (статичний) гібрид CF та CBF з фіксованими вагами:

$$\hat{r}_{Simple}(u,i) = \alpha \times \hat{r}_{CF}(u,i) + \beta \times \hat{r}_{CB}(u,i), \qquad (4.3)$$

де в експериментах використано $\alpha = 0.6, \beta = 0.4$. Ці коефіцієнти не змінюються в часі та не залежать від користувача.

4) **Hybrid-XGB (запропонований динамічний гібрид)** - вдосконалена гібридна модель, у якій остаточний рейтинг прогнозується за допомогою метамоделі XGBoost, що динамічно комбінує результати CF та CBF з урахуванням додаткових ознак[31].

4.1.3. Формування мета-ознак для динамічної гібридної моделі

Запропонований метод динамічного визначення ваг реалізовано через навчання **регресійної моделі XGBoost (метамоделі)**, яка на вході отримує:

- R_{CF}- прогноз CF-моделі для пари (u, i);
- R_{CB}- прогноз CBF-моделі;
- Rel_{CF}- відносна “надійність” CF для користувача:

$$Rel_{CF}(u) = \alpha \times \frac{\textit{кількість оцінок користувача } u}{\textit{загальна кількість фільмів}}, \qquad (4.4)$$

- Rel_{CB}- відносна “надійність” CBF для фільму:

$$Rel_{CB}(i) = \beta \times \left(1 - \frac{\textit{кількість оцінок фільму } i}{\textit{максимальна популярність у каталозі}}\right). \qquad (4.5)$$

Інтуїтивно:

- CF більше довіряємо для активних користувачів (багато оцінок);
- CBF - для менш популярних фільмів, де колаборативний сигнал слабкий.

Для навчання метамоделі було відібрано випадкову підвибірку 50 000 записів (user-movie-rating) з датасету. Для кожного запису обчислювалися вищенаведені 4 ознаки, а цільовою змінною був фактичний рейтинг R_{actual} . Отримано навчальний набір:

- $X_{meta} \in \mathbb{R}^{50000\times4}$,

- $y_{meta} = R_{actual}$.

Для моделі XGBoost використовували параметри[31]:

n_estimators = 300,

max_depth = 6,

learning_rate = 0.07,

tree_method = 'hist',

subsample = 0.8,

colsample_bytree = 0.8,

objective = 'reg:squarederror'.

Таким чином, метамодель навчається **не просто як ще один предиктор рейтингу**, а як оптимальний комбінатор базових моделей з урахуванням структури даних (активність користувача, популярність фільму).

4.1.4. Тестовий вибірковий набір та метрики оцінювання

Для об'єктивного порівняння якості моделей сформовано **тестову підвибірку** розміром **20 000 записів** (user-movie-rating), випадково обрану з усього датасету після фільтрації. Для кожної пари (u, i) обчислювалися прогнози:

- $\hat{r}_{CF}$ - CF;
- $\hat{r}_{CB}$ - CBF;
- $\hat{r}_{Simple}$ - простий гібрид;
- $\hat{r}_{Hybrid-XGB}$ - запропонована гібридна модель;

Для оцінки якості використовувалися такі метрики[17]:

- **MAE (Mean Absolute Error)** - середня абсолютна похибка:

$$\mathrm{MAE} = \frac{1}{N}\sum_{k=1}^{N}\left|r_k - \hat{r}_k\right|. \qquad (4.6)$$

- **RMSE (Root Mean Squared Error)** - корінь із середньоквадратичної похибки:

$$\mathrm{RMSE} = \sqrt{\frac{1}{\mathrm{N}}\sum_{\mathrm{k}=1}^{\mathrm{N}}\left(\mathrm{r_k} - \hat{\mathrm{r}}_k\right)^2}. \qquad (4.7)$$

- **Accuracy (класифікаційна точність)** - частка правильно класифікованих прикладів при бінаризації рейтингів за порогом:

  $\mathrm{r_k} \geq 4.0 \rightarrow$ позитивний клас, $\mathrm{r_k} < 4.0 \rightarrow$ негативний.

- **F1-score** - гармонійне середнє між precision та recall для позитивного класу.

Таким чином, MAE та RMSE вимірюють **якість регресійного прогнозу рейтингу**, а Accuracy і F1 - **якість класифікації "подобається/не подобається"**, що більш інтерпретовано з точки зору практики стрімінгових сервісів[17].

4.2. Результати експериментів та їх аналіз

4.2.1. Порівняння якості базових та гібридних моделей

У таблиці 4.1 наведені результати порівняння чотирьох моделей: CF, CBF, простого гібриду (Hybrid-Simple) та запропонованої гібридної метамоделі (Hybrid-XGB).

Таблиця 4.1

Порівняння якості моделей на тестовій вибірці (N = 20 000)

Модель	MAE	RMSE	Accuracy	F1-score
CF	0.5321	0.6985	0.7129	0.6300
CBF	1.7037	1.9524	0.4945	0.0008
Hybrid-Simple	0.8669	1.0288	0.5013	0.0281
Hybrid-XGB	0.4855	0.6402	0.7539	0.7022

4.2.2. Аналіз результатів

1) **Колаборативна модель (CF)** демонструє досить хорошу якість:

- MAE ≈ 0.53, RMSE ≈ 0.70;
- Accuracy ≈ 0.713, F1 ≈ 0.63.

