Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення

”Допущено до захисту”
Завідувач кафедри кібербезпеки
та програмного забезпечення
д.т.н., професор
_____ Олексій СМІРНОВ
“___” _____ 20__ р.

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
за другим (магістерським) рівнем вищої освіти

на тему

“Дослідження та програмна реалізація системи створення персоналізованої реклами для користувачів веб-сайту”

Виконав здобувач вищої освіти
ІІ курсу, групи _____
ОПП «Комп’ютерна інженерія»
спеціальності 123 «Комп’ютерна інженерія»
_____ Голодок О.Д.
«____» _____ 20 р.

Керівник проекту
кандидат технічних наук, доцент
_____ Роман МИНАЙЛЕНКО
«____» _____ 20 р.
Рецензент _____
_____

м. Кропивницький

Кафедра_КБПЗ_2022рік

Центральноукраїнський національний технічний університет

Факультет *Механіко-технологічний*

Кафедра *Кібербезпеки та програмного забезпечення*

Освітній ступінь *бакалавр*

Галузь знань *12 "Інформаційні технології"*

Спеціальність *123 "Комп'ютерна інженерія"*

Освітньо-професійна (освітньо-наукова) програма *"Комп'ютерна інженерія"*

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., проф.
_____Олексій СМІРНОВ
«__ »_____ 20__ року

ЗАВДАННЯ НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ ЗА ДРУГИМ (МАГІСТЕРСЬКИМ) РІВНЕМ ВИЩОЇ ОСВІТИ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ

Голодку Олексію Дмитровичу
(прізвище, ім'я, по батькові)

1. Тема роботи *Дослідження та програмна реалізація системи створення персоналізованої реклами для користувачів веб-сайту*

2. Керівник роботи *Минайленко Роман Миколайович, кандидат техн. наук, доцент*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу №_____ від __.__.20__ року

3. Строк подання роботи до захисту *10.12.2022 р.*

4. Мета та завдання випускної кваліфікаційної роботи: *Дослідження та програмна реалізація системи створення персоналізованої реклами для користувачів веб-сайту*

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

1. Призначення та область використання.
2.Перегляд аналогічних існуючих систем.
3. Опис і обґрунтування проектних рішень.
4. Етапи програмування системи.
5. Впровадження системи в промислову експлуатацію.
6. Наукова новизна
7. Економічна ефективність розробленої програми.
8. Заходи з охорони праці та техніки безпеки.
9. Висновки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

Наукова новизна	*1 аркуш*
Структурна схема системи	*1 аркуш*
Функціональна схема системи	*1 аркуш*
Блок-схема алгоритму роботи додатку	*2 аркуша*
Діаграма процесів	*1 аркуш*
Показники економічної ефективності	*1 аркуш*

Кафедра_КБПЗ_2022рік

6. Консультанти по роботі, із зазначенням розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Савеленко Г.В., к.т.н., доцент	25.10.2022	11.11.2022
Охорона праці	Оришака О.В., к.т.н., доцент	04.11.2022	21.11.2022

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти	Строк виконання етапів випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти	Примітка
1.	Аналіз існуючих систем	10.10.2022 р.	
2.	Постановка задачі, оформлення ТЗ	15.10.2022 р.	
3.	Розробка моделі компонента	20.10.2022 р.	
4.	Розробка структур даних	25.10.2022 р.	
5.	Розробка алгоритмів зв’язку та відображення	30.10.2022 р.	
6.	Програмування алгоритмів	10.11.2022 р.	
7.	Розрахунок економічної ефективності	13.11.2022 р.	
8.	Розрахунки з охорони праці та техніки безпеки	15.11.2022 р.	
9.	Оформлення ПЗ	17.11.2022 р.	
10.	Попередній захист роботи	10.12.2022 р.	

Дата видачі завдання
«__»_____20 р.

Підпис керівника

_____ _____
(прізвище та ініціали)

Завдання прийнято до виконання
«__»_____20 р.

Підпис здобувача

_____ _____
(прізвище та ініціали)

Кафедра_КОПЗ_2022рік

АНОТАЦІЯ

Голодок О.Д. Дослідження та програмна реалізація системи створення персоналізованої реклами для користувачів веб-сайту. 123 Комп'ютерна інженерія. Центральноукраїнський національний технічний університет. Кропивницький. 2022.

В даній магістерській роботі розроблено програмне забезпечення, яке призначено для реалізації системи створення персоналізованої реклами для користувачів веб-сайту.

Метою розробки є дослідження та програмна реалізація системи створення персоналізованої реклами для користувачів веб-сайту.

Об'єктом дослідження є процес створення персоналізованої реклами для користувачів веб-сайту.

Предметом дослідження є методи та алгоритми колаборативної фільтрації даних для створення персоналізованої реклами.

Методи дослідження базуються на методах аналізу даних, методах математичної статистики та методах розробки програмного забезпечення.

Результат роботи – програмна реалізація системи створення персоналізованої реклами для користувачів веб-сайту.

В процесі роботи над програмною моделлю виконано аналіз існуючих апаратних та програмних засобів. В повній мірі описані всі компоненти розробленого програмного забезпечення.

Розроблено зручний інтерфейс користувача. Наведені інструкції по роботі з програмними засобами.

Програма може використовуватися на ПЕОМ архітектури IBM PC з ОС Windows 10/11.

Програму розроблено на мові програмування Python.

Ключові слова: комп'ютерна інженерія, аналіз даних, колаборативна фільтрація, персоналізована реклама

Кафедра_КОПЗ_2022рік

ABSTRACT

Holodok O.D. Research and software implementation of a system for creating personalized advertising for website users. 123 Computer engineering. Central Ukrainian National Technical University. Kropyvnytskyi 2022.

In this master's thesis, software designed for a system for creating personalized advertising for website users.

The purpose of the development is the research and program implementation of a system for creating personalized advertising for website users.

The object of the research is the process of creating personalized advertising for website users.

The subject of the research is the methods and algorithms of collaborative data filtering for creating personalized advertising.

Research methods are based on data analysis methods, mathematical statistics methods, and software development methods.

The result of the work is a system for creating personalized advertising for website users.

In the process of working on a software model an analysis of existing hardware and software was performed. All components of the software developed are fully described.

User-friendly user interface is developed. These are instructions for working with software.

The program can be used on PCs of IBM PC architecture with Windows 10/11 OS.

The program was developed in the Python programming language.

Keywords: computer engineering, data analysis, collaborative filtering, personalized advertising

Кафедра_КОПЗ_2022рік

ЗМІСТ

Кафедра_КоПЗ_2022рік

Кафедра_КбПЗ_2022рік

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

БД – база даних.

Колаборативна фільтрація – процес фільтрації інформації, який використовує відомі уподобання групи користувачів для визначення невідомих уподобань деякого користувача.

ОС – операційна система.

ПЗ – програмне забезпечення.

ПК – персональний комп’ютер.

Рекомендатор – алгоритм, що створює персоналізовану рекламу.

СУБД – система управління базами даних.

Таргетована реклама – реклама, спрямована на деяку цільову аудиторію, яка відповідає певному набору параметрів, заданих рекламодавцем.

Кафедра_КбПЗ_2022рік

ВСТУП

Актуальність теми. В онлайн-рекламі дуже важливо показувати релевантну рекламу цільовим користувачам. Однак підбір корисної та цікавої реклами різним категоріям користувачів – це складний процес. Хоча звичайні методи підбору реклами дають хороші результати на загальну аудиторію, вони не можуть ефективно підбирати персоналізовану рекламу для певних груп користувачів. В той же час методи колаборативної фільтрації даних успішно застосовуються для підбору персоналізованого контенту в рекомендаційних системах, цей досвід різні дослідники та розробники починають успішно перенести на створення персоналізованої онлайн-реклами. Системи рекомендацій стають центральною частиною індустрії таргетованої реклами у мережі Інтеренет, яка швидко зростає на трильйони доларів. Тож, розробка нових методів та алгоритмів створення персоналізованої реклами на веб-сайтах на основі колабративної фільтрації є актуальною задачею.

Мета й завдання дослідження. Метою роботи є дослідження та програмна реалізація системи створення персоналізованої реклами для користувачів веб-сайту.

Для досягнення поставленої мети визначена програма дослідження, що складається з наступних завдань:

– Дослідження існуючих систем створення персоналізованої реклами для користувачів веб-сайту.

– Розробка методів та алгоритмів системи створення персоналізованої реклами для користувачів веб-сайту.

– Програмна реалізація системи створення персоналізованої реклами для користувачів веб-сайту.

Об'єктом дослідження є процес створення персоналізованої реклами для користувачів веб-сайту.

Кафедра_КбПЗ_2022рік

Предметом дослідження є методи та алгоритми колаборативної фільтрації даних для створення персоналізованої реклами.

Методи дослідження базуються на методах аналізу даних, методах математичної статистики та методах розробки програмного забезпечення.

Наукова новизна отриманих результатів. У процесі рішення завдань, обумовлених цілями дослідження, отримані наступні результати:

1. Удосконалено метод колаборативної фільтрації даних на основі матричної факторизації, який відрізняється від відомих запропонованим методом захисту від «бульбашки фільтрів».

2. Розроблено вітчизняний продукт генерації персоналізованої реклами для користувачів веб-сайту, який має більш широкі можливості, на відміну від існуючих аналогів та може бути використаний на різних веб-ресурсах.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми колаборативної фільтрації на основі матричної факторизації з захистом від «бульбашки фільтрів», що дозволяють успішно вирішувати задачі створення персоналізованої реклами для користувачів будь-якого веб-сайту.

Достовірність наукових результатів підтверджена теоретичними викладеннями, даними комп'ютерного моделювання, коректними дослідженнями параметрів на функціонуючій обчислювальній мережі, а також відповідністю отриманих результатів окремим результатам, наведеним у науковій літературі.

Виходячи з вищерозглянутого, дослідження та програмна реалізація системи створення персоналізованої реклами для користувачів веб-сайту, є актуальною науково-практичною задачею, яка потребує вирішення у цій випускній кваліфікаційній роботі.

Кафедра_КОПЗ_2022рік

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

Персоналізована (або таргетована) онлайн-реклама використовується у пошукових системах, контекстній рекламі, банерах, соціальних мережах та багатьх інших веб-ресурсах. Її призначення полягає у наданні рекламодавцю можливості задати або автоматично визначати необхідні параметри користувачів, за яким відбуватиметься відбір цільової аудиторії, враховуючи її інтереси.

Персоналізована реклама є більш релевантною для користувачів і дозволяє рекламодавцям підвищити рентабельність інвестицій у свої веб-ресурси. Припущення про інтереси користувачів, ґрунтуються на інформації про їх зафіксовані попередні дії на веб-сайті чи групі веб-сайтів, що обмінюються інформацією. Прогнозовані різними методами штучного інтелекту інтереси користувачів дозволяють рекламодавцям враховувати їх у своїх кампаніях, завдяки чому покращується сприйняття реклами та підвищується її ефективність.

Онлайн-реклама вважається персоналізованою (таргетованою), якщо при її підборі враховується зібрана раніше інформація чи статистика про дії користувачів. До такої інформації можуть належати пошукові запити користувачів, їх дії в Інтернеті, історія відвідування сайтів та робота з додатками, лайки та оцінки виставлені конетенту веб-сайту, а також демографічні та геолокаційні дані. Подібна інформація та статистика дій користувача дає можливість для персоналізації реклами застосовувати демографічне націлення, націлення на категорії інтересів, націлення на типи користувачів тощо.

Таким чином персоналізована реклама налаштовується на людей, які виявляють інтерес до певного контенту, підвищує кількість кліків на рекламних оголошеннях та збільшує лояльність до веб-сайту, адже користувач зауважує, що реклама відповідає його інтересам.

Кафедра_КПЗ_2022рік

1.2 Область застосування

Персоналізована реклама буває різних типів в залежності від очікуваного результату, технічних можливостей та розміру рекламного бюджету. Чим вищий рівень персоналізації, тим складніші технічні рішення та кращий результат.

Персоналізовану рекламу можна використати у наступних випадках:

– Оголошення в соціальних мережах.

– Оголошення в мобільних додатках.

– Рекламні банери на веб-сайті.

– Оголошення в пошукових системах.

– Поштові розсилки.

– тощо.

Виділяють наступні види сучасної персоналізованої реклами:

– Локальна.

– Соціально-демографічна.

– Поведінкова.

– Часова.

– За інтересами.

– За кількістю запитів.

Усі вони відрізняються набором критеріїв для ранжування цільової аудиторії. Хороші найкращі результати дають рекламні компанії, що використовують та поєднують всі види персоналізованої реклами. У даній роботі основні дослідження та розробки були присвячені методам створення персоналізованої реклами за інтересами користувачів.

Кафедра_КбПЗ_2022рік

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

2.1 Огляд існуючих систем, технологій, архітектур, програмних рішень за профілем теми магістерської роботи

Для підбору персоналізованої реклами на основі інтересів користувачів найчастіше застосовуються різні методи фільтрації даних, зокрема, методи контентої та колаборативної фільтрації. Було проведено дослідження даних методів створення персоналізованої реклами, його результати наведені нижче.

Контентна фільтрація

Методи засновані на контентній фільтрації, використовують різну інформацію про користувачів та/або елементи контенту. Наприклад, додатковою інформацією можуть бути вік, стать, робота або будь-яка інша особиста інформація користувачів з їх профілю, а також властивості та ключові слова контенту, який вони переглядають.

Основна ідея методів, заснованих на контентній фільтрації, полягає в тому, щоб спробувати побудувати модель на основі доступних «функцій», які пояснюють спостережувану взаємодію між користувачем і елементом.

Рекомендатор (алгоритм, що створює персоналізовану рекламу) на основі контентної фільтрації працює з даними, які надає користувач явно (лайки, оцінки) або неявно (клацнувши посилання). На основі цих даних створюється профіль користувача, який потім використовується для надання пропозицій користувачеві. Оскільки користувач надає більше вхідних даних або виконує дії відповідно до цих рекомендацій, механізм стає все точнішим.

Рекомендатор має вибрати між двома методами доставки інформації, надаючи користувачеві персоналізовану рекламу:

– *Експлуатація*. Система вибирає об’єкти, подібні до тих, яким користувач уже віддав перевагу.

Кафедра_КБПЗ_2022рік

– *Розвідка*. Система вибирає об’єкти, у яких профіль користувача не містить доказів для прогнозування реакції користувача.

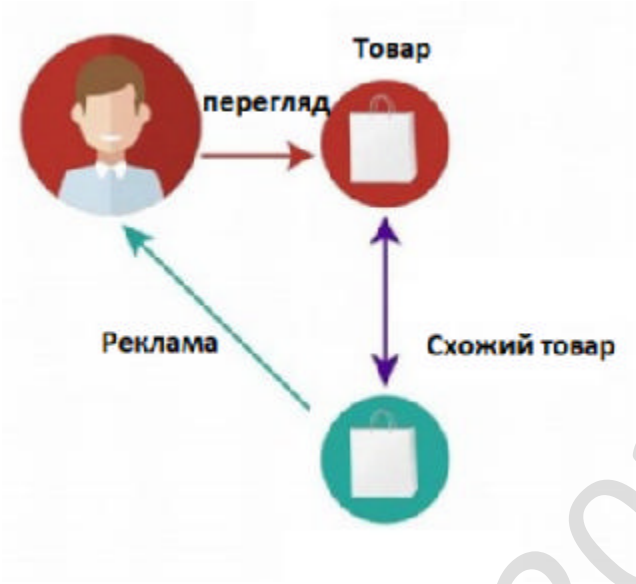

Рисунок 2.1 – Схема контентної фільтрації

Існує 3 складові процедури створення реклами цим методом:

– Створення кандидатів: цей метод відповідає за створення менших підмножин кандидатів для рекомендації користувачеві, враховуючи величезний пул із тисяч елементів.

– Системи підрахунку балів: Генерацію кандидатів можуть виконувати різні генератори, тому потрібно все стандартизувати та спробувати призначити оцінку кожному з елементів у підмножинах. Це робиться за системою підрахунку балів.

– Системи повторного рейтингу: після підрахунку балів система враховує інші додаткові обмеження для створення остаточного рейтингу.

Система рекомендацій на основі контенту працює двома методами, обидва з яких використовують різні моделі та алгоритми. Один використовує метод векторного інтервалу та називається методом 1, а інший використовує модель класифікації та називається методом 2.

1. Метод векторного простору. На основі оцінок користувача створюється його вектор, який ранжує елементи. Після цього створюється вектор елементів, у якому елементи ранжуються.

Кафедра_КОПЗ_2022рік

2. Метод класифікації. У ньому ми можемо створити дерево рішень і дізнатися, чи захоче користувач придбати товар чи ні.

Мета контентної фільтрації полягає в тому, щоб класифікувати елементи за певними ключовими словами, дізнатися, що подобається клієнтам, знайти ці терміни в базі даних, а потім порекомендувати подібні речі.

Переваги:

– Легко масштабується для великої кількості клієнтів, оскільки дані інших користувачів не потрібні, щоб рекомендувати щось конкретному користувачеві.

