Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет _____Комп'ютерних наук_____
(повна назва)

Кафедра _____ Інформаційних управляючих систем_____
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти _____другий (магістерський)_____
Дослідження методів динамічної адаптації рекомендацій в IT-проектах електронної комерції
(тема)

Виконав:
студент _2_ курсу, групи _УПГІТм-21-1_____
_____Станіслав СТУКАЛОВ_____
(власне ім'я, прізвище)

Спеціальність 122 Комп'ютерні науки_____
(код і повна назва спеціальності)
Тип програми _____освітньо-наукова_____
(освітньо-професійна або освітньо-наукова)
Освітня програма _Управління проєктами в галузі інформаційних технологій_____
(повна назва освітньої програми)

Керівник д.т.н., професор каф. ІУС Сергій ЧАЛИЙ
(посада, власне ім'я, прізвище)

Допускається до захисту

Зав. кафедри _____ ___Костянтин ПЕТРОВ__
(підпис) (власне ім'я, прізвище)

2023 р.

Харківський національний університет радіоелектроніки

Факультет _____Комп'ютерних наук_____

Кафедра _____Інформаційних управляючих систем_____

Рівень вищої освіти _____другий (магістерський)_____

Спеціальність _____122 Комп'ютерні науки_____
(код і повна назва)
Тип програми _____освітньо-наукова_____
(освітньо-професійна або освітньо-наукова)
Освітня програма_Управління проєктами в галузі інформаційних технологій
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_03_» _квітня_ 20 23_ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові_____Стукалову Станіславу Віталійовичу_____
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження методів динамічної адаптації рекомендацій в IT-проектах електронної комерції
затверджена наказом університету від _03_ _квітня_2023_ р. № _319Ст_
2. Термін подання студентом роботи до екзаменаційної комісії _18_ ___травня___ 2023 р.
3. Вихідні дані до роботи методична та технічна література, матеріали конференцій, дані інтернет-мережі, програма Microsoft Excel, програма Microsoft Project.

4. Перелік питань, що потрібно опрацювати в роботі Провести огляд літератури та загальний аналіз методів динамічної адаптації рекомендацій в IT-проектах електронної комерції. Провести збір та аналіз даних про IT-проекти електронної комерції. Дослідити розроблення та застосування методів динамічної адаптації рекомендацій в IT-проектах електронної комерції. Провести аналіз та узагальнення результатів дослідження.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання для кваліфікаційної роботи	03.04.2023	Виконано
2	Аналіз предметної області для кваліфікаційної роботи	04.04.2023 – 08.04.2023	Виконано
3	Формування плану роботи на кваліфікаційною роботою	09.04.2023 – 11.04.2023	Виконано
4	Робота над теоретичними відомостями для кваліфікаційної роботи	12.04.2023 – 19.04.2023	Виконано
5	Аналіз існуючих методів побудови рекомендацій	20.04.2023 – 22.04.2023	Виконано
6	Аналіз методів для рекомендаційних систем на практиці	22.04.2023 – 25.04.2023	Виконано
7	Оформлення звіту та висновків	25.04.2023 – 30.04.2023	Виконано
8	Захист кваліфікаційної роботи	19.05.2023	Виконано

Дата видачі завдання ___3___ __квітня__ 2023 р.

Студент _____
(підпис)

Керівник роботи _____ д.т.н, професор каф. ІУС Сергій ЧАЛИЙ
(підпис) (посада, власне ім’я, прізвище)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи магістра: 93 стор., 17 рис., 17 табл., 32 джерела, 1 додаток.

ДИНАМІЧНА АДАПТАЦІЯ, НАБІР ДАНИХ, ПЕРСОНАЛІЗАЦІЯ, ПРОГНОЗУВАННЯ, РЕКОМЕНДАЦІЙНА СИСТЕМА

Об’єкт дослідження – процеси побудови рекомендацій в ІТ-проектах електронної комерції.

Предмет дослідження – методи динамічної адаптації рекомендацій в ІТ-проектах електронної комерції.

Мета дослідження – проаналізувати властивості рекомендаційних систем, існуючи методи побудови рекомендацій та обґрунтувати необхідність адаптувати рекомендації по мірі зміни запитів та вподобань користувачів.

ABSTRACT

Explanatory note to the master's thesis: 93 pp, 17 figures, 17 tables, 32 sources, 1 appendix.

DYNAMIC ADAPTATION, DATA SET, FORECASTING, PERSONALIZATION, RECOMMENDER SYSTEM

The object of research are the processes of building recommendations in e-commerce IT projects.

The subject of research are the methods of dynamic adaptation of recommendations in e-commerce IT projects.

The purpose of the study is to analyze the properties of recommender systems, existing methods of building recommendations and to justify the need to adapt recommendations as user requests and preferences change.

ЗМІСТ

Скорочення та умовні познаки 7
Вступ 8
1 Рекомендаційні системи в іт-проектах електронної комерції 10
1.1 Аналіз іт-проектів електронної комерції які використовують рекомендаційні системи 10
1.2 Аналіз рекомендаційних систем 11
1.3 Аналіз методів побудови рекомендацій 16
1.4 Постановка задачі дослідження 32
2 Метод динамічної адаптації рекомендацій з використанням рейтингів користувачів 36
2.1 Загальний підхід до уточнення рекомендацій з використанням рейтингів користувачів 36
2.2 Метод динамічної адаптації рекомендацій з використанням колаборативної фільтрації 37
3 Розробка програмного проекту модулю динамічної адаптації рекомендацій в системі електронної комерції 44
3.1 Визначення життєвого циклу проекту, його окремі фази та етапи 44
3.2. Структуризація проекту 45
3.3 Побудова організаційної структури виконавців 46
3.4 Ув’язка WBS з OBS на основі побудови матриці відповідальності 47
3.5 Побудова структури споживаних ресурсів 49
3.6 Створення структури вартості 50
3.7 Склад та порядок розробки проектно-кошторисної документації 52
3.8 Логіко-інформаційна схема розробки проектно-кошторисної документації 53
3.9 Розробка моделі проекту 54
4 Реалізація та експериментальна перевірка вдосконаленого методу динамічної адаптації рекомендацій 59
4.1 Вибір метрики оцінки ефективності методу динамічної адаптації 59
4.2 Експериментальна перевірка ефективності розроблених методів 68
Висновки 77
Перелік джерел посилання 79
Додаток А Слайди презентації .....**Error! Bookmark not defined.**

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

МНС – метод найближчого сусіда;

РС – рекомендаційна система;

ЖЦ – життевий цикл;

CBF – content-based filtering;

CF – collaborative filtering;

IT – інформаційні технології;

OBS – organization breakdown structure;

WBS – work breakdown structure.

ВСТУП

Поширення цифрових платформ та експоненціальне зростання обсягів даних, створених користувачами, призвели до проблеми інформаційного перевантаження. Користувачі стикаються з величезною кількістю варіантів вибору, що робить пошук релевантного контенту або продуктів, які відповідають їх інтересам, дедалі складнішим завданням. Рекомендаційні системи пропонують вирішення цієї проблеми, використовуючи можливості аналізу даних і алгоритмів машинного навчання для надання персоналізованих рекомендацій. Пропонуючи товари або контент, які відповідають уподобанням користувача, ці системи мають на меті покращити користувацький досвід, підвищити рівень залученості та полегшити процес прийняття рішень.

Рекомендаційні системи відіграють ключову роль у сучасних додатках і сервісах. На платформах електронної комерції вони допомагають користувачам знаходити продукти, які відповідають їх інтересам, тим самим збільшуючи продажі та задоволеність клієнтів. В індустрії розваг системи рекомендацій допомагають користувачам знаходити фільми, музику чи книги, які відповідають їх уподобанням, що сприяє підвищенню залученості та утриманню користувачів. Платформи соціальних мереж використовують системи рекомендацій для надання користувачам персоналізованого контенту, покращуючи взаємодію між користувачами. Вплив рекомендаційних систем виходить за рамки користувацького досвіду, оскільки компанії отримують вигоду від збільшення конверсії, підвищення лояльності клієнтів та цінну інформацію про вподобання і тенденції користувачів.

Але існуючи методи побудови рекомендацій мають ряд недоліків які можуть привести до втрати довіри до рекомендаційної системи та втрати лояльності користувачів внаслідок постійної зміни вподобань користувачів.

Нами був удосконалений метод орієнтований на адаптацію рекомендованого переліку предметів з тим, щоб рекомендація відповідала останнім запитам.

Перевірка удосконаленого методу свідчить про можливість ефективного коригування рекомендацій з урахуванням старіння знань щодо вподобання користувачів з часом, що дає можливість підвищити привабливість рекомендаційної системи і лояльність клієнтів.

1 РЕКОМЕНДАЦІЙНІ СИСТЕМИ В ІТ-ПРОЕКТАХ ЕЛЕКТРОННОЇ КОМЕРЦІЇ

1.1 Аналіз ІТ-проектів електронної комерції які використовують рекомендаційні системи

Хоча автоматична рекомендація виникла як самостійна галузь досліджень і стала популярною в 1990-х роках, можна сказати, що вона почала розвиватися ще наприкінці 1970-х років в когнітивних науках, де використовувалися стереотипи для моделювання користувачів. Деякі роботи з теорії пошуку інформації та прогнозування можна вважати початком рекомендацій як дослідницької галузі.

Товари досвіду визначають активи, які необхідно спожити, перш ніж дізнатися про рівень задоволення від них. Культурний контент можна розглядати як квінтесенцію досвіду, який створює багатомільярдну глобальну індустрію, що базується на незліченній кількості жанрів і незліченній кількості митців[1].

Споживачі стикаються з важким завданням використання своїх обмежених бюджетів для придбання деяких з цих видів контенту, не маючи повного уявлення про те, наскільки вони є повноцінними. У таких ситуаціях рекомендації можуть суттєво покращити процес прийняття рішення про те, що купувати. Інтернет-магазини, які використовують систему рекомендацій, стають дедалі більш конкурентоспроможними порівняно з традиційними магазинами. Останнє десятиліття продемонструвало історичне зростання кількості веб-сайтів, що пропонують онлайн-послуги.

Від рекомендацій виграють не лише сервіси електронної комерції, але й будь-які сервіси, що базуються на великій кількості контенту. Дійсно, існує кілька мультимедійних платформ, що містять величезну кількість культурних продуктів, таких як Spotify (музика), Netflix (фільми) і так далі. Рекомендаційні

системи мають на меті запропонувати споживачам найцікавіші для них товари з великого каталогу продуктів.

Ці системи розробляються залежно від галузі та доступного джерела даних. Наприклад, користувачі Netflix ставлять оцінки переглянутим фільмам за шкалою від 1 (не сподобався) до 5 (сподобався). Крім того, система може мати доступ до специфічних для користувача та конкретного товару атрибутів профілю, таких як демографічні дані та опис товару відповідно[2].

На сьогоднішній день системи рекомендацій та їх оцінка в реальних умовах є дуже активною сферою досліджень.

1.2 Аналіз рекомендаційних систем

Система рекомендацій може надавати пропозиції (рекомендації) користувачам у різних контекстах, наприклад, коли вони роблять вибір серед великого каталогу товарів або коли вони хочуть отримати поради[3].

Виділяють 4 ключові функції:

– допомога прийняття рішення: прогнозування рейтингу для користувача для товару;

– допомога порівняння: ранжування списку елементів у персоналізований для користувача спосіб;

– допомога пошуку: надати користувачеві невідомі товари, які будуть оцінені;

– допомога вивчення: надати елементи, схожі на заданий цільовий елемент.

Більшість рекомендаційних систем застосовуються на веб-сайтах електронної комерції. На сайті кінцевому користувачеві відображається список рекомендованих товарів.

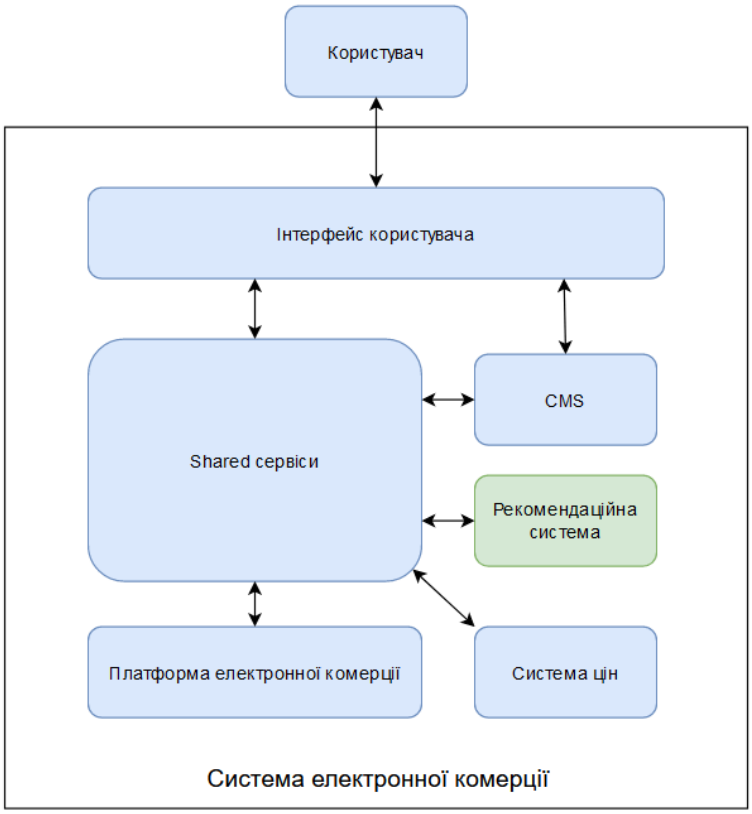


Рисунок 1.1 – Схема системи електронної комерції

На рисунку 1.1 зображена структура системи електронної комерції, у вигляді інтернет-магазину.

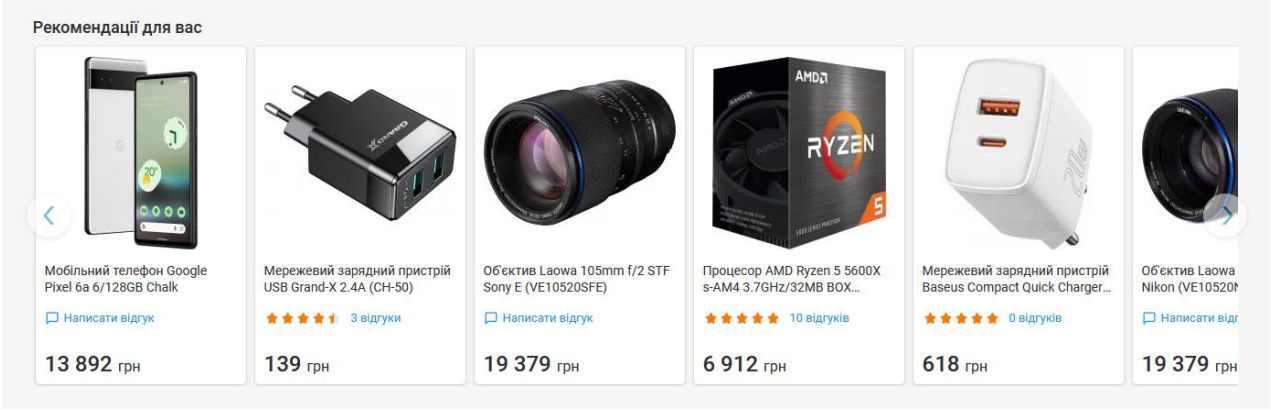


Рисунок 1.2 – Результат роботи рекомендаційної системи

На рисунку 1.2 зображений результат роботи рекомендаційної системи у вигляді списку рекомендованих товарів.

Система рекомендацій може бути визначена як така, що здатна вивчати вподобання користувачів щодо різних товарів і використовувати ці вподобання, щоб пропонувати нові товари, які можуть зацікавити користувачів[4].

Інше визначення описує систему рекомендацій як таку, що прогнозує, які товари можуть відповідати смакам і потребам певних користувачів.

Також можна сказати, що рекомендаційна система повинна бути здатною надавати індивідуалізовані рекомендації та вести користувачів у персоналізований спосіб. Це визначення додає нові поняття, такі як індивідуалізація та персоналізація[5].

Перейдемо до функцій рекомендаційних систем.

Розглянемо функції РС з точки зору користувача і постачальника послуг. Наприклад, система рекомендацій ресторанів чи готелів зазвичай використовується посередником (наприклад, TripAdvisor) для того, щоб підвищити коефіцієнт конверсії, тобто збільшити кількість людей, які відвідують певний ресторан, або продати більше готельних номерів.

З іншого боку, мотивація користувача, який користується такою системою, як TripAdvisor, полягає в тому, щоб знайти ресторан чи готель, який відповідає його смакам і потребам, і таким чином підвищити рівень його задоволеності[6].

Постачальники послуг використовують системи рекомендацій щоб:

– збільшити дохід. Іншими словами, це означає збільшення кількості проданих товарів. Ця функція, швидше за все, є найважливішою в промисловому контексті (комерційні РС). Мета полягає в тому, щоб продати більше товарів, ніж було б продано без будь-яких рекомендацій. Щоб досягти цієї мети, система рекомендує товари, які, як очікується, задовольнять смаки та потреби користувача. Однак слід розрізняти прогнозування інтересу користувачів до товару та ймовірність того, що користувачі дійсно виберуть рекомендований товар;

– збільшити різноманітність товарів, що продаються. Мета цієї функції – спонукати користувачів обирати товари, які без рекомендацій залишилися б

невідомими. Наприклад, у випадку книжкової РС (наприклад, книжковий магазин Amazon), постачальник послуг хоче мати можливість продавати книги з усього каталогу, а не лише 5 найпопулярніших;

– покращити користувацький досвід. Якщо система працює правильно і розроблена належним чином, вона може підвищити задоволеність користувачів. Адже, отримуючи цікаві, різноманітні та релевантні пропозиції, користувач буде цінувати досвід роботи на сайті або в додатку;

– зрозуміти користувачів. Ще однією важливою функцією системи самооцінки є можливість описати вподобання користувачів. Ці вподобання можуть бути зібрані в явному вигляді або шляхом прогнозування. Ці дані можуть бути використані постачальником послуг для кращого управління виробництвом або запасами.

Добре спроектована РС повинна мати хороший баланс між користувачами та постачальниками послуг[7].

Тепер розглянемо функції, які можуть бути цікавими для користувачів при використанні РС:

– ранжування списку товарів. Це, мабуть, одна з найважливіших функцій для РС, яка полягає в тому, щоб надавати поточному користувачеві деякі хороші товари відповідно до прогнозів рейтингу. Іншими словами, рекомендувати товари, які повинні сподобатися користувачеві;

– рекомендація послідовності. Ця функція більше спрямована на те, щоб відповідати довгостроковим уподобанням користувачів. Принцип полягає в тому, щоб генерувати послідовні рекомендації замість того, щоб надавати послідовність незалежних рекомендацій. Наприклад, має сенс рекомендувати «Матриця 2: Перезавантаження» після рекомендації «Матриця 1»;

– надання анотацій. У цьому випадку, замість того, щоб створювати список елементів, розглянемо вже існуючий контекст, наприклад, щотижневий реліз фільмів у кінотеатрах. Функція РС полягатиме в тому, щоб підкреслити фільми, які з більшою ймовірністю відповідають смакам та уподобанням користувача;

– покращення навігації. За наявності великого каталогу, РС може покращити користувацький досвід, допомагаючи користувачу знаходити елементи, які відповідають його смакам та потребам.

Перейдемо до розгляду джерела даних для рекомендаційних систем.

Системи рекомендацій самі наповнюють себе даними. Вони збирають різні типи даних, такі як дуже прості і базові дані (рейтинги/оцінки користувачів), більш залежні від знань (онтологічний опис) або соціальні зв'язки і діяльність користувачів. Незалежно від того, яким є джерело даних, зазвичай виділяються три сутності: об'єкти, користувачі та зв'язки між користувачами та об'єктами[8].

Найзручніші дані – це високоякісні явні відгуки, які включають явну інформацію від користувачів про їхню зацікавленість у продуктах. Наприклад, Netflix збирає оцінки фільмів, а користувачі TiVo вказують свої вподобання щодо телепередач, натискаючи кнопки «палець вгору» і «палець вниз».

Зазвичай, явний зворотній зв'язок складається з розрідженої матриці, оскільки кожен окремий користувач, швидше за все, оцінив лише невеликий відсоток можливих елементів.

За відсутності явного зворотного зв'язку системи рекомендацій можуть робити висновки про вподобання користувача за допомогою неявного зворотного зв'язку, який опосередковано відображає думку користувача, спостерігаючи за його поведінкою, включаючи історію пошуку, історію переглядів, шаблони пошуку або навіть рухи миші[9].

Перейдемо до класифікації рекомендаційних систем.