Це підтверджує, що матрична факторизація SVD є сильною базовою моделлю для рейтингів у MovieLens[9,17].

2) **Контентно-орієнтована модель (CBF)** показала значно гірші результати:

- MAE > 1.7, RMSE ≈ 1.95;
- Accuracy ≈ 0.495 (практично на рівні випадкового вгадування за порогом 4.0);
- F1 ≈ 0.0008 (майже повна відсутність коректних позитивних прогнозів).

Це пояснюється тим, що CBF у даній реалізації працює лише з **жанровими мітками**, без додаткових ознак (опис, актори, рік, ембеддинги тощо). Тобто сигнал від контенту є дуже грубим і недостатнім для точного передбачення конкретного числового рейтингу.

3) **Простий гібрид (Hybrid-Simple)**, що лінійно комбінує CF і CBF з фіксованими вагами, **не покращує**, а навпаки **погіршує** результати порівняно з чистим CF:

- MAE зростає до ≈ 0.87,
- RMSE - до ≈ 1.03,
- Accuracy ≈ 0.50,
- F1 ≈ 0.03.

Якщо одна з базових моделей (CBF) має низьку якість, то **наївне додавання її прогнозів «у середньому» погіршує гібрид**. Статична гібридизація без урахування якості складових не є оптимізацією, а може стати деградацією[16,23,24].

4) **Запропонована модель Hybrid-XGB** демонструє **найкращі результати серед усіх порівнюваних моделей**:

- MAE зменшується до 0.4855 (проти 0.5321 у CF);
- RMSE - до 0.6402 (проти 0.6985 у CF);
- Accuracy зростає до 0.7539 (проти 0.7129 у CF);
- F1-score - до 0.7022 (проти 0.6300 у CF).

Якщо порівнювати з CF:

- MAE зменшено приблизно на 8.7 %;
- RMSE - приблизно на 8.3 %;
- Accuracy зросла на ≈ 4.1 відсоткових пункти (з 71.3 % до 75.4 %);
- F1-score - на ≈ 7.2 в.п. (з 0.63 до 0.70).

Тобто, навіть за умови, що контентна модель CBF сама по собі є слабкою, запропонований мета-гібрид **уміє «правильно її використовувати»**, знижуючи її вплив там, де вона шкодить, і підсилюючи - там, де сигнал корисний (наприклад, для менш популярних фільмів)[24].

4.2.3. Інтерпретація результатів з точки зору оптимізації гібридної моделі

Отримані результати експериментів дозволяють зробити кілька важливих висновків:

1) **Проста гібридизація ≠ оптимізація**. Статичне зважене усереднення (Hybrid-Simple) не враховує ні якості сигналів від CF і CBF, ні контексту/характеристик користувача, що в даному випадку призводить до відчутного погіршення результатів.

2) **Динамічне, ML-обґрунтоване комбінування моделей дає відчутний виграш**. Hybrid-XGB використовує метамодель машинного навчання для:

- врахування активності користувача (наскільки надійні CF-прогнози);
- врахування популярності фільму (де CBF може бути кориснішим);

Це і є **реалізація методу динамічного визначення ваг** у гібридній моделі.

3) **Оптимізація алгоритмів рекомендацій відбувається не тільки за рахунок «кращого CF» чи «кращого CBF»**, а через:

- побудову **метарівневої ML-моделі**, що оптимально комбінує результати базових алгоритмів;
- використання додаткової інформації (активність користувачів, популярність фільмів) в якості ознак.

Таким чином, експериментально підтверджено, що запропонований підхід:

- відповідає темі роботи **«Оптимізація алгоритмів персоналізованих рекомендацій у стрімінгових сервісах на основі гібридних моделей машинного навчання»**;
- забезпечує кількісно кращі результати, ніж класична CF-модель та прості схеми гібридизації;
- демонструє практичну доцільність застосування **динамічного методу визначення ваг** у гібридних рекомендаційних системах.

ВИСНОВКИ

Магістерська робота присвячена проблемі оптимізації алгоритмів персоналізованих рекомендацій у стрімінгових сервісах з використанням гібридних моделей машинного навчання. Актуальність теми зумовлена стрімким зростанням обсягів мультимедійного контенту, конкуренцією між платформами за увагу користувача та необхідністю забезпечення високого рівня персоналізації при обмежених обчислювальних ресурсах, зокрема у мобільних застосунках.

У вступі сформульовано мету роботи, яка полягає у розробці та дослідженні ефективності методу динамічного коригування вагових коефіцієнтів у гібридній рекомендаційній моделі для стрімінгових сервісів з метою підвищення точності та адаптивності рекомендацій, а також частково подолання проблеми «холодного старту». Для досягнення цієї мети були визначені й послідовно вирішені такі основні завдання:

- проаналізувати існуючі підходи до персоналізованих рекомендацій у стрімінгових сервісах;
- дослідити можливості використання машинного навчання та мобільних ML-платформ для покращення персоналізації;
- розробити математичну модель гібридної системи рекомендацій із динамічним визначенням ваг між базовими алгоритмами;
- реалізувати запропонований метод програмно та провести експериментальну оцінку його ефективності на реальному датасеті.