– Оскільки рекомендації базуються на повсякденній діяльності користувача, усі вподобання та параметри пропозицій точно налаштовані відповідно до вибору користувача. Таким чином, модель може рекомендувати конкретні ніші, які можуть не зацікавити інших користувачів.

– Найновіші елементи можна запропонувати відразу після їх запуску, не чекаючи перепису, оскільки функції доступні з самого початку.

Недоліки:

– Створення механізму рекомендацій на основі вмісту потребує багато знань предметної області, оскільки вибір функцій елементів здебільшого жорстко закодований у системі. Таким чином, модель хороша настільки, наскільки добре знання того, хто її створює.

– Модель може рекомендувати нові елементи на основі поточного інтересу користувача. Отже, виявити та розширити нові шляхи, які можуть зацікавити користувача, неможливо.

– Проблема холодного старту є значним недоліком, оскільки двигун не має достатньо інформації про нового користувача, щоб почати робити пропозиції.

– Важко давати нові рекомендації не дуже активним користувачам.

Колаборативна фільтрація на основі пам’яті

Колаборативна фільтрація на основі пам’яті заснована на знаходженні коефіцієнтів подоби між користувачами або між елементами на основі оцінок які

Кафедра_КОПЗ_2022рік

користувачі ставлять елементам, та відповідно поділяється на види: «користувач-користувач» і «елемент-елемент».

Колаборативна фільтрація на основі пам'яті «користувач-користувач»

Щоб надати нову рекомендацію користувачеві, метод «користувач-користувач» приблизно намагається ідентифікувати користувачів із найбільш подібним «профілем взаємодії» (найближчими сусідами), щоб запропонувати елементи, які є найпопулярнішими серед цих сусідів (і це «нове» для нашого користувача). Цей метод називають «орієнтованим на користувача», оскільки він представляє користувачів на основі їх взаємодії з елементами та оцінює відстані між користувачами.

Припустімо, що ми хочемо дати рекомендацію для певного користувача. По-перше, кожен користувач може бути представлений своїм вектором взаємодії з різними елементами («своєю лінією» в матриці взаємодії). Тоді ми можемо обчислити якусь «схожість» між цим користувачем і всіма іншими користувачами. Ця міра подібності така, що двох користувачів, які однаково взаємодіють з тими самими елементами, слід вважати близькими. Після обчислення подібності для кожного користувача ми можемо зберегти *k*-найближчих сусідів для нашого користувача, а потім запропонувати найпопулярніші елементи серед них (переглядаючи лише елементи, з якими наш контрольний користувач ще не взаємодіяв).

Під час обчислення подібності між користувачами слід уважно враховувати кількість «загальних взаємодій» (скільки елементів уже розглянули обидва користувачі) і справді, у більшості випадків ми хочемо уникнути того, щоб хтось, хто має лише одну спільну взаємодію з досліджуваним користувачем, міг мати 100% збіг і вважався «ближчим», ніж той, хто має 100 спільних взаємодій і погоджується «лише» на 98% з них. Таким чином, ми вважаємо, що два користувачі схожі, якщо вони взаємодіяли з багатьма спільними елементами однаково (подібний рейтинг, схожий час взаємодії…).

Кафедра_КОПЗ_2022рік

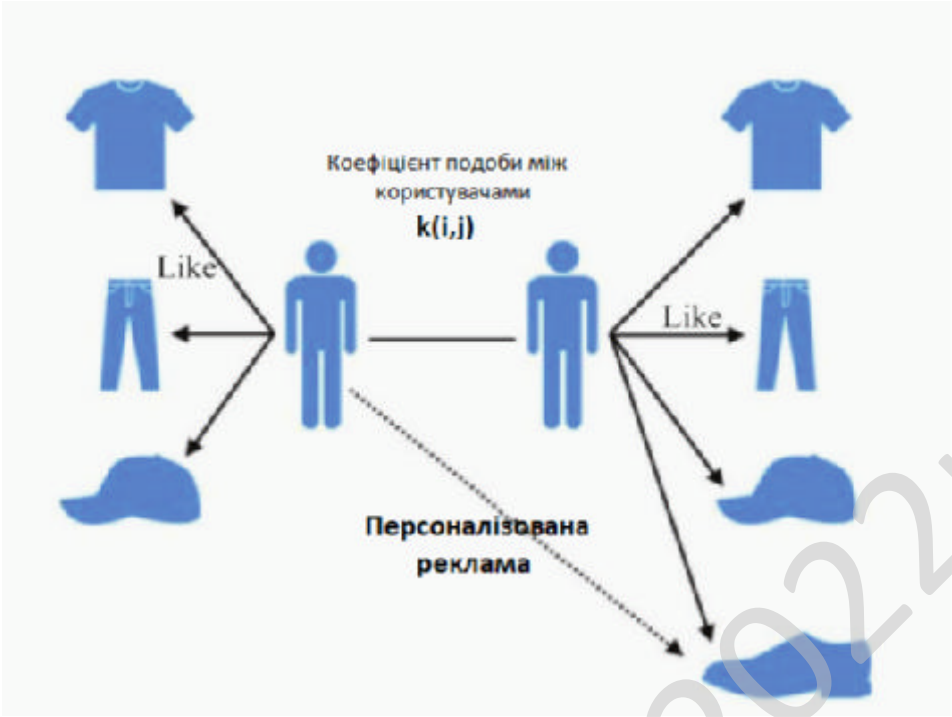

Рисунок 2.2 – Схема колаборативної фільтрації на основі пам’яті «користувач-користувач»

Колаборативна фільтрація на основі пам’яті «елемент-елемент»

Ідея методу «елемент-елемент» полягає в тому, щоб знайти елементи, схожі на ті, з якими користувач уже «позитивно» взаємодіяв. Два елементи вважаються подібними, якщо більшість користувачів, які взаємодіяли з ними обома, робили це подібним чином. Цей метод називається «орієнтованим на предмет», оскільки він представляє елементи на основі взаємодії користувачів з ними та оцінює відстані між цими елементами.

Припустімо, що ми хочемо дати рекомендацію для певного користувача. Спочатку ми розглядаємо предмет, який сподобався цьому користувачеві найбільше, і представляємо його (як і всі інші елементи) його вектором взаємодії з кожним користувачем («його стовпець» у матриці взаємодії). Тоді ми можемо обчислити подібність між «найкращим предметом» і всіма іншими елементами. Після обчислення подібності ми можемо зберегти *k*-

Кафедра_КбПЗ_2022рік

найближчих сусідів для вибраного «найкращого елемента», які є новими для нашого користувача, і рекомендувати ці елементи.

Зауважте, що для отримання релевантніших рекомендацій ми можемо виконати цю роботу не лише для улюбленого продукту користувача, а натомість розглянути *n* бажаних елементів. У цьому випадку ми можемо рекомендувати товари, близькі до кількох із цих бажаних позицій.

Рисунок 2.3 – Схема колаборативної фільтрації на основі пам’яті «елемент-елемент»

Порівняння методів «користувач-користувач» і «елемент- елемент»

Метод «користувач-користувач» заснований на пошуку схожих користувачів за взаємодією з елементами. Оскільки, як правило, кожен користувач взаємодіяв лише з кількома елементами, це робить метод досить чутливим до будь-яких записаних взаємодій (висока дисперсія). З іншого боку, оскільки остаточна рекомендація ґрунтується лише на взаємодії, зареєстрованій для користувачів, подібних до користувача, який нас цікавить, ми отримуємо більш персоналізовані результати (низьке упередження).

Кафедра_КОПЗ_2022рік

Навпаки, метод «елемент-елемент» базується на пошуку подібних елементів з точки зору взаємодії користувача з елементом. Оскільки, як правило, багато користувачів взаємодіяли з елементом, пошук сусідів набагато менш чутливий до окремих взаємодій (менша дисперсія). На противагу цьому в рекомендації враховуються взаємодії з усіма типами користувачів (навіть користувачів, які сильно відрізняються від нашого контрольного користувача), що робить метод менш персоналізованим (більш упередженим). Таким чином, цей підхід менш персоналізований, ніж підхід «користувач-користувач», але більш надійний.

Один із найбільших недоліків методів колаборативної фільтрації на основі пам'яті полягає в тому, що вони важко масштабуються: створення нової рекомендації може зайняти надзвичайно багато часу для великих систем. Дійсно, для систем з мільйонами користувачів і мільйонами елементів крок пошуку найближчих сусідів може стати нерозв'язним, якщо не бути ретельно розробленим (алгоритм KNN має складність O(*ndk*) з *n* кількістю користувачів, *d* кількістю елементів і *k* кількість розглянутих сусідів). Щоб зробити обчислення більш зручними для величезних систем, ми можемо скористатися перевагами розрідженості матриці взаємодії під час розробки алгоритму або використовувати методи наближених найближчих сусідів (ANN).

У більшості алгоритмів рекомендацій необхідно бути надзвичайно обережними, щоб уникнути ефекту «багаті-стають багатшими» для популярних елементів і щоб уникнути застрягання користувачів у так званій «області інформаційного обмеження» (бульбашці фільтрів). Іншими словами, ми не хочемо, щоб наша система все частіше рекомендувала лише популярні товари, а також ми не хочемо, щоб наші користувачі отримували лише рекомендації щодо товарів, дуже близьких до тих, що їм уже сподобалися, без можливості побачити нові предмети, які їм також можуть сподобатися (оскільки ці предмети не «достатньо близькі», щоб їх пропонувати). Ці проблеми можуть виникнути в більшості алгоритмів створення персоналізованої реклами, це особливо актуально для колаборативних методів фільтрації на основі пам'яті.

Кафедра_КОПЗ_2022рік

Методи колаборативної фільтрації, засновані на моделі

Методи колаборативної фільтрації, засновані на моделі, покладаються лише на інформацію про взаємодію між користувачем і елементом і припускають приховану модель, яка повинна пояснити ці взаємодії. Наприклад, алгоритми факторизації матриць полягають у розкладанні величезної та розрідженої матриці взаємодії користувача з елементами на добуток двох менших і щільних матриць: матриці факторів користувача (що містить представлення користувачів), яка множить матрицю факторів на елементи (містить представлення елементів).

Колаборативна фільтрація на основі матричної факторізації

Основне припущення, що лежить в основі матричної факторизації, полягає в тому, що існує досить низьковимірний латентний простір ознак, у якому ми можемо представити як користувачів, так і елементи, і такий, що взаємодія між користувачем і елементом може бути отримана шляхом обчислення скалярного добутку відповідних щільних векторів в цьому просторі.

Наприклад, у нас є матриця рейтингів фільмів, які використовують користувачі. Щоб змоделювати взаємодію між користувачами та фільмами, ми можемо припустити, що:

- існують деякі особливості, які досить добре описують (і розрізняють) фільми.
- ці функції також можна використовувати для опису вподобань користувача (високі значення для функцій, які подобаються користувачеві, низькі значення в іншому випадку)

Однак ми не хочемо явно надавати ці функції нашій моделі (як це можна було б зробити для підходів на основі вмісту, які ми опишемо пізніше). Натомість ми вважаємо за краще дозволити системі самостійно виявляти ці корисні функції та створювати власні представлення як користувачів, так і елементів. Оскільки вони вивчаються, а не надаються, одержані ознаки, взяті окремо, мають математичне значення, але не мають інтуїтивного тлумачення (і тому їх важко, якщо не зовсім неможливо, зрозуміти людині). Однак не є чимось незвичайним,

Кафедра_КБПЗ_2022рік

коли в кінцевому підсумку отримуються структури, що випливають із такого типу алгоритму, надзвичайно близькі до інтуїтивної декомпозиції, про яку людина може думати. Дійсно, наслідком такої факторизації є те, що близькі користувачі з точки зору переваг, а також близькі елементи з точки зору характеристик в кінцевому підсумку мають близькі представлення в прихованому (латентному) просторі.

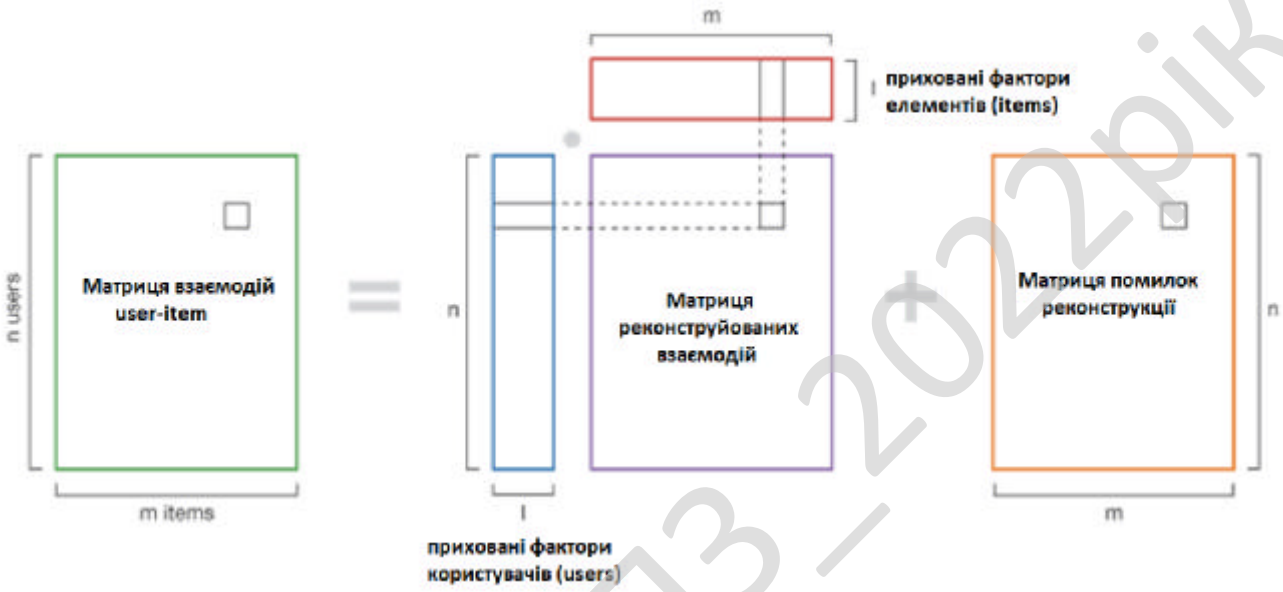

Рисунок 2.3 – Схема колаборативної фільтрації на основі матричної факторизації

Давайте розглянемо матрицю взаємодії M (nxm) рейтингів, де лише деякі елементи були оцінені кожним користувачем (для більшості взаємодій відсутня оцінка). Ми хочемо розкласти цю матрицю на множники так, що:

$$M \approx \{X.Y^T$$

де X – «матриця користувачів» (nxl), рядки якої представляють n користувачів, а Y – «матриця елементів» (mxl), рядки якої представляють m елементів:

$$user_i \equiv X_i, \ \forall_i \in \{1, \dots, n\}$$

$$item_i \equiv Y_i \ \forall_j \in \{1, \dots, m\}$$

Тут l – розмір прихованого простору, в якому будуть представлені користувачі та елемент. Отже, ми шукаємо матриці X і Y, скалярний добуток яких найкраще наближає існуючі взаємодії. Позначаючи E ансамбль пар (i,j) так, що

Кафедра_КбПЗ_2022рік

встановлено M_{ij} (а не None), ми хочемо знайти X і Y, які мінімізують «помилку реконструкції рейтингу»:

$$(X,Y) = \underset{X,Y}{arg\min} \sum_{(i,j)\in E} [(X_i)(Y_j)^T - M_{ij}]^2$$

Додавши регуляризаційний коефіцієнт і поділивши на 2, отримаємо:

$$(X,Y) = \underset{X,Y}{arg\min} \frac{1}{2} \sum_{(i,j)\in E} \left[(X_i)(Y_j)^T - M_{ij}\right]^2 + \frac{\lambda}{2}(\sum_{i,k}(X_{ik})^2 + \sum_{j,k}(Y_{jk})^2)$$

Потім матриці X і Y можна отримати після процесу оптимізації градієнтним спуском, для якого ми можемо помітити дві речі. По-перше, градієнт не потрібно обчислювати для всіх пар у просторі E на кожному кроці, і ми можемо розглядати лише підмножину цих пар, щоб оптимізувати нашу цільову функцію «пакетно». По-друге, значення в X і Y не потрібно оновлювати одночасно, а градієнтний спуск можна виконувати альтернативно для X і Y на кожному кроці (таким чином ми вважаємо одну матрицю фіксованою та оптимізуємо іншу, перш ніж робити протилежне на наступній ітерації.

Після того, як матрицю було розкладено на множники, ми можемо просто помножити вектор користувача на будь-який вектор елемента, щоб оцінити відповідний рейтинг. Зауважте, що ми також можемо використовувати методи «користувач-користувач» і «елемент-елемент» із цими новими представленнями користувачів і елементів: (приблизні) пошуки найближчих сусідів здійснюватимуться не за величезними розрідженими векторами, а за маленькими щільними, що зробить деякі методи апроксимації більш зручними.