Найпоширенішим способом опису та ідентифікації різних типів рекомендаційних систем є наступний:

– системи колаборативної фільтрації;

– системи фільтрації на основі вмісту;

– гібридні системи.

Типова рекомендація на основі колаборативної фільтрації (наприклад, на Amazon) повністю ґрунтується на оцінці користувача (наприклад, користувач оцінив фільм на 4 зірки, або користувачеві «подобається» фільм).

Коли обчислюється схожість між товарами, використовується лише історія оцінок усіх користувачів. Тому схожість між елементами обчислюється на основі оцінок, а не метаданих вмісту елемента.

Та навпаки, суть фільтрації на основі вмісту полягає в використанні метаданих як користувача, так і предмета. Наприклад, фільм буде представлено як його оцінку та жанр. Для профілю користувача можна зробити те ж саме, базуючись на його улюблених жанрах, кінозірках, тощо. Тоді схожість користувача і товару можна обчислити, використовуючи будь яку формулу подібності[10].

1.3 Аналіз методів побудови рекомендацій

Класифікація існуючих методів рекомендацій, визначає вхідні дані кожного методу та алгоритм, що використовується. Визначається п'ять типів методів рекомендацій:

– колаборативна фільтрація;

– на основі вмісту;

– демографічні;

– на основі корисності;

– на основі знань.

Визначення корисних для користувачів елементів є основною функцією рекомендаційної системи[11]. Для того, щоб це зробити, РС повинна передбачити корисність цих елементів. Потім, на основі отриманих результатів, система вирішує, які елементи є рекомендованими.

Розглянемо колаборативну фільтрацію між користувачами.

Колаборативна фільтрація між користувачами базується на центральній ідеї, що користувачі, які цікавляться тими самими об'єктами і мають схожі рейтинги, матимуть схожі вподобання. Враховуючи подібну поведінку рейтингу, система рекомендацій здатна передбачити, чи може користувач зацікавитись невидимим товаром[12].

Процес типової колаборативної фільтрації користувач-користувач зазвичай поділяється на три етапи:

– вимірювання схожості. Система рекомендацій обчислює схожість між активним користувачем та іншими користувачами, які оцінили ті ж елементи. Цей крок є добре дослідженою областю в галузі рекомендаційних систем;

– сусідство. Схожі користувачі перегруповуються в підмножину, а саме «сусідство» активного користувача. Як правило, це оточення складається з найбільш схожих користувачів, обчислених на кроці 1;

– прогнозування та генерація рекомендацій. На третьому і останньому кроці система збирає інформацію з оточення активного користувача. Потім алгоритм генерує список товарів, які можна рекомендувати. У реальному контексті веб-сайтів електронної комерції він називається списком рекомендацій «Top-N». Крім того, для конкретного товару можна розрахувати прогноз рейтингу.

Найсильнішою перевагою колаборативної фільтрації є її незалежність від домену. Схожість між користувачами обчислюється за допомогою лише рейтингових даних, робить систему адаптованою до будь-якого типу продуктів. Однак основний недолік цієї методики полягає в тому, що схожість, яка обчислюється лише на основі даних рейтингу, не може враховувати причини, які призвели до гарної чи поганої оцінки. Таким чином, двом користувачам може сподобатися один і той самий товар, але з абсолютно різних причин.

Розглянемо колаборативну фільтрацію по елементах.

Колаборативна фільтрація по елементах чимось схожа на колаборативну фільтрацію між користувачами і може розглядатися як такий же підхід, але з точки зору елементів[13].

Ми дотримуємося такого принципу: товари, які були оцінені однаково, ймовірно, мають схожі характеристики, тому користувачам, яким сподобався один з них, мають сподобатися й інші товари, які отримали аналогічну оцінку.

Amazon є, мабуть, найвідомішим веб-сайтом електронної комерції, який використовує колаборативну фільтрацію товарів. Завдяки високоефективній та добре функціонуючій технології рекомендацій Amazon отримав значний приріст прибутку. Оскільки вона також вимагає лише даних про рейтинг, то колаборативна фільтрація по елементах має ті ж самі сильні сторони та недоліки, що й фільтрація між користувачами.

Розглянемо фільтрацію на основі вмісту (CBF).

В основі контент-орієнтованих підходів лежить побудова моделі (або профілю) інтересів користувачів на основі характеристик об'єктів, які вони оцінили. Для цього система аналізує опис товарів, які користувачі найчастіше оцінюють. Процес надання рекомендацій полягає в тому, щоб знайти відповідність між профілем користувача та характеристиками товарів.

Ця методика має такі переваги, як незалежність користувача, оскільки системи CBF використовують оцінки лише активного користувача для побудови своєї моделі, на відміну від методів колективної фільтрації, які покладаються на «сусідів». Крім того, коли з'являється новий товар, який ще не був оцінений, системи CBF можуть рекомендувати його. Це метод колаборативної фільтрації, добре відомий під назвою «холодний старт» або проблема «першого оцінювача».

Однак ці методи страждають від надмірної спеціалізації, оскільки вони не здатні знаходити несподівані товари – користувач отримуватиме рекомендації товарів, схожих на ті, які він оцінював раніше. Ця проблема новизни також відома як проблема випадковості.

Важливо зазначити, що методи фільтрації на основі вмісту, насправді не ґрунтуються на реальному вмісті товарів. Наприклад, у контексті книжкових рекомендацій система CBF насправді не покладається на зміст книги. У кращому випадку вона базується лише на описі, ключових словах або анотації. Здебільшого «вміст» – це лише жанр, автор, редактор або інші метадані. Крім того, ці методи не беруть до уваги текстовий контент, який був написаний про об'єкти користувачами в блогах, або соціальних мережах[14].

Ці підходи інтегрують лінгвістичні знання в процес вивчення профілів користувачів. Основним недоліком цих методів є те, що лінгвістичні знання походять виключно з лексичної онтології WordNet. Тому так звана «смислова» репрезентація профілю спирається і залежить від деяких детермінованих концепцій: вона насправді не вивчається з того, що користувачі пишуть про об'єкти. Більше того, ці методи залежать від мови та предметної області.

Демографічні системи здатні рекомендувати товари відповідно до демографічної інформації користувачів. Цей тип систем ґрунтується на гіпотезі, що кожному демографічному класу слід рекомендувати різні товари. Це простий і ефективний метод персоналізації. Веб-сайти можуть легко адаптувати мову відображення відповідно до країни або мови користувача. Рекомендації можуть враховувати вік користувача. Ці методи більш вивчені в маркетинговій літературі, ніж у сфері РС.

Системи на основі знань рекомендують товари відповідно до певної інформації про відповідність характеристик даного товару потребам конкретного користувача. У системах на основі знань, схожість дає міру якості відповідності між вподобаннями (або потребами) користувача і рекомендованими товарами. У цьому випадку схожість може бути прирівняна до корисності для користувача[15].

Системи статичного узагальнення базуються на ідеї, що люди зазвичай довіряють статистиці. Найчастіше використовуються такі методи статистичного узагальнення, як популярність та середнє значення. Ці методи популярні тому,

що вони ефективно пояснюють користувачам причини надання конкретних рекомендацій.

Методи рекомендацій на основі популярності ґрунтуються на простій ідеї. Як правило, платформи електронної комерції показують список найбільш продаваних товарів на головній сторінці. Система рекомендацій, враховує рейтинги кожного користувача і з цього загального середнього значення вибираються товари з найвищим рейтингом.

Простий і прагматичний метод – використання середнього значення в поєднанні з популярністю. Внутрішня якість і популярність товару якимось чином зосереджені в його середньому рейтингу.

Наприклад, база даних фільмів в Інтернеті (IMDb.com) представляє середній рейтинг для кожного фільму. Це дозволяє користувачам отримати уявлення про загальну думку (позитивну чи негативну) про даний фільм.

На Vodkaster, французькій відео-платформі та соціальній мережі для фільмів, представлений ще один тип середнього показника – середня оцінка друзів. Це одночасно і статистична, і соціальна інформація. Вважається, що цей тип середнього показника є дуже переконливим[16].

Техніка соціальної навігації ґрунтується на простому принципі відображення активності інших користувачів (наприклад, «Користувач 1 вподобав/купив товар 3»). Вважається, що користувачі надають значення тому, що подобається їх однолітkам.

Простий інтерфейс соціальної навігації покращує досвід рекомендацій користувача, оскільки його сусіди фактично є потенційними рекомендаторами.

Соціальна система рекомендацій використовує рекомендації на основі контенту і на основі соціальних мереж. Основна ідея полягає в тому, щоб рекомендувати високоякісні документи, пов'язані з темами і змістом запитів, які зберігаються у «друзів» користувачів.

Методи, які були представлені вище, можна об'єднати у гібридну систему рекомендацій. Таким чином, слабкі сторони та недоліки кожного окремого методу можуть нівелювати один одного і покращиться продуктивність. Наприклад, методи колаборативної фільтрації стикаються з проблемою нових елементів, тобто вони не можуть рекомендувати елементи, які ще не були оцінені. Однак, характеристики нових елементів є загальнодоступними і можуть бути використані з методами на основі вмісту.

Розглянемо методи інтелектуального аналізу даних для рекомендаційних систем.

Методи найближчого сусіда є одними з найпоширеніших у сфері рекомендацій. Як правило, базуючись на бінарних або дійсних даних, ці підходи використовують деякі принципи правил асоціації та узагальнюють їх для порівняння об'єктів (наприклад, товарів або користувачів). Навіть якщо ці методи дуже добре пристосовані для рекомендацій від елемента до елемента, вони страждають від недостатньої масштабованості – час пошуку сусідів зростає в квадратичній залежності від кількості елементів.

Зазвичай, існують три основні компоненти підходу до колаборативної фільтрації з використанням МНС:

– міра подібності;

– функція, яка знаходить сусідів, використовуючи міру подібності;

– функція прогнозування рейтингу на основі рейтингів сусідів.

Залежно від того, яка асоціація використовується, можна виділити два методи МНС: на основі об'єктів та на основі користувачів.

Розглянемо МНС на основі елементів.

У цьому підході прогноз рейтингу певного користувача u на певному об'єкті i обчислюється на основі попередніх рейтингів u на об'єктах, схожих на i. МНС-підходи на основі об'єктів мають значну перевагу з точки зору обчислювальної складності, тому що час обчислення сусідства пропорційний квадрату кількості об'єктів, що порівнюються. Кількість користувачів майже

завжди набагато більша за кількість об'єктів. Тому підходи на основі елементів часто є більш ефективними. Однак, можна стверджувати, що це не так у каталогах, створених користувачами, таких як Youtube, оскільки кількість відео є такою, що матриці «елемент-елемент» є величезними.

Найвідомішим прикладом підходу MHC на основі товарів є Amazon і його знамените «люди, які купили цей товар, також купили ці товари».

Розглянемо MHC на основі користувача.

У цьому підході прогноз рейтингу заданого користувача u на заданому елементі i обчислюється з використанням попередніх рейтингів сусідів u на елементі i. Таким чином, необхідно обчислити матрицю схожості між користувачами. Як і раніше, потрібно визначити сусідство і вибрати набір з K найближчих сусідів. Після цього рейтинги сусідів об'єднуються у функцію прогнозування рейтингу.

Перейдемо до методів матричної факторизації.

У своїй базовій формі матрична факторизація характеризує як товари, так і користувачів за допомогою векторів факторів, виведених з моделей оцінювання товарів. Висока відповідність між факторами товарів і користувачів призводить до рекомендації. Рекомендаційні системи покладаються на різні типи вхідних даних, які часто розміщуються у матриці, один вимір якої представляє користувачів, а інший – об'єкти, що представляють інтерес.

Моделі матричної факторизації – це методи, які проектують користувачів та об'єкти на спільний простір латентних факторів розмірності k. У цьому новому просторі взаємодії між користувачем та об'єктом представлені як внутрішні продукти. Кожен товар i асоціюється з вектором $q_i \in$, а кожен користувач u асоціюється з вектором $p_u \in$.

Щоб оцінити рейтинг користувача u для товару i, ми обчислюємо точковий продукт $q_i^T p_u$, який позначимо $r_{u,i}$:

$$\widehat{r_{u,i}} = q_i^T p_u, \qquad (1.1)$$

Цей підхід чимось схожий на розкладання за одиничними значеннями. В інформаційному пошуку цей підхід часто використовується для виявлення латентних семантичних факторів. У контексті колаборативної фільтрації необхідно факторизувати матрицю оцінок «користувачі-об'єкти». Такі матриці зазвичай є неповними через пропущені значення і тому є відносно розрідженими.

Можна також заповнити матрицю і зробити її щільною. Основною проблемою цього підходу є значне збільшення обсягу даних і викривлення, якщо інтерполяція не є точною[17].

Можна моделювати лише спостережувані рейтинги і використовувати регуляризатори в своїй моделі. Таким чином, факторні вектори p_u та q_i визначаються таким чином, щоб регуляризована квадратична похибка на множині відомих оцінок була мінімальною:

$$\min_{q,p} \sum_{(u,i)\in K} \left(r_{u,i} - q_i^T \dot{p}_u\right)^2 + \lambda\left(\|q_i\|^2 + \|p_u\|^2\right), \qquad (1.2)$$

де k – множина пар (u, i);

$r_{u,i}$ – навчальна множина.

Модель налаштовується на виставлених оцінках. Оскільки система повинна бути здатна передбачати невідомі оцінки, ці попередні спостереження повинні бути узагальнені. Таким чином, вивчені параметри повинні бути регуляризовані. λ (константа) – задає інтенсивність регуляризації. Для визначення цієї константи зазвичай використовують перехресну перевірку.

Ці методи пропонують хорошу масштабованість і точність прогнозування, але їх класичні сучасні версії не дозволяють здійснювати динамічну адаптацію.

Розглянемо підходи до розрахування міри подібності.

Подібність за Пірсоном між елементами i та j обчислюється наступним чином:

$$\text{Pearson}(i,j) = \frac{\sum_{u \in T_i \cap T_j} \left(r_{u,i} - \bar{r_i}\right)\left(r_{u,j} - \bar{r_j}\right)}{\sqrt{\left(\sum_{u \in T_i \cap T_j} \left(r_{u,i} - \bar{r_i}\right)^2\right)\left(\sum_{u \in T_i \cap T_j} \left(r_{u,j} - \bar{r_j}\right)^2\right)}}, \quad (1.3)$$

де T_i – множина користувачів, які оцінили об'єкт i;

T_j – множина користувачів, які оцінили об'єкт j;

$r_{u,i}$ – оцінка користувача u за об'єкт i;

r_i – середня оцінка об'єкта i;

r_i – середня оцінка об'єкта j.

Однак у цієї міри подібності існує проблема у випадку, коли два елементи мають лише одного спільного користувача. Припустимо, що цей користувач поставив однакову оцінку обом позиціям, тоді подібність дорівнюватиме максимальному значенню 1. Крім того, вона буде такою ж, якщо тисяча користувачів дадуть однакові оцінки товарам i та j[18].

Косинусна міра подібності визначається наступним чином:

$$\text{Cosine}(i,j) = \frac{\sum_{u \in T_i \cap T_j} r_{u,i} \times r_{u,j}}{\sqrt{\left(\sum_{u \in T_i \cap T_j} r_{u,i}^2\right)\left(\sum_{u \in T_i \cap T_j} r_{u,j}^2\right)}}, \quad (1.4)$$

Вона страждає від тієї ж похибки, що і Пірсона. Більше того, у випадку колінеарних векторів косинусна подібність є максимальною. У чисто геометричному контексті цей результат може мати сенс, але в нашому випадку це не так.

Наприклад, припустимо, що у нас є два елементи (a, b) і $T_a \cap T_b = (u_1, u_2, u_3, u_4, u_5)$. Припустимо, що ми маємо наступні рейтинги (таблиця 1.1):

Таблиця 1.1 – Приклад помилки косинусної міри подібності

	u_1	u_2	u_3	u_4	u_5
Елемент a	1	1	1	1	1
Елемент b	5	5	5	5	5

Швидкий розрахунок призводить до результату $Cosine(a,b) = 1$, що є максимальною подібністю. Проте, a та b абсолютно різні. Альтернативою цій проблемі є взяття об'єднання замість перетину в знаменнику. Це справедливо як для формули косинусів, так і для формули Пірсона.

Подібність Жаккара може бути корисною для роботи з наборами даних, в яких переносяться лише бінарні події. У таких випадках ні Пірсон, ні косинус не є цікавими, оскільки вони дорівнюють нулю. Подібність Жаккара також корисна для порівняння двох елементів на основі їх метаданих, якщо такі є[19]. Вона також може бути адаптована до інших ситуацій, відмінних від двійкових даних.

$$\text{Jaccard}(i,j) = \frac{|T_i \cap T_j|}{|T_i \cup T_j|}, \tag{1.5}$$

Розглянемо приклади подібностей.

Для того щоб показати відмінності між різними подібностями візьмемо двох користувачів u та v з набору даних Netflix, які поставили відповідно 204 та 205 оцінок. Вони мають 30 спільних елементів. Ми також маємо r_u = 3.81 та r_v = 3.29.

В таблиці 1.2. показане порівняння оцінок цих користувачів на їх спільних елементах. Поглянувши побіжно, можна інтуїтивно здогадатися, що ці два користувачі схожі, оскільки вони не мають багато сильних розбіжностей, таких як $r_{u,i} = 5$ та $r_{v,i} = 1$.

Таблиця 1.2 – Порівняння оцінок користувачів

Предмети	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8	i_9	i_{10}	i_{11}	i_{12}	i_{13}	i_{14}	i_{15}
u	5	3	4	3	5	4	4	5	4	1	5	3	5	4	2
v	5	5	5	2	4	5	4	4	3	2	4	4	5	5	1
Предмети	i_{16}	i_{17}	i_{18}	i_{19}	i_{20}	i_{21}	i_{22}	i_{23}	i_{24}	i_{25}	i_{26}	i_{27}	i_{28}	i_{29}	i_{30}
u	5	5	5	4	2	5	5	3	1	5	4	5	3	2	3
v	4	4	5	2	4	4	5	1	3	3	2	3	4	2	2

Розрахуємо їх схожість за допомогою різних методів:

– пірсон(u, v) = 0.504;

– косинус(u, v) = 0.948;

– жаккард(u, v) = 0.082.

Результати розрахування подібності цих користувачів сильно відрізняються одне від одного. Цю проблему можна вирішити виведенням нової мірі подібності яка буде компромісом між занадто суворою подібністю, як Пірсона, і занадто м'якою, як подібність косинусів. Це також допоможе нам уникнути згаданих недоліків цих мір.

Перейдемо до прогнозування оцінок.

Розглянемо об'єкт i та користувача u. Оскільки користувачі здебільшого не мають права ставити кілька оцінок одному й тому ж об'єкту, припустимо, що пара (u, i) є унікальною. Це правило застосовується до більшості соціальних мереж або відео платформ. Рекомендувати чи не рекомендувати користувачеві u елемент i означає передбачити рейтинг $r(u, i)$. Потім, залежно від прогнозу рейтингу, система вирішує, рекомендувати чи не рекомендувати елемент i користувачеві u[20].

Інформацією, доступною для системи, є попередні оцінки користувача u та оцінки товару i, надані іншими користувачами $v \in T_i$. З цієї точки зору, намагаючись передбачити $r(u, i)$ потрібно спочатку подивитися, чи є серед користувачів v, які оцінили i раніше, такі, що мають спільні пункти з u. Якщо

так, то ми можемо обчислити схожість між u та кожним з цих користувачів. Якщо ні, то подібність буде дорівнювати нулю.

Обчислимо середньозважене значення попередніх оцінок, виставлених іншими користувачами, де вагами є схожість між u та $v \in T_i$.

$$\text{rating}(u,i) = \frac{\sum_{v \in T_i} \text{Sim}(u,v) \times r_{v,i}}{\sum_{v \in T_i} |\text{Sim}(u,v)|}, \quad (1.6)$$

де Sim - деяка функція подібності.

Цей підхід орієнтований на користувача, тому що враховується саме схожість між користувачами.

Симетрична формула $rating(i,u)$ орієнтована на продукти, виглядатиме так:

$$\text{rating}(i,u) = \frac{\sum_{j \in S_u} \text{Sim}(i,j) \times r_{u,j}}{\sum_{j \in S_u} |\text{Sim}(i,j)|}, \quad (1.7)$$

Щоб врахувати обидві точки зору, робимо лінійну комбінацію.

$$\hat{r}_{u,i} = \beta \times \text{rating}(u,i) + (1-\beta) \times \text{rating}(i,u) \quad (1.8)$$

Для того, щоб збалансувати прогноз, потрібно врахувати r_u і r_i. Ці два середні значення комбінуються з двома коефіцієнтами: m_i для r_i та m_u для r_u, причому $m_i + m_u = 1$. Таким чином, зважена функція прогнозування рейтингу:

$$\hat{r}w_{u,i} = \gamma \hat{r}_{u,i} + (1-\gamma)\left(\alpha \bar{r_u} + (1-\alpha)\bar{r_i}\right), \quad (1.9)$$

Перейдемо до врахування упереджень користувачів та об'єктів.