У першому розділі було розглянуто теоретичні основи персоналізації контенту, класифікацію рекомендаційних систем та особливості їх застосування у стрімінгових сервісах. Окрему увагу приділено аналізу ролі персоналізованих рекомендацій у підвищенні залученості та утриманні користувачів, а також огляду сучасних технологічних рішень для

впровадження машинного навчання у мобільних застосунках, зокрема TensorFlow Lite та Firebase ML Kit.

У другому розділі виконано детальний аналіз класичних моделей фільтрації - контентно-орієнтованої (CBF) та колаборативної (CF). Показано їх математичний апарат, принципи формування профілів користувачів, обчислення схожості та прогнозування рейтингів. На основі порівняльного аналізу виявлено сильні й слабкі сторони кожного підходу: CBF добре працює з новим контентом і забезпечує пояснюваність, тоді як CF дає змогу виявляти приховані закономірності та формувати «сюрпризні» рекомендації, але потерпає від проблеми cold-start і розрідженості даних.

Далі у другому розділі розглянуто гібридні рекомендаційні системи, наведено їх класифікацію (weighted, switching, mixed, cascade, feature combination, meta-level тощо), а також сучасні ML-орієнтовані гібриди на основі глибинних нейронних мереж (Two-Tower, DNN-ранжувальники, DeepFM). Показано, що, попри значний прогрес, більшість існуючих рішень залишаються статичними щодо комбінування базових моделей, слабо враховують сесійний контекст і динаміку змін інтересів та часто потребують значних обчислювальних ресурсів. На цій основі обґрунтовано необхідність розробки динамічного гібридного методу, здатного адаптивно змінювати ваги між CF та CBF залежно від стану користувача та характеристик контенту.

У третьому розділі було реалізовано основну наукову новизну роботи. Запропоновано математичну модель гібридних рекомендацій, у якій підсумковий рейтинг формується як зважена комбінація оцінок, отриманих від колаборативної та контентної моделей. На відміну від класичних схем із фіксованими вагами, у роботі розроблено метод динамічного визначення ваг, де вагові коефіцієнти не задаються вручну, а визначаються за допомогою окремої метамоделі машинного навчання (XGBoost), яка працює з набором мета-ознак, що характеризують активність користувача, популярність фільму та якість прогнозів базових моделей.

Було розроблено алгоритм роботи запропонованого методу, що включає: отримання даних користувача, обчислення рекомендацій CF та CBF, формування поведінкових та контекстних ознак, обчислення ваг, побудову гібридного рейтингу та оновлення ваг за результатами реакції користувача. На базі відкритого датасету MovieLens 25M реалізовано прототип системи, що включає: колаборативну модель SVD, контентно-орієнтовану модель на основі жанрових ознак, простий статичний гібрид CF+CBF та запропонований динамічний Hybrid-XGB.

Експериментальні результати показали, що:

- контентна модель CBF у спрощеній конфігурації (лише жанри) істотно поступається за точністю колаборативній моделі;
- простий статичний гібрид CF+CBF не тільки не покращує якість, а й погіршує її через неконтрольований вплив слабшої моделі;
- запропонована динамічна гібридна модель Hybrid-XGB демонструє найкращі показники MAE, RMSE, Accuracy та F1-score, перевершуючи чисту CF-модель і статичний гібрид.

Це підтверджує, що оптимізація алгоритмів персоналізованих рекомендацій може бути досягнута шляхом метарівневого комбінування моделей та адаптивного керування вагами між ними, а не лише покращенням окремих компонент.

Наукова новизна роботи полягає у поєднанні:

- гібридного підходу (CF + CBF);
- метамоделі машинного навчання для динамічного визначення ваг;
- практично орієнтованої реалізації та експериментів на великому реальному датасеті.

Практичне значення одержаних результатів полягає в тому, що запропонований метод може бути інтегрований у реальні стрімінгові сервіси як окремий модуль ранжування, а також адаптований для роботи в мобільних застосунках з використанням TensorFlow Lite та суміжних платформ. Це

дозволяє підвищити якість персоналізованих рекомендацій без необхідності радикального перегляду всієї інфраструктури рекомендаційної системи.

Подальші дослідження можуть бути спрямовані на:

- розширення набору контентних ознак (опис, актори, візуальні та аудіо-ембеддинги);
- включення сесійних і послідовних моделей (RNN/Transformer) для врахування короткострокової динаміки;
- перенесення метамоделі на мобільний пристрій та експериментальну перевірку її роботи в умовах реального мобільного застосунку;
- оптимізацію методу з точки зору обчислювальної складності та енергоспоживання на користувацьких пристроях.

У підсумку, робота демонструє, що гібридні моделі машинного навчання з динамічним визначенням ваг є перспективним напрямом розвитку рекомендаційних систем у стрімінгових сервісах, дозволяючи підвищити точність і адаптивність рекомендацій, що безпосередньо сприяє покращенню користувацького досвіду.

ПЕРЕЛІК ПОСИЛАНЬ

1. How to personalize content. Adobe Experience Cloud. URL: https://business.adobe.com/blog/basics/how-to-personalize-content

2. Why personalization matters in content marketing. Content Creation Platform - Interactive Content 100% on Brand - Foleon. URL: https://www.foleon.com/blog/why-personalization-matters-in-content-marketing

3. Content Based Filtering and Collaborative Filtering: Difference | Aman Kharwal. thecleverprogrammer. URL: https://thecleverprogrammer.com/2023/04/20/content-based-filtering-and-collaborative-filtering-difference/?utm_source

4. 4.S R. Recommender Systems: Collaborative Filtering and Content-Based Filtering. Medium. URL: https://ogre51.medium.com/recommender-systems-collaborative-filtering-and-content-based-filtering-20785d50d56f.