Порівняння – фільтрації з контентною фільтрацією:

– Методи контентної фільтрації вимагають значної кількості інформації про функції елемента, а не про його взаємодію з користувачем. Для таких товарів, як одяг, такими ознаками можуть бути розмір, колір, бренд, матеріал тощо, а у випадку фільмів – актори, жанр, режисер, рік випуску тощо.

– Колаборативна фільтрація, з іншого боку, використовує історію взаємодії між користувачами та елементами, щоб об’єднати користувачів зі схожими

Кафедра_КОТЗ_2022рік

смаками та запропонувати цільовому користувачеві нові продукти, популярні в групі.

– Моделі фільтрації на основі контенту значною мірою базуються на знаннях предметної області, оскільки функції елементів вручну впроваджуються в систему. Для колаборативної фільтрації не потрібні такі глибокі знання предметної області, оскільки всі вбудовування вивчаються автоматично.

– Системи колаборативної фільтрації вимагають лише даних про поведінку користувача, тоді як методи, засновані на фільтрації контенту, вимагають і даних про користувачів, і даних про елементи.

2.2 Обґрунтування вибору засобів для побудови системи та мови програмування

Для розробки багатоагентної системи створення персоналізованої реклами для користувачів веб-сайту було обрано мову програмування Python.

За останні роки Python став однією з найпопулярніших мов програмування у світі. Його використовують у всьому, від машинного навчання до створення веб-сайтів і тестування програмного забезпечення.

Python – це високорівнева мова програмування загального призначення, що означає, що її розроблено для використання в різних програмах, включаючи аналіз даних, веб-розробку, створення рекомендацій. Таким чином вона ідеально підходить для вирішення задач створення персоналізованої онлайн-реклами.

Аналіз даних і машинне навчання

Python став основним продуктом науки про дані, дозволяючи аналітикам даних та іншим професіоналам використовувати цю мову для проведення складних статистичних обчислень, створення візуалізацій даних, створення алгоритмів машинного навчання, обробки та аналізу даних, а також виконання інших завдань, пов’язаних із даними.

Кафедра_КПЗ_2022рік

Python може створювати широкий спектр різних візуалізацій даних, як-от лінійні та стовпчасті діаграми, секторні діаграми, гістограми та 3D-графіки. Python також має низку бібліотек, які дозволяють програмістам писати програми для аналізу даних і машинного навчання швидше й ефективніше, наприклад TensorFlow і Keras.

Веб-розробка

Python часто використовується для розробки скриптів веб-сайту. Роль Python у веб-розробці може включати надсилання даних на сервери та з них, обробку даних і зв'язок із базами даних, маршрутизацію URL-адрес і забезпечення безпеки. Python пропонує декілька фреймворків для веб-розробки. Найбільш часто використовувані включають Django і Flask.

Переваги та причини популярності Python

Python популярний з кількох причин. Ось глибший погляд на те, що робить його таким універсальним і простим у використанні для програмістів.

- Він має **простий синтаксис**, який імітує природну мову, тому його легше читати та розуміти. Це дозволяє швидше створювати проекти та швидше їх покращувати.
- Він **універсальний**. Python можна використовувати для багатьох різних завдань, від веб-розробки до машинного навчання.
- Він **зручний для початківців**, що робить його популярним для програмістів початкового рівня.
- Він має **відкритий сирцевий код**, що означає, що його можна безкоштовно використовувати та розповсюджувати навіть у комерційних цілях.
- Архів **модулів і бібліотек** Python – наборів коду, створених сторонніми користувачами для розширення можливостей Python – величезний і постійно зростає.
- Python має **велике й активне співтовариство**, яке вносить свій внесок у пул модулів і бібліотек Python і діє як корисний ресурс для інших програмістів. Велика спільнота підтримки означає, що якщо кодери стикаються з

Кафедра_КбПЗ_2022рік

каменем спотикання, знайти рішення відносно легко; хтось обов'язково стикався з такою ж проблемою раніше.

2.3 Розгорнута постановка завдання

Наступним пунктом є аналіз поставленої задачі в усіх напрямках та виокремлення основних положень, що потребуватимуть розгляду.

Згідно з навчальними матеріалами, а також з технічним завданням на кваліфікаційну магістерську роботу, підлягає реалізації програмне забезпечення системи створення персоналізованої реклами для користувачів веб-сайту.

В процесі розробки даного додатку необхідно провести групування виокремлених пунктів у деякий план та паставити правильні задачі.

Отже, при постановці завдання були виділені такі пункти плану:

- проведення аналізу існуючих систем, а також виявити їх позитивні та негативні сторони;
- визначити методику створення системи створення персоналізованої реклами для користувачів веб-сайту для використання її у різних сферах. Побудувати структурну та функціональну схему;
- розробити програмне забезпечення для виконання поставленої задачі. Побудувати блок-схеми алгоритмів і підалгоритмів;
- розробити рекомендації по організаційних та методичних заходах, які забезпечать впровадження системи в промислову експлуатацію та її подальшу успішну експлуатацію;
- сформувати висновки про разроблену систему та методики, що були використані при розробці.

Кафедра_КОПЗ_2022рік

3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Опис функціонування системи

У розробленій системі для створення персоналізованої реклами корситувачам веб-сайту було вирішено застосувати колаборативну фільтрацію на основі матричної факторизації, а також розробити вдосконалення до неї, додаванням певної випадкової інформації для вирішення проблеми «бульбашки фільтрів».

Для формування персоналізованої реклами на основі колаборативної фільтрації перш за все треба зібрати статистику про дії користувачів. У найбільш простому випадку збираються дані про поставлені користувачами оцінки та записуються у матрицю рейтингів (рис. 3.1).

В матриці рейтингів значення є оцінками конкретного користувача для конкретного об’єкту. Відсутні значення в таблиці рейтингів є невідомими, тобто, для певного об’єкту певний користувач не виставив оцінку.

	Елемент 1	**Елемент 2**	**…**	**Елемент *i***
Користувач 1	2	5	…	5
Користувач 2	3	–	…	4
…	…	…	…	…
Користувач *u*	–	5	…	3

Рисунок 3.1 – Матриця рейтингів для колаборативної фільтрації

Методи колаборативної фільтрації з матричною факторизацією здійснюють визначення інтересів користувачів на основі прихованих факторів елементів системи, що впливають на вподобання користувачів та можуть бути одержані за допомогою факторизації матриці рейтингів та отримання змішаної матриці, що містить рейтинги та ознаки елементів веб-сайту. Визначені інтереси користувачів

Кафедра_КБПЗ_2022рік

і дозволять створити таргетовану рекламу.

Факторизація матриці здійснює декомпозицію матриці в набір інших матриць (факторів), добуток яких дає початкову матрицю.

Алгоритм факторизації матриці рейтингів виглядає наступним чином:

Крок 1. Ініціалізуємо матриці M_u та M_i випадковими значеннями.

Крок 2. Перемножуємо матриці M_u та M_i і порівнюємо результат з матрицею рейтингів R, обчислюємо помилки для кожної комірки та загальну помилку.

Крок 3. Мінімізуємо обчислені помилки за допомогою градієнтного спуску, поки загальна помилка не знизиться до задовільного значення.

Потім треба визначити базові зміщення $b_{u,i}$, які складаються з базових зміщень окремих користувачів b_u і базових зміщень окремих елементів b_i, а також загального середнього рейтингу елементів у системі μ:

$$b_{u,i} = \mu + b_u + b_i. \quad (3.1)$$

Визначення рейтингу для пари користувач-елемент здійснюється за такою формулою:

$$\hat{r}_{\mathrm{u,i}} = \mu + b_u + b_i + q_i \cdot p_u. \quad (3.2)$$

де q_i – вектор прихованих факторів елементу i, а p_u – вектор прихованих факторів користувача u.

Далі треба обчислити глобальну середню оцінку та усі зміщення. Потім треба знайти найкращі зміщення та приховані фактори, що дозволяють визначати інтереси користувачів з найменшою помилкою.

Помилка визначається за наступною формулою:

$$E = \sum_{(u,i)\in D}(r_{u,i} - \hat{r}_{u,i})^2, \quad (3.3)$$

$$E = \sum_{(u,i)\in D}(r_{u,i} - \mu - b_u - b_i - q_i \cdot p_u)^2, \quad (3.4)$$

де $r_{u,i}$ – справжній рейтинг об’єкту i у користувача u; $\hat{r}_{u,i}$ – прогнозований рейтинг.

Ця функція оптимізується методом градієнтного спуску. Беруться часткові похідні за кожним аргументом, рух під час процедури градієнтного спуску відбувається у зворотному напрямку до цих похідних.

Щоб вирішити можливу проблему перенавчання системи треба

Кафедра_КПЗ_2022рік

використати регуляризацію. Регуляризуючий вираз $R(f)$ додається до значення помилки:

$$f_* = \underset{f}{argmin} \sum_{i=1}^{n} E(f(\hat{x}_i),\ \hat{y}_i) + \lambda \cdot R(f), \tag{3.5}$$

де E – функція визначення помилки обчислення інтересів користувачів $f(x)$ для значень y, параметр λ коефіцієнт важливості регуляризації.

Під час градієнтного спуску використовуються наступні правила для оптимізації змінних, що впливають на результат:

$$b_u = b_u + \gamma\left(e_{u,i} - \lambda b_u\right), \tag{3.6}$$

$$b_i = b_i + \gamma\left(e_{u,i} - \lambda b_i\right), \tag{3.7}$$

$$q_{i,k} = q_{i,k} + \gamma\left(e_{u,i} \cdot p_{u,k} - \lambda q_{i,k}\right), \tag{3.8}$$

$$p_{u,k} = p_{u,k} + \gamma\left(e_{u,i} \cdot q_{i,k} - \lambda p_{u,k}\right), \tag{3.9}$$

де $e_{u,i} = r_{u,i} - \hat{r}_{u,i}$ – помилка при навчанні моделі; γ – швидкість навчання.

Алгоритм градієнтного спуску був застосований у цій роботі для навчання колаборативної фільтрації в процесі факторизації матриці рейтингів.

Градієнтний спуск – це алгоритм оптимізації, який використовується для різноманітного навчання моделі машинного навчання. Він найкраще підходить для завдань, в яких є велика кількість функцій і дуже багато зразків, щоб помістити в пам'яті моделі машинного навчання. У цій статті я познайомлю вас із алгоритмом градієнтного спуску в машинному навчанні та його реалізацією з використанням Python.

Алгоритм градієнтного спуску

У машинному навчанні градієнтний спуск – це алгоритм оптимізації, здатний знаходити найбільш сприятливі рішення для кола проблем. Він працює через повторення параметрів для мінімізації функції вартості.

Найважливішим параметром цього алгоритму є розмір кроків, що визначається швидкістю навчання. Якщо швидкість навчання дуже низька, алгоритм проходитиме безліч ітерацій, щоб зійтися, що зрештою займе дуже багато часу.

Кафедра_КОПЗ_2022рік

Але якщо швидкість навчання дуже висока, він розділить алгоритм великі значення, що допоможе знайти хороше рішення. При використанні алгоритму градієнтного спуску необхідно переконатися, що всі функції мають однаковий масштаб, інакше знадобиться багато часу, щоб зійтися.

Типи алгоритмів градієнтного спуску

Щоб реалізувати алгоритм градієнтного спуску, нам необхідно обчислити градієнт функції вартості, що призводить до трьох типів алгоритмів градієнтного спуску, які відрізняються обсягом даних, що використовуються для обчислення значення градієнта функції вартості.

Пакетний градієнтний спуск

Він використовує весь пакет даних кожному етапі ітерації, у результаті досить повільно обробляє великі набори навчальних даних. Це краще, тільки якщо ви хочете обробити всі екземпляри в навчальному наборі. Єдине обмеження полягає в тому, що якщо навчальний набір дуже великий, він буде дуже повільним. Давайте подивимося, як можна реалізувати його за допомогою Python:

```
import numpy as np
eta = 0.1
n_iter = 1000
m = 100

X = 2 * np.random.rand(100, 1)
y = 4 + 3 * X + np.random.randn(100, 1)
X_b = np.c_[np.ones((100, 1)), X]  # add x0 = 1 to each instance
theta = np.random.randn(2,1) #ініціалізація
for iteration in range(n_iter):
    gradients = 2/m * X_b.T.dot(X_b.dot(theta)-y)
    theta = theta - eta * gradients
print(theta)
```

Основна функція градієнта полягає у вимірі змін у кожній вазі щодо зміни помилок. Можна думати про градієнти як про нахил функції. Нахил буде крутішим залежно від висоти градієнта – це сприятлива умова для моделей, оскільки вони можуть швидко вчитися. Однак модель перестане вчитися, якщо нахил стане нульовим. З математичної точки зору, градієнт найкраще описувати як обмежену похідну щодо його вхідних даних.

Кафедра КбПЗ 2022рік

Вкрай важливо встановити швидкість навчання на відповідні значення, щоб допомогти градієнтному спуску досягти місцевих мінімумів. Тому краще не встановлювати їх надто високі чи низькі. Це дуже важливо, оскільки досягнення мінімуму може ускладнитися надмірно довгими кроками. Тому, якщо ми встановимо швидкість навчання на менші значення, то градієнтний спуск, зрештою, може досягти своїх локальних мінімумів. Однак це може тривати деякий час.

Стохастичний градієнтний спуск

Пакетний градієнтний спуск працює дуже повільно, оскільки він використовує всі навчальні дані для обчислення градієнтів на кожному етапі ітерації. Це обмеження долається з допомогою алгоритму стохастичного градієнтного спуску. Цей метод отримує випадковий екземпляр навчального набору на кожному кроці ітерації та обчислює градієнти на основі цього випадкового екземпляра. Таким чином, робота з одним екземпляром триває швидше.

Міні-пакетний градієнтний спуск

Останній тип алгоритму градієнтного спуску – міні-пакетний градієнтний спуск. Він обчислює градієнти для невеликих випадкових наборів екземплярів, відомих як міні-пакети, замість обчислення набору навчальних даних або одного випадкового екземпляра.

Проблеми методів колаборативної фільтрації

При застосуванні алгоритмів колаборативної фільтрації та багатьох інших алгоритмів фільтрації можуть виникати наступні проблеми:

– **Розрідженість даних.** Як правило, більшість рекомендаторів ґрунтується на великій кількості даних (товарів), тоді як більшість користувачів не ставить оцінки товарам. У результати цього матриця «елемент-користувач» виходить дуже великою і розрідженою, що становить проблеми при обчисленні вподобань. Ця проблема особливо гостра нових, щойно виникли систем. Також розрідженість даних посилює проблему холодного старту.

Кафедра_КоПЗ_2022рік

– **Масштабованість.** Зі збільшенням кількості користувачів у системі, з'являється проблема масштабованості. Наприклад, маючи 10 мільйонів покупців, алгоритм колаборативної фільтрації занадто складний для розрахунків. Також, багато систем повинні моментально реагувати на онлайн запити від усіх користувачів, незалежно від історії їх покупок та оцінок, що потребує ще більшої масштабованості.

– **Проблема холодного старту.** Нові елементии або користувачі становлять велику проблему для рекомендаторів. Частково проблему допомагає вирішити підхід, заснований на аналізі вмісту, оскільки він покладається не так на оцінки, але в атрибути, що допомагає включати нові предмети у рекомендації користувачам. Проте проблему з наданням рекомендації нового користувача вирішити складніше.

– **Синонімія** – Синонімією називається тенденція схожих та однакових предметів мати різні імена. Більшість рекомендаторів не здатні виявити ці приховані зв'язки і тому ставляться до цих предметів як різних. Наприклад, «фільми для дітей» і «дитячий фільм» відносяться до одного жанру, але система сприймає їх як різні.

– **Шахрайство.** У рекомендаційних системах, де кожен може ставити оцінки, люди можуть давати позитивні оцінки своїм предметам та погані своїм конкурентам. Також, таргетована реклама стала сильно впливати на продаж та прибуток, відколи отримали широке застосування в комерційних сайтах. Це призводить до того, що недобросовісні постачальники намагаються шахрайським чином піднімати рейтинг своїх товарів хороших і знижувати рейтинг своїх конкурентов.

– **Різноманітність.** Колаборативна фільтрація спочатку визнана збільшити різноманітність, щоб дозволяти відкривати користувачам нові продукти з безлічі. Однак деякі алгоритми, зокрема основні на продажах та рейтингах, створюють дуже складні умови для просування нових та маловідомих продуктів, оскільки їх заміняють популярні продукти, які давно знаходяться на ринку. Це своє чергу

Кафедра_КОПЗ_2022рік

лише підвищує ефект «багаті стають ще багатшими» і призводить до меншої різноманітності.

– **Білі ворони.** До «білих ворон» належать користувачі, чия думка постійно не збігається з більшістю інших. Через їх унікальний смак їм неможливо щось рекомендувати. Проте, такі мають проблеми з отриманням рекомендацій й у реальному житті, тому пошуки вирішення цієї проблеми нині не ведутся.