Деякі користувачі мають тенденцію ставити вищі оцінки, ніж інші і, відповідно, деякі об'єкти отримують вищі оцінки, ніж інші[21].

Позначимо через μ загальний середній рейтинг. Базовий прогноз для невідомого рейтингу $r(u,i)$ позначимо через b_{ui} і врахуємо вплив користувача та товару:

$$b_{ui} = \mu + b_u + b_i, \tag{1.10}$$

Параметри b_uта b_i вказують на спостережувані відхилення користувача u та товару i від середнього значення. Такий підхід дозволяє виокремити те, що є дійсно важливим для взаємодії користувача з предметом.

Використовуючи цю загальну ідею, визначаємо більш точну функцію, а саме функцію Gap, яка дозволяє нам налаштувати прогноз рейтингу для кожного користувача. В цій функції «розрив» не розраховується відносно загального середнього значення, а виводиться з кожного відповідного відхилення від середнього значення оцінених елементів. Формула розрахування «розриву»:

$$\mathrm{Gap}(u) = \frac{\sum_{i \in S_u} (r_{u,i} - \bar{r}_i)}{|Su|}, \tag{1.11}$$

Функція $\mathrm{Gap}(u)$ дозволяє визначити, чи має користувач певну тенденцію ставити оцінки вище (більш толерантні) або нижче (більш суворі) від середнього значення. Ця функція використовується під час процесу прогнозування рейтингу, щоб скоригувати індивідуально для кожного користувача середній рейтинг позиції. Функція $Gap(u)$ постійно оновлюється, отже, якщо є якась зміна в поведінці людини, система одразу ж це врахує.

Те ж саме стосується і товарів:

$$\mathrm{Gap}(i) = \frac{\sum_{u \in S_i} \left(r_{u,i} - \bar{r_u}\right)}{|Si|}, \tag{1.12}$$

Можливі випадки при обчисленні функції Gap показані в таблиці 1.3.

Таблиця 1.3 – Можливі випадки при обчисленні функції «розриву»

	Високо оцінений продукт	Низько оцінений продукт
Дружній користувач	$Gap(u) > 0, Gap(i) > 0$	$Gap(u) > 0, Gap(i) < 0$
Суворий користувач	$Gap(u) < 0, Gap(i) > 0$	$Gap(u) < 0, Gap(i) < 0$

Наприклад, візьмемо суворого користувача Антона та фільм з високим рейтингом «Зоряні війни: Епізод V – Імперія завдає удару у відповідь». Включивши «розриви», система врахує цю інформацію і компенсує прогноз рейтингу.

Функція Gap інтегрується у функцію зваженого рейтингу $\hat{r}w_{u,i}$ (формула 1.9). Вона коригує обидва середні значення наступним чином:

$$m_i\left(\bar{r_i} + Gap(u)\right) + m_u\left(\bar{r_u} + Gap(i)\right), \qquad (1.13)$$

Середнє для товару коригується функцією $Gap(u)$, а середнє для користувача – функцією $Gap(i)$.

Перейдемо до автоматичного налаштування через випадковість.

Багато задач оптимізації можна вирішити за допомогою методів випадкового пошуку. Ці методи стають конкурентоспроможними в деяких специфічних обставинах, наприклад, коли характеристики функції важко обчислити, коли доступна лише обмежена пам'ять комп'ютера, коли функція, яку потрібно мінімізувати, дуже «нерівномірна», коли дуже бажано знайти глобальний мінімум функції, що має багато локальних мінімумів, тощо[22].

Процес такої оптимізації виглядає наступним чином:

– запустити n дослідів з набором випадкових параметрів (суддя);

– оцінити кожного суддю, використовуючи метрику, яку потрібно оптимізувати;

– вибрати K найкращих суддів;

– запустити на тестовому наборі;

– об'єднати для кожної пари (u, i) оцінки, передбачені K суддями;

– оцінити.

У цьому методі K прогнозів оцінок $r_k\ (u, i)$ об'єднуються за допомогою простого середнього. Для більшої точності використаємо зважене середнє надавши більшу вагу найкращим суддям. Ці ваги W_k прямо пропорційні оцінці відповідного судді. Таким чином, остаточний прогноз рейтингу розраховується наступним чином:

$$\hat{r}_{u,i} = \frac{\sum_k r_k(u,i) \times W_k}{\sum_k W_k}, \tag{1.14}$$

Розглянемо підхід конкретних суддів.

У цьому підході вибирають конкретних суддів, які добре підходять для конкретної частини популяції користувачів та/або конкретного набору об'єктів. Для цього спочатку розділяють користувачів та об'єкти відповідно до заданого показника. Цей показник може бути взятий з метаданих, наприклад – вік, стать, країна, тощо.

В таблиці 1.4. показані результати використання підходу конкретних суддів на наборі даних Netflix.

Таблиця 1.4 – Приклад підходу конкретних суддів на наборі даних Netflix

	RMSE	MAE	α	β	γ
Суддя 1	0.919	0.713	0.609	0.315	0.085
Суддя 2	0.919	0.715	0.123	0.505	0.068
Суддя 3	0.919	0.716	0.094	0.434	0.162
Суддя 4	0.919	0.715	0.489	0.075	0.093
Суддя 5	0.919	0.715	0.234	0.176	0.080
Суддя 6	0.919	0.714	0.670	0.536	0.071
Суддя 7	0.919	0.715	0.175	0.555	0.164
Суддя 8	0.920	0.714	0.374	0.619	0.185
Суддя 9	0.920	0.716	0.560	0.848	0.105
Суддя 10	0.920	0.713	0.604	0.388	0.074

В таблиці 1.4:

– α – вага r_u, $(1 - \alpha)$ для r_i;

– β – вага $rating_{u,i}$, $(1 - \beta)$ для $rating_{i,u}$;

– γ – вага моделі на основі подібності, $(1 - \gamma)$ для усередненої моделі.

Можна побачити, що залежно від судді співвідношення між параметрами можуть суттєво змінюватися. Для всіх суддів показник γ достатньо невеликий (приблизно 0,1). Це означає, що більшість обраних тут моделей спирається на середні (або базові) прогнози.

У суддів 3 і 6, можна побачити велику різницю в α. Це означає, що суддя 3 покладається лише на середнє значення елементів і не бере до уваги середнє значення користувачів. Суддя 6, навпаки, надає більшої ваги середньому значенню користувачів.

Судді 4 і 9 також дуже відрізняються, якщо звернути увагу на параметр β. Це означає, що суддя 4 більше покладається на $rating_{i,u}$, тоді як суддя 9 більше покладається на $rating_{u,i}$.

1.4 Постановка задачі дослідження

Буле проведено соціологічне дослідження користувачів MovieLens та Vodkaster. Мета цього дослідження – краще зрозуміти поведінку користувачів та визначити їх типології.

Ці два набори даних доповнюють один одного, оскільки набір даних MovieLens надає демографічну інформацію про користувачів, а набір даних Vodkaster містить текстові відгуки, пов'язані з рейтингами.

Перейдемо до наших загальних спостережень та розглянемо динаміку середньої оцінки користувачів як функцію часу.

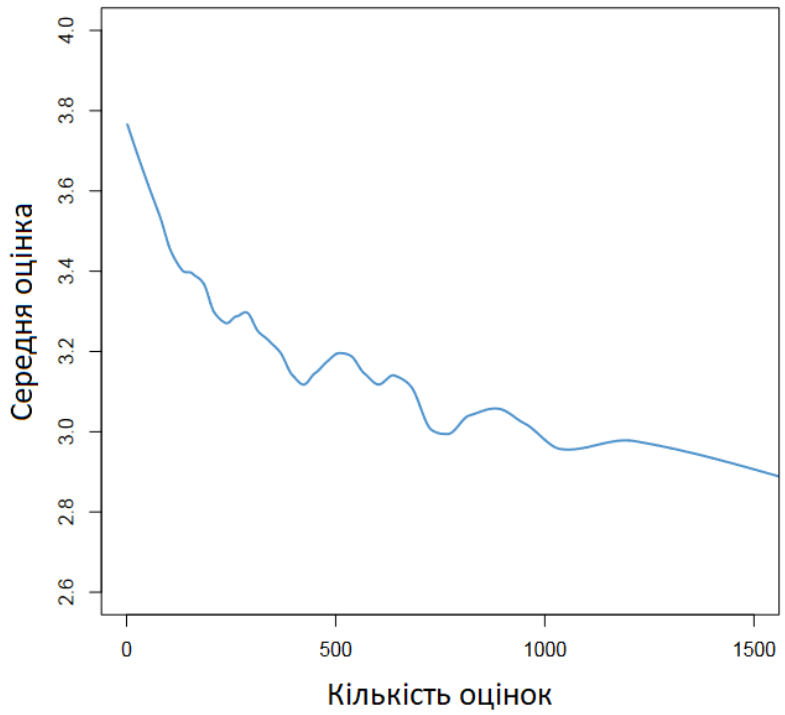


Рисунок 1.3 – Середня оцінка користувачів як функція від кількості оцінок

На цьому графіку (рисунок 1.3) час представлений неявно через зростання кількості оцінок з часом. Оцінки впорядковані в хронологічному порядку, тому можна чітко побачити еволюцію в часі.

Можна побачити, що перші оцінки вищі за решту. Це спостерігається в середньому для кожного користувача.

Припустимо, що користувачі мають певну тенденцію оцінювати фільми, які їм подобаються, першими. Це дозволяє нам ввести поняття «фільмів пантеону». Тобто, чудові фільми, не обов'язково найкращі, з якими користувачі мають особистий зв'язок і люблять їх настільки, що вони визначають смаки користувачів. Це дає нам деяке уявлення про те, які фільми, як правило, оцінюються першими[23].

Також можна побачити еволюцію суворості користувачів. Глобальною тенденцією є зниження середньої оцінки, оскільки користувачі оцінюють більше фільмів. Це спостереження показує, що чим більше користувачі дивляться фільми, тим більше вони стають суворими та критичними.

Перш ніж йти далі, введемо поняття «розрив».

Розрив - це індивідуальний параметр, який визначається як середнє відхилення оцінки користувача від середнього рейтингу фільмів, які він оцінює. Таким чином, розрив є хорошим індикатором для визначення місця користувача відносно середньої оцінки фільмів.

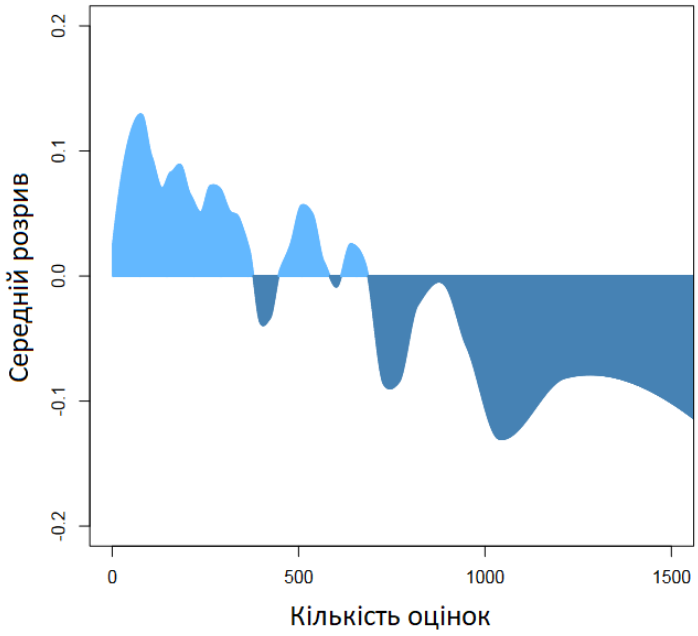


Рисунок 1.4 – Середній розрив як функція від кількості оцінок

Цей графік (рисунок 1.4) показує динаміку розриву в залежності від кількості оцінених фільмів.

Позитивний розрив (більше 0) є відображенням людини, яка в цілому ставить оцінки вищі, ніж середня оцінка об'єктів, які вона оцінює. І навпаки, людина з від'ємним розривом (менше 0) в цілому ставить оцінки нижче, ніж середня оцінка об'єктів, які вона оцінює. Це середній розрив, розрахований для всіх користувачів, отже, ми спостерігаємо глобальну поведінку користувачів.

На рисунку 1.4 можна помітити основну тенденцію - чим більше фільмів люди оцінюють, тим їхній розрив стає від'ємним і зменшується (темно-синя область).

Однак ця тенденція не виглядає лінійною. Наприклад, коли користувачі оцінили приблизно 350-400 фільмів, розрив досить різко зменшується і переходить через нульову лінію. Це означає, що люди стають більш суворими.

Цікаво також спостерігати зростання одразу після падіння. Можна припустити, що люди визначилися зі своїми смаками і відповідно до них обирають фільми.

Розглянемо динаміку середньої оцінки як функції по віку.

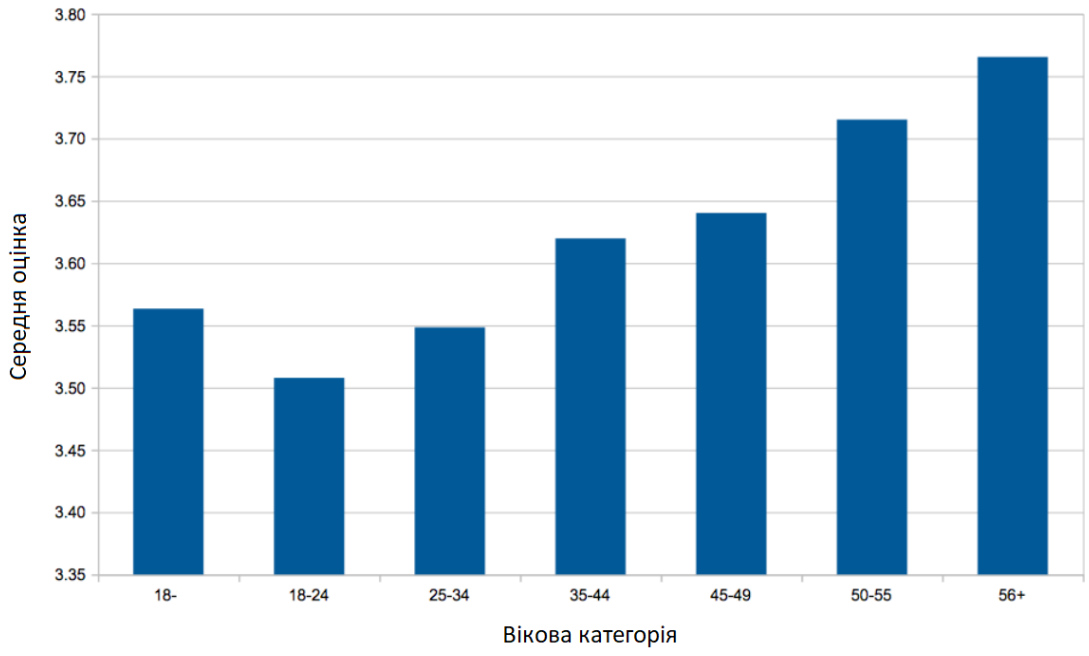


Рисунок 1.5 – Середня оцінка як функція по віку

На цьому графіку (рисунок 1.5) можна побачити, що люди віком 18-24 роки виглядають більш суворими, ніж середній показник. Можливо, з віком користувачі стають все більш толерантними та знижують свої очікування. Або, вони все більше знають свої смаки і не дуже ризикують. Це явище також може бути пов'язане з поєднанням цих двох гіпотез.

Категорія 18-24 роки, схоже, перебуває на стадії дослідження. Люди можуть шукати свої смаки, формуючи свою культурну особистість. Як наслідок, вони схильні відстоювати свої культурні вподобання. Їх молодий культурний вік змушує їх перебувати в парадигмі, де вони або люблять, або ненавидять кіно, бо все ще намагаються визначити, хто вони є[24].

Як результат проведеного соціологічного дослідження, було виявлено, що користувачів слід розглядати як динамічні об'єкти, які розвиваються з часом.

Метою цієї роботи було проаналізувати властивості рекомендаційних систем, існуючи методи побудови рекомендацій та обґрунтувати необхідність адаптувати рекомендації по мірі зміни запитів та вподобань користувачів.

При розробці системи задачею було вирішення наступних проблем:

– люди звикають отримувати рекомендації, але після кількох застарілих рекомендацій, користувачі втратять довіру до рекомендаційної системи;

– смаки та настрої користувачів змінюються з часом і рекомендаційні системи не можуть достатньо швидко реагувати на ці зміни.

Для досягнення поставленої мети вирішуються наступні задачі:

– аналіз існуючих рекомендаційних систем;

– аналіз існуючих методів побудови рекомендацій;

– доповнення методів побудови рекомендацій можливостями динамічною адаптації з урахуванням старіння знань про продукти та послуги а також зміни вподобань користувачів;

– порівняння рекомендацій без динамічної адаптації та з такою адаптацією з метою оцінки ефективності останньої.

2 МЕТОД ДИНАМІЧНОЇ АДАПТАЦІЇ РЕКОМЕНДАЦІЙ З ВИКОРИСТАННЯМ РЕЙТИНГІВ КОРИСТУВАЧІВ

2.1 Загальний підхід до уточнення рекомендацій з використанням рейтингів користувачів

Системи рекомендацій призначені для того, щоб пропонувати користувачам відповідні товари з великої бази даних продуктів. Ці системи адаптуються індивідуально, використовуючи профіль для кожного користувача, створений на основі аналізу попередніх оцінок. Найпоширенішими методами автоматичної рекомендації є фільтрація на основі вмісту (Content-Based Filtering, CBF) та колаборативна фільтрація (Collaborative Filtering, CF). Гібридні системи поєднують методи колаборативної фільтрації та фільтрації на основі вмісту, таким чином використовуючи переваги обох методів[25].

Пропонується ввести механізм «реактивності» в підхід колаборативної фільтрації на основі схожості, який оновлює міри схожості між користувачами та між елементами з певним фактором забування, що дозволяє зменшити важливість старих оцінок. Також пропонується метод адаптивного заповнення матриць, який робить систему дуже гнучкою щодо динамічної поведінки. Матриці факторів динамічно і безперервно оновлюються, щоб надавати рекомендації більше пов’язані з недавнім минулим.

Розглядається випадок, коли немає ніякої іншої інформації, крім набору кортежів <користувач, товар, оцінки>, отже, проблема «холодного старту», коли з'являється абсолютно новий користувач або новий товар, без жодної пов'язаної з ним інформації, не розглядається. Принцип методу полягає в тому, що при отриманні нового спостереження (кортеж <користувач, товар, оцінка>) оновлюються відповідні записи (рядки і стовпці) факторних матриць. Оновлення, має бути якомога ближчим до нового спостереження і бути гладким і послідовним щодо попередніх записів.

2.2 Метод динамічної адаптації рекомендацій з використанням колаборативної фільтрації

Розроблений метод працює з матрицею рекомендацій (рисунок 2.1), між користувачами(u) та продуктами(i). Вона відображає оцінки які користувачі поставили продуктам.

	i_1	i_2	i_3	i_4	i_5
u_1		4		2	1
u_2	3		5	1	
u_3	4		2		1
u_4		1		3	
u_5	1		2		1

Рисунок 2.1 – Приклад матриці рекомендацій

Задача. Побудова рекомендацій для користувача.

Етап 1. Пошук сусідів заданого користувача.

Шаг 1. Розрахування схожості користувачів та товарів до інших.

Міра подібності буда виведена з відстані Мангеттена. Для того щоб отримати міру подібності, ми доповнюємо її до одиниці. Отже, чим ближче оцінки та чим коротша відстань, тим більше схожість прагне до 1. А отже, тим більше користувачі або предмети вважаються схожими.

$$\text{Manhattan}(x, y) = 1 - \frac{\sum_{k \in T_x \cap T_y} |r_{k,x} - r_{k,y}|}{|T_x \cap T_y| \times D}, \quad (2.1)$$

де k – користувач або предмет;

x, y – пара користувачів або предметів;

D – константа, що означає максимальну різницю між двома оцінками.

Як згадувалося раніше, користувачі можуть бути в середньому більш або менш суворими. Щоб врахувати ці відмінності у поведінці, до формули додається різниця $r_x - r_y$ з коефіцієнтом F.