5. Banerjee A. Pros and cons of collaborative filtering. Medium. URL: https://medium.com/@ashmi_banerjee/pros-and-cons-of-collaborative-filtering-9c3aa4ce44f6

6. Content-based filtering | Machine Learning | Google for Developers. Google for Developers. URL: https://developers.google.com/machine-learning/recommendation/content-based/basics?utm_source.com

7. IBM. What is content-based filtering? | IBM. IBM - United States. URL: https://www.ibm.com/think/topics/content-based-filtering#:~:text=Content-based%20filtering%20is%20an,user%20expresses%20interest%20into%20account.

8. What Is Content-Based Filtering? Benefits and Examples in 2025. Upwork. URL: https://www.upwork.com/resources/what-is-content-based-filtering

9. Thandapani S. P. Recommendation systems: collaborative filtering using matrix factorization - simplified. Medium. URL: https://medium.com/sfu-cspmp/recommendation-systems-collaborative-filtering-using-matrix-factorization-simplified-2118f4ef2cd3

10. How TensorFlow Lite helps you from prototype to product. The TensorFlow Blog URL: https://blog.tensorflow.org/2020/04/how-tensorflow-lite-helps-you-from-prototype-to-product.html

11. What's new in TensorFlow Lite for NLP. The TensorFlow Blog. URL: https://blog.tensorflow.org/2020/09/whats-new-in-tensorflow-lite-for-nlp.html

12. Szczepanik M. Firebase ML Kit - overview. Medium. URL: https://medium.com/mobilepeople/firebase-ml-kit-overview-2c9ea7f4ee13

13. ML kit for firebase. Firebase. URL: https://firebase.google.com/docs/ml-kit

14. Sert O. A brief introduction to firebase ML kit. Medium. URL: https://medium.com/trendyol-tech/a-brief-introduction-to-firebase-ml-kit-6a018785b949

15. Adomavicius G., Tuzhilin A. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. IEEE Transactions on Knowledge and Data Engineering, 2005. URL: https://home.cs.colorado.edu/~martin/csci5417/recommender-review.pdf

16. Burke R. Hybrid Recommender Systems: Survey and Experiments. User Modeling and User-Adapted Interaction, 12(4), 2002. URL: https://link.springer.com/article/10.1023/A:1021240730564

17. Jannach D., Zanker M., Felfernig A., Friedrich G. Recommender Systems: An Introduction. Cambridge University Press, 2011. URL: https://pzs.dstu.dp.ua/DataMining/recom/bibl/1jannach_dietmar_zanker_markus_felfernig_alexander_friedrich.pdf

18. Zhang S., Yao L., Sun A., Tay Y. Deep Learning based Recommender System: A Survey and New Perspectives. ACM Computing Surveys, 2018. URL: https://dl.acm.org/doi/10.1145/3285029

19. Covington P., Adams J., Sargin E. Deep Neural Networks for YouTube Recommendations. Proc. of RecSys'16, 2016. URL: https://research.google.com/pubs/archive/45530.pdf

20. Collaborative filtering | Machine Learning | Google for Developers. URL: https://developers.google.com/machine-learning/recommendation/collaborative/basics

21. Deep neural network models. Softmax DNN for recommendation. Google Developers, Recommendation systems. URL: https://developers.google.com/machine-learning/recommendation/dnn/softmax

22. Chiang J. 7 Types of Hybrid Recommendation System. Analytics Vidhya (Medium), 2021. URL: https://medium.com/analytics-vidhya/7-types-of-hybrid-recommendation-system-3e4f78266ad8

23. Gohari F.S., Tarokh M.J. Classification and Comparison of the Hybrid Collaborative Filtering Systems. International Journal of Research in Industrial Engineering, 6(3), 2017. URL: https://www.riejournal.com/article_49158.html

24. Bakker T.F. Improving Hybrid Recommendation Systems for Solving the Cold Start User Problem. Master Thesis, Erasmus University Rotterdam, 2023. URL: https://thesis.eur.nl/pub/70117

25. Harper F.M., Konstan J.A. The MovieLens Datasets: History and Context. ACM Transactions on Interactive Intelligent Systems, 5(4), 2015. URL: https://files.grouplens.org/papers/harper-tiis2015.pdf

26. MovieLens 25M Dataset. GroupLens Research. URL: https://grouplens.org/datasets/movielens/25m/

27. Recommendation System for Movies - MovieLens. The Startup / Medium. URL: https://medium.com/swlh/recommendation-system-for-movies-movielens-grouplens-171d30be334e

28. David R. et al. TensorFlow Lite Micro: Embedded Machine Learning on TinyML Systems. Proc. MLSys 2021. URL: https://arxiv.org/abs/2010.08678

29. Rashidi M. Application of TensorFlow Lite on Embedded Devices: A Hands-on Practice of TensorFlow Model Conversion to TensorFlow Lite Model and its Deployment on Smartphone. Bachelor Thesis, Mid Sweden University, 2022. URL: https://www.diva-portal.org/smash/get/diva2:1698946/FULLTEXT01.pdf