– **Бульбашка фільтрів** (англ. Filter bubble) – поняття, розроблене Інтернет-активістом Ілаєм Парайзером, це негативна сторона персоналізованої реклами та персоналізованого пошуку, коли веб-сайти починають пропонувати користувачу дуже невелику частину контенту на основі його попередніх дій, звужуючи його поле зору. В результаті веб-сайти показують лише інформацію, яка узгоджується з минулими поглядами цього користувача. Вся інша інформація, зазвичай, користувачеві не виводиться.

Для подолання проблеми «бульбашки фільтрів» було вирішено додати випадковість та різноманітність до рекомендацій оголошень персоналізованої реклами. Нові елементи у системі можуть ще не мати оцінок і не бути популярними, але вони можуть бути цікавими користувачам через свою новизну. Але нові елементи ще не мають оцінок, і невідомо кому їх рекомендувати. Новизна – характеристика елемента протилежна його популярності, і може визначатися формулою:

$$novelty(i) = -\log_2 p(i), \tag{3.10}$$

де $p(i)$ – ймовірність того, що елемент i потрапить у список рекламних оголошень деяким користувачам і сподобається їм.

У цій роботі пропонується визначати для кожного елемента його новизну, обирати N найновіших елементів та додавати з ймовірністю 0,3 їх у списки оголошень персоналізованої реклами.

Кафедра_КОПЗ_2022рік

3.2 Розробка структурної схеми

Структурна схема розробленого програмного забезпечення зображена на рисунку 3.2.

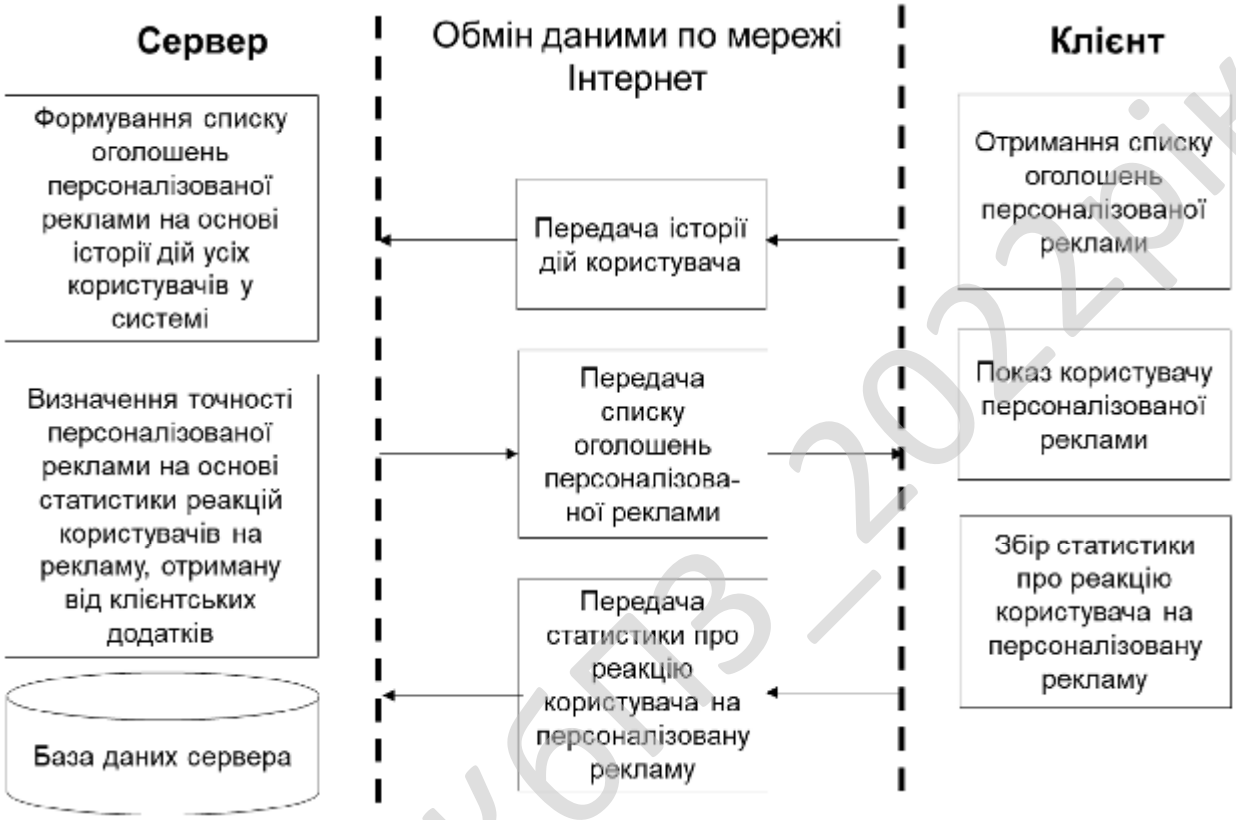

Рисунок 3.2 – Структурна схема системи

На рисунку видно, що система складається з наступних частин:

- Сервер;
- Обмін даними по мережі Інтернет;
- Клієнт.

У свою чергу серверна частина складається з наступних елементів:

– Блок формування списку оголошень персоналізованої реклами на основі історії дій усіх користувачів у системі.

– Блок визначення точності персоналізованої реклами на основі статистики реакцій користувачів на рекламу, отриману від клієнтських додатків.

Кафедра_КБІЗ_2022рік

– База даних сервера.

А клієнтська частина складається з наступних елементів:

– Отримання списку оголошень персоналізованої реклами.

– Показ користувачу персоналізованої реклами.

– Збір статистики про реакцію користувача на персоналізовану рекламу.

При роботі системи між клієнтами та сервером постійно відбувається обмін інформацією у такому форматі:

1. «Клієнт серверу»: передача історії дій користувача.

2. «Сервер клієнту»: передача списку оголошень персоналізова-ної реклами.

3. «Клієнт серверу»: передача статистики про реакцію користувача на персоналізовану рекламу.

3.3 Розробка функціональної схеми

Функціональна схема системи наведена на рис. 3.3.

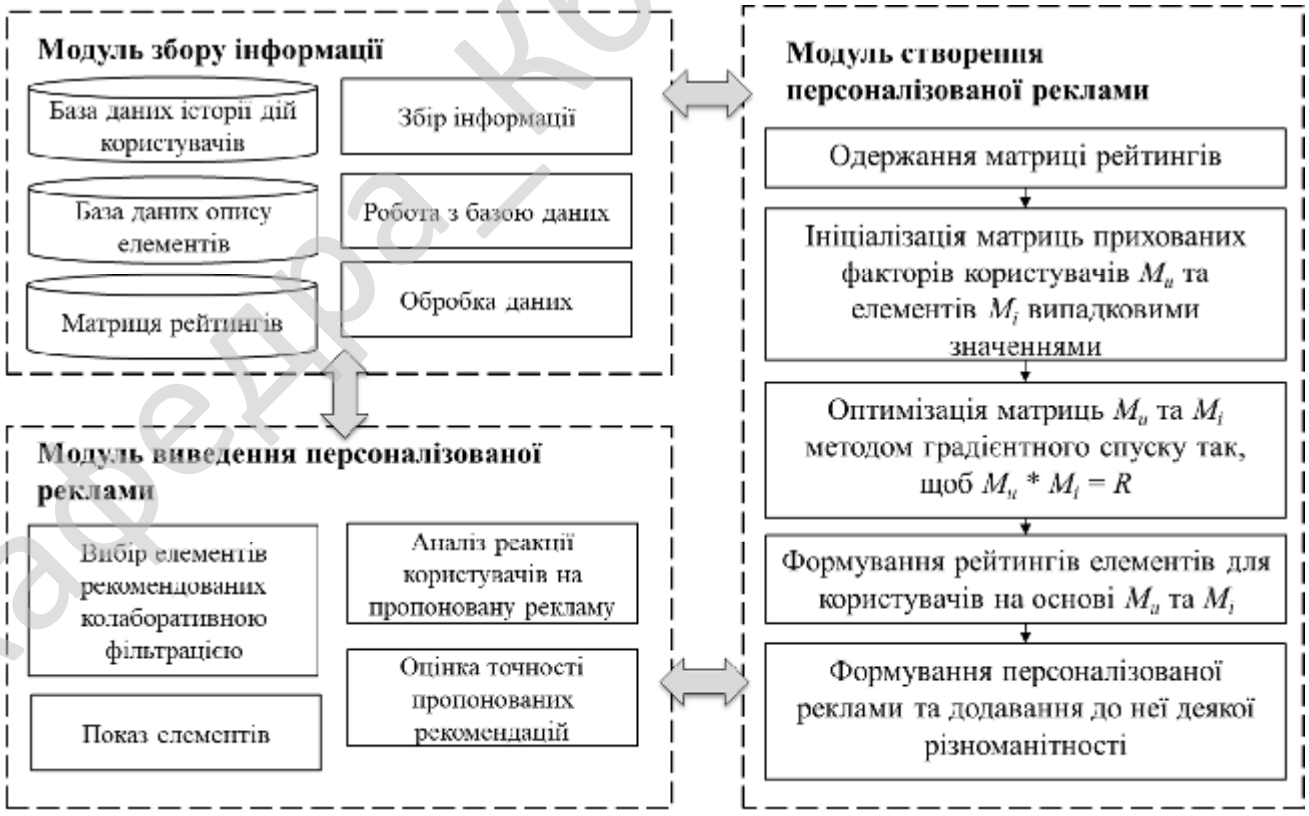

Рисунок 3.3 – Функціональна схема системи

Кафедра_КбПЗ_2022рік

Система складається з наступних модулів:

– Модуль збору інформації.

– Модуль виведення персоналізованої реклами.

– Модуль створення персоналізованої реклами.

У модулі збору інформації містяться наступні елементи:

– База даних історії дій користувачів.

– База даних опису елементів.

– База даних з матрицею рейтингів.

– Збір інформації.

– Робота з базою даних.

– Обробка даних.

У модулі виведення персоналізованої реклами містяться наступні елементи:

– Вибір елементів рекомендованих колаборативною фільтрацією.

– Показ елементів.

– Аналіз реакції користувачів на пропоновану рекламу.

– Оцінка точності пропонованих рекомендацій.

У модулі створення персоналізованої реклами виконуються наступні дії:

– Одержання матриці рейтингів.

– Ініціалізація матриць прихованих факторів користувачів M_u та елементів M_i випадковими значеннями.

– Оптимізація матриць M_u та M_i методом градієнтного спуску так, щоб $M_u * M_i = R$.

– Формування рейтингів елементів для користувачів на основі M_u та M_i.

– Формування персоналізованої реклами та додавання до неї деякої різноманітності.

Кафедра_КПЗ_2022рік

3.4 Розробка діаграми процесів

На рисунку 3.4 зображена діаграма процесів, що описує наявні у системі процеси та їх взаємодію.

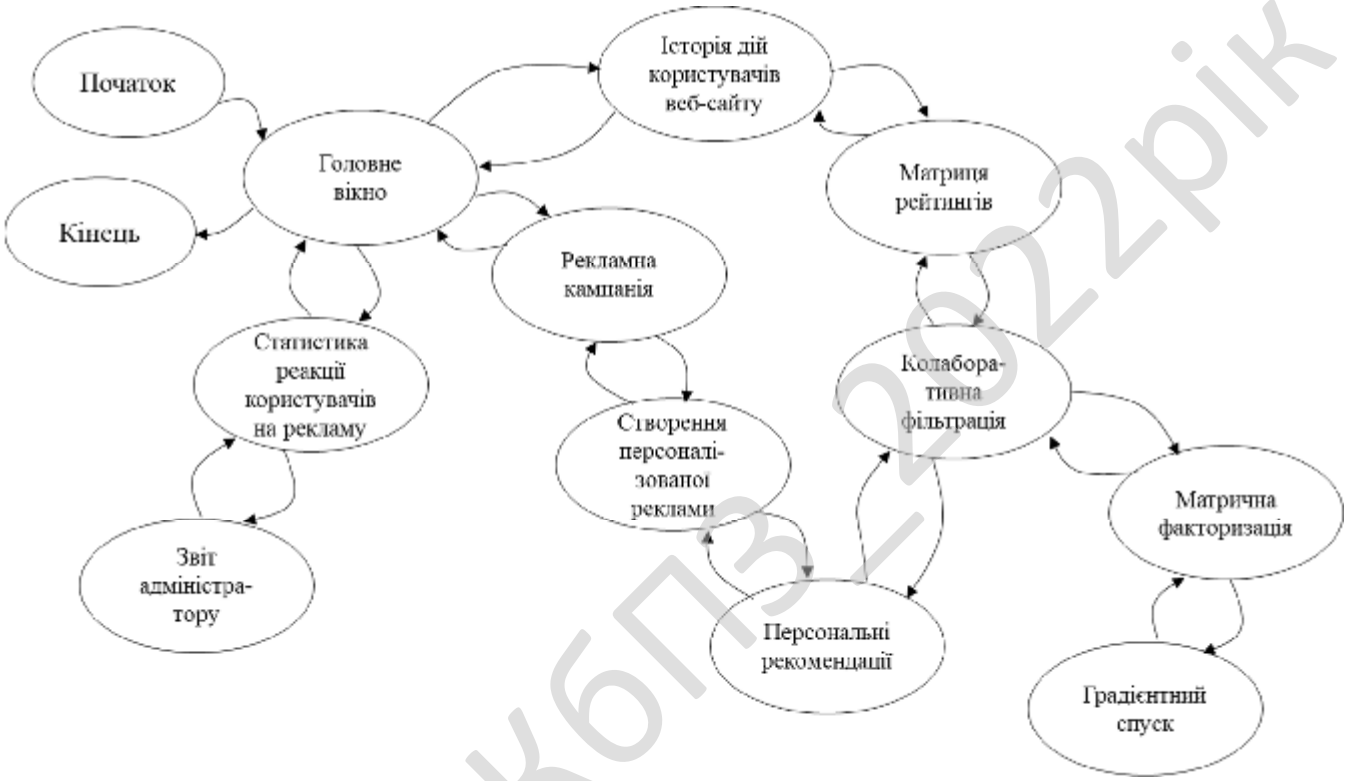

Рисунок 3.4 – Діаграма процесів системи

У розробленій системі наявні наступні основні процеси:

– Головне вікно.

– Історія дій користувачів веб-сайту.

– Матриця рейтингів.

– Рекламна кампанія.

– Створення персоналізованої реклами.

– Персональні рекомендації.

– Колаборативна фільтрація.

– Матрична факторизація.

– Градієнтний спуск.

Кафедра_КбПЗ_2022рік

– Статистика реакції користувачів на рекламу.

– Звіт адміністратору.

Перший процес у системі – Головне вікно програми, з нього можна перейти до найважливіших процесів системи: Історія дій користувачів веб-сайту, Рекламна кампанія, Статистика реакції користувачів на рекламу. Всі інші процеси забезпечують їх виконання.

Кафедра_КбПЗ_2022рік

4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ПРАВИЛЬНІСТЬ ПРОЕКТНИХ РІШЕНЬ

4.1 Блок-схеми та опис алгоритмів функціонування системи

Принцип роботи основної програми наведено на блок-схемі з рисунку 4.1.

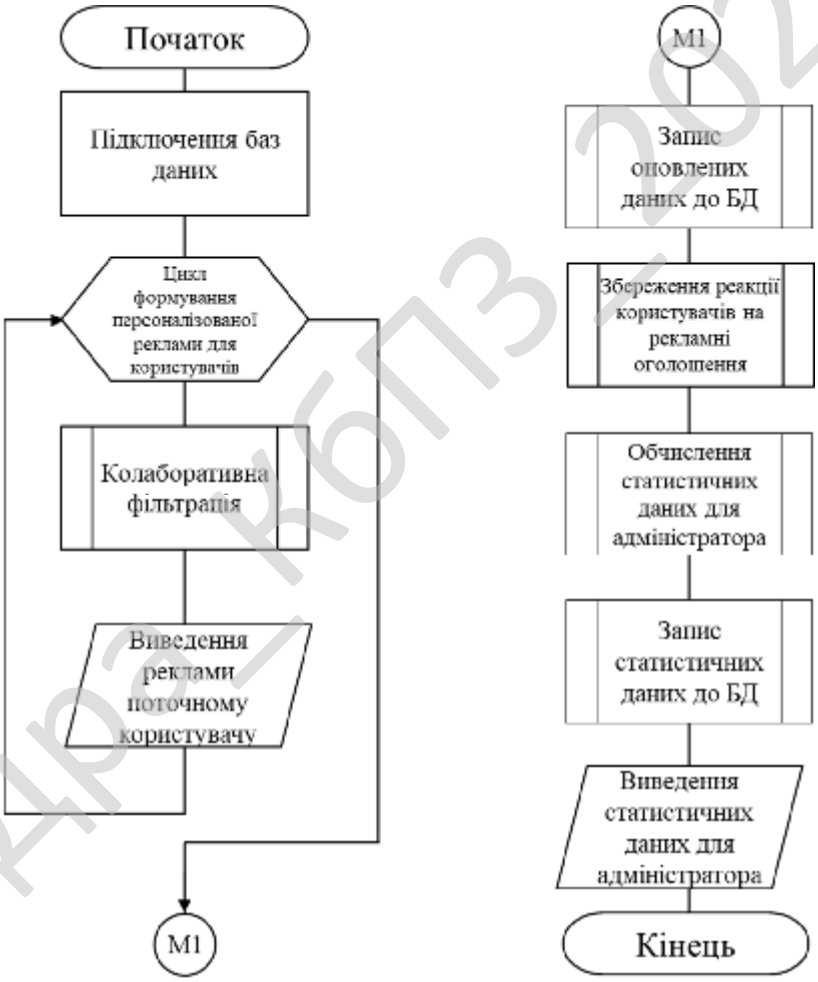

Рисунок 4.1 – Блок-схема основної програми

Як видно з рисунку алгоритм основної програми складається з наступних етапів:

Етап 1. Підключення баз даних.