Наприклад, давайте подивимося на схожість між користувачем a і користувачем b. Користувач a має певну схильність бути дуже суворим до об'єктів, які він оцінює. Користувач b, навпаки, більш поблажливий. Ця різниця в поведінці між a та b певним чином пов'язана з різницею в середніх оцінках. Ця неоднорідність враховується в подібності за допомогою коефіцієнта F.

$$\text{Manhattan}(x,y) = 1 - \frac{\sum_{k \in T_x \cap T_y} |r_{k,x} - r_{k,y}| + F\left|\bar{r_x} - \bar{r_y}\right|}{(|T_x \cap T_y| + F)D}, \qquad (2.2)$$

Використаємо коефіцієнт, пропорційний кардинальності перетину $T_x \cap T_y$, як міру довіри. Таким чином, ми надаємо більшу вагу елементам, які мають більшу кількість користувачів.

$$Sim(x,y) = \text{Manhattan}(x,y) \times \left(1 + \epsilon - \frac{1}{|T_x \cap T_y|^{\alpha}}\right), \qquad (2.3)$$

Ця функція призначена для вирішення проблеми реактивності. Таким чином, ми зменшили складність на один ступінь, роблячи нашу систему дуже добре пристосованою до динамічної адаптації.

Шаг 2. Позначення найбільше схожих користувачів сусідами по інтересах.

Оцінки сусідів по інтересах користувача для якого ми генеруємо рекомендації будуть використані в прогнозуванні рейтингу.

Етап 2. Прогнозування рейтингу.

Шаг 1. Зменшення ваги рейтингів залежно від часу їх публікації.

Для того щоб система краще адаптувалася, до рейтингів було застосовано функцію зважування на основі часу. Таким чином, оцінки виставлені пізніше

матимуть більшу вагу в прогнозуванні рейтингу для даного користувача і даного товару.

Визначаємо функцію штрафу за час P:

$$P(r_x) = \frac{1}{(t-t_x)^{\alpha}}, \tag{2.4}$$

де r_x – оцінка;

t_x – дата, коли вона була виставлена;

t – поточна дата.

Складність полягає в тому, яку одиницю виміру використовувати для різниці в часі та α. Було використано місяці та α = 0,33.

Місяці були обрані, тому що це найпростіша шкала часу для інтерпретації. Параметр α був визначений таким чином, що оцінка шестимісячної давності має вдвічі меншу вагу, ніж щойно виставлена оцінка.

Шаг 2. Адаптація до помилок у прогнозуванні рейтингу.

Після прогнозування рейтингу, отримаємо доступ до реального рейтингу. Це дає чіткий зворотній зв'язок про те, наскільки добре працює розроблена система. Тому для кожного користувача ми можемо обчислити середню похибку e_u. Вона визначається наступним чином:

$$e_u = \frac{1}{|S_u|}\sum_{i \in S_u} r_{u,i} - \hat{r}_{u,i}, \tag{2.5}$$

де S_u – множина об'єктів, оцінених користувачем u;

$r_{u,i}$ – оцінка користувача u за об'єкт i;

$\hat{r}_{u,i}$ – прогнозована оцінка користувача u за об'єкт i.

У цьому випадку S_u включає навчальну вибірку, щоб мати більше даних для обчислення оцінки e_u.

Застосувавши цей коригувальний коефіцієнт до зваженої функції рейтингу (рівняння 1.9), представленого в Розділі 1.3, отримаємо наступне рівняння:

$$\hat{r}w_{u,i} = \gamma\hat{r}_{u,i} + (1-\gamma)\left(\alpha\bar{r_u} + (1-\alpha)\bar{r_i}\right) + K \times e_u, \qquad (2.6)$$

де $\hat{r}_{u,i}$ – прогнозована оцінка користувача u за об'єкт i;

r_u – середнє значення користувача;

r_i – середнє значення об'єкта;

α – вага r_u, $(1 - \alpha)$ для r_i;

γ – вага моделі на основі подібності.

Етап 3. Надання рекомендацій.

Використовуючи прогнозовані рейтинги рекомендуємо користувачу продукти з найбільшими прогнозованими рейтингами.

Перейдемо до адаптивного заповнення матриці рекомендацій.

Розглянемо один з методів матричної факторизації в системах колаборативної фільтрації. Цей метод матричної факторизації полягає в тому щоб розкласти матрицю рекомендацій на добуток матриці факторів користувачів (вподобання користувачів) та матриці факторів товарів (характеристики товарів), а потім вирішити задачу мінімізації середньоквадратичної похибки на проставлених оцінках (рисунок 2.2).

В результаті отримаємо заповнену приближеними оцінками матрицю рекомендацій.

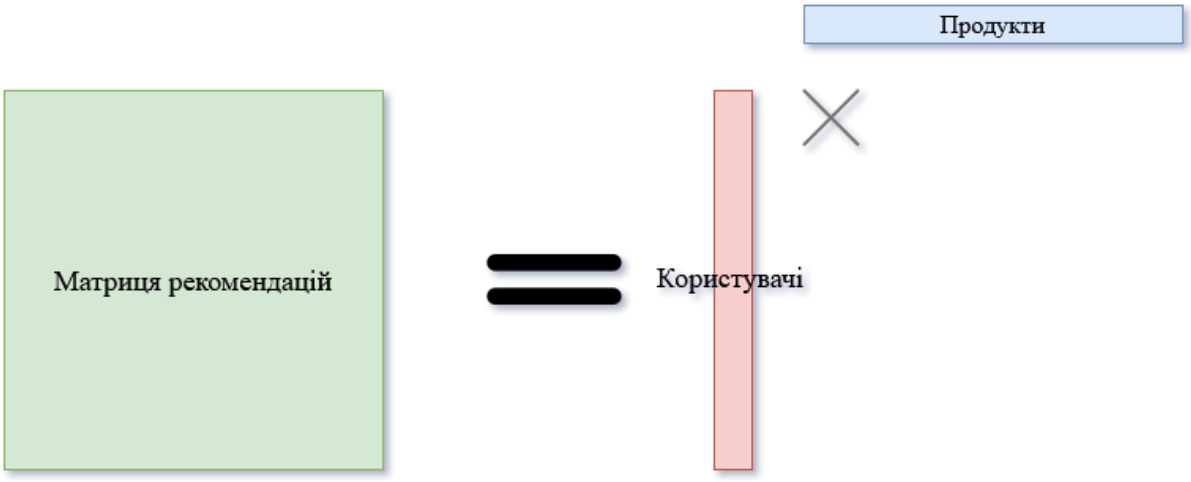


Рисунок 2.2 – Матрична факторизація

Нехай R – це матриця оцінок $n \times m$ (n користувачів, m позицій), в якій, багато пропущених даних. Використовуючи підхід матричної факторизації апроксимуємо матрицю R, матрицею низького рангу L, та вводимо специфічні для користувача та елемента упередження (суб'єктивне упередження користувача та популярність елемента), позначені як a і b. Отримаємо наступну задачу мінімізації:

$$min \sum_{(i,j)\in\omega} \left(r_{i,j} - m - a_i - b_j - \sum_{k=1}^{K} L_{i,k}R_{j,k}\right)^2 + \mu_a \| \mathbf{a} \|^2 + \mu_b \| \mathbf{b} \|^2 + \mu_L \| \mathbf{L} \|_F^2 + \mu_R \| \mathbf{R} \|_F^2, \quad (2.7)$$

де ω – множина доступних кортежів оцінок;

m – середня оцінка за ω;

a_i, b_j, $r_{i,j}$, $L_{i,k}$ та $R_{j,k}$ – відповідно елементи a, b, R, L та R, що відповідають користувачу i, товару j та латентному фактору k;

$\|M\|_F^2$ - норма Фробеніуса (квадратний корінь сум квадратів модулів елементів матриці розміру m × n) матриці M.

Розглянемо адаптацію a_i та b_j.

Спочатку розглянемо просту модель, яка включає лише упередження продукту та користувача, а потім опишемо її розширення до повної моделі на

основі матричної факторизації. При отриманні нового кортежу $< i, j, r_{i,j} >$ оновлюємо a_i та b_j, мінімізуючи наступний критерій:

$$min\left(r_{i,j} - m - a_i - b_j\right)^2 + \alpha_1(a_i - \tilde{a}_i)^2 + \beta_1\left(b_j - \tilde{b}_j\right)^2, \qquad (2.8)$$

де $\tilde{a}_i$ та $\tilde{b}_j$ – значення до адаптації.

Цей критерій є компромісом між якістю апроксимації по відношенню до нового спостереження та плавністю еволюції зміщень. Для нових користувачів та об'єктів $\tilde{a}_i$ та $\tilde{b}_j$ дорівнюють 0. Значення α_1 та β_1 отримуються шляхом сіткового пошуку на навчальній вибірці, яка є хронологічно пізнішою за навчальну.

Розв'язання цієї оптимізаційної задачі приводить до наступних простих рівнянь оновлення:

$$a_i = \frac{(\alpha_1 + \alpha_1/\beta_1)\tilde{a}_i + r_{i,j} - m - \tilde{b}_j}{1 + \alpha_1 + \alpha_1/\beta_1}, \qquad (2.9)$$

$$b_j = \frac{(\beta_1 + \beta_1/\alpha_1)\tilde{b}_j + r_{i,j} - m - \tilde{a}_i}{1 + \beta_1 + \beta_1/\alpha_1}, \qquad (2.10)$$

L та R також адаптуються відповідно до тієї самої ідеї: отримуємо новий кортеж $< i, j, r_{i,j} >$ а потім оновлюємо L_i та R_j (відповідно i-й рядок L та j-й рядок R) таким чином, що:

$$min\left(\widehat{r_{i,j}} - \sum_k L_{i,k}R_{j,k}\right)^2 + \alpha_2\|L_i - \tilde{L_i}\|_F^2 + \beta_2\|R_j - \tilde{R}_j\|_F, \qquad (2.11)$$

де $\widehat{r_{i,j}} = r_{i,j} - m - a_i - b_j$ (залишковий рейтинг);

L_i та R_j – значення відповідних рядків до адаптації.

Для нових користувачів та товарів значення $\widetilde{L_\iota}$ та $\tilde{R}_j$ дорівнюють 0. Значення α_2 та β_2 отримано шляхом сіткового пошуку на навчальній вибірці.

Рівняння оновлення виглядають так:

$$L_i^{(t)} = \left(\alpha_2 I + R_j^{(t-1)} \cdot R_j^{(t-1)}\right)^{-1} \cdot \left(\alpha_2 L_i^{(t-1)} + \widehat{r_{\iota,J}} \cdot R_j^{(t-1)}\right), \quad (2.12)$$

$$R_j^{(t)} = \left(\beta_2 I + L_i^{\prime(t-1)} \cdot L_i^{(t-1)}\right)^{-1} \cdot \left(\beta_2 R_j^{(t-1)} + \widehat{r_{\iota,J}} \cdot L_i^{(t)}\right), \quad (2.13)$$

де $L_i^{(0)} = \widetilde{L_\iota}$;

$R_j^{(0)} = \tilde{R}_j$.

Експериментально, для всіх наборів даних, які були використані і відповідних значень α_2 і β_2, було достатньо двох або трьох ітерацій для збіжності.

3 РОЗРОБКА ПРОГРАМНОГО ПРОЕКТУ МОДУЛЮ ДИНАМІЧНОЇ АДАПТАЦІЇ РЕКОМЕНДАЦІЙ В СИСТЕМІ ЕЛЕКТРОННОЇ КОМЕРЦІЇ

3.1 Визначення життєвого циклу проекту, його окремі фази та етапи

Життєвий цикл – це стадії, які проходить програмний продукт від появи ідеї до її реалізації в коді, імплементації в бізнес та подальшої підтримки. Моделі життєвого циклу багато в чому визначають методології розробки[26].

Існує певна варіативність у проходженні етапів ЖЦ під час розробки та застосування товару на ринку. Для кожного продукту це відбувається по-своєму, але щоб процесом якось управляти були сформульовані моделі життєвого циклу – спрощене та узагальнене уявлення про те, як розвивається продукт. В таблиці 3.1 зображено зміст фаз життєвого циклу проекту.

Таблиця 3.1 – Зміст фаз життєвого циклу проекту

Фаза	Ініціація	Планування	Виконання та контроль	Завершення
Початок фази	31.01.2022	07.02.2022	07.03.2022	07.04.2022
Закінчення фази	04.02.2022	04.03.2022	04.04.2022	11.04.2022
Перелік основних робіт	Визначення цілей, специфікації, задач, відповідальності, команди проекту.	Визначення розкладу, бюджету, ресурсів, ризиків та учасників проекту.	Розробка програми та проведення тестування.	Презентація виконаного проекту замовнику, передача йому документації.
Ключові віхи	Затвердження вимог та команди проекту.	Затвердження розкладу, бюджету та інших ресурсів.	Готовність до релізу програми.	Прийняття замовником готового продукту.
Складності	Чітке визначення задач та вимог стейкхолдерів.	Планування необхідних ресурсів в ході реалізації проекту.	Дотримання термінів в ході розробки та тестування системи.	Презентація виконаної роботи.

Наведений життєвий цикл проекту допомагає оптимізувати час, поліпшити комунікацію між командою та замовниками, бути впевненим, що команда слідує запланованим термінам, а також, можна керувати ризиками та мінімізувати їх.

Перейдемо до визначення складу учасників проекту.

Учасники проекту реалізують різні інтереси у процесі здійснення проекту, формують власні вимоги відповідно до цілей та мотивації і впливають на проект, виходячи зі своїх інтересів, компетенцій та ступеню залучення до проекту

Зацікавлених сторін проекту зазвичай поділяють на зовнішніх та внутрішніх, але замовником нашого проекту є генеральний директор нашої компанії і розробляється він для одного з існуючих проектів нашої компанії[27].

3.2. Структуризація проекту

Уточнення дерева цілей та створення ієрархічної структури декомпозиції робіт (WBS) (рисунок 3.1).

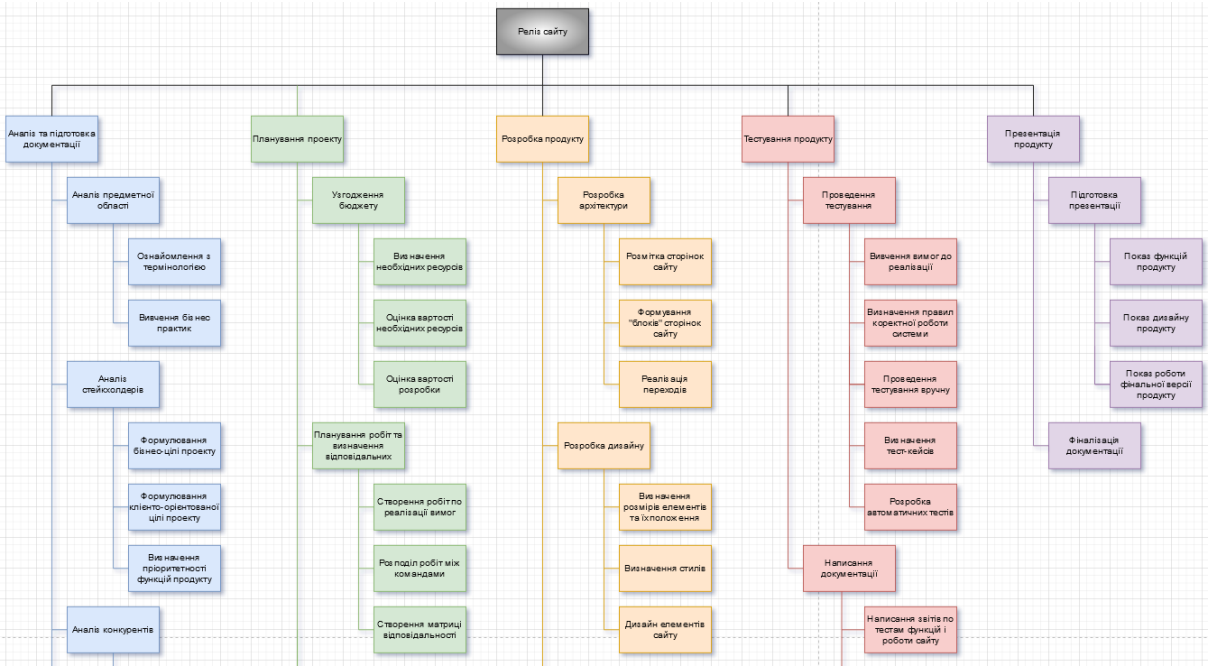


Рисунок 3.1 – Ієрархічна структура декомпозиції робіт

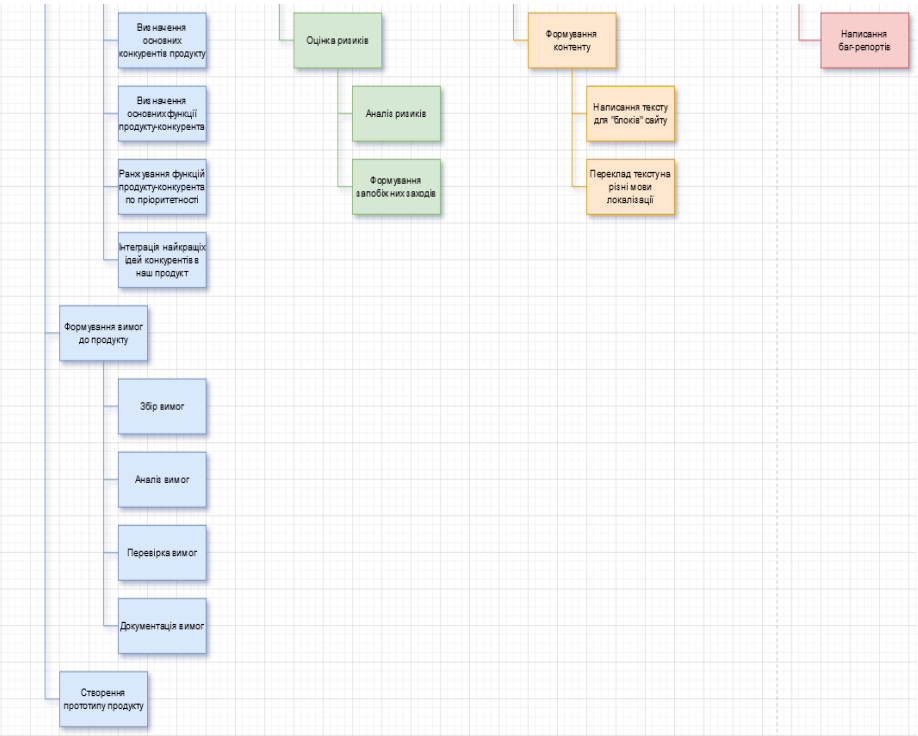


Рисунок 3.1, аркуш 2

Зазначена вище діаграма являє собою уточнення першого варіанту дерева цілей у вигляді ієрархічної структури декомпозиції робіт, а саме WBS. WBS являє собою ієрархічну декомпозицію повного змісту робіт, що виконуються командою проекту для досягнення цілей проекту і створення необхідних результатів, що поставляються.

3.3 Побудова організаційної структури виконавців

Організаційна ієрархічна структура (OBS) схожа на WBS, але організована не за результатами проекту, а відповідно до наявної структури підрозділів організації (відділів, груп або команд). Під кожним відділом наведено список операцій проекту або пакети робіт. Таким чином, конкретний функціональний відділ може дізнатися про всі свої обов'язки проекту (наприклад, відділ інформаційних технологій або відділ закупівель), вивчивши свою частину OBS.

На рисунку 3.2 наведена організаційна ієрархічна структура проекту.

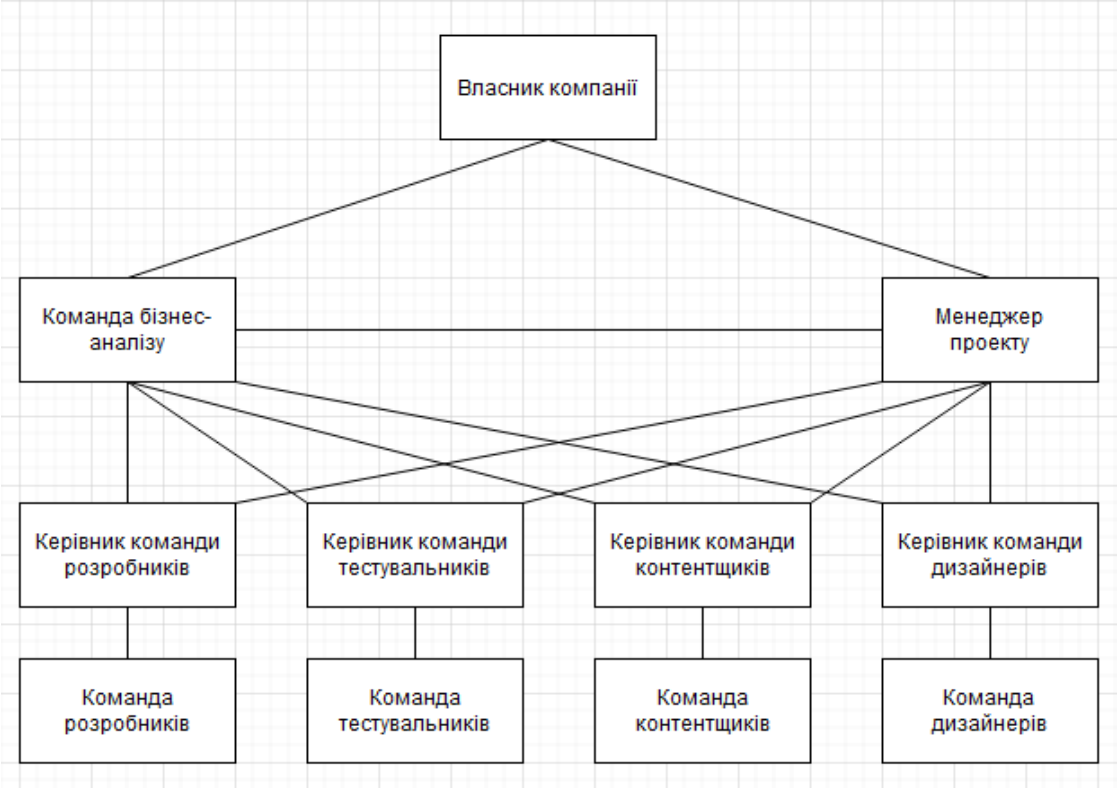


Рисунок 3.2 – Організаційна ієрархічна структура проекту

Після формування ієрархічної структури декомпозиції робіт та організаційної ієрархічної структури проекту можна сформувати матрицю відповідальності.