30. Redkar T.T. TensorFlow Lite in Mobile Development. International Research Journal of Modernization in Engineering, Technology and Science, 2025. URL: https://www.irjmets.com/upload_newfiles/irjmets71100109236/paper_file/irjmets71100109236.pdf

31. Chen T., Guestrin C. XGBoost: A Scalable Tree Boosting System. Proc. 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016. URL: https://arxiv.org/abs/1603.02754

ДОДАТОК А. ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Магістерська робота

«Оптимізація алгоритмів персоналізованих рекомендацій у стрімінгових сервісах на основі гібридних моделей машинного навчання»

Виконав: студент групи ПДМ-61 Артем СТЕПАНЧЕНКО

Керівник: канд. техн. наук, доцент кафедри ІТ Наталія ТРІНТІНА

Київ - 2025

МЕТА, ОБ'ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

Мета роботи: розробити та дослідити метод динамічного коригування вагових коефіцієнтів у гібридній рекомендаційній моделі для стрімінгових сервісів, спрямований на підвищення точності рекомендацій (зокрема, показників Accuracy та F1-Score) та покращення їх адаптивності, особливо в умовах проблеми «холодного старту».

Об'єкт дослідження: процес формування персоналізованих рекомендацій у стрімінгових платформах із застосуванням алгоритмів машинного навчання.

Предмет дослідження: моделі та алгоритми оптимізації процесу гібридизації рекомендаційних систем (зокрема, метод динамічного зважування) на основі факторів користувацької поведінки та сесійного контексту.

ПОРІВНЯЛЬНА ХАРАКТЕРИСТИКА

Критерій	Content-Based Filtering (CBF)	Collaborative Filtering (CF)	Hybrid Filtering
Основні джерела даних	Метадані, атрибути та ознаки елементів	Поведінка користувачів (рейтинги, кліки, історія переглядів)	Комбінація поведінкових та контентних даних
Залежність від інших користувачів	Відсутня	Висока	Помірна або низька (залежить від моделі)
Cold-start для нових елементів	Добре справляється, якщо є описи	Погано, елементам потрібна історія взаємодій	Значно зменшується завдяки комбінуванню джерел даних
Cold-start для нових користувачів	Проблема, бо потрібно зібрати початкові вподобання	Проблема, бо бракує взаємодій	Найкраще рішення, бо використовує анкети, контентні сигнали та мінімальну поведінку
Новизна / серендипітність	Низька, тому що рекомендує схожі об'єкти	Вища, але можливі несподівані збіги поведінки	Висока, тому що комбінує схожість та поведінкові патерни
Різноманітність рекомендацій	Обмежена	Вища	Найвища, можна керувати балансом
Інтерпретованість	Висока (зрозуміло, на яких ознаках базується)	Середня або низька	Середня (залежить від моделі, як правило складніша)
Масштабованість	Залежить від кількості атрибутів	Залежить від кількості користувачів та взаємодій	Може бути оптимізована, особливо у модель-орієнтованих гібридах

1

МАТЕМАТИЧНА МОДЕЛЬ ГІБРИДНИХ РЕКОМЕНДАЦІЙ

$$R_{hyb}(u,i) = w_{CB}(u,s) \times R_{CB}(u,i) + w_{CF}(u,s) \times R_{CF}(u,i)$$

Де:

$R_{CB}(u,i)$ – оцінка контентної моделі

$R_{CF}(u,i)$ – оцінка колаборативної моделі

$w_{CB}(u,s), w_{CF}(u,s)$ – ваги моделей, які залежать:

- від користувача u,
- від сесії s,
- від контексту c (час, тип пристрою, жанрові зрушення)

Обмеження:

$$w_{CB} + w_{CF} = 1,\ w_{CB} + w_{CF} \in [0,1]$$

1

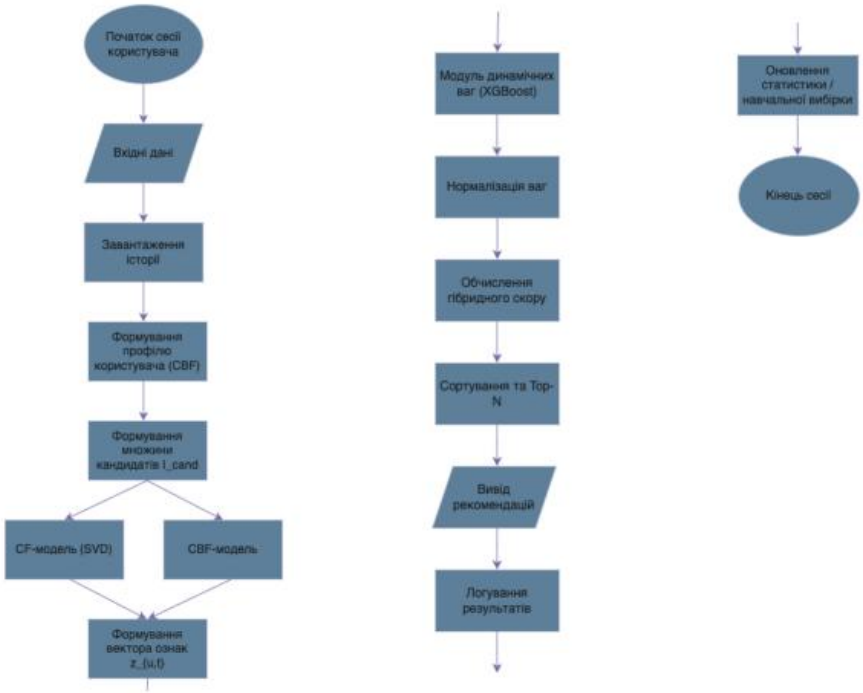
МЕТОД ДИНАМІЧНОГО ВИЗНАЧЕННЯ ВАГ

(1) Аналіз поведінки → (2) Аналіз контексту → (3) Розрахунок релевантності → (4) Нормалізація ваг → w_{CB}, w_{CF}