Кафедра КбПЗ 2022рік

Етап 2. Цикл формування персоналізованої реклами для кожного з користувачів, у якому відбувається запуск колаборативної фільтрації та виведення реклами поточному користувачу.

Етап 3. Запис оновлених даних до БД.

Етап 4. Збереження реакції користувачів на рекламні оголошення.

Етап 5. Обчислення статистичних даних для адміністратора.

Етап 6. Запис статистичних даних до БД.

Етап 7. Виведення статистичних даних для адміністратора.

На рисунку 4.2 показана блок-схема підпрограми колаборативної фільтрації.

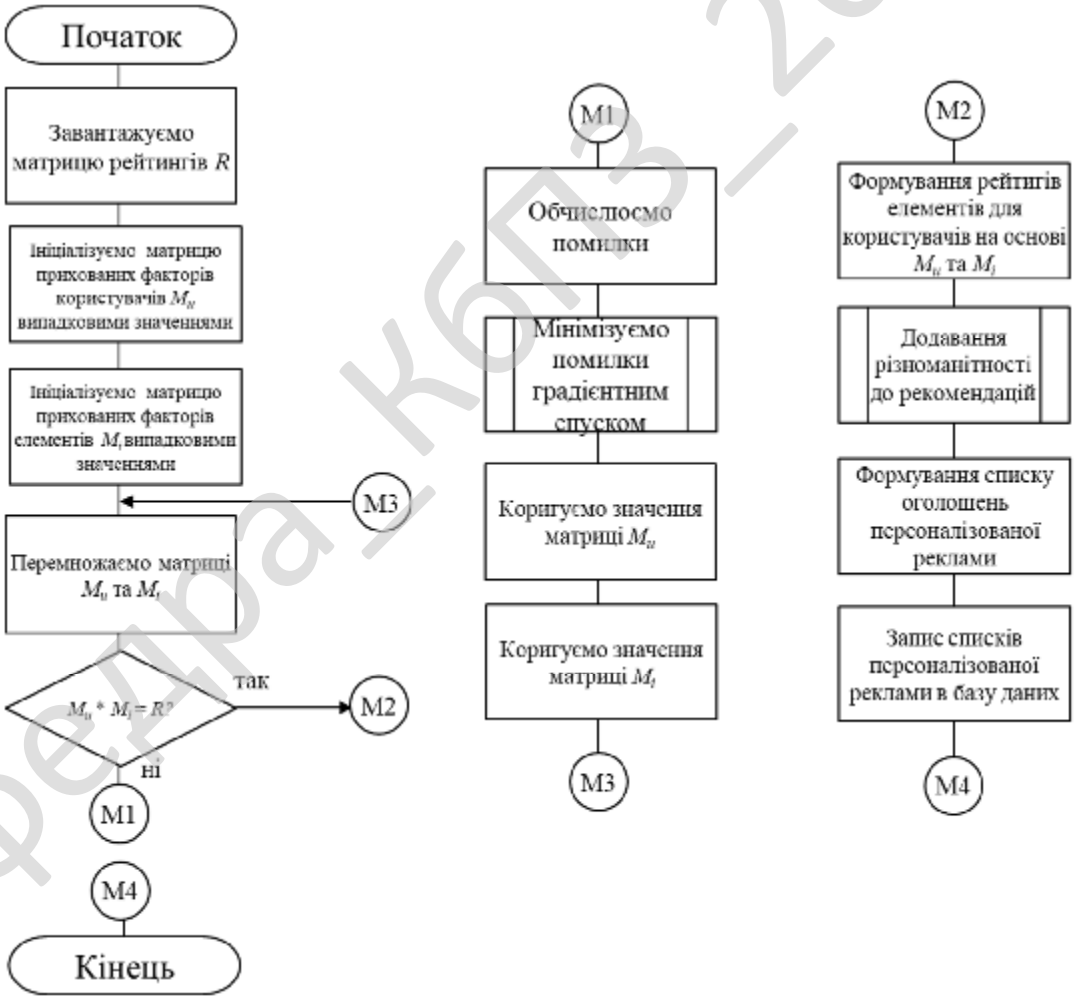

Рисунок 4.2 – Блок-схема підпрограми колаборативної фільтрації

Кафедра_КбПЗ_2022рік

Як видно з рисунку алгоритм колаборативної фільтрації складається з наступних етапів:

Етап 1. Завантажуємо матрицю рейтингів *R*.

Етап 2. Ініціалізуємо матрицю прихованих факторів користувачів M_u випадковими значеннями.

Етап 3. Ініціалізуємо матрицю прихованих факторів елементів M_i випадковими значеннями.

Етап 4. Перемножаємо матриці M_u та M_i.

Етап 5. Перевірка чи $M_u * M_i = R$?

Етап 6. Поки $M_u * M_i$ не дорівнює R так, то обчислюємо помилки, Мінімізуємо помилки градієнтним спуском Коригуємо значення матриці M_u та M_i.

Етап 7. Якщо $M_u * M_i$ дорівнює R, то Формування рейтигів елементів для користувачів на основі M_u та M_i.

Етап 8. Додавання різноманітності до рекомендацій.

Етап 9. Формування списку оголошень персоналізованої реклами.

Етап 10. Запис списків персоналізованої реклами в базу даних.

У вік інформаційного вибуху такі методи створення персоналізованих рекомендацій, як колаборативна фільтрація, дуже корисні, оскільки кількість об'єктів навіть в одній категорії (такі як фільми, музика, книги, новини, веб-сайти) стало настільки багато, що окрема людина не здатна переглянути їх всі, щоб вибрати відповідні.

Колаборативна фільтрація у розробленому програмному забезпеченні реалізована наступною функцією:

```
def AlgSVD(IdUsers, IdItems, RatingMatrix):

    MinRating = 0.5
    MaxRating = 5

    def ForecastingRating(BiasUser,
                          BiasItem,
                          VectorHiddenUserFactors,
                          VectorHiddenItemFactors,
                          GlobalAverageRating):

        Prediction = GlobalAverageRating + BiasUser + BiasItem
        Prediction += numpy.dot(VectorHiddenUserFactors,
VectorHiddenItemFactors)
```

Кафедра_КБПЗ_2022рік

```
        return Prediction


    def RMSECalc(list1, list2):
        n = len(list1)
        rmse = 0
        for i in range(n):
            rmse += math.pow((list1[i] - list2[i]), 2)


        rmse = rmse / n
        rmse = math.sqrt(rmse)

        return rmse


    def TrainingSystem(RatingMatrix,
                       NumberOfHiddenFactors = 70,
                        LearningRate = 0.005,
                        RegularizationFactor = 0.02,
                        NumberOfEpochsOfTraining = 100,
                        StopCondition = 0.001):

        # Ініціалізація

        NumUsers = len(RatingMatrix)
        NumItems = len(RatingMatrix[0])

        k = 0
        GlobalAverageRating = 0

        for user in range(len(RatingMatrix)):
            for item in range(len(RatingMatrix[0])):
                if RatingMatrix[user][item] != 0:
                    k += 1
                    GlobalAverageRating += RatingMatrix[user][item]

        GlobalAverageRating = GlobalAverageRating / k

        print("GlobalAverageRating", GlobalAverageRating)

        UserBiasVector = [0.01] * NumUsers
        ItemBiasVector= [0.01] * NumItems
        for ii in range(NumUsers):
            UserBiasVector[ii] * ii
        for ii in range(NumItems):
            ItemBiasVector[ii] * ii


# зміщення об’єктів
        for item in range(len(RatingMatrix[0])):
            sumDRi = 0
            ki = 0
            for user in range(len(RatingMatrix)):
                if RatingMatrix[user][item] != 0:
                    ki += 1
                    sumDRi += RatingMatrix[user][item] - GlobalAverageRating
            if ki != 0:
                ItemBiasVector[item] = (sumDRi / ki ) # * 1.5

# зміщення користувачів
```

Кафедра_КБПЗ_2022рік

```
        for user in range(len(RatingMatrix)):
            sumDRu = 0
            ku = 0
            for item in range(len(RatingMatrix[0])):
                if RatingMatrix[user][item] != 0:
                    ku += 1
                    sumDRu += RatingMatrix[user][item] - GlobalAverageRating -
ItemBiasVector[item]
            if ku != 0:
                UserBiasVector[user] = (sumDRu / ku) # * 1.5



        MatrixHiddenFactorsUsers = numpy.random.normal(0, 0.1,
                                                                    (NumUsers,

NumberOfHiddenFactors))

        MatrixHiddenFactorsItems = numpy.random.normal(0, 0.1,
                                                                   (NumItems,

NumberOfHiddenFactors))


        RMSEDic = []



        Set1 = [i for i in range(len(RatingMatrix))]
        random.shuffle(Set1)


        for epoch in range(NumberOfEpochsOfTraining):
            errList = []
            ratingList = []
            predList = []

            for user in Set1:




                Set2 = [i for i in range(len(RatingMatrix[0]))]
                random.shuffle(Set2)

                for item in Set2:
                    if True:
                        BiasUser = UserBiasVector[user]
                        BiasItem = ItemBiasVector[item]
                        VectorHiddenUserFactors =
MatrixHiddenFactorsUsers[user]
                        VectorHiddenItemFactors =
MatrixHiddenFactorsItems[item]

                        # прогнозування вподобань
                        Prediction = ForecastingRating(BiasUser,
                                                                BiasItem,

VectorHiddenUserFactors,

VectorHiddenItemFactors,
                                                               GlobalAverageRating)
```

Кафедра_КбПЗ_2022рік

```
                        # помилка прогнозування
                        ErrorPrediction = RatingMatrix[user][item] -
Prediction

                        errList.append(ErrorPrediction)

                        predList.append(Prediction)
                        ratingList.append(RatingMatrix[user][item])


                        # оновлення прихованих факторів
                        for factor in range(NumberOfHiddenFactors):
                            MSFP_f = MatrixHiddenFactorsUsers[user][factor]
                            MSFPr_f = MatrixHiddenFactorsItems[item][factor]


                            MatrixHiddenFactorsUsers[user][factor] +=
LearningRate * (ErrorPrediction * MSFPr_f - RegularizationFactor * MSFP_f)
                            MatrixHiddenFactorsItems[item][factor] +=
LearningRate * (ErrorPrediction * MSFP_f - RegularizationFactor * MSFPr_f)
                            GSP = UserBiasVector[user]
                            GSPr = ItemBiasVector[item]
                            UserBiasVector[user] += LearningRate *
(ErrorPrediction - RegularizationFactor * GSP)
                            ItemBiasVector[item] += LearningRate *
(ErrorPrediction - RegularizationFactor * GSPr)

            RMSEDic.append( RMSECalc(predList, ratingList))

        print(RMSEDic)
        print(RMSECalc(predList, ratingList) )



        return (UserBiasVector,
                ItemBiasVector,
                MatrixHiddenFactorsUsers,
                MatrixHiddenFactorsItems,
                GlobalAverageRating,
                NumUsers,
                NumItems,
                RMSEDic)


    (UserBiasVector,
     ItemBiasVector,
     MatrixHiddenFactorsUsers,
     MatrixHiddenFactorsItems,
     GlobalAverageRating,
     NumUsers,
     NumItems,
     listRMSE) = TrainingSystem(RatingMatrix,
                                                NumberOfHiddenFactors = 70,
                                                 LearningRate = 0.005,
                                                 RegularizationFactor = 0.02,
                                                 NumberOfEpochsOfTraining =
100,
                                                 StopCondition = 0.0001)
```

Кафедра КОТЗ 202 рік

```
MatrixPredictions = []
for user in range(NumUsers):
    MatrixPredictions.append([0] * NumItems)


for user in range(NumUsers):
    for item in range(NumItems):
        if RatingMatrix[user][item] == 0:
            BiasUser = UserBiasVector[user]
            BiasItem = ItemBiasVector[item]
            VectorHiddenUserFactors = MatrixHiddenFactorsUsers[user]
            VectorHiddenItemFactors = MatrixHiddenFactorsItems[item]

            # prediction рейтинга
            Prediction = ForecastingRating(BiasUser,
                                           BiasItem,
                                           VectorHiddenUserFactors,
                                           VectorHiddenItemFactors,
                                           GlobalAverageRating)

            MatrixPredictions[user][item] = Prediction

return (MatrixPredictions,
        listRMSE,
        MatrixHiddenFactorsUsers,
        MatrixHiddenFactorsItems,
        UserBiasVector,
        ItemBiasVector)
```

4.2 Захист розробленого програмного забезпечення

Для захисту програмного забезпечення від несанкціонованого використання та поширення існує багато різних технологій. Оскільки сирцевий код на Python не компілюється, а інтерпретується, то найлегшим підходом до захисту відкритого коду розроблених бібліотек є заплутування коду. Найчастіше для цього використовується обфускація. Але існують і більш просунуті технології заплутування коду. Тож, у даній роботі пропонується використати псевдокод та поліморфні технології.

Будь-яка обфускація лише на рівні машинного коду зменшує швидкість виконання програми. Адже в кращому випадку для заплутування коду доводиться додавати туди непотрібні інструкції (втім, що не впливають на результат). Для більш просунутої захисту використовується технологія так званого псевдокоду. Для звичайного x86-процесора розробляється певна псевдомова, що оперує зовсім іншими командами та логічними конструкціями, після чого вихідний код

Кафедра_КбПЗ_2022рік

захищеного додатку інструкція за інструкцією перетворюється на псевдокод. Для виконання цього псевдокоду генерується "віртуальна машина", яка на льоту перетворює команди псевдокоду на машинні інструкції та виконує їх. Результат залишається той самий, але код програми змінюється до невпізнання, і "відреверсувати" шукані алгоритми захищеної програми можна лише після розкриття всієї логіки псевдокоду.

Для підвищення стійкості псевдокоду можуть використовуватися поліформні технології. Це означає, що псевдокод щоразу новий та віртуальна машина для його виконання теж генерується миттєво на льоту.

Ці технології самі по собі не захищають програму від нелегального копіювання, але дозволяють надійно приховати алгоритми та ноу-хау розробника, а також інші важливі місця програми (на зразок того, де здійснюється обмін з електронним ключем або процедура проведення та перевірки онлайн-активації).