3.4 Ув’язка WBS з OBS на основі побудови матриці відповідальності

Після проектування WBS та OBS можна розробити їх ув’язку та побудувати матрицю відповідальності.

Визначені ролі проекту:

– ВО – відповідальна особа;

– В – виконавець;

– К – контроль;

– У – узгодження.

Скорочення:

– ВК – Власник компанії;

– БА – Команда бізнес-аналізу;

– МП – Менеджер проекту;

– ККР – Керівник команди розробників;

– КР – Команда розробників;

– ККТ – Керівник команди тестувальників;

– КТ – Команда тестувальників;

– ККК – Керівник команди контентщиків;

– КК – Команда контентщиків;

– ККД – Керівник команди дизайнерів;

– КД – Команда дизайнерів.

Для ув’язки пакетів робіт (WBS) з організаціями-виконавцями (OBS) була побудована матриці відповідальності наведена у таблиці 3.2.

Таблиця 3.2 – Матриця відповідальності

№	Етапи реалізації проекту	Учасники проекту										
		ВК	БА	МП	ККР	КР	ККТ	КТ	ККК	КК	ККД	КД
1	Аналіз та підготовка документації	У,К	ВОВ									
1.1	Аналіз предметної області	У,К	ВОВ									
1.2	Аналіз стейкхолдерів	У,К	ВОВ		К				К		К	
1.3	Аналіз конкурентів	У,К	ВОВ									
1.4	Формування вимог	У,К	ВОВ	У,К	К				К		К	
1.5	Створення прототипу	У	ВОВ		К				К		К	
2	Планування проекту	У,К	У,К	ВОВ	У,К		У		У,К		У,К	

Кінець таблиці 3.2

№	Етапи реалізації проекту	Учасники проекту										
		ВК	БА	МП	ККР	КР	ККТ	КТ	ККК	КК	ККД	КД
2.1	Узгодження бюджету	У,К	У,К	ВОВ	У		У		У		У	
2.2	Планування робіт	У	У,К	ВОВ	У,К		У,К		У,К		У,К	
2.3	Оцінка ризиків	У,К	У,К	ВОВ	У		У		У		У	
3	Розробка продукту	У	У,К	ВОУ,К	ВО	В	У,К		ВО	В	ВО	В
3.1	Розробка архітектури	У	К	У,К	ВО	В	У		У		У,К	
3.2	Розробка дизайну	У,К	К	У,К	У		У		У,К		ВО	В
3.3	Формування контенту	У,К	К	У,К	У		У		ВО	В	У	
4	Тестування продукту	У	У	У	У,К		ВО	В	У		У	
4.1	Проведення тестування	У	У	У	У		ВО	В	У		У	
4.2	Написання документації	У	У	У			ВО	В	У		У	
5	Презентація продукту	У	ВОВ	У,К	У		У		У		У	
5.1	Підготовка презентації	У	ВОВ	У,К	У		У		У		У	
5.2	Фіналізація документації	У	ВОВ	У,К	У		У		У		У	

3.5 Побудова структури споживаних ресурсів

Структура розбивки ресурсів RBS описує організацію ресурсів проекту і може використовуватися разом із WBS, для визначення призначень робочого пакету. Структура RBS наведена на рисунку 3.3.

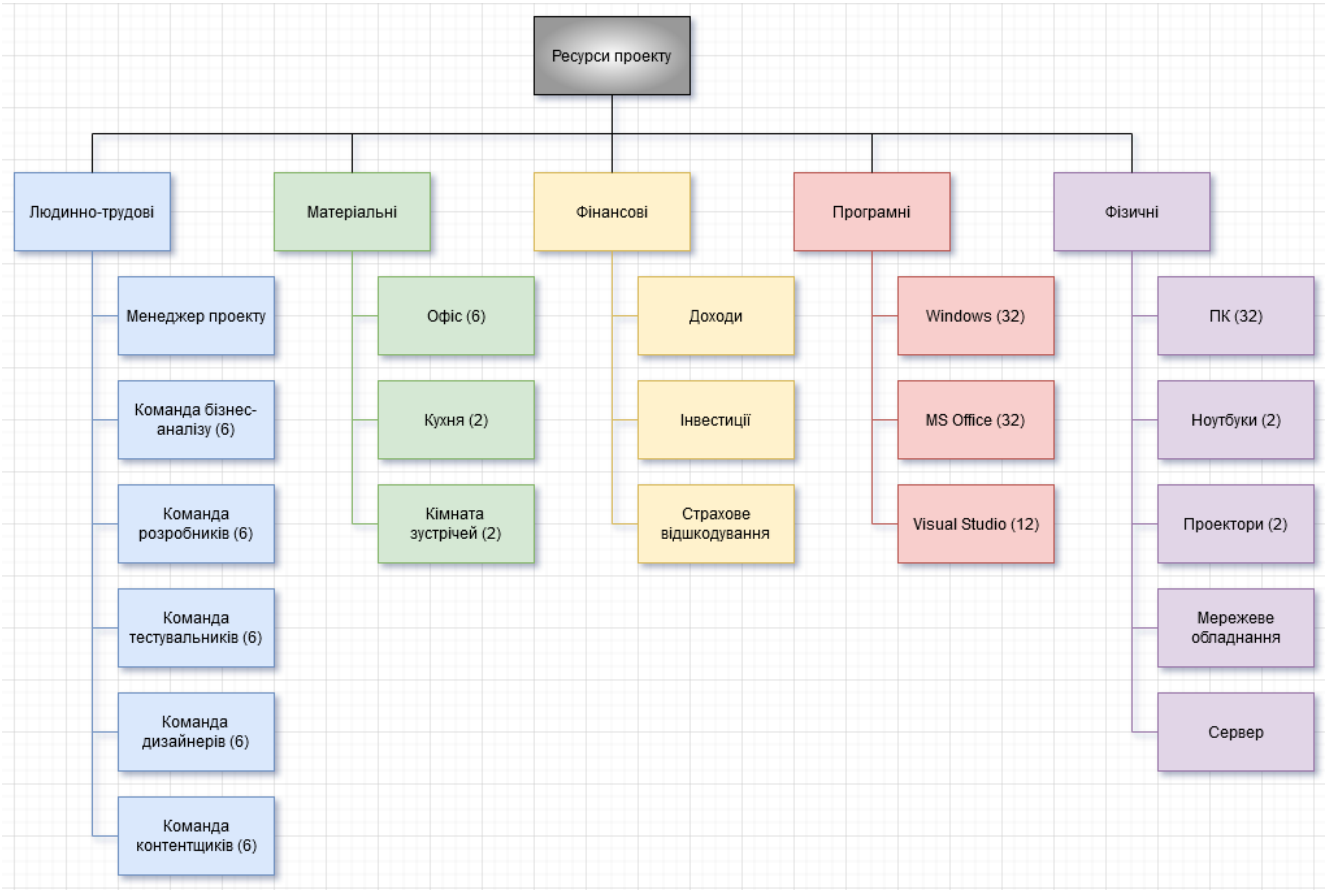


Рисунок 3.3 – Структура споживаних ресурсів

В результаті були виявлені основні ресурси необхідні для успішної та продуктивної реалізації проекту.

3.6 Створення структури вартості

Фінансові ресурси утворюють структуру вартості ABS, яка наведена на рисунку 3.4.

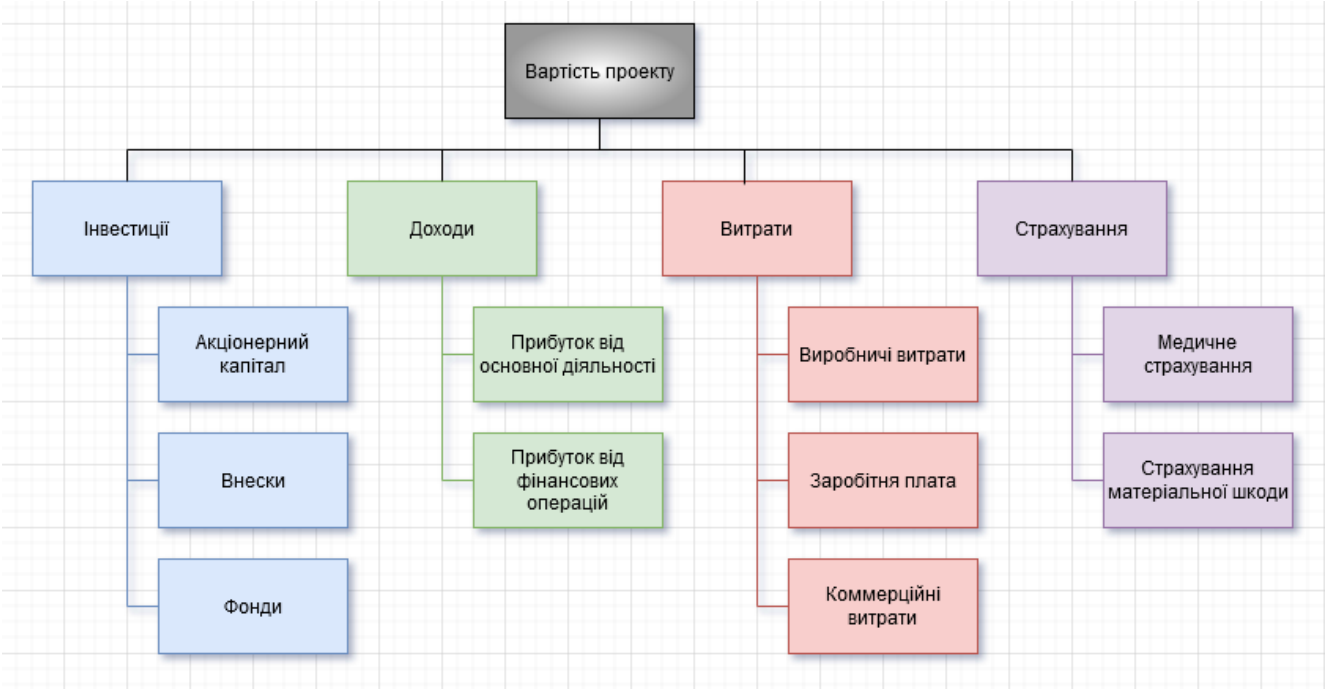


Рисунок 3.4 – Структура вартості проекту

В результаті були наведені основні пункти щодо вартості розробляємого проекту.

3.7 Склад та порядок розробки проектно-кошторисної документації

В цьому проекті за рахунок засобів замовника здійснюється підготовка документації та фінансування діяльності. Робоча документація для розробки сайту розробляється відповідно до державних стандартів. Склад робочої документації може уточнюватися замовником і виконавцем робіт, що обумовлюється договором, звичайно, це також залежить від специфіки проектованої системи й умов здійснення проектування. Посилальні документи до складу робочої документації не входять і можуть передаватися замовнику, якщо це обумовлено та передбачено в договорі.

Усі роботи, у тому числі і проектні, виконуються на підставі контрактів або договорів, укладених між компанією та замовником. Контракт, як правило, можна укладати на виконання передпроектних робіт, комплексу проектних робіт, дослідницьких робіт, окремих стадій та розділів проекту. Розробка проектної документації без дослідження основних вимог системи не допускається.

Проектування системи здійснюється за дотриманням законодавства України на підставі вихідних даних.

Порядок розробки проектної документації поділяється на:

– розробка проектної документації (проектування) може здійснюватися фізичними та юридичними особами, які мають ліцензії на відповідні види діяльності у сфері інформаційних технологій, тестування, планування та проектування;

– розробка проектної документації здійснюється на підставі контракту на виконання проектних робіт, що укладається між замовником та компанією виконавцем у порядку, встановленому законодавством, а також на підставі завдання на проектування, що складається відповідно до норм та затверджується замовником;

– у контракті обов'язково вказуються вид та обсяг послуг, які керівник команди має намір передати на виконання розробникам та тестувальникам;

– завдання на проектування складається замовником або його уповноваженою особою та затверджується замовником;

– завдання на проектування за дорученням замовника може бути підготовлене виконавцем (розробником, керівником проекту) та стає обов'язковим для сторін з моменту його затвердження замовником;

– разом із завданням на проектування замовник видає підрядний проектній організації (генеральному проектувальнику) вихідні матеріали (дані) для розробки проектної документації.

Наступним кроком є створення логіко-інформаційної схеми розробки проектно-кошторисної документації.

3.8 Логіко-інформаційна схема розробки проектно-кошторисної документації

Логіко-інформаційна схема графічно зображує процес управління підприємством, що дозволяє регламентувати цей процес за вхідною та вихідною інформацією, виконавцям, споживачам, показати зв'язки між ними та відобразити особливості тієї чи іншої арбітражної процедури.

В таблиці 4.1 наведена логіко-інформаційна схема управління процесом проектування системи.

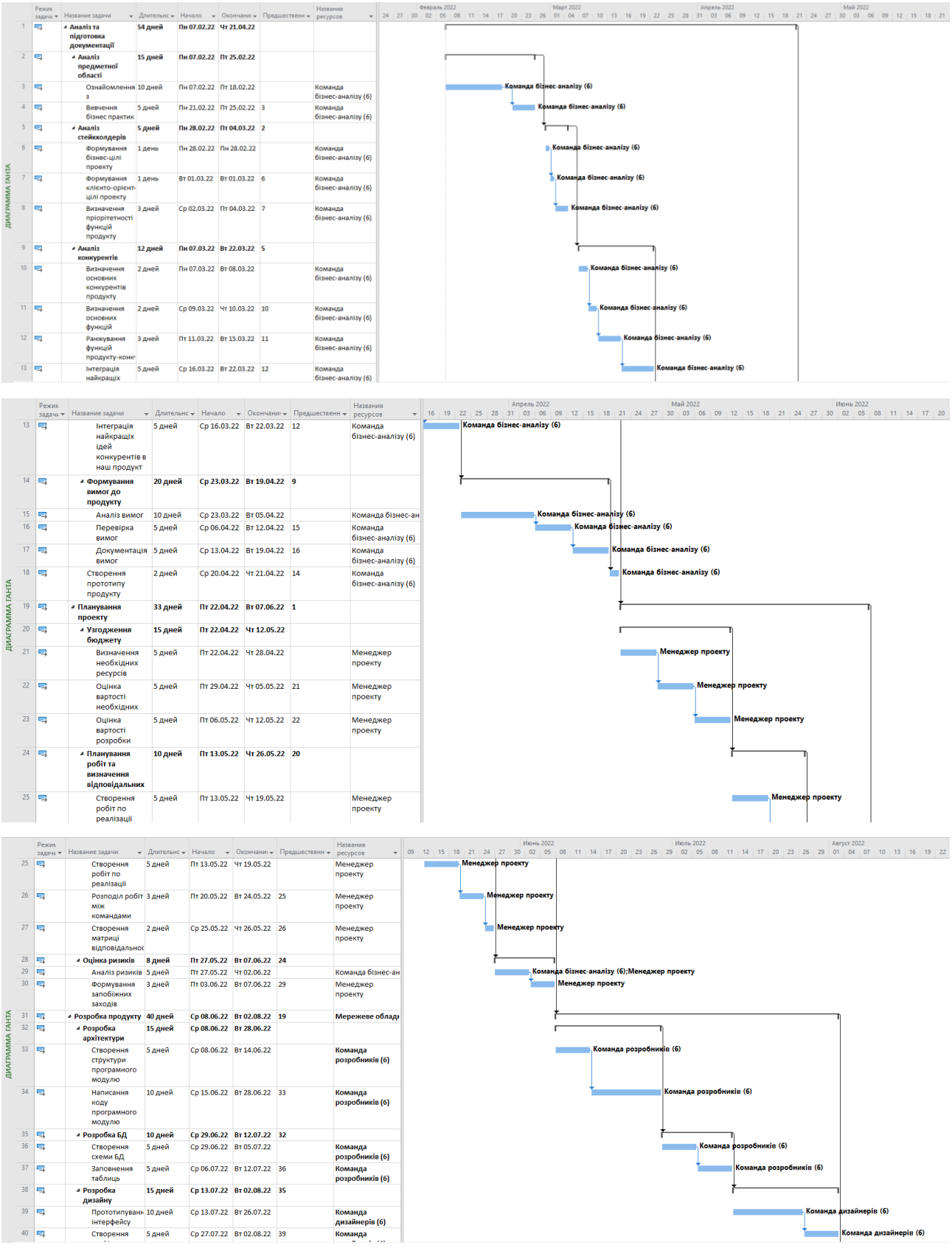
Таблиця 3.3 – Логіко-інформаційна схема управління процесом проектування

№	Задачі процесу проектування	Вихідна інформація	Виконавець задачі	Документ, що є результатом виконання задачі	Споживач результату
1.	Обґрунтування необхідності створення системи	Інформація про необхідність створення системи	Замовник	Заявка на проведення розробки системи	Компанія виконавець
2.	Формування команди розробки системи	Інформація про вимоги щодо команди фахівців	Керівник проекту, менеджер проекту	Документ про склад робітників у проекті	Замовник
3.	Визначення основних вимог системи	Перелік конкретних вимог щодо функціональності системи	Замовник, менеджер проекту, керівник проекту	Документ про вимоги до інформаційної системи	Розробники, тестувальники
4.	Аналіз предметної області	Інформація про аналоги системи, аналіз їх переваг та недоліків	Команда розробників	Документ, що описує сучасний стан предметної області	Керівник проекту
5.	Планування ресурсів	Інформація про вимоги проекту	Власник компанії, менеджер проекту	Документ про визначені ресурси проекту	Замовник
5.	Проектування системи	Інформація про вимоги щодо роботи системи	Команда розробників	Документ з описом розробленої системи	Замовник
5.	Проведення тестування системи	Інформація про необхідні тести для системи	Команда тестувальників	Документ про результати тестів	Керівник проекту
6.	Презентація готової системи	Інформація про розроблену систему	Команда розробників, керівник проекту	Документ про закінчення розробки системи	Замовник

3.9 Розробка моделі проекту

Діаграма Ганта – це популярний тип стовпчастих діаграм, який використовується для ілюстрації плану, графіка робіт з будь-якого проекту. Є одним із методів планування проектів. Використовується в програмах управління проектами. Такі діаграми дуже розповсюджені і їх порівняно легко читати[27].

За допомогою MS Project була розроблена діаграма Ганта проекту (рисунок 3.5).