$$rel_{CB} = \alpha \times sim_{content} + \beta \times novelty$$
$$rel_{CF} = \gamma \times denisty + \delta \times sim_{users}$$
$$w_{CB} = \frac{rel_{CB}}{rel_{CB} + rel_{CF}}$$
$$w_{CF} = 1 - w_{CB}$$

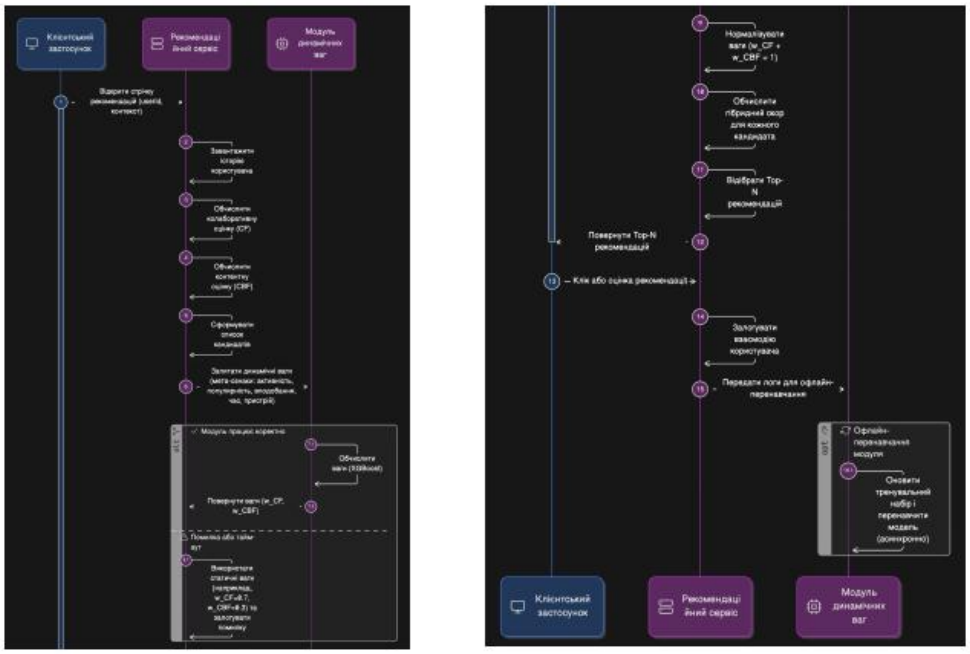
1

АЛГОРИТМ РОБОТИ ЗАПРОПОНОВАНОГО МЕТОДУ



1

ДІАГАРАМА ДІЯЛЬНОСТІ



1

ПРАКТИЧНА РЕАЛІЗАЦІЯ ТА СТРУКТУРА МОДЕЛЮВАННЯ

Практична реалізація проводилася у середовищі Python/Jupyter Notebook. Використано такі бібліотеки: pandas, numpy для обробки даних, surprise – для реалізації CF на основі SVD, xgboost – для реалізації метамоделі динамічного зважування, scikit-learn – для обчислення метрик якості. Як дані використано відкритий датасет MovieLens 25M, який містить близько 25 мільйонів оцінок.

```
# !pip install scikit-surprise xgboost numpy tqdm pandas scikit-learn

import pandas as pd
import numpy as np
import os
from surprise import Dataset, Reader, SVD
from surprise.model_selection import train_test_split
from tqdm import tqdm
from sklearn.metrics import mean_absolute_error, mean_squared_error, accuracy_score, f1_score
import xgboost as xgb
import pickle

# завантаження ML-25M та створення DATAFRAME
base_path = "Documents/Дуплет/ml-25m"

ratings_df_25m = pd.read_csv(os.path.join(base_path, 'ratings.csv'))
movies_df_25m = pd.read_csv(os.path.join(base_path, 'movies.csv'))

data_df_25m = pd.merge(ratings_df_25m, movies_df_25m, on='movieId')
data_df_25m = data_df_25m.drop('timestamp', axis=1)

# фільтрація рідкісних(поодиноких) користувачів и фільмів
MIN_RATING_COUNT = 5
user_counts = data_df_25m['userId'].value_counts()
movie_counts = data_df_25m['movieId'].value_counts()
data_df_25m = data_df_25m[data_df_25m['userId'].isin(user_counts[user_counts >= MIN_RATING_COUNT].index)]
data_df_25m = data_df_25m[data_df_25m['movieId'].isin(movie_counts[movie_counts >= MIN_RATING_COUNT].index)]

print(f"Датасет очищений. Залишилось {data_df_25m.shape[0]} оценок.")

# ONE-HOT ENCODING ЖАНРІВ
all_genres_list = data_df_25m['genres'].str.split('|').explode().dropna().unique()
all_genres = set(all_genres_list)

for genre in tqdm(all_genres, desc="Создание колонок жанров"):
    data_df_25m[genre] = data_df_25m['genres'].apply(lambda x: 1 if genre in x else 0)
```