Для роботи з файлами наборів даних розроблено наступні функції:

```
        def SelectNUsersFromTable(sDirData, NumUsers, NumSeries, FileNameFrom,
Scope = 500):


            n = NumUsers
            nn = NumSeries
            p = []*n
            l = []
            x = 0
            for i in range(n):
                r = x + random.randint(1, Scope)
                p.append(r)
                x = r

            FileToData = open(sDirData + "ratings"+str(n)+"_"+str(nn)+".csv", "w",
newline="", encoding='utf-8-sig')
            FileToListID = open(sDirData +
"ratings"+str(n)+"_users"+str(nn)+".csv", "w", newline="", encoding='utf-8-sig')


            with open(FileNameFrom + "ratings.csv", "r", newline="", encoding='utf-
8-sig') as FileFrom:
                reader1 = csv.reader(FileFrom, delimiter=',')

                i = p[0]
                j = 0

                next(reader1)
                for row in reader1:

                    i = int(row[0])
```

Кафедра_КбПЗ_2022рік

```
                    if (i != j) and (i in p):


                        l.append(row[0])

                    if i > p[n-1]:
                        break

                    if i in p:
                        FileToData.write(row[0] + ';' + row[1] + ';' + row[2] + ';'
+ row[3] + '\n')

                    j = i


            FileToListID.write(str(l))
            FileToListID.write(';' + str(len(l)))



            FileToData.close()
            FileToListID.close()
            FileFrom.close()

            print("З датасету вибрано користувачів: ", len(l))

            return l

        def SelectNUsersFromTablePlus(sDirData, NumUsers, NumSeries, FileNameFrom):


            n = NumUsers
            nn = NumSeries
            p = []*n
            l = []
            lm = []
            x = 0
            for i in range(n):
                r = x + random.randint(1, 500)
                p.append(r)
                x = r

            FileToData = open(sDirData + "ratings"+str(n)+"_"+str(nn)+".csv", "w",
newline="", encoding='utf-8-sig')
            FileToListID = open(sDirData +
"ratings"+str(n)+"_users"+str(nn)+".csv", "w", newline="", encoding='utf-8-sig')


            with open(sDirData + "ratings.csv", "r", newline="", encoding='utf-8-
sig') as FileFrom:
                reader1 = csv.reader(FileFrom, delimiter=',')

                i = p[0]
                j = 0

                next(reader1)
                for row in reader1:

                    i = int(row[0])

                    if (i != j) and (i in p):
```

Кафедра_КбПЗ_2022рік

```
                    l.append(row[0])

                if i > p[n-1]:
                    break

                if i in p:
                    FileToData.write(row[0] + ';' + row[1] + ';' + row[2] + ';'
+ row[3] + '\n')

                    if row[1] not in lm:
                        lm.append(row[1])

                j = i


        FileToListID.write(str(l))
        FileToListID.write(';' + str(len(l)))

        FileFrom.close()
        with open(sDirData + "ratings.csv", "r", newline="", encoding='utf-8-
sig') as FileFrom:
            reader1 = csv.reader(FileFrom, delimiter=',')
            next(reader1)
            for row in reader1:
                x = random.randint(0,100)
                if (row[1] in lm) and (row[0] not in l) and x <= 2:
                    FileToData.write(row[0] + ';' + row[1] + ';' + row[2] + ';'
+ row[3] + '\n')



        FileToData.close()
        FileToListID.close()
        FileFrom.close()

        print("З датасету вибрано користувачів: ", len(l))

        return l

    def SelectNUsersFromTableConsistently(sDirData, NumUsers, NumSeries,
FileNameFrom):


        n = NumUsers
        nn = NumSeries
        p = []*n
        l = []
        x = 0
        for i in range(n):
            r = i
            p.append(r)
            x = r

        FileToData = open(sDirData + "ratings"+str(n)+"_"+str(nn)+".csv", "w",
newline="", encoding='utf-8-sig')
        FileToListID = open(sDirData +
"ratings"+str(n)+"_users"+str(nn)+".csv", "w", newline="", encoding='utf-8-sig')


        with open(sDirData + "ratings.csv", "r", newline="", encoding='utf-8-
sig') as FileFrom:
            reader1 = csv.reader(FileFrom, delimiter=',')
```

Кафедра_КбПЗ_2022рік

```
                i = p[0]
                j = 0

                next(reader1)
                for row in reader1:

                    i = int(row[0])

                    if (i != j) and (i in p):


                        l.append(row[0])

                    if i > p[n-1]:
                        break

                    if i in p:
                        FileToData.write(row[0] + ';' + row[1] + ';' + row[2] + ';'
+ row[3] + '\n')

                    j = i


            FileToListID.write(str(l))
            FileToListID.write(';' + str(len(l)))



            FileToData.close()
            FileToListID.close()
            FileFrom.close()

            print("З датасету вибрано користувачів: ", len(l))

            return l



        def SelectNUsersFromTableВДиапазоне(sDirData, First, Last, NumSeries,
FileNameFrom):


            n = Last - First + 1
            nn = NumSeries
            p = []*n
            l = []
            x = 0
            for i in range(First, Last+1):
                r = i
                p.append(r)
                x = r

            FileToData = open(sDirData + "ratings"+str(n)+"_"+str(nn)+".csv", "w",
newline="", encoding='utf-8-sig')
            FileToListID = open(sDirData +
"ratings"+str(n)+"_users"+str(nn)+".csv", "w", newline="", encoding='utf-8-sig')


            with open(FileNameFrom + "ratings.csv", "r", newline="", encoding='utf-
8-sig') as FileFrom:
                reader1 = csv.reader(FileFrom, delimiter=',')

                i = p[0]
```

Кафедра_КбПЗ_2022рік

```
                j = 0

                next(reader1)
                for row in reader1:

                    i = int(row[0])

                    if (i != j) and (i in p):


                        l.append(row[0])

                    if i > p[n-1]:
                        break

                    if i in p:
                        FileToData.write(row[0] + ';' + row[1] + ';' + row[2] + ';'
+ row[3] + '\n')

                    j = i


            FileToListID.write(str(l))
            FileToListID.write(';' + str(len(l)))



            FileToData.close()
            FileToListID.close()
            FileFrom.close()

            print("З датасету вибрано користувачів: ", len(l))

            return l



        def ReadyListOfUsers(sDirData, NumUsers, NumSeries):


            n = NumUsers
            nn = NumSeries
            l = [0]


            FileToListID = open(sDirData +
"ratings"+str(n)+"_users"+str(nn)+".csv", "r", newline="", encoding='utf-8-sig')

            reader1 = csv.reader(FileToListID, delimiter=';')

            for row in reader1:
                l = row[0]

            FileToListID.close()

            if "#" in l:
                l = re.findall(r'(\#\w\d+)\,*', l)
            else:
                l = re.findall(r'(\d+)\,*', l)

            return l

        def ReadyMovieList(NumMovies, NumSeries):
```

Кафедра_КбПЗ_2022рік

```
            n = NumMovies
            nn = NumSeries
            l = [0]


            FileListId = open(sDirData +
"movies/movies"+str(n)+"_id"+str(nn)+".csv", "r", newline="", encoding='utf-8-
sig')

            reader1 = csv.reader(FileListId, delimiter=';')

            for row in reader1:
                l = row[0]

            FileListId.close()

            l = re.findall(r'(\d+)\,*', l)

            return l


        def SelectNMoviesFromTable(NumMovies, NumSeries, Directory, FileName):

            n = NumMovies
            nn = NumSeries
            p = []*n
            l = []
            x = 0

            for i in range(n):
                r = x + random.randint(1, 100)
                p.append(r)
                x = r


            FileToData = open(Directory + "movies" + str(n)+"_"+str(nn)+".csv",
"w", newline="", encoding='utf-8-sig')
            FileToListID = open(Directory + "movies" + str(n)+"_id"+str(nn)+".csv",
"w", newline="", encoding='utf-8-sig')


            with open(Directory + FileName, "r", newline="", encoding="ISO-8859-1")
as FileFrom:
                reader1 = csv.reader(FileFrom, delimiter=';')

                t = 0


                next(reader1) # пропустити заголовок таблиці
                for row in reader1:

                    if t in p:
                        try:

                            FileToData.write(row[0] + ';' + row[1] + ';' + row[2] +
';' + row[3] + '\n')
                            l.append(row[0])

                        except:
                            print("помилка")
```

КафедраКбПЗ_2022рік

```
            t += 1


        FileToListID.write(str(l))
        FileToListID.write(';' + str(len(l)))



        FileToData.close()
        FileToListID.close()
        FileFrom.close()

        print("З датасету вибрано фільмів: ", len(l))

        return l
```

Кафедра_КбПЗ_2022рік

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

Для запуску програми слід підключити всі бібліотеки, розміщені у пакеті RecommendationSystem, зокрема, Graph, Table та Time, та запустити файл Main.py.

Розроблене програмне забезпечення на вхід отримує історію дій користувачів у файлах формату .csv, що містять наступні стовпчики:

– Ідентифікатор користувача;

– Ідентифікатор елемента;

– Оцінка користувача елементу;

– Часова мітка.

На виході система формує для кожного користувача ранжований список елементів для списку оголошень персональної реклами.

Режими роботи розробленої системи:

– Навчання колаборативної фільтрації на історії дій користувачів.

– Формування списків персоналізованої реклами.

– Тестування точності визначення вподобань користувачів.

Програмне забезпечення не має візуального інтерфейсу та виконується у командному рядку. Так як пропонується застосовувати його як бібліотеку для розробки скриптів веб-сайтів.

Рекомендується використовувати розроблену систему для створення персоналізованої реклами на веб-сайтах, де користувачі можуть виставляти контенту оцінки та реєструються і мають свої акаунти.

Перш ніж починати застосування системи, треба провести її навчання на попередньо зібраних даних з веб-сайту, де вона в подальшому буде використовуватися.

Режим тестування точності роботи системи може використовуватися для перевірки того, наскільки якісно пройшов процес навчання. Якщо результати тестування незадовільні – треба знову навчати систему на більшій кількості даних.

Кафедра_КоПЗ_2022рік

6 НАУКОВА НОВИЗНА

У магістерській роботі розроблено програмне забезпечення, яке призначено для системи створення персоналізованої реклами для користувачів веб-сайту.

Метою роботи є дослідження та програмна реалізація системи створення персоналізованої реклами для користувачів веб-сайту.

Для досягнення поставленої мети визначена програма дослідження, що складається з наступних завдань:

– Дослідження існуючих систем створення персоналізованої реклами для користувачів веб-сайту.

– Розробка методів та алгоритмів системи створення персоналізованої реклами для користувачів веб-сайту.

– Програмна реалізація системи створення персоналізованої реклами для користувачів веб-сайту.

Об'єктом дослідження є процес створення персоналізованої реклами для користувачів веб-сайту.

Предметом дослідження є методи та алгоритми колаборативної фільтрації даних для створення персоналізованої реклами.

Методи дослідження базуються на методах аналізу даних, методах математичної статистики та методах розробки програмного забезпечення.

Наукова новизна отриманих результатів. У процесі рішення завдань, обумовлених цілями дослідження, отримані наступні результати:

1. Удосконалено метод колаборативної фільтрації даних на основі матричної факторизації, який відрізняється від відомих запропонованим методом захисту від «бульбашки фільтрів».

2. Розроблено вітчизняний продукт генерації персоналізованої реклами для користувачів веб-сайту, який має більш широкі можливості, на відміну від існуючих аналогів та може бути використаний на різних веб-ресурсах.

Кафедра_КБПЗ_2022рік

7 ДАНІ ПРО ЕКОНОМІЧНУ ЕФЕКТИВНІСТЬ РОЗРОБЛЕНОЇ ПРОГРАМИ

Техніко-економічне обґрунтування теми магістерської роботи

Після ознайомлення з підприємством та засобами розробки програмної продукції був розроблений план розробки програми. Був підрахований необхідний час для розробки та впровадження програми. Цей час склав 60 днів (три місяці).

В магістерській роботі було проведене дослідження та виконана програмна реалізація системи створення персоналізованої реклами для користувачів веб-сайту.

Розроблене програмне забезпечення має достатню надійність і задовольняє усім поставленим умовам, а саме:

а) невеликий розмір;

б) невеликі системні потреби;

в) незалежність від встановлених на комп'ютері баз даних;

г) зручність у користуванні та надійність

Таблиця 7.1 - Початкові данні

Показники	Позначення	Характеристика або величина
1	2	3
1. Кількість розроблених програм період, шт	N	1
2. Кількість екземплярів програм, шт	Ne	40
3. Запланований термін розробки, днів	Fpq	60 (3 місяці)
4. Група задачі підсистеми управління (1-6)	–	1
5. Ступінь новизни задачі (А, Б, В, Г)	–	Б
6. Складність алгоритму (1, 2, 3)	–	2
7. Кількість макетів вхідної інформації	–	3

Кафедра_КОПЗ_2022рік

Продовження табл. 7.1

1	2	3
8. Кількість форм вихідної інформації.	–	4
9. Мова програмування (1-6)	–	2
10. Попередній досвід (1-6)	–	3
11. Гнучкість проекту ПП (1-6)	–	3
12. Детальність проекту ПП (1-6)	_	2
13. Рівень спрацьованості колективу (1-6)	_	2
14. Ступінь вимірності процесів (1-6)	_	3
15. Необхідна надійність програмного забезпечення (1-6)	_	2
16. Розмір бази даних (порівняно з розміром програми) (1-6)	_	2
17. Складність кінцевого програмного продукту (1-6)	_	2
18. Необхідний рівень забезпечення повторного використання (1-6)	_	2
19. Документованість відповідно до планованого життєвого циклу (1-6)	_	2
20. Вимоги до швидкодії ПП (1-6)	_	2
21. Обмеження на розміри основного сховища даних (1-6)	_	2
22. Різноманітність використовуваних обчислювальних платформ (1-6)	_	2
23. Професійний рівень аналітиків (1-6)	_	2
24. Професійний рівень програмістів (1-6)	_	2
25. Постійність складу команди розробників (1-6)	_	2
26. Досвід розробки додатків (1-6)	_	2
27. Досвід роботи з обчислювальною платформою (1-6)	_	2

Кафедра КбПЗ 2022рік

Продовження табл. 7.1

1	2	3
28. Досвід роботи з мовою і інструментами середовища розробки (1-6)	–	2
29. Досвід роботи з програмними інструментами розробки (1-6)	–	3
30. Розробка ПО для декількох серверів одночасно (1-6)	–	2
31. Вимоги до дотримання встановленого графіка робіт (1-6)	–	2
32. Вартість ПЗ у розробника (НМА), грн	–	40000
33. Норматив додаткової зарплати, % :	$Н_д$	10
34. Норматив відрахувань у соціальні фонди, %	Нс	22
35. Норматив загальногосподарських витрат, %	Нг	15
36. Норматив витрат на освоєння нових мов програмування, %	Нп	15
37. Рівень рентабельності програмної продукції, %	Ре	55
38. Ставка податку на додану вартість, %	Ндв	20

7.2 Розрахунок трудомісткості розробки програмної продукції

Значення трудомісткості розробки програмного забезпечення для стадій ТЗ, ЕК, ТП та ВП визначаємо по типовим нормам часу приведеним в додатках МВ. Стадія РП є найбільш тривалою і трудомісткою, що робить значний вплив на інші стадії проекту.

Визначимо трудомісткість розробки ПЗ для стадії РП.

Обчислюємо номінальні трудовитрати, люд-міс.:

$$T_{ном} = A \, \text{Size}^B , \tag{7.1}$$

де A - коефіцієнт Боема, A=2,45; Size - загальний об'єм відлагодженого програмного коду, тис. рядків; B - показник ступеня, що визначається співвідношенням:

$$B = 1{,}01 + 0{,}001 \sum W_i \tag{7.2}$$

де Wi - сумарне значення п'яти показників (МВ, додаток 2), що відображають особливості розробки проекту програмного продукту (ПП) і колективу розробників.

Кафедра_КбПЗ_2022рік

$$B=1,01+0,001(2,43+3,64+3,38+3,95+2,73) = 1,026$$

$$T_{ном} = 2,45 \cdot 2,7^{1,026} = 6,78 \text{ люд-міс.}$$

Визначаємо уточнені (з урахуванням приведених в МВ додатку 3 сімнадцяти додаткових коефіцієнтів) трудовитрати, люд-міс.:

$$T_{уточн}=T_{ном}\, ПV_j, \qquad (7.3)$$

де *ПVj* - добуток сімнадцяти додаткових коефіцієнтів, приведених в МВ додатку 3.

$$T_{уточн}= 6,78 \cdot (0,88 \cdot 0,93 \cdot 0,88 \cdot 0,91 \cdot 0,95 \cdot 1 \cdot 1 \cdot 0,87 \cdot 1,22 \cdot 1,16 \cdot 1,1 \cdot 1,1 \cdot$$
$$\cdot 1,12 \cdot 1,1 \cdot 1 \cdot 1,1 \cdot 1,1) = 9,37 \text{ люд-міс.}$$

Ці коефіцієнти дозволяють диференційовано оцінювати результати роботи програмістів, беручи до уваги швидкодію програми, використання різноманітних обчислювальних платформ і інструментів розробки, взаємодію декількох серверів, вимоги до об'ємів баз даних і ін.

Визначаємо підсумкові трудовитрати по стадії робочий проект, люд-дні:

$$T_{РП} = 0,3 C T_{уточн}^{0,33+0,2(B-1,01)} S, \qquad (7.4)$$

де С - визначений емпірично коефіцієнт, запропонований авторами методики, (МВ, додаток 4); S - коефіцієнт стиснення (або подовження) графіка робіт %, що дозволяє коректувати терміни розробки ПО згідно встановленим вимогам. Вибираємо в межах (25…350)%

$$T_{РП}=0,3 \cdot 2,66 \cdot 9,37^{0,33+0,2(1,026-1,01)} \cdot 85 = 143 \text{ люд/день}$$

Для зручності визначення загальної трудомісткості на розробку програмного забезпечення результати розрахунків по стадіям зводимо до таблиці 7.2.

Таблиця 7.2 - Визначення трудомісткості розробки програмного забезпечення

Стадії розробки	**Трудомісткість за типовими нормами та розрахунками**	
	Величина, люд/дні	Підстава
Технічне завдання	9	Д5
Ескізний проект	10	Д6

Кафедра_КОПЗ_2022рік

Продовження табл. 7.2

Стадії розробки	**Трудомісткість за типовими нормами та розрахунками**	
	Величина, люд/дні	Підстава
Технічний проект	9	Д7
Робочий проект	143	Ф 7.1-7.4
Впровадження	13	Д13
Всього	184	–

7.3 Визначення чисельності виконавців і планового фонду зарплати

Чисельність ставок інженерів-програмістів для розробки програмного забезпечення визначається за формулою

$$Ч = \frac{T_{пз} N}{F_{рq} - H_{ев}}, \quad (7.5)$$

де Fpq - плановий фонд робочого часу одного спеціаліста, днів, Тпз – трудомісткість розробки програмного забезпечення люд-дні,

$$Ч = \frac{184 \cdot 1}{60\text{-}5} = 3,3 \text{ ставки}$$

Чисельність інженерів-електронщиків для проведення технічного обслуговування та ремонту комп'ютерних мереж визначається в залежності від наявності технічних засобів і норм витрат часу на виконання профілактичних робіт на протязі року.