	Режим задач	Название задачи	Длительнс	Начало	Окончани	Предшественн	Названия ресурсов
1		**Аналіз та підготовка документації**	**54 дней**	**Пн 07.02.22**	**Чт 21.04.22**		
2		**Аналіз предметної області**	**15 дней**	**Пн 07.02.22**	**Пт 25.02.22**		
3		Ознайомлення з	10 дней	Пн 07.02.22	Пт 18.02.22		Команда бізнес-аналізу (6)
4		Вивчення бізнес практик	5 дней	Пн 21.02.22	Пт 25.02.22	3	Команда бізнес-аналізу (6)
5		**Аналіз стейкхолдерів**	**5 дней**	**Пн 28.02.22**	**Пт 04.03.22**	**2**	
6		Формування бізнес-цілі проекту	1 день	Пн 28.02.22	Пн 28.02.22		Команда бізнес-аналізу (6)
7		Формування клієнто-орієнт цілі проекту	1 день	Вт 01.03.22	Вт 01.03.22	6	Команда бізнес-аналізу (6)
8		Визначення пріорітетності функцій продукту	3 дней	Ср 02.03.22	Пт 04.03.22	7	Команда бізнес-аналізу (6)
9		**Аналіз конкурентів**	**12 дней**	**Пн 07.03.22**	**Вт 22.03.22**	**5**	
10		Визначення основних конкурентів продукту	2 дней	Пн 07.03.22	Вт 08.03.22		Команда бізнес-аналізу (6)
11		Визначення основних функцій	2 дней	Ср 09.03.22	Чт 10.03.22	10	Команда бізнес-аналізу (6)
12		Ранжування функцій продукту-конк	3 дней	Пт 11.03.22	Вт 15.03.22	11	Команда бізнес-аналізу (6)
13		Інтеграція найкращіх	5 дней	Ср 16.03.22	Вт 22.03.22	12	Команда бізнес-аналізу (6)
13		Інтеграція найкращіх ідей конкурентів в наш продукт	5 дней	Ср 16.03.22	Вт 22.03.22	12	Команда бізнес-аналізу (6)
14		**Формування вимог до продукту**	**20 дней**	**Ср 23.03.22**	**Вт 19.04.22**	**9**	
15		Аналіз вимог	10 дней	Ср 23.03.22	Вт 05.04.22		Команда бізнес-ан
16		Перевірка вимог	5 дней	Ср 06.04.22	Вт 12.04.22	15	Команда бізнес-аналізу (6)
17		Документація вимог	5 дней	Ср 13.04.22	Вт 19.04.22	16	Команда бізнес-аналізу (6)
18		Створення прототипу продукту	2 дней	Ср 20.04.22	Чт 21.04.22	14	Команда бізнес-аналізу (6)
19		**Планування проекту**	**33 дней**	**Пт 22.04.22**	**Вт 07.06.22**	**1**	
20		**Узгодження бюджету**	**15 дней**	**Пт 22.04.22**	**Чт 12.05.22**		
21		Визначення необхідних ресурсів	5 дней	Пт 22.04.22	Чт 28.04.22		Менеджер проекту
22		Оцінка вартості необхідних	5 дней	Пт 29.04.22	Чт 05.05.22	21	Менеджер проекту
23		Оцінка вартості розробки	5 дней	Пт 06.05.22	Чт 12.05.22	22	Менеджер проекту
24		**Планування робіт та визначення відповідальних**	**10 дней**	**Пт 13.05.22**	**Чт 26.05.22**	**20**	
25		Створення робіт по реалізації	5 дней	Пт 13.05.22	Чт 19.05.22		Менеджер проекту
25		Створення робіт по реалізації	5 дней	Пт 13.05.22	Чт 19.05.22		Менеджер проекту
26		Розподіл робіт між командами	3 дней	Пт 20.05.22	Вт 24.05.22	25	Менеджер проекту
27		Створення матриці відповідальнос	2 дней	Ср 25.05.22	Чт 26.05.22	26	Менеджер проекту
28		**Оцінка ризиків**	**8 дней**	**Пт 27.05.22**	**Вт 07.06.22**	**24**	
29		Аналіз ризиків	5 дней	Пт 27.05.22	Чт 02.06.22		Команда бізнес-ан
30		Формування запобіжних заходів	3 дней	Пт 03.06.22	Вт 07.06.22	29	Менеджер проекту
31		**Розробка продукту**	**40 дней**	**Ср 08.06.22**	**Вт 02.08.22**	**19**	Мережеве облад
32		**Розробка архітектури**	**15 дней**	**Ср 08.06.22**	**Вт 28.06.22**		
33		Створення структури програмного модулю	5 дней	Ср 08.06.22	Вт 14.06.22		**Команда розробників (6)**
34		Написання коду програмного модулю	10 дней	Ср 15.06.22	Вт 28.06.22	33	**Команда розробників (6)**
35		**Розробка БД**	**10 дней**	**Ср 29.06.22**	**Вт 12.07.22**	**32**	
36		Створення схеми БД	5 дней	Ср 29.06.22	Вт 05.07.22		**Команда розробників (6)**
37		Заповнення таблиць	5 дней	Ср 06.07.22	Вт 12.07.22	36	**Команда розробників (6)**
38		**Розробка дизайну**	**15 дней**	**Ср 13.07.22**	**Вт 02.08.22**	**35**	
39		Прототипуванн інтерфейсу	10 дней	Ср 13.07.22	Вт 26.07.22		**Команда дизайнерів (6)**
40		Створення	5 дней	Ср 27.07.22	Вт 02.08.22	39	**Команда**

Рисунок 3.5 – Діаграма Ганта

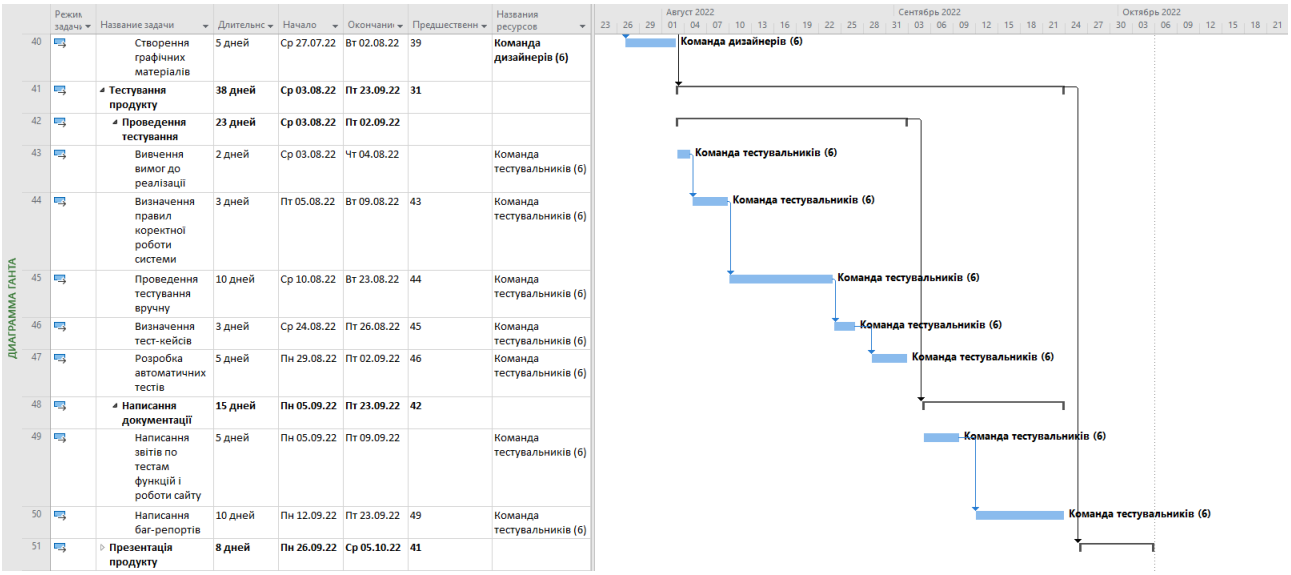


Рисунок 3.5, аркуш 2

В таблиці 3.4 наведена декомпозиція робіт діаграми Ганта.

Таблиця 3.4 – Декомпозиція робіт діаграми Ганта

Назва роботи	Тривалість	Початок	Кінець	Попередні роботи	Назви ресурсів
Аналіз та підготовка документації	**54 дні**	**Пн 07.02.22**	**Чт 21.04.22**		
Аналіз предметної області	**15 днів**	**Пн 07.02.22**	**Пт 25.02.22**		
Ознайомлення з термінологією	10 днів	Пн 07.02.22	Пт 18.02.22		Команда бізнес-аналізу (6)
Вивчення бізнес практик	5 днів	Пн 21.02.22	Пт 25.02.22	3	Команда бізнес-аналізу (6)
Аналіз стейкхолдерів	**5 днів**	**Пн 28.02.22**	**Пт 04.03.22**	**2**	
Формування бізнес-цілі проекту	1 день	Пн 28.02.22	Пн 28.02.22		Команда бізнес-аналізу (6)
Формування клієнто-орієнтованої цілі проекту	1 день	Вт 01.03.22	Вт 01.03.22	6	Команда бізнес-аналізу (6)
Визначення пріорітетності функцій продукту	3 дня	Ср 02.03.22	Пт 04.03.22	7	Команда бізнес-аналізу (6)
Аналіз конкурентів	**12 днів**	**Пн 07.03.22**	**Вт 22.03.22**	**5**	
Визначення основних конкурентів продукту	2 дня	Пн 07.03.22	Вт 08.03.22		Команда бізнес-аналізу (6)
Визначення основних функцій продукту-конкурента	2 дня	Ср 09.03.22	Чт 10.03.22	10	Команда бізнес-аналізу (6)

Продовження таблиці 3.4

Назва роботи	Тривалість	Початок	Кінець	Попередні роботи	Назви ресурсів
Ранжування функцій продукту-конкурента по пріоритетності	3 дня	Пт 11.03.22	Вт 15.03.22	11	Команда бізнес-аналізу (6)
Інтеграція найкращіх ідей конкурентів в наш продукт	5 днів	Ср 16.03.22	Вт 22.03.22	12	Команда бізнес-аналізу (6)
Формування вимог до продукту	**20 днів**	**Ср 23.03.22**	**Вт 19.04.22**	**9**	
Аналіз вимог	10 днів	Ср 23.03.22	Вт 05.04.22		Команда бізнес-аналізу (6)
Перевірка вимог	5 днів	Ср 06.04.22	Вт 12.04.22	15	Команда бізнес-аналізу (6)
Документація вимог	5 днів	Ср 13.04.22	Вт 19.04.22	16	Команда бізнес-аналізу (6)
Створення прототипу продукту	2 дня	Ср 20.04.22	Чт 21.04.22	14	Команда бізнес-аналізу (6)
Планування проекту	**33 дня**	**Пт 22.04.22**	**Вт 07.06.22**	**1**	
Узгодження бюджету	**15 днів**	**Пт 22.04.22**	**Чт 12.05.22**		
Визначення необхідних ресурсів	5 днів	Пт 22.04.22	Чт 28.04.22		Менеджер проекту
Оцінка вартості необхідних ресурсів	5 днів	Пт 29.04.22	Чт 05.05.22	21	Менеджер проекту
Оцінка вартості розробки	5 днів	Пт 06.05.22	Чт 12.05.22	22	Менеджер проекту
Планування робіт та визначення відповідальних	**10 днів**	**Пт 13.05.22**	**Чт 26.05.22**	**20**	
Створення робіт по реалізації вимог	5 днів	Пт 13.05.22	Чт 19.05.22		Менеджер проекту
Розподіл робіт між командами	3 дня	Пт 20.05.22	Вт 24.05.22	25	Менеджер проекту
Створення матриці відповідальності	2 дня	Ср 25.05.22	Чт 26.05.22	26	Менеджер проекту
Оцінка ризиків	**8 днів**	**Пт 27.05.22**	**Вт 07.06.22**	**24**	
Аналіз ризиків	5 днів	Пт 27.05.22	Чт 02.06.22		Команда бізнес-аналізу (6);Менеджер проекту
Формування запобіжних заходів	3 дня	Пт 03.06.22	Вт 07.06.22	29	Менеджер проекту
Розробка продукту	**40 днів**	**Ср 08.06.22**	**Вт 02.08.22**	**19**	**Мережеве обладнання[1];Ноутбуки (2)[1];ПК (32)[1];Проектори (2)[1]**
Розробка архітектури	**15 днів**	**Ср 08.06.22**	**Вт 28.06.22**		
Створення структури програмного модулю	5 днів	Ср 08.06.22	Вт 14.06.22		**Команда розробників (6)**

Кінець таблиці 3.4

Назва роботи	Тривалість	Початок	Кінець	Попередні роботи	Назви ресурсів
Написання коду програмного модулю	10 днів	Ср 15.06.22	Вт 28.06.22	33	**Команда розробників (6)**
Розробка БД	**10 днів**	**Ср 29.06.22**	**Вт 12.07.22**	**32**	
Створення схеми БД	5 днів	Ср 29.06.22	Вт 05.07.22		**Команда розробників (6)**
Заповнення таблиць	5 днів	Ср 06.07.22	Вт 12.07.22	36	**Команда розробників (6)**
Розробка дизайну	**15 днів**	**Ср 13.07.22**	**Вт 02.08.22**	**35**	
Прототипування інтерфейсу	10 днів	Ср 13.07.22	Вт 26.07.22		**Команда дизайнерів (6)**
Створення графічних матеріалів	5 днів	Ср 27.07.22	Вт 02.08.22	39	**Команда дизайнерів (6)**
Тестування продукту	**38 днів**	**Ср 03.08.22**	**Пт 23.09.22**	**31**	
Проведення тестування	**23 дні**	**Ср 03.08.22**	**Пт 02.09.22**		
Вивчення вимог до реалізації	2 дня	Ср 03.08.22	Чт 04.08.22		Команда тестувальників (6)
Визначення правил коректної роботи системи	3 дня	Пт 05.08.22	Вт 09.08.22	43	Команда тестувальників (6)
Проведення тестування вручну	10 днів	Ср 10.08.22	Вт 23.08.22	44	Команда тестувальників (6)
Визначення тест-кейсів	3 дня	Ср 24.08.22	Пт 26.08.22	45	Команда тестувальників (6)
Розробка автоматичних тестів	5 днів	Пн 29.08.22	Пт 02.09.22	46	Команда тестувальників (6)
Написання документації	**15 днів**	**Пн 05.09.22**	**Пт 23.09.22**	**42**	
Написання звітів по тестам функцій і роботи сайту	5 днів	Пн 05.09.22	Пт 09.09.22		Команда тестувальників (6)
Написання баг-репортів	10 днів	Пн 12.09.22	Пт 23.09.22	49	Команда тестувальників (6)
Презентація продукту	**8 днів**	**Пн 26.09.22**	**Ср 05.10.22**	**41**	

4 РЕАЛІЗАЦІЯ ТА ЕКСПЕРИМЕНТАЛЬНА ПЕРЕВІРКА ВДОСКОНАЛЕНОГО МЕТОДУ ДИНАМІЧНОЇ АДАПТАЦІЇ РЕКОМЕНДАЦІЙ

4.1 Вибір метрики оцінки ефективності методу динамічної адаптації

Для оцінки ефективності роботи системи дані були розділені на датасети по часу публікації відгуків. Таким чином прогнозування використовує оцінки виставлені раніше і ці прогнози потім порівнюються з оцінками які користувачі виставили пізніше.

Точність прогнозування вимірює близькість прогнозів системи до реальних оцінок користувачів шляхом простого порівняння оціночного значення $\hat{r}_{u,i}$ для заданої пари (u, i) з істинним рейтингом $r_{u,i}$.

Середньоквадратична похибка ($RMSE$).

Ця міра часто використовується для оцінки різних методів, що застосовуються в рекомендаційних системах[28]. Нехай R – множина прогнозованих оцінок, тоді $RMSE$ визначається наступним чином:

$$RMSE = \sqrt{\frac{1}{|R|}\sum_{(u,i,r)\in R}\left(\hat{r}_{u,i} - r_{u,i}\right)^2}, \quad (4.1)$$

Прийнято вважати, що зменшення $RMSE$ призводить до підвищення релевантності та точності рекомендацій. Однак, можна стверджувати, що цей показник страждає від переривчастої поведінки. $RMSE$ надає більшої ваги помилкам, більшим за одиницю і меншої тим, що менші за одиницю.

Формула середньої абсолютної похибки (MAE) виглядає так:

$$MAE = \frac{1}{|R|}\sum_{(u,i,r)\in R}\left|\hat{r}_{u,i} - r_{u,i}\right|, \quad (4.2)$$

На відміну від $RMSE$, ця метрика є парною. Тому її також можна критикувати за те, що вона надто м'яко реагує на великі похибки. Може бути корисно змішати MAE для похибок менших за 1 і $RMSE$ для більших за 1.

Середня абсолютна відсоткова похибка ($MAPE$), це міра точності побудови рекомендацій у часі, наприклад в оцінюванні трендів. Зазвичай точність виражається у відсотках і визначається за формулою:

$$\text{MAPE} = \frac{1}{n}\sum_{t=1}^{n} \left|\frac{\hat{r}_{u,i} - r_{u,i}}{r_{u,i}}\right|, \tag{4.3}$$

де $r_{u,i}$ – фактичне значення;

$\hat{r}_{u,i}$ – прогнозоване значення.

Різниця між $r_{u,i}$ та $\hat{r}_{u,i}$ знову ділиться на фактичне значення $r_{u,i}$. Абсолютне значення в цьому розрахунку підсумовується для кожного встановленого або прогнозованого моменту часу і знову ділиться на кількість встановлених точок n. Множення на 100 дає похибку у відсотках.

Розглянемо метрики на основі прийняття рішень.

Метрики, що базуються на прийнятті рішень, оцінюють топ-N рекомендацій для користувача. Зазвичай рекомендації – це ранжований список елементів, впорядкований за зменшенням релевантності. Однак метрики, засновані на прийнятті рішень, не беруть до уваги позицію (або ранг) елемента в списку результатів[28].

Існує чотири різні випадки, які слід враховувати:

– істинно позитивний (ІП). Система рекомендує позицію, яка цікавить користувача;

– хибнопозитивний (ХП). Система рекомендує позицію, яка не цікавить користувача;

– істинно негативний (ІН). Система не рекомендує товар, який не цікавить користувача;

– хибно негативний (ХН). Система не рекомендує товар, який цікавить користувача.

Таблиця 4.1 – Матриця плутанини

	Цікаво	Не цікаво
Рекомендовано	Істинно позитивний	Хибно позитивний
Не рекомендовано	Хибно негативний	Істинно негативний

Точність і повернення отримують з матриці плутанини (таблиця 4.1).

Ці показники потребують порогового значення t, яке визначає, що є рекомендованим, а що ні.

Точність ($Precision$) вимірює частку релевантних елементів, з рекомендованих:

$$\text{Precision} = \frac{TP}{TP+FP}, \tag{4.4}$$

Точність можна також оцінити на заданому рівні відсікання, враховуючи лише n найкращих рекомендацій. Цей показник називається точністю при n. При оцінюванні найкращих n результатів системи рекомендацій досить часто використовують саме цю міру:

$$\text{Precision} = \frac{|\text{ hitset }|}{N}, \tag{4.5}$$

де $|hitset| = |test \cap topN|$.

Повернення ($Recall$) вимірює охоплення рекомендованих елементів і визначається наступним чином:

$$\text{Recall} = \frac{TP}{TP+FN}, \tag{4.6}$$

Знову ж таки, оцінюючи топ-N результатів рекомендаційної системи, можна використовувати цю міру:

$$\text{Recall} = \frac{|\text{ hitset }|}{|\text{ test }|}, \tag{4.7}$$

F-міра поєднує результати P і R, використовуючи зважене гармонійне середнє. Загальна формула (для невід'ємного реального значення β) така:

$$F_\beta = \frac{(1+\beta^2)\cdot(\text{ precision } \cdot \text{ recall })}{(\beta^2 \cdot \text{ precision } + \text{ recall })}, \tag{4.8}$$

Двома поширеними F-мірами є F_1 та F_2. У F_1 повернення та точність рівномірно зважені, в той час як F_2 важить повернення вдвічі більше, ніж точність.

Основним недоліком метрик, заснованих на прийнятті рішень є те, що вони не враховують рейтинг рекомендованих товарів. Таким чином, стаття в топ-1 має таку ж релевантність, як і стаття в топ-20. Щоб уникнути цього обмеження, можна використовувати рангові метрики[29].

З цими метриками потрібно бути обережними, оскільки немає сенсу обчислювати їх, якщо користувач з'являється лише один раз у тестовій вибірці. Крім того, оскільки базові дані можуть мати зв'язки, рангові метрики іноді важко інтерпретувати.

Коефіцієнт ρ Спірмена обчислює рангову кореляцію Пірсона для двох ранжованих списків. Він порівнює прогнозований список з вподобаннями користувача і визначається наступним чином:

$$\rho = \frac{1}{n_u} \frac{\sum_i (r_{ui} - \bar{r})(\hat{r}_{ui} - \hat{\bar{r}})}{\sigma(r)\sigma(\hat{r})}, \tag{4.9}$$

Коефіцієнт τ Кендалла також порівнює рекомендований список (top-N) зі списком елементів, яким користувач надає перевагу. Коефіцієнт Кендалла визначається наступним чином:

$$\tau = \frac{C^+ - C^-}{\frac{1}{2}N(N-1)}, \tag{4.10}$$

де C^+ – кількість конкордантних пар;

C^- – кількість дискордантних пар у наборі даних.

Середній взаємний ранг (MRR) визначається наступним чином:

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{Q} \frac{1}{\text{rank}_i}, \tag{4.11}$$

Рекомендації, які знаходяться на початку списку top-N, мають більшу вагу, ніж ті, що знаходяться в кінці списку.