1

РЕЗУЛЬТАТИ МОДЕЛЮВАННЯ ТА ПОРІВНЯЛЬНИЙ АНАЛІЗ

Модель	MAE	RMSE	Accuracy	F1-score
CF	0.5321	0.6985	0.7129	0.6300
CBF	1.7037	1.9524	0.4945	0.0008
Hybrid-Simple	0.8669	1.0288	0.5013	0.0281
Hybrid-XGB	0.4855	0.6402	0.7539	0.7022

ВИСНОВКИ

1. Проаналізовано класичні методи CF та CBF, а також сучасні гібридні моделі; показано, що статичні схеми комбінування не забезпечують достатньої адаптивності для стрімінгових сервісів.

2. Розроблено математичну модель гібридної рекомендаційної системи з динамічними ваговими коефіцієнтами між CF та CBF.

3. Запропоновано та реалізовано метод динамічного визначення ваг на основі поведінкових та контекстних ознак користувача, у вигляді ML-метамоделі (XGBoost).

4. На датасеті MovieLens 25M показано, що запропонований Hybrid-XGB перевершує базову CF-модель та простий гібрид за метриками MAE, RMSE, Accuracy та F1-score.

5. Отримані результати підтверджують доцільність застосування динамічних гібридних моделей машинного навчання для оптимізації алгоритмів персоналізованих рекомендацій у стрімінгових сервісах.

1

ПУБЛІКАЦІЇ ТА АПРОБАЦІЯ РОБОТИ

Тези доповідей:

1. Степанченко А.О., Трінтіна Н.А. Дослідження алгоритмів Content-Based, Collaborative Filtering та їх комбінованих моделей для побудови персоналізованих рекомендацій. Всеукраїнська науково-технічна конференція «Застосування програмного забезпечення в ІКТ», 24 квітня 2025 р., Київ, Державний університет інформаційно-комунікаційних технологій. Збірник тез. К.: ДУІКТ, 2025. С.372-374.
2. Степанченко А.О., Трінтіна Н.А. Впровадження машинного навчання у мобільні додатки: порівняння можливостей TensorFlow Lite та Firebase ML Kit. Всеукраїнська науково-технічна конференція «Застосування програмного забезпечення в ІКТ», 24 квітня 2025 р., Київ, Державний університет інформаційно-комунікаційних технологій. Збірник тез. К.: ДУІКТ, 2025. С.628-630.