Визначаємо затрати часу на виконання профілактичних робіт по обслуговуванню обладнання за період розробки. Результати розрахунку зводимо до таблиці 7.3

Кафедра_КБПЗ – 2022рік

Таблиця 7.3 - Затрати часу на виконання профілактичних робіт по обслуговуванню обладнання за розрахунковий період

Найменування обладнання	**Профілактичне обслуговування**			
	Кількість хв. на один. обл.	**Кількість облад-нання**	**Затрати часу в хв.**	**Затрати часу в год.**
Системний блок ПК	90	*9*	*810*	*13,5*
Монітор	60	9	*540*	*9*
Клавіатура	30	9	*270*	*4,5*
Маніпулятор «мишка»	30	9	*270*	*4,5*
Принтер матричний	60	*0*	*0*	*0,0*
Принтер лазерний	120	*1*	*120*	*2*
Принтер струминний	60	*1*	*60*	*1*
Сканер	20	*1*	*20*	*0,33*
Концентратор–маршрутизатор	30	*2*	*60*	*1*
Кабельні господарства ЛВС на 1 м. п.	2,5	*100*	*250*	*4,17*
Копіювальний апарат	140	*1*	*140*	*2,33*
Усього за рік:			**$З_ч$**	**42,33**

Час на профілактику обладнання в загальному балансі робочого часу інженерів-електронщиків неповинен складати більше 10%

Виходячи з цього фонд робочого часу інженерів-електронщиків складає:

$$Ф_{др}^{с} = \frac{З_ч \cdot n_{міс}}{1{,}2} \qquad (7.6)$$

Кафедра_КОПЗ_2022рік

$$Ф_{др}^{с} = \frac{42 \cdot 3}{1,2} = 105 \text{ год}$$

Визначаємо необхідну кількість ставок штатного персоналу сектора ТО:

$$Ч_{ел} = \frac{Ф_{др}^{с}}{F_{др} \cdot T_{зм}} \quad (7.7)$$

$Ч_{ел}$=105/(60·8)=0,2 ставки

Для забезпечення нормального технічного обслуговування засобів ТО та мереж, необхідно прийняти найбільше ціле значення розрахункової чисельності інженерів – електронщиків. Чисельність інженерів-системотехніків, адміністраторів мережі, дизайнерів WEB вузлів, системних програмістів (аналітиків), бухгалтерів-економістів визначається за потребою в залежності від функціональних обов'язків. Після визначення чисельності персоналу складається штатний розклад.

Таблиця 7.4 - Розрахунок чисельності штатного персоналу сектору системного та адміністративного обслуговування засобів ОТ та комп'ютерних мереж

Посада	Вид роботи	Час	К-сть штатних одиниць
Адміністратор загальної мережі, аналітик	Адміністрування локальної мережі, поштового та серверу DNS (ОС FreeBSD), маршрутизатора Cisco, серверу доступу АДСЛ (ОС Linux), Wi-Fi налаштування ADSL, VPN, PPPoE, Frame Relay	2	0,5
	Налаштування і конфігурування базової станції безпровідного зв'язку (CMTS)	0,5	
	Розробка та впровадження проектів з організації зв'язку між віддаленими об'єктами, ЛОМ	0,5	
	Забезпечення цілодобової роботи зв'язку клієнтів до мережі Інтернет	1	
Всього		4	

Кафедра_КБПЗ_2022рік

Продовження таблиці 7.4

Посада	Вид роботи	Час	Кіль-кість штатних одиниць
Продакт-менеджер	Презентації нової продукції, пошук каналів збуту	1	0,25
	Підтримка постійних клієнтів	0,5	
	Оформлення договорів, ведення тендерів	0,25	
	Контроль взаєморозрахунків з постачальниками	0,25	
Всього		2	
Дизайнер WEB	Розробка концепції оформлення та інтерфейсу сайту, оптимізація дизайну існуючих, проектує їх структуру та навігацію	1	0,25
	Створення графічних і стилістичних елементів сайту	0,5	
	Оформлення банерів і промо-сторінок	0,25	
	Розміщення графіки і контенту на Інтернет сторінках	0,25	
Всього		2	
Інженер верстальник	Розробка та верстка макетів рекламної продукції та технічної документації	1	0,25
	Верстка друкованих видань	0,5	
	Додрукова підготовка макетів	0,25	
	Розміщення графіки і контенту на Інтернет сторінках	0,25	
Всього		2	

КафедРа_КПЗ_2022рік

Складемо штатний розклад виконавців:

Таблиця 7.5 - Штатний розклад виконавців

Посада	Кількість ставок	Середньомісячний оклад, грн.	Всього за період розробки, грн.
Керівник (IT-менеджер)	1	12310	36930
Продакт-менеджер	0,25	8000	6000
Інженер-програміст	3,3	8600	85140
Інженер-електронщик	0,2	8000	4800
Інженер-системотехнік	0,25	8000	6000
Адміністратор мережі	0,5	8000	12000
Системний програміст	0,25	8000	6000
Дизайнер WEB	0,25	8000	6000
Інженер-верстальник	0,25	8000	6000
Бухгалтер-економіст	0,5	10000	15000
Всього за період розробки	$R_{сп}$=6,75	-	$Ф_{роб}$=183870

Розрахуємо середньоденну зарплату одного виконавця:

$$З_{сд} = \frac{Ф_{роб}}{R_{сп} F_{рд}}, \quad (7.8)$$

де Фроб – загальна сума зарплати за плановий період, грн.

$$З_{сд} = \frac{183870}{6,75 \cdot 60} = 454 \; грн.$$

7.4 Розрахунок капітальних вкладень та амортизаційних відрахувань у розробника

Балансова вартість будівель визначається з урахуванням кількості робочих місць виконавців, питомої площі на одне робоче місце, та вартості одного квадратного метра виробничої площі.

Кафедра_КБПЗ_2022рік

$$Б_{уд} = R^1_{сп} S_у Ц_{пл}, \quad (7.9)$$

де $R^1_{сп}$ – кількість робочих місць виконавців, шт. Приймаємо 8 робочих місць. $S_у$ – питома площа на одне робоче місце, $м^2$; $Ц_{пл}$ – вартість одного квадратного метра площі, грн.

Згідно даних інтернет ресурсу DOM.RIA (https://dom.ria.com) ціна одного квадратного метра площі, вік якої не перевищує 30 років, по місту складає 500…1600 $у.о./м^2$. Враховуючи, що курс складає 1 у.о. = 38 грн. приймаємо для розрахунку вартість одного метра квадратного рівною 20000 грн./м^2. На кожне робоче місце у середньому потрібно 8 $м^2$. З урахуванням цього:

$$Б_{уд} = 8 \cdot 8 \cdot 20000 = 1280000 \text{ грн.}$$

Вартість передавальних пристроїв складає 10% від вартості будівель, і у даному випадку вона складе: 128000 грн.

Балансова вартість інвентарю розраховується за нормою 3500 грн на одне робоче місце. Тобто

$$І_{нв} = R^1_{сп} \cdot Ц_м, \quad (7.10)$$

де $Ц_м$ – ціна меблів для одного робочого місця, грн.

$$І_{нв} = 8 \cdot 3500 = 28000 \text{ грн}$$

Балансова вартість обчислювальної техніки визначається по оптовим цінам постачальника з врахуванням витрат на транспортування.

Специфікація на обчислювальну техніку наведена в таблиці 7.7.

Дані по оптовій ціні на обладнання та комплектуючі вибирались по прайсу Інтернет магазину Компбест за 28.10.22 – джерело https://compbest.com.ua.

Кафедра_КБПЗ_2022рік

Таблиця 7.6 – Специфікація

Найменування комплектуючої або обладнання	Тип	Оптова ціна
Персональний комп’ютер		10947
Системний блок	Fujitsu P720 Tower	7347
Процесор	Intel Core i3-4130 (2 (4) ядра по 3.40 GHz); Cache Memory 3	-
Системна плата	Fujitsu D3221-A1 Intel Haswell с TDP до 95 Вт. classic desktop series, Socket LGA 1150, Intel® Q85, Intel® HD Graphics (depends on processor, DP, DVI,VGA), PCI Express Gen3	-
Відеокарта	AMD Radeon RX 550 4GB GDDR5 Re Dragon PowerColor (AXRX 550 4GBD5-DH)	-
Жорсткий диск	HDD Seagate Barracuda 750 Gb 7200 32Mb ST3750528AS (ST3750528AS)	-
Оперативна пам’ять	DIMM 4096Mb DDR3 PC3-10600 CL9 Transcend JetRam, non-Reg., no-ECC , CL 9 (2 модулі)	-
DVD-привод	DVD -RW/+RW , LG SATA SuperMulti Bulk 22x, SecurDisc, black	-
Корпус	GRESSO GE-7525, 500W (120mm big fan), 2xIDE, full-ATX,БЖ 2xSATA, 1xFDD, Air Duct, 2xUSB 2.0, Mic+Audio, silver/black	-
Кардрідер внутрішній	USB 3.0 Card reader STORM CR-35U1A4-B, int. 3.5", 1*USB3.0+AUDIO+1394, multi: All Type Cards, black	-
інше	Клавіатура, мишка	-

Кафедра_КПТВ_2022рік

Продовження таблиці 7.6

Найменування комплектуючої або обладнання	Тип	Оптова ціна
Монітор	Samsung S22R350FHI (LS22R350FHIXCI) / 21.5" (1920x1080) IPS LED / HDMI, VGA	3600
Принтер лазерний	Canon i-SENSYS LBP6030W	2700
Принтер струменевий	Epson Stylus Photo P50 (C11CA45341) + USB cable	5500
Копіювальний апарат	Canon i-SENSYS MF217W with Wi-Fi	5965

Витрати на транспорт, монтаж та випробування можуть бути прийняті в межах до 10% від оптової ціни.

Для визначення необхідної кількості капітальних вкладень складемо таблицю 7.8.

Таблиця 7.7 - Балансова вартість обчислювальної техніки

Найменування обчислювальної техніки	Кількість,шт.	Ціна за одиницю, грн.	Витрати на транспортування, монтаж та випробовування.	Загальна вартість, грн.
Персональні комп’ютери	8	10947	8757,6	96333,6
Принтер лаз.	2	2700	540	5940
Принтер струм.	1	5500	550	6050
Копіюв. апарат	1	5965	596,5	6561,5
Всього	–	–	–	114885,1

Кафедра_КБПЗ_2022рік

Таблиця 7.8 – Вартість основних фондів та амортизаційні відрахування розробника

Групи та види основних фондів	Балансова вартість, грн.	Амортизація	
		Норма, %	Відрахування, грн.
1	2	3	4
Група 3			
1. Будівлі	1280000	-	-
2. Передавальні пристрої	128000	-	-
Всього по групі	1408000	5	70400
Група 4			
3. Обчислювальна техніка	114885	-	-
Всього по групі	114885	50	57442,5
Група 5			
4. Вимірювальні пристрої	5190	-	-
5. Господарський інвентар	28000	-	-
Всього по групі	33190	25	8297,5
Нематеріальні активи			
6. Нематеріальні активи	40000	10	4000
Разом	$К_р$= 1596075		$А_р$= 140140

7.5 Визначення собівартості розробки та ціни програмної продукції

Визначимо основну зарплату виконавців

$$З_о = \frac{З_{сд} \cdot Т_{пз}}{N_e}, \quad (7.11)$$

Кафедра_КбПЗ_2022рік

де Ne – Кількість екземплярів програм, шт.

$$З_о = 454 \cdot 184/40 = 2090 \text{ грн}$$

Визначимо додаткову зарплату (оплата відпусток, виконання державних та суспільних обов'язків) на рівні 10%

$$З_д = З_о \cdot Н_q \cdot 0{,}01, \quad (7.12)$$

де $Н_q$ – норматив додаткової зарплати, %

$$З_д = 2090 \cdot 10 \cdot 0{,}01 = 209 \text{ грн}$$

Відрахування на соціальні потреби за нормативом Нс=22% від суми основної та додаткової зарплати

$$С_{оц} = 0{,}01 \cdot Н_с (З_о + З_д), \quad (7.13)$$

де $Н_с$ – відрахування на соціальні потреби, %

$$С_{оц} = 0{,}01 \cdot 22(2090+209) = 506 \text{ грн}$$

Визначимо загальногосподарські витрати (електроенергію, ремонт і утримання приміщень і т.д) за нормативом Нг=15% від основної зарплати

$$Г_{осп} = З_о \cdot Н_г \cdot 0{,}01, \quad (7.14)$$

де $Н_г$ – загальногосподарські витрати, %

$$Г_{осп} = 2090 \cdot 15 \cdot 0{,}01 = 314 \text{ грн}$$

Визначимо витрати на матеріали для розробки програмної продукції за нормами споживання та діючими цінами за одиницю виміру:

$$З_М = (З_{М1} + З_{М2} + З_{М3})/N_е, \quad (7.15)$$

де $З_{М1}$ – вартість паперу, грн., $З_{М2}$ – вартість запам'ятовуючих пристроїв, грн., $З_{М3}$ – вартість фарби, картріджей, тонеру, грн., $N_е$ – кількість екземплярів програм, шт.

Згідно прийнятих норм на підприємстві $n_{вит}$ приймаємо 0,5 пачки паперу на період розробки. Тоді, враховуючи, що вартість пачки паперу складає $Ц_п$=210 грн., визначаємо вартість паперу за період розробки:

$$З_{М1} = Ц_п \cdot N_м \, n_{міс}. \quad (7.16)$$

$$З_{М1} = 210 \cdot 1 \cdot 0{,}5 = 105 \text{ грн}.$$

Кафедра_КБПЗ_2022рік

Згідно прийнятих норм по комплектації до вартості запам'ятовуючих пристроїв входить вартість CD/DVD дисків. Їх кількість дорівнює кількості коробочних версій запропонованого продукту (приймаємо 10):

$$З_{М2} = \sum Ц_{д}., \quad (7.17)$$

де: $Ц_{д}$ – вартість дисків CD/DVD: CDR box – 23 грн./шт., DVD-R box – 49,2 грн./шт.

$$З_{М2} = 49,2 \cdot 10 = 492 \text{ грн.}$$

Згідно норм одноразовій заправці підлягають усі друкуючі пристрої і становить:

$$З_{М3} = \sum Ц_{з}., \quad (7.18)$$

де: $Ц_{з}$ – вартість розхідних матеріалів друкуючих пристроїв: відновлення та заправка картриджу для Canon i-SENSYS LBP6030W – 574 грн.; картридж для Epson Stylus Photo P50 – 558 грн.; відновлення картриджу для MF217W – 570 грн.

$$З_{М3} = 574+558+570=1702 \text{ грн.}$$

$$З_{М} = (105 + 492 + 1702)/40=57 \text{ грн.}$$

Визначимо витрати на освоєння нових мов програмування або операційних систем за нормативом (Нп = 15%) від основної зарплати виконавців

$$О_{п} = З_{о} \cdot Н_{п} \cdot 0,01, \quad (7.19)$$

де Нп - норматив витрат на освоєння нових мов програмування, %

$$О_{п} = 2090 \cdot 15 \cdot 0,01=314 \text{ грн}$$

Визначимо витрати на амортизацію основних фондів з урахуванням загальної річної суми амортизаційних відрахувань та кількості екземплярів програм ($N_{е}$ = 40 прим.)

$$А_{м} = \frac{А_{р} \cdot N_{міс}}{N_{е} \cdot 12}, \quad (7.20)$$

де Ар – загальна річна сума амортизаційних відрахувань, грн.

$$А_{м} = 140140 \cdot 3/(40 \cdot 12)=876 \text{ грн}$$

Повна собівартість ПЗ визначається як сума витрат за попередніми статтями калькуляції

Кафедра_КОПЗ_2022рік

$$C_п = З_о + З_д + С_{оц} + Г_{осп} + З_м + О_п + А_м. \quad (7.21)$$

$$C_п = 2090+209+506+314+57+314+876 = 4366 \text{ грн.}$$

Визначимо плановий прибуток за рівнем рентабельності (Рп) програмної продукції, яка залежить від складності програми та ступеня новизни задачі.