Середньоарифметична точність (MAP) визначається наступним чином:

$$\text{MAP} = \frac{\sum_{q=1}^{Q} AP(q)}{Q}, \tag{4.12}$$

де Q – кількість запитів;

AP – середня точність.

AP дорівнює:

$$\mathrm{AP} = \frac{\sum_{k=1}^{n} (P(k) \times \mathrm{rel}(k))}{\text{number of relevant documents}}, \qquad (4.13)$$

де $P(k)$ – точність на вершині k;

$rel(k)$ – індикаторна функція, що дорівнює 1, якщо елемент на вершині є релевантним документом, і нулю в протилежному випадку.

Рекомендації, що знаходяться на початку списку «top-N», мають більшу вагу, ніж ті, що знаходяться в кінці списку.

Перейдемо до вибору метрик оцінки.

Показники Спірмена та Кендалла не підходять, оскільки вони не враховують зв'язки. У контексті рекомендацій дуже часто зустрічаються однакові оцінки через дискретний характер шкали (зазвичай цілі числа від 1 до 5 включно).

Середній взаємний це хороша міра, яка базується на ранжируванні, але він не враховує порядок у прогнозованому списку елементів та їх ранг. Отже, оскільки нам потрібно мати можливість оцінити здатність нашої системи впорядковувати і ранжувати список елементів, MRR не буде використовуватися.

Середньоарифметична точність, однак, враховує порядок і ранг елементів у прогнозованому списку. MAP буде єдиною метрикою, що базується на ранжируванні, в нашому протоколі оцінювання.

Розглянемо дані для оцінки ефективності системи.

Оцінка системи проводилася за допомогою трьох різних наборів даних: Netflix, Vodkaster та MovieLens. Netflix та MovieLens мають перевагу в тому, що вони є загальнодоступними. Vodkaster дозволяє працювати на семантичному рівні, оскільки набір даних містить не лише рейтинги, але й текстові коментарі. Як і в Netflix та MovieLens, користувачі оцінюють та коментують фільми.

Для даних Netflix і MovieLens лог має таку структуру: userID, itemID, оцінка, дата. Для Vodkaster він включає текст або відгук про

товар. Крім того, користувачі та елементи ідентифікуються не за числовим ідентифікатором, а за іменем користувача та назвою. Таким чином, журнал для Vodkaster має такий вигляд: userID, itemID, рейтинг, відгук, дата.

На відміну від Netflix і MovieLens, на Vodkaster не було системи рекомендацій. Тому розподіл пар (u, i) може бути більш усередненим на Vodkaster. Той факт, що на Netflix і MovieLens працює система рекомендацій, міг вплинути на поведінку користувачів. Проте можна припустити, що вплив на смаки користувачів є незначним.

Розглянемо Vodkaster.

Користувачі Vodkaster публікують мікрорецензії, щоб висловити свою думку про фільм і поставити йому оцінку, в межах 140 символів. На цей час, датасет містить близько 200 тис. мікрорецензій та 2 млн оцінок. Він був розділений на три частини, відсортовані в хронологічному порядку: навчання (Tr), розвиток (D) і тест (T) (таблиця 4.2).

Таблиця 4.2 – Статистика датасету Vodkaster

	Tr	D	Tr+D	T	Всього
Розмір	2 млн.	20 тис.	2,02 млн.	20 тис.	2,04 млн.
Фільми	26097	5520	26248	5683	26344
Користувачі	19922	1426	2041	1412	20213

Перейдемо до Netflix.

Набір даних, використаний для Netflix, дуже великий (100 млн оцінок, 400 тис. користувачів, 17 тис. фільмів). Щоб мати змогу порівняти результати, отримані на наборах даних Vodkaster та MovieLens, було вирішено зберегти 6000 випадкових користувачів з принаймні 50 оцінками. Потім дані були розділені вибірки як і в попередньому випадку(таблиця 4.3).

Таблиця 4.3 – Статистика датасету Netflix

	Tr	D	Tr+D	T	Всього
Розмір	1,3 млн.	20 тис.	1,32 млн.	20 тис.	1,5 млн.
Фільми	15345	4416	15419	4562	26344
Користувачі	6000	6000	6000	6000	6000

Розглянемо MovieLens.

Цей набір даних містить 1 млн анонімних оцінок, приблизно 3900 фільмів, зроблених 6040 користувачами MovieLens (таблиця 4.4).

Таблиця 4.4 – Статистика датасету MovieLens

	Tr	D	Tr+D	T	Всього
Розмір	980 тис.	10 тис.	990 тис.	10 тис.	1 млн.
Фільми	3701	2503	3703	2432	3706
Користувачі	6039	356	6039	348	6040

Перейдемо до моделювання «холодного старту».

Для того, щоб явно спостерігати ефект динамічної адаптації, потрібні дві умови. Перша полягає в тому, що і навчальні, і тестові набори повинні бути відсортовані в хронологічному порядку, від старих оцінок до нових. Друга умова полягає в тому, щоб на етапі тестування не виникало «холодних стартів» для користувачів і товарів, або і для тих, і для інших[30].

Динамічна адаптація має сенс на реальних даних. Тобто на даних, які не були змінені взагалі. Єдиний реальний і повний набір даних, який ми маємо – дані з Vodkaster. Далі можна побачити, що в наслідок цього вплив динамічної адаптації набагато більший на Vodkaster.

Це значить, що не потрібно штучно інтенсифікувати «холодний старт» для Vodkaster, оскільки набір даних не був змінений, а отже, відображає реальну ситуацію в світі. Тому було вирішено взяти кількість «холодних стартів» на

Vodkaster за еталонну, тобто 178 для користувачів і 114 для товарів. Екстремальний випадок, коли новий користувач оцінює новий товар, трапляється лише 9 разів на всьому наборі даних. Після обробки така сама частка (1%) «холодних стартів» була додана до наборів даних MovieLens та Netflix.

Розглянемо розподіл оцінок.

Рисунок 4.1 показує, що в цілому розподіл оцінок на MovieLens (синім), Netflix (помаранчевим) і Vodkaster (зеленим) досить схожий (4 зірки є найпоширенішою оцінкою). Однак можна помітити, що користувачі Vodkaster рідше ставлять оцінку 5 зірок. Вони також частіше ставлять 1 та 2 зірки. Це має сенс, оскільки користувачі Vodkaster є кіноманами, а тому більш суворі в оцінці фільмів.

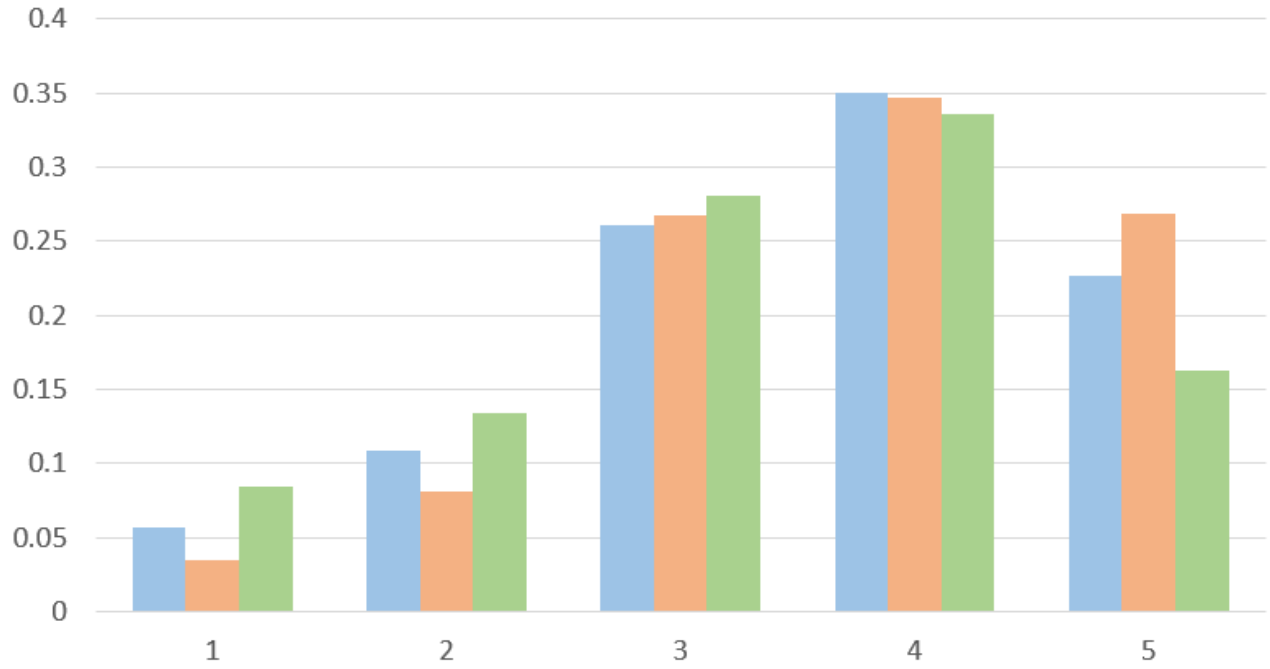


Рисунок 4.1 – Розподілення оцінок

4.2 Експериментальна перевірка ефективності розроблених методів

Далі представлені результати, отримані з адаптацією та без неї на наборах даних Vodkaster, Netflix та MovieLens. Для порівняння наведені результати без адаптації для методів колаборативної фільтрації та матричної факторизації.

Експерименти були проведені на 3 наборах даних: MovieLens , Vodkaster та Netflix, розділених на 3 хронологічно впорядковані частини: Train, Dev та Test. Ці набори даних мають дуже різні характеристики: Netflix має велику кількість користувачів і розподілений за короткий проміжок часу порівняно з іншими. MovieLens має велику кількість користувачів і розподілений на тривалий період часу. Vodkaster має низьку кількість користувачів і поширений протягом короткого періоду часу, але користувачі дуже "лояльні" та активні.

Розглянемо вплив адаптації на різні метрики та на різні набори даних при повному покритті за допомогою класичного методу колаборативної фільтрації.

У таблицях 4.1,4.2 та 4.3 ми бачимо, що всі метрики було покращено. Приріст становить приблизно 10% для RMSE та MAE. Результати з динамічною адаптацією на наборах для розробки та тесту близькі, і це справедливо для всіх наборів даних. Отже, динамічна адаптація є надійним підходом.

Таблиця 4.1 – Оцінка ефективності роботи системи на датасеті Netflix

Netflix	Dev		Test	
Метод оцінки ефективності	Без адаптації	З адаптацією	Без адаптації	З адаптацією
MAE	0.753	0.713	0.772	0.728
RMSE	0.986	0.920	0.999	0.927

Кінець таблиці 4.1

Netflix	Dev		Test	
Метод оцінки ефективності	Без адаптації	З адаптацією	Без адаптації	З адаптацією
Precision користувачів	0.807	0.833	0.795	0.822
Recall користувачів	0.904	0.931	0.896	0.925
F-міра користувачів	0.841	0.868	0.831	0.858
Precision товарів	0.832	0.849	0.824	0.843
Recall товарів	0.879	0.898	0.875	0.894
F-міра товарів	0.850	0.868	0.843	0.862
MAP	0.629	0.648	0.621	0.633

Таблиця 4.2 – Оцінка ефективності роботи системи на датасеті MovieLens

MovieLens	Dev		Test	
Метод оцінки ефективності	Без адаптації	З адаптацією	Без адаптації	З адаптацією
MAE	0.759	0.708	0.793	0.692
RMSE	0.984	0.908	1.035	0.885

Кінець таблиці 4.2

MovieLens	Dev		Test	
Метод оцінки ефективності	Без адаптації	З адаптацією	Без адаптації	З адаптацією
Precision користувачів	0.754	0.756	0.778	0.784
Recall користувачів	0.876	0.868	0.882	0.882
F-міра користувачів	0.801	0.800	0.820	0.824
Precision товарів	0.695	0.673	0.712	0.701
Recall товарів	0.798	0.770	0.801	0.779
F-міра товарів	0.733	0.708	0.744	0.730
MAP	0.549	0.527	0.512	0.483

Таблиця 4.3 – Оцінка ефективності роботи системи на датасеті Vodkaster

Vodkaster	Dev		Test	
Метод оцінки ефективності	Без адаптації	З адаптацією	Без адаптації	З адаптацією
MAE	0.992	0.634	0.771	0.652
RMSE	1.293	0.830	1.007	0.834

Кінець таблиці 4.3

Vodkaster	Dev		Test	
Метод оцінки ефективності	Без адаптації	З адаптацією	Без адаптації	З адаптацією
Precision користувачів	0.726	0.784	0.769	0.794
Recall користувачів	0.846	0.910	0.891	0.916
F-міра користувачів	0.770	0.831	0.815	0.841
Precision товарів	0.653	0.788	0.709	0.757
Recall товарів	0.713	0.857	0.787	0.833
F-міра товарів	0.674	0.813	0.737	0.784
MAP	0.743	0.719	0.748	0.742

Ефект адаптації набагато сильніший для набору даних Vodkaster. RMSE без адаптації становить 1,293, а з адаптацією – 0,830, що означає підвищення точності на 35%. Це можна пояснити тим, що дані, витягнуті з Vodkaster, є реальними немодифікованими данимиі безперервними з хронологічної точки зору. Крім того, набір даних є повним, а не підмножиною більшого набору даних.

Крім того, поведінка користувачів може бути більш неоднорідною, оскільки на Vodkaster немає рекомендацій, а MovieLens і Netflix завжди мали рекомендації, що призводило до більш стабільної, контрольованої та однорідної

поведінки користувачів. Отже, для цих наборів даних ефект динамічної адаптації є менш важливим, оскільки смаки користувачів є більш статичними, а отже, передбачуваними.

Інша відмінність може полягати в тому, що Vodkaster – спільнота кіноманів. Користувачі можуть більш суворими, коли оцінюють фільм, а також можуть ризикувати і досліджувати нові жанри фільмів. Тому їх смаки можуть змінюватися швидше, ніж в інших наборах даних, таких як Netflix та MovieLens.

Перейдемо до графіків, які дозволяють чітко показати ефект динамічної адаптації.

Рисунки 4.2, 4.3, 4.4 чітко відображають ефект динамічної адаптації в часі. Кожна точка координат $x = n$ відповідає середньому $RMSE$ після спостереження за n оцінками користувача (починаючи з початку тестового набору), причому середнє значення обчислюється для користувачів, які оцінили щонайменше n елементів у тестовому наборі. Це відповідає відносній шкалі часу, орієнтованій на користувача, і показує, що без адаптації помилки прогнозування зростають, тоді як з адаптацією вони стабілізуються до значно нижчого значення.

Зауважте, що чим більше зростає x, тим менше користувачів залишається. Тому потрібно встановити нижній поріг кількості користувачів, нижче якого немає сенсу обчислювати середнє значення. Було вирішено зупинитися, коли залишиться лише 25 користувачів. Ось чому, залежно від набору даних, x_{max} відрізняється.

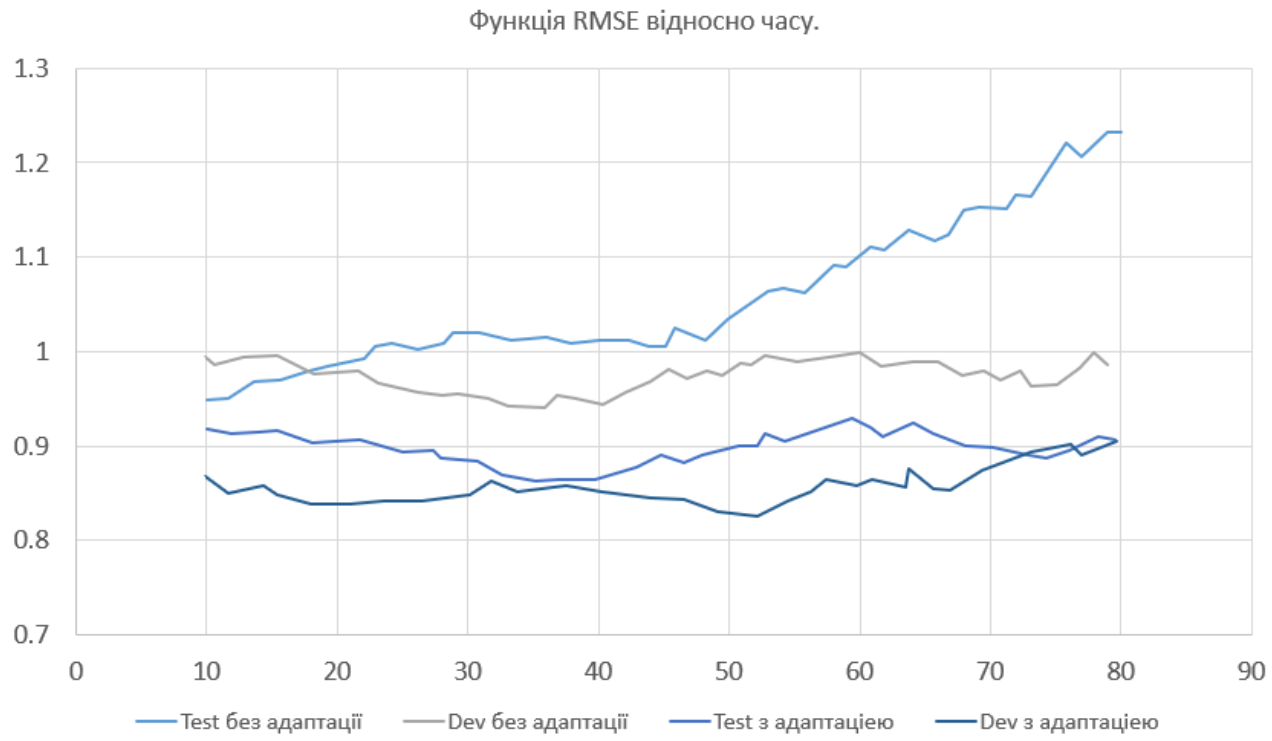


Рисунок 4.2 – Функція RMSE відносно часу. Датасет MovieLens

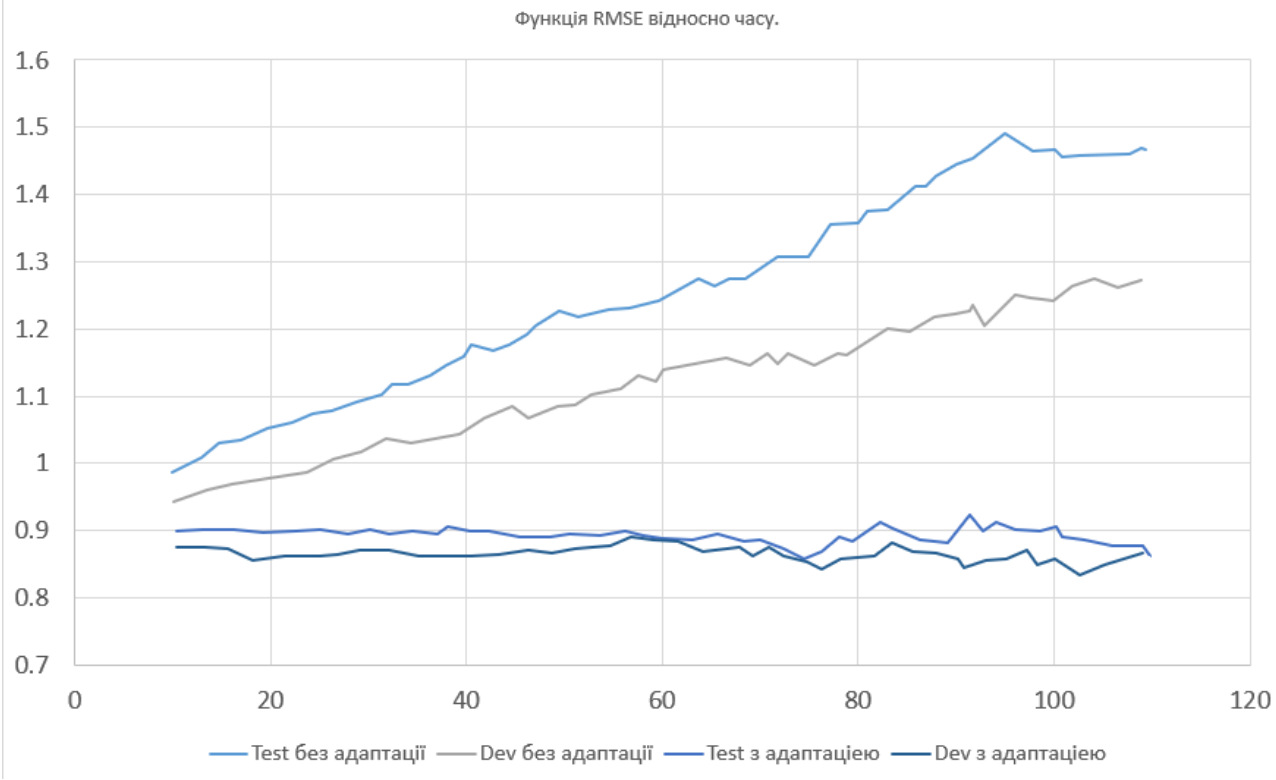


Рисунок 4.3 – Функція RMSE відносно часу. Датасет Netflix

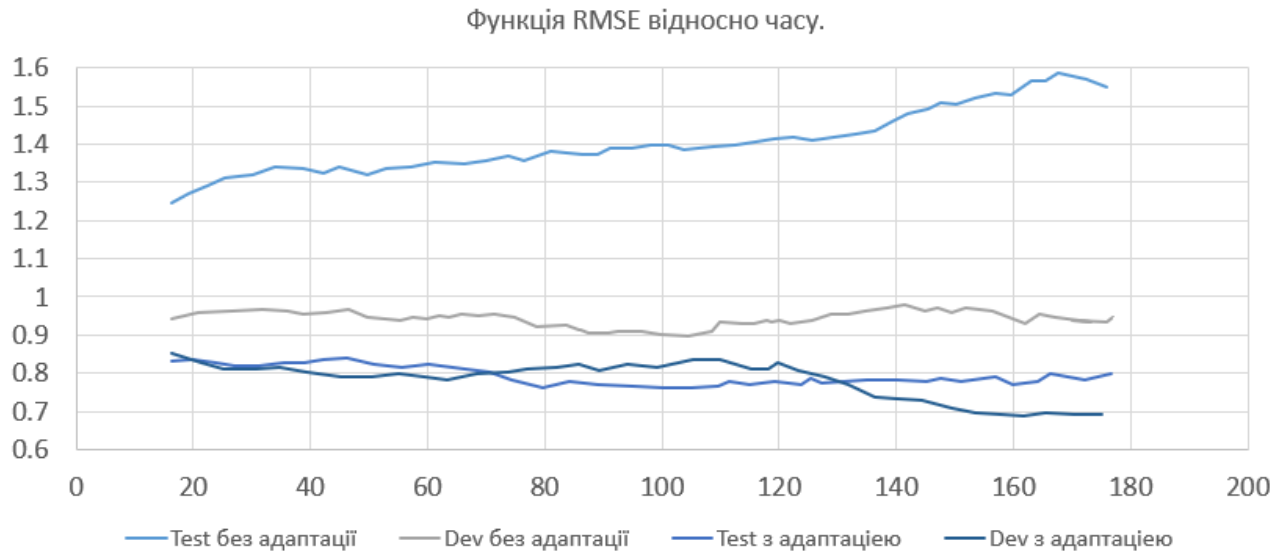


Рисунок 4.4 – Функція RMSE відносно часу. Датасет Vodkaster

Розглянемо результати розробленого методу з використанням адаптивного заповнення матриці рекомендацій (таблиця 4.4).