1

ДОДАТОК Б. ЛІСТИНГИ ОСНОВНИХ МОДУЛІВ

```
import os
import pickle
import joblib
import numpy as np
import pandas as pd

from tqdm import tqdm
from surprise import Dataset,
Reader, SVD
from sklearn.metrics import (
mean_absolute_error,
mean_squared_error,
accuracy_score,
f1_score
)
import xgboost as xgb

# Завантаження MovieLens 25M т
а підготовка даних
base_path = "Documents/Дуікт/ml-
25m"

ratings_df =
pd.read_csv(os.path.join(base_path,
"ratings.csv"))
movies_df =
pd.read_csv(os.path.join(base_path,
"movies.csv"))

data_df = (
ratings_df
.merge(movies_df, on="movieId")
.drop(columns=["timestamp"])
)

MIN_RATING_COUNT = 5

user_counts =
data_df["userId"].value_counts()
movie_counts =
data_df["movieId"].value_counts()

data_df = data_df[
data_df["userId"].isin(user_counts[user_co
unts >= MIN_RATING_COUNT].index)
&
data_df["movieId"].isin(movie_counts[mo
vie_counts >=
MIN_RATING_COUNT].index)
]

print(f"Датасет очищено. Кількість оціно
к: {data_df.shape[0]}")

# One-Hot Encoding жанрів
all_genres = set(
data_df["genres"].str.split("|").explode().dr
opna().unique()
)

for genre in tqdm(all_genres, desc="Кодув
ання жанрів"):
data_df[genre] =
data_df["genres"].apply(lambda x:
int(genre in x))

data_df = data_df.drop(columns=["genres",
"title"])

NON_GENRE_COLS = ["userId",
"movieId", "rating"]
genre_cols = [c for c in data_df.columns if
c not in NON_GENRE_COLS]

# Матриця ознак елементів (Item
Features)
item_features_df = (
data_df
.drop(columns=["userId", "rating"])
```

```
.drop_duplicates("movieId")
.set_index("movieId")
)

item_features_matrix =
item_features_df[genre_cols]
print("Матриця ознак елементів:",
item_features_matrix.shape)

# Профілі користувачів
ALPHA = 0.6
BETA = 0.4

def create_user_profile(user_id):
user_data =
data_df[data_df["userId"] ==
user_id]
if user_data.empty:
return np.zeros(len(genre_cols),
dtype=np.float32)

genres =
user_data[genre_cols].values.astyp
e(np.float32)
ratings =
user_data["rating"].values.reshape(-
1, 1)

return (genres *
ratings).sum(axis=0) / ratings.sum()

profiles_path =
"user_profiles_25m.pkl"

if os.path.exists(profiles_path):
user_profiles_df =
pd.read_pickle(profiles_path)
else:
profiles = {
uid: create_user_profile(uid)
for uid in
tqdm(data_df["userId"].unique(),
desc="Побудова профілів")
}
user_profiles_df =
pd.DataFrame.from_dict(
profiles, orient="index",
columns=genre_cols
)
user_profiles_df.to_pickle(profiles_path)

# Контентна модель (CBF)

user_profiles = user_profiles_df.values
item_features =
item_features_matrix.values

user_id_map = {uid: i for i, uid in
enumerate(user_profiles_df.index)}
movie_id_map = {mid: i for i, mid in
enumerate(item_features_matrix.index)}

global_mean = data_df["rating"].mean()

def cosine_similarity(u, i):
denom = np.linalg.norm(u) *
np.linalg.norm(i)
return 0 if denom == 0 else np.dot(u, i) /
denom

def predict_cbf(user_id, movie_id):
try:
u = user_profiles[user_id_map[user_id]]
i =
item_features[movie_id_map[movie_id]]
except KeyError:
return global_mean

sim = cosine_similarity(u, i)
return np.clip(1 + sim * 4, 0.5, 5.0)

# Колаборативна модель (SVD)
svd_path = "svd_model.pkl"

reader = Reader(rating_scale=(0.5, 5.0))
data_surprise = Dataset.load_from_df(
data_df[["userId", "movieId", "rating"]],
reader
)
```

```
trainset =
data_surprise.build_full_trainset()

if os.path.exists(svd_path):
algo_cf =
pickle.load(open(svd_path, "rb"))
else:
algo_cf = SVD(
n_factors=100,
n_epochs=12,
lr_all=0.007,
reg_all=0.02,
random_state=42
)
algo_cf.fit(trainset)
pickle.dump(algo_cf,
open(svd_path, "wb"))

# Формування мета-ознак та навч
ання XGBoost

SAMPLE_SIZE = 50_000
META_MODEL_PATH =
"xgb_meta_25m.pkl"

sample_df = (
data_df[["userId", "movieId",
"rating"]]
.sample(n=SAMPLE_SIZE,
random_state=42)
.reset_index(drop=True)
)

user_activity =
data_df["userId"].value_counts()
item_popularity =
data_df["movieId"].value_counts()

total_items =
data_df["movieId"].nunique()
max_pop = item_popularity.max()

meta_rows = []

for u, m, r in
tqdm(sample_df.itertuples(index=False),
total=SAMPLE_SIZE):
r_cf = algo_cf.predict(u, m).est
r_cb = predict_cbf(u, m)

meta_rows.append({
"R_CF": r_cf,
"R_CB": r_cb,
"Rel_CF": ALPHA * user_activity.get(u, 0)
/ total_items,
"Rel_CB": BETA * (1 -
item_popularity.get(m, 0) / max_pop),
"R_actual": r
})

meta_df = pd.DataFrame(meta_rows)

X_meta =
meta_df.drop(columns=["R_actual"])
y_meta = meta_df["R_actual"]

xgb_model = xgb.XGBRegressor(
objective="reg:squarederror",
n_estimators=300,
max_depth=6,
learning_rate=0.07,
subsample=0.8,
colsample_bytree=0.8,
random_state=42,
n_jobs=-1
)

xgb_model.fit(X_meta, y_meta)
joblib.dump(xgb_model,
META_MODEL_PATH)

# Оцінювання моделей
TEST_SIZE = 20_000
THRESHOLD = 4.0

test_df = (
data_df[["userId", "movieId", "rating"]]
.sample(n=TEST_SIZE, random_state=42)
.rename(columns={"rating": "R_actual"})
```

```
)

def calc_metrics(y_true, y_pred):
y_true = np.asarray(y_true,
dtype=float)
y_pred = np.asarray(y_pred,
dtype=float)

return {
"MAE":
mean_absolute_error(y_true,
y_pred),
"RMSE":
np.sqrt(mean_squared_error(y_true,
y_pred)),
"Accuracy":
accuracy_score(y_true >=
THRESHOLD, y_pred >=
THRESHOLD),
"F1": f1_score(y_true >=
THRESHOLD, y_pred >=
THRESHOLD, zero_division=0)
}

pred_cf, pred_cb, pred_simple,
pred_meta = [], [], [], []

for u, m, r in
tqdm(test_df.itertuples(index=False)
, total=TEST_SIZE):
r_cf = algo_cf.predict(u, m).est
r_cb = predict_cbf(u, m)
pred_cf.append(r_cf)
pred_cb.append(r_cb)
pred_simple.append(np.clip(ALPHA * r_cf
+ BETA * r_cb, 0.5, 5.0))

features = pd.DataFrame([{
"R_CF": r_cf,
"R_CB": r_cb,
"Rel_CF": ALPHA * user_activity.get(u, 0)
/ total_items,
"Rel_CB": BETA * (1 -
item_popularity.get(m, 0) / max_pop)
}])

pred_meta.append(
np.clip(xgb_model.predict(features)[0], 0.5,
5.0)
)

results = pd.DataFrame({
"CF": calc_metrics(test_df["R_actual"],
pred_cf),
"CBF": calc_metrics(test_df["R_actual"],
pred_cb),
"Hybrid-Simple":
calc_metrics(test_df["R_actual"],
pred_simple),
"Hybrid-XGB":
calc_metrics(test_df["R_actual"],
pred_meta)
}).T

print(results)
```