Таблиця 4.4 – Перевірка ефективності розробленого методу адаптивного заповнення матриці рекомендацій

		RMSE	MAE	MAPE
Vodkaster	Без адаптації	1.1158	0.8924	0.5296
	З адаптацією	0.7805	0.5993	0.3031
Netflix	Без адаптації	1.0484	0.8669	0.3209
	З адаптацією	0.8685	0.6678	0.2506
MovieLens	Без адаптації	1.0976	0.9096	0.3850
	З адаптацією	0.8435	0.6528	0.2576

Результати показують, що адаптивні методи покращують показники за метриками RMSE, MAE та MAPE. Виграш у RMSE набагато сильніший для для Vodkaster (від 1,158 до 0,7805), ніж для Netflix (від 1,0484 до 0,8685) і MovieLens (від 1,0976 до 0,8435).

Графічне відображення функції RMSE відносно часу зображено на рисунках 4.4,4.5 та 4.6.

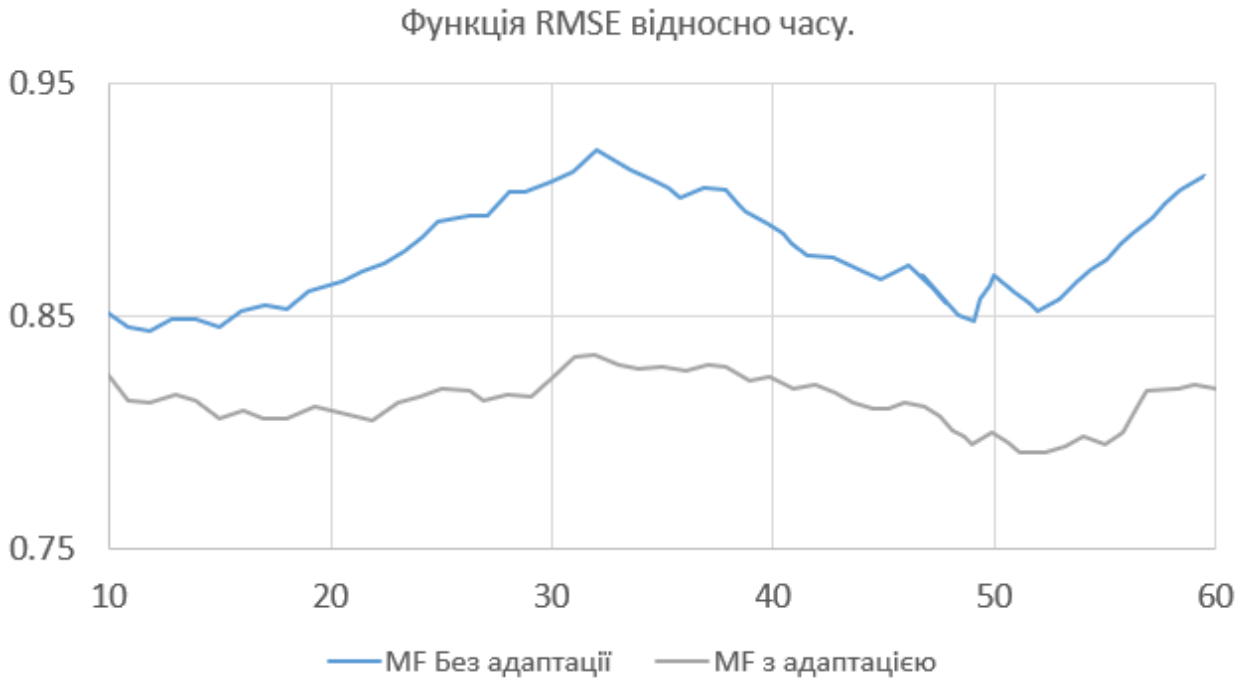


Рисунок 4.5. Функція RMSE відносно часу. Датасет MovieLens.

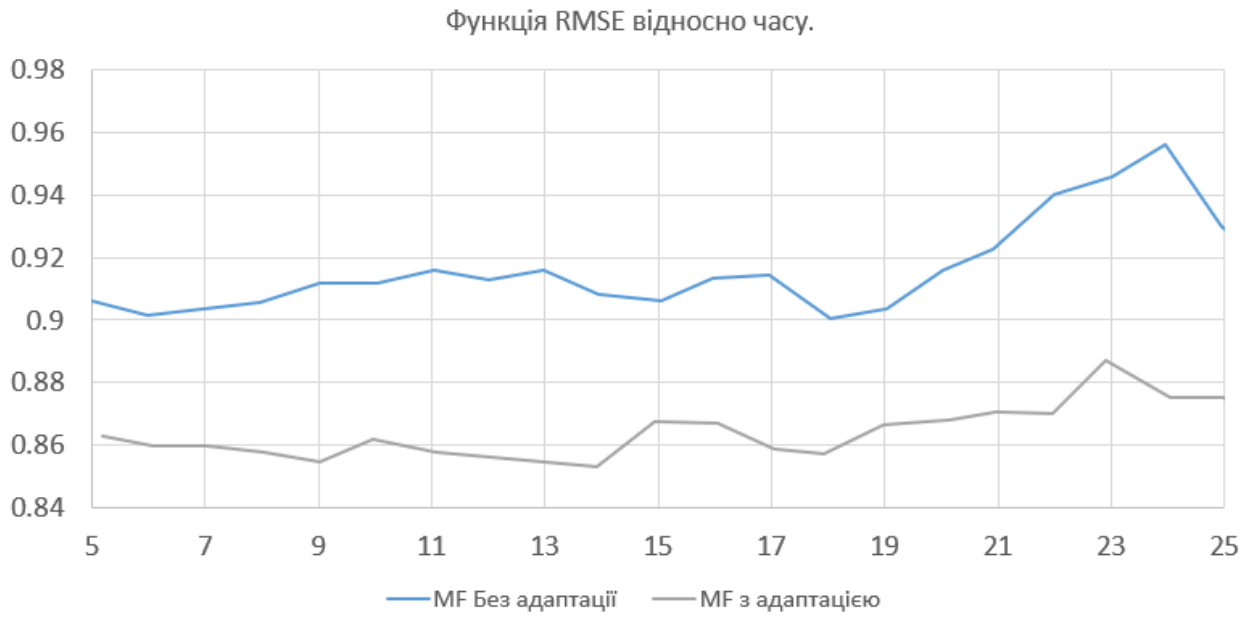


Рисунок 4.5 – Функція RMSE відносно часу. Датасет Netflix

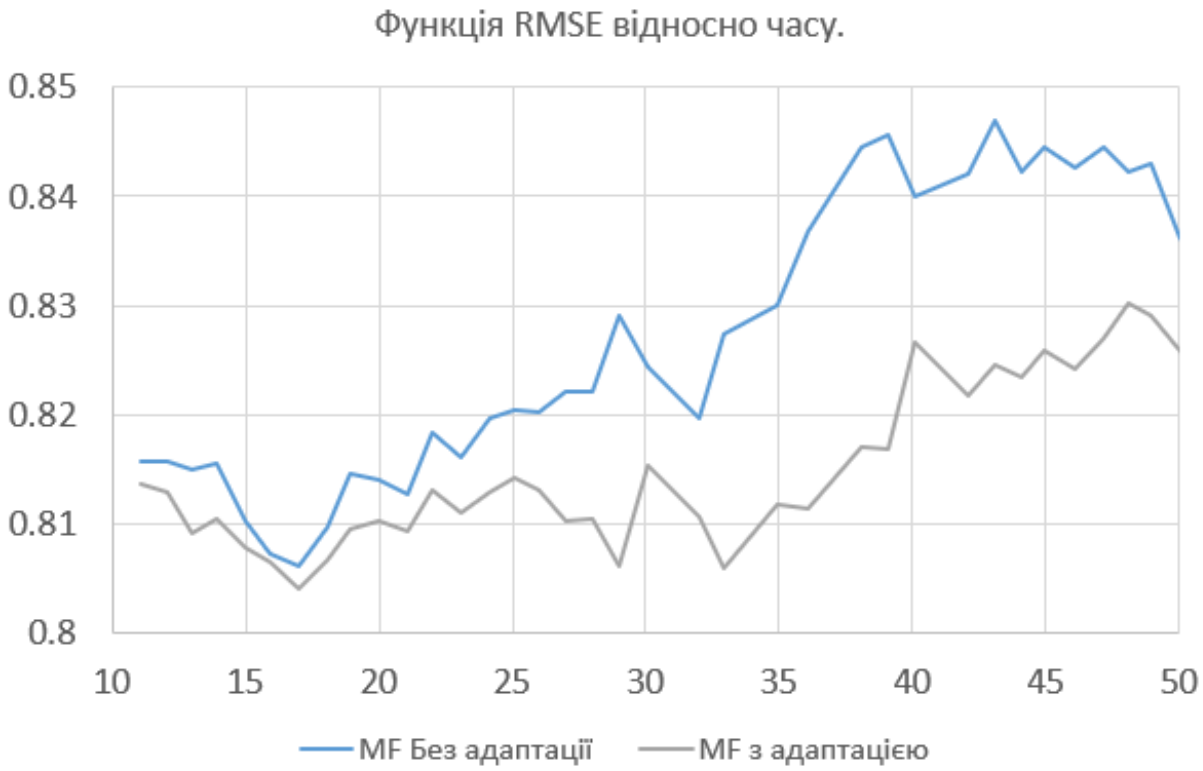


Рисунок 4.6 – Функція RMSE відносно часу. ДатасетVodkaster

Вплив динамічної адаптації можна спостерігати на трьох наборах даних, на яких ми тестували нашу систему і для обох методів які ми використовували, а саме класичної колаборативної фільтрації та матричної факторизації.

Можна побачити, що динамічна адаптація значно покращує продуктивність на всіх наборах даних і що методи адаптованої матричної факторизації перевершують класичний метод колаборативної фільтрації.

ВИСНОВКИ

В ході виконання даної кваліфікаційної роботи було проведено аналіз існуючих рекомендаційних систем, та існуючих методів побудови рекомендацій.

В результаті аналізу можна зробити висновок, що існуючі методи побудови рекомендацій враховують інтегральну інформацію що до вибору користувача. Однак у випадку зміни інтересів користувача, ця інформація стає застарілою, що зменшує точність рекомендацій. Тому важливою є розробка методів, які б давали можливість динамічно адаптувати рекомендації в такій ситуації.

Було розроблено новий метод динамічної адаптації, який дозволяє системі рекомендацій адаптуватися до змін вподобань користувачів з кожним новим рейтингом.

Для того, щоб отримати швидку «реактивність», було запропоновано нове застосування міри подібності на основі відстані Мангеттена. Це призводить до значного зменшення складності та дозволяє ввести динамічну адаптацію. Таким чином, стає можливим оновлювати параметри рекомендаційної системи ітеративно, щоразу, коли з'являється новий рейтинг.

Також було запропоновано метод адаптивного заповнення матриці, який дозволяє рекомендаційним системам бути дуже динамічними. Експериментальні результати показали, що цей метод значно підвищує точність прогнозованих оцінок.

В результаті порівняння ефективності базових методів побудови рекомендацій та розробленого методу динамічної адаптації, можна зробити висновок, що динамічна адаптація сильно покращує точність рекомендацій і ефективність роботи системи.

Під час написання дипломної роботи також були написані та опубліковані тези на тему «Дослідження методів динамічної адаптації рекомендацій в IT-проектах електронної комерції».

Кваліфікаційна робота виконана згідно з методичними вказівками щодо розробки та оформлення магістерської атестаційної роботи за спеціальністю 122 Комп’ютерні науки (освітня програма «Управління проєктами в галузі інформаційних технологій» освітньо-кваліфікаційного рівня «магістр»[31].

Перелік джерел посилання оформлено згідно з державним стандартом ДСТУ 8302:2015 «Бібліографічне посилання. Загальні положення та правила складання» [32].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Rendle, S. (2019). Neural factorization machines for sparse predictive analytics. In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 355-364). ACM.

2. Okura, S., Tagami, Y., & Murata, T. (2017). Embedding-based news recommendation for millions of users. In Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 295-304). ACM.

3. Oosterhuis, H., & de Rijke, M. (2018). Learning to recommend with social contextual information. In Proceedings of the 27th ACM International Conference on Information and Knowledge Management (pp. 185-194). ACM.

4. Kaya, M., & Benli, B. (2019). Hybrid recommendation algorithms for e-commerce platforms. Journal of Intelligent Information Systems, 53(1), 53-76.

5. Bhowmik, D., & Chakraborty, D. (2019). A review on dynamic recommender systems. Journal of Intelligent Information Systems, 53(3), 427-452.

6. Chen, Y., Wang, S., Sun, Y., Zhang, H., & Cheng, X. (2020). A dynamic collaborative filtering algorithm with trust-awareness and temporal influence for online recommendation. Information Sciences, 506, 39-59.

7. Shen, Y., Guo, Y., Liu, F., Chen, Y., & Wang, Y. (2018). Dynamic news recommendation based on user behavior and text. In Proceedings of the 27th ACM International Conference on Information and Knowledge Management (pp. 2077-2080). ACM.

8. Zuo, Y., & Song, J. (2019). A dynamic personalized recommendation algorithm based on long-term and short-term interests. Journal of Intelligent Information Systems, 52(3), 485-501.

9. Zhou, T., Cao, L., Zuo, W., & Li, L. (2019). A dynamic personalized recommendation algorithm based on behavior evolution. Journal of Ambient Intelligence and Humanized Computing, 10(11), 4237-4250.

10. Zhao, J., Li, C., Li, Y., & Li, Y. (2019). A dynamic recommendation algorithm based on user preference prediction. Journal of Ambient Intelligence and Humanized Computing, 10(8), 2957-2968.

11. Liu, Z., & Yao, X. (2021). A dynamic item recommendation algorithm based on the user's long-term and short-term interests. IEEE Transactions on Industrial Informatics, 17(1), 267-275.

12. Baltrunas, L., & Moling, O. (2019). Towards dynamic adaptation of recommender systems: A workshop on the impact of demographic changes on recommender systems. In Proceedings of the 13th ACM Conference on Recommender Systems (pp. 532-533).

13. Castagnos, S. G., & Bellogin, A. (2020). Dynamic adaptation in recommender systems: A systematic mapping study. Information Processing & Management, 57(4), 102-159.

14. Chirita, P. A., & Nejdl, W. (2009). Exploiting user context for dynamic adaptation of search results ranking. Journal of Web Semantics, 7(4), 277-286.

15. Chen, L., & Pu, P. (2012). Context-aware dynamic adaptation of information retrieval systems. ACM Transactions on Information Systems, 30(4), Article 23.

16. Чалий С. Ф., Лещинський В. О., Лещинська І. О. (2019). Моделювання пояснень щодо рекомендованого переліку об’єктів з урахуванням темпорального аспекту вибору користувача. Системи управління, навігації та зв’язку. 6(58). 97-101.

17. Чалий, С. Ф., Лещинський, В. О., & Лещинська, І. О. (2018). Інтеграція локальних контекстів споживачів в рекомендаційних системах на основі відношень еквівалентності, схожості та сумісності. In Process mining Materials of

the VII International Scientific Conference «Information-Control System and Technologies (pp. 142-144).

18. Chalyi, S., Leshchynskyi, V., & Leshchynska, I. (2020). Модель інтерфейсу пояснень з темпоральними параметрами в рекомендаційній системі. Системи управління, навігації та зв'язку. Збірник наукових праць, 2(60), 105-109.

19. Chalyi, S., & Pribylnova, I. (2019). Ситуаційна модель користувацького вибору в рекомендаційній системі. Системи управління, навігації та зв'язку. Збірник наукових праць, 2(54), 159-163.

20. Chalyi, S., Leshchynskyi, V., & Leshchynska, I. (2019). Концепція формування пояснень в рекомендаційних системах за принципом білого ящику. Системи управління, навігації та зв'язку. Збірник наукових праць, 3(55), 156-160.

21. Yu, F., Liu, Y., & Song, H. (2019). Dynamic adaptation in personalized recommender systems: An overview. Frontiers of Computer Science, 13(2), 193-209.

22. Ricci, F., & Rokach, L. (2018). Introduction to recommender systems handbook. In F. Ricci, L. Rokach, B. Shapira, & P. B. Kantor (Eds.), Recommender Systems Handbook (2nd ed., pp. 1-34). Springer.

23. Said, A., Bellogín, A., Hurley, N., & Tikk, D. (2013). Recommender systems evaluation: A 3D benchmark. In Proceedings of the 7th ACM Conference on Recommender Systems (pp. 297-300).

24. Adomavicius, G., & Tuzhilin, A. (2005). 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, 17(6), 734-749.

25. Yuan, Q., Cong, G., Ma, Z., & Sun, A. (2016). Time-aware point-of-interest recommendation. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 1085-1094).

26. Burke, R. (2002). Hybrid recommender systems: Survey and experiments. User Modeling and User-Adapted Interaction, 12(4), 331-370.

27. Shi, Y., & Larson, M. (2012). Learning context-aware music similarity from last.fm scrobbles. In Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR) (pp. 559-564).

28. Jannach, D., Zanker, M., Felfernig, A., & Friedrich, G. (2010). Recommender systems: An introduction. Cambridge University Press.

29. Konstan, J. A., & Riedl, J. (2012). Recommender systems: From algorithms to user experience. User Modeling and User-Adapted Interaction, 22(1-2), 101-123.

30. Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems, 22(1), 5-53.

31. Методичні вказівки щодо розробки та оформлення магістерської атестаційної роботи за спеціальністю 122 Комп'ютерні науки (освітня програма «Управління проектами в галузі інформаційних технологій» освітньо-кваліфікаційного рівня «магістр» / Упоряд.: Петров К.Е., Левикін В.М., Чалий С.Ф., Євланов М.В., Саєнко В.І., Міхнов Д.К., Міхнова А.В., Чала О.В. – Харків: ХНУРЕ, 2019. – 28 с.

32. ДСТУ 8302:2015. Інформація та документація. Бібліографічні посилання. Загальні положення та правила складання. – Чинний від 04.03.2016. – Київ: ДП «УкрНДНЦ», 2016. – 20 с.