Для даного програмного забезпечення рівень рентабельності складає 55%

$$П_р = 0{,}01 \cdot Р_п \cdot С_п, \quad (7.22)$$

де Рс – рівень рентабельності, %

$$П_р = 0{,}01 \cdot 55 \cdot 4366 = 2401 \text{ грн.}$$

Величини ціна підприємства, податок на додану вартість, відпускна ціна програмної продукції визначаються за формулами, приведеними в таблиці 7.9

Таблиця 7.9 – Нормативна калькуляція собівартості розробки програмного забезпечення задачі

Найменування статей витрат	**Позначення**	**Величина, грн.**
1. Основна зарплата виконавців	$З_о$	2090
2. Додаткова зарплата виконавців	$З_д$	209
3. Відрахування на соціальні потреби	$С_{оц}$	506
4. Загальногосподарські витрати	$Г_{осп}$	314
5. Витрати на матеріали	$З_М$	57
6. Освоєння нових операційних систем, мов програмування	$О_п$	314
7. Амортизація основних фондів	$А_м$	876
8. Повна собівартість програмного забезпечення	$С_п$	4366
9. Плановий прибуток	$П_р$	2401
10.Ціна підприємства $Ц_п=С_п+П_р$	$Ц_п$	6767
11.Податок на додану вартість $ПДВ = 0.01 \cdot Н_{дв} \cdot Ц_п$	$ПДВ$	1353,4
12.Відпускна ціна програмної продукції Ц = Цп + ПДВ	$Ц$	9168

Кафедра_КбПЗ_2022рік

7.6 Визначення об'єму капітальних вкладень у споживача програмної продукції

Об'єм капітальних вкладень у споживача програмної продукції визначаємо на основі балансової вартості основних фондів, яка враховує ціну, транспортно-заготівельні витрати, вартість будівель, монтажних та пусконалагоджувальних робіт, а також витрати на випробування у виробничих умовах. Результати розрахунків зводимо у таблицю 7.10.

Таблиця 7.10 - Розрахунок об'єму капітальних вкладень у споживача програмної продукції

Найменування капітальних вкладень	**Сума за варіантами, грн**	
	Базовий	**Новий**
Вартість програмної продукції	–	9168
Всього капітальних витрат	–	9168

7.7 Визначення експлуатаційних витрат

Експлуатаційні витрати у споживача програмної продукції визначаємо при умові роботи підсистеми на протязі року. Результати зводимо до таблиці 7.11.

Таблиця 7.11 - Розрахунок експлуатаційних витрат у споживача програмної продукції

Найменування статей витрат	**Позна-чення**	**Сума витрат за варіантами, грн.**	
		Базовий	**Новий**
1. Витрати на техн. обслуговування	$З_р$	105213	26303
2. Витрати на електроенергію	$З_{ел}$	2099	1049
3. Витрати на амортизацію	$З_{ам}$	0	4584
Всього витрат за рік	$І$	107312	31936

Кафедра_КбПЗ_2022рік

Витрати на оплату праці:

$$З_р = Т_р \cdot З_г \cdot (1 + 0{,}01 \cdot Н_q) \cdot (1 + 0{,}01 \cdot Н_с), \quad (7.23)$$

де $Т_р$ – кількість годин обслуговування системи за рік, *год.*; $З_г$ - заробітна плата обслуговуючого персоналу, *грн/год*.

Після купівлі нового програмного забезпечення кількість профілактичних годин робіт зменшилася з 800 годин на рік до 200 годин на рік, тому витрати на технічне обслуговування зменшилися з

$$З_{р\,баз} = 800 \cdot 98 \cdot 1{,}1 \cdot 1{,}22 = 105213 \text{ грн.}$$

до

$$З_{р\,нов} = 200 \cdot 98 \cdot 1{,}1 \cdot 1{,}22 = 26303 \text{ грн.}$$

Витрати на електроенергію визначаються з урахуванням споживаємої потужності ($П_{ел}$) в кіловатах, часу експлуатації технічних засобів ($Т_р$) в годинах та ціни однієї кіловат-години ($Ц_{ел}$).

$$З_{ел} = П_{ел} \cdot Т_р \cdot Ц_{ел}. \quad (7.24)$$

Зел баз=·0,475·2455·1,8 = 2099 грн

Зел нов=·0,475·1227·1,8 = 1049 грн

Витрати по амортизації визначаються на основі норм амортизаційних відрахувань, вартості програмної продукції і основних фондів. Для розрахунку складаємо таблицю 7.12.

Таблиця 7.12 - Розрахунок амортизаційних відрахувань

Групи основних фондів	**Норма амортизації %**	**Балансова вартість, грн., за варіантами**		**Сума відрахувань, грн., за варіантами**	
		Базовий	**Новий**	**Базовий**	**Новий**
Програмна продукція	50	–	9168	–	4584
Всього відрахувань	-	–	9168	–	4584

Кафедра_КбПЗ_2022рік

7.8 Визначення економічної ефективності програмної продукції

Економічна ефективність програмного забезпечення визначається для виготовлювача і споживача за такими показниками.

Величина економічного ефекту при виготовленні програмної продукції, розраховуємо за формулою

$$E_в = (Ц_п - C_п) \cdot N_е - \sum_{i=1}^{m} E_{p_m} \cdot K_{p_m}, \qquad (7.25)$$

де: $K_р$ – балансова вартість основних фондів розробника, грн.; $E_р$ – розрахунковий коефіцієнт капіталовкладень.

$$Ев = (6767 - 4366) \cdot 40 - (0,05 \cdot 1408000 + 0,5 \cdot 114885 +$$
$$+0,25 \cdot 33190 + 0,1 \cdot 40000) \cdot 3/12 = 61005 \text{ грн.}$$

Визначимо період окупності додаткових капітальних вкладень у виробника програмної продукції:

$$T_в = \frac{K_р^*}{(Ц_п - C_п) \cdot N_е}, \qquad (7.26)$$

де: $K_р^*$ – балансова вартість основних фондів розробника без врахування вартості ОФ третьої групи, так як їх строк служби на порядок більший ніж період розробки ПЗ.

$$T_в = \frac{188075}{(6767 - 4366) \cdot 40 \cdot 12 / 3} = 0,5 \text{ років}.$$

Визначимо величину економічного ефекту у користувача програмної продукції за формулою:

$$E_{сп} = (I_б - I_н) - E_н(K_н - K_б), \qquad (7.27)$$

де $I_б$, $I_н$ – величина експлуатаційних витрат за базовим и новим варіантом відповідно, $K_б$, $K_н$ – об'єм капітальних вкладень за варіантами, що порівнюються

$$E_{сп} = (107312 - 31936) - 0,5 \cdot 9168 = 70792 \text{ грн.}$$

Визначимо період окупності додаткових капітальних вкладень у споживача програмної продукції за рахунок зниження експлуатаційних витрат

Кафедра_КбПЗ_2022рік

$$T_{сп} = \frac{K_н - K_б}{I_б - I_н} \quad (7.28)$$

$$T_{сп} = \frac{9168}{107312 - 31936} = 0{,}12 \ року$$

Показники економічної ефективності програмної продукції зводимо до таблиці 7.13.

Таблиця 7.13 - Показники економічної ефективності програмної продукції

Найменування показників	**Одиниця виміру**	**Величина**
1. Кількість екземплярів програми	Прим.	40
2. Повна собівартість розробленої програми	Грн	4366
3. Ціна розробленої програми	Грн.	6767
4. Плановий прибуток від реалізації розробленої програми	Грн.	2401
5. Рентабельність програмної продукції	%	55
6. Об'єм додаткових капітальних вкладень у виробника програмної продукції	Грн.	1596075
7. Загальний прибуток від реалізації програмної продукції	Грн.	96040
8. Величина економічного ефекту при виготовлені програмної продукції	Грн.	61005
9. Період окупності додаткових капітальних вкладень у виробника програмної продукції	Років.	0,5
10.Об'єм додаткових капітальних вкладень у споживача програмної продукції	Грн.	9168
11.Величина економічного ефекту у користувача програмної продукції	Грн.	70792
12.Період окупності додаткових капітальних вкладень у користувача програмної продукції	Років	0,12

Кафедра_КБПЗ_2022рік

Висновки

Розроблена програма економічно вигідна. За рахунок впровадження програмного забезпечення досягається скорочення часу обробки інформації, підвищується культура праці, підвищення якості приймаючих управлінських рішень.

Кафедра_КбПЗ_2022рік

8 ЗАХОДИ З ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ

8.1 Вступ

Охорона праці − це: система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів та засобів, спрямованих на збереження життя, здоров'я і працездатності людини в процесі трудової діяльності.

Охорона праці є складовою частиною безпеки життєдіяльності [53, 54, 61].

Законом України “Про охорону праці” [53] регламентуються загальні положення державної політики в галузі охорони праці, а конкретизуються ці положення нормативно-правовими актами про охорону праці, зокрема Наказом Міністерства соціальної політики України 14.02.2018 № 207, який зареєстровано в Міністерстві юстиції України 25 квітня 2018 р. за №508/31960 «Про затвердження Вимог щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями» [55], яким затверджено нормативно-правовий акт з охорони праці НПАОП 0.00-7.15-18, «Правила охорони праці під час експлуатації електронно-обчислювальних машин», та «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» ДСанПіН 3.3.2-007-98.

Програмісти у процесі роботи мають негативний вплив на органи зору, а також мають значну розумову напругою і нервово-емоційне навантаження. Руки (суглоби пальців та м'язи рук) при роботі з клавіатурою мають теж істотне навантаженням. До шкідливих факторів, які впливають на робитників галузі інформаційних технологій (ІТ) спеціалісти відносять високочастотні електромагнітні коливання (випромінювання) роботи апаратної частини ЕОМ та виділення шкідливих газів.

Ці шкідливі фактори можуть привести до професійних захворювань.

Кафедра_КОПЗ_2022рік

Розглянемо шкідливі чинники роботи програмістів керуючись наступними нормативно-правовими актами: «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» ДСанПіН 3.3.2-007-98 [55], та «Вимоги щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями» НПАОП 0.00-7.15-18.

Умови праці програміста включають наступні фактори:

- параметри повітряного середовища в приміщенні;
- вентиляція приміщення;
- освітлення приміщення;
- параметри повітряного середовища в приміщенні, тощо.

Щоб запропонувати заходи щодо зменшення негативного впливу комп'ютера на організм людини визначемо фактори, які можуть викликати професійне захворювання і впливають на працездатність програміста.

8.2 Шкідливі і небезпечні фактори при роботі з комп'ютером

Електронно-обчислювальна машин (ЕОМ) та інше обладнання є джерелами небезпеки ураження електричним струмом. Так як робота програміста характеризується істотним зоровим навантаженням, то вимагає належного освітлення. У приміщенні, в якому працюють люди (у т.ч. програмісти) необхідно створити належний мікроклімат, праметри якого регламентуються, Державними санітарними правилами і нормами, зокрема ДСанПіН 3.3.2.007-98.

При роботі з використанням ЕОМ відзначають наступні небезпечні та шкідливі фактори:

ризик виникнення надзвичайних ситуацій природного або штучного характеру на об'єкті або території.

- ризик виникнення пожежі;
- негативний вплив на органи зору людини;
- ризики ураження електричним струмом;

Кафедра_КБПЗ_2022рік

- недостатня, або надмірна освітленість робочого місця;

- електромагнітні (у т.ч. високочастотні) електромагнітні випромінювання

- несприятливі мікрокліматичні умови;

- нервово-емоційна напруженість праці;

- інтелектуальні навантаження;

- монотонність праці;

невідповідність ергономічних показників робочого місця діючим вимогам;

- шум;

- статичні навантаження на кістково-м’язовий апарат;

8.3 Аналіз санітарно-гігієнічних умов праці на робочому місці програміста

Розглянемо умови праці у приміщенні, в якому працюють програмісти. Геометричні розміри приміщення наведено у таблиці 8.1.

Таблиця 8.1 - Розміри приміщення

Найменування	
Ширина	
Довжина	
Висота	

Таблиця 8.2 - Площа та обсяг приміщення, на одного працюючого*

Геометрична характеристика	Одиниця виміру	Нормативне значення*	
Площа, S	$м^2$	не менше	
Об'єм, V	$м^3$	не менше	

Кафедра_КБПЗ_2022рік

* Згідно ДСанПіН 3.3.2.007-98 (Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин).

У зазначеному приміщенні працюють двоє людей. За даними, які наведено у табл. 8.1, та табл. 8.2, можна зробити висновок, що площа та об'єм приміщення у розрахунку на одно робоче місце програміста не відповідають нормативним вимогам ДСанПіН 3.3.2-007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» [55], але відповідають нормативним вимогам Наказу Міністерства соціальної політики України № 207, від 14.02.2018 «Про затвердження Вимог щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями» [55] та НПАОП 0.00-1.28-10 «Правила охорони праці під час експлуатації електронно-обчислювальних машин»). Таним чином можна зробити висновок, що санітарно-гігієнічні умови праці на робочому місці програміста відповідають вимогам.

Температура повітря в приміщенні визначається впливом температури зовнішнього повітря і тепловою енергією, яка виділяється всередині приміщення. Джерелами виділення теплоти в даному приміщенні є електроустаткування, освітлювальні прилади, а також люди. У світлий час доби джерелом надлишкового тепла є сонячна радіація. Згідно Постанови № 42 від 01.12.1999 Головного державного санітарного лікаря України, робота, виконувана в даному приміщенні, відноситься до категорії Іа. В цьому випадку людина витрачає енергії до 120 ккал у годину. Вологість повітря в приміщенні визначається впливом багатьох факторів, серед яких: вологість атмосферного повітря, виділення вологи людьми (при диханні та випарами з поверхні шкіри).

Мікроклімат повітряного середовища в приміщенні характеризується запиленістю та загазованістю повітря. Мікроклімат приміщення визначається діючим на організм людини поєднанням, вологості, температури, швидкості руху повітря та інтенсивності теплового випромінювання. Аналіз мікроклімату складається з визначення зазначених вище факторів і порівняння результатів із встановленими нормами.

Кафедра_КОПЗ_2022рік

У таблиці 8.3 наведено оптимальні та фактичні значення параметрів мікроклімату як для категорії ваги робіт Іа, так і розглянутого приміщення. У приміщеннях, де встановлено ЕОМ, рекомендується застосування тільки оптимальних значень показників мікроклімату.

Таблиця 8.3 - Оптимальні і фактичні значення параметрів мікроклімату

Пора року	Оптимальні для Іа			Фактичні		
	Температура, °С	Воло-гість,%	Швидкість повітря, м/с	Температура, °С	Воло-гість%	Швидкість повітря, м/с
Холодна						
Тепла						

Проведений аналіз показує, що показники мікроклімату в приміщенні відповідають установленим нормам. Штучне опалення застосовується у холодний період року.

В літню пору застосовується кондиціонер.

Для боротьби з пилом робляться регулярні провітрювання та вологі прибирання приміщенні.

У приміщенні знаходяться наступні джерела шуму: принтер *Xerox*

Кафедра_КбПЗ_2022рік

Одним з найважливіших факторів, які впливають на ефективність трудової діяльності людини, та попереджають травматизм і професійні захворювання програмістів є освітлення на робочому місці.

З 2019 року діють Державні будівельні норми україни "Природне і штучне освітлення" – ДБН В.2.5-28:2018 [51], у яких прописані вимоги до використання всіх освітлювальних приладів, у т.ч. світлодіодних.

Працю працівника, який постійно працює за комп'ютером, згідно ДБН В.2.5-28:2018 [51], можна віднести до роботи з малою точністю (найменший розмір об'єкта розрізнення від 1 до 5 мм) V-го розряду зорової роботи, з великою контрастністю об'єкта розрізнення (символів на екрані дисплея), з темним тлом (під розряд зорової роботи В). Приміщення можна віднести до 1-ої групи приміщень, у яких проводиться розрізнення об'єктів зорової роботи при фіксованому напрямку лінії зору того, що працює на робочу поверхню. Для такого типу приміщень і розряду зорової роботи нормоване значення коефіцієнта природної освітленості становити не більше 1,5%, освітленість при штучному висвітленні повинна становити 300 Лк. [51], Крім того все поле зору повинне бути освітлено достатньо рівномірно - ця основна гігієнічна вимога. Так як яскраве світло на ділянці периферійного зору значно збільшує напруженість очей і, як наслідок, призводить до їх швидкої стомлюваності, ступінь освітлення приміщення і яскравість екрану комп'ютера повинні бути приблизно однаковими.

8.4 Розробка заходів з умов поліпшення охорони праці

Згідно аналізу умов праці в розглянутому приміщенні, ми одержали наступні результати:

- розмірі приміщення, у розрахунку на одному працюючого, відповідають нормативам;

- мікроклімат відповідає нормативному значенню;

Кафедра_КОПЗ_2022рік

- акустичні умови роботи не перевищують нормативних значенн;

Таким чином можна припустити, що основною причиною можнивого зниження працездатності програміста є психофізіологічний фактор, тому основна пропозиція буде така: дотримання позитивної психологічної атмосфери в колективі та регламентованого режиму праці та відпочинку, організація робочого місця з урахуванням ергономічних вимог.

Рекомендовані заходи: регулярні періодичні наочні огляди персоналом шляхів для евакуації людей із приміщення, відповідно до плану евакуації (який повинен розташовуватись на видному місці у приміщенні), включення до колективного договору мінімально можливого вмісту аптечок з обов'язковою наявністю масок-клапанів, або іншого спорядження для штучного дихання. Регулярна періодична перевірка параметрів заземлення та занулення (вимірювання опору ланцюга).

Регулярна наочне знайомство персоналу із шляхами для евакуації людей із приміщення відповідно до плану евакуації, забезпечення розподільних щитів спеціальними розетками з заземлюючими контактами; організація заземлення всіх приладів і пристроїв, які працюють при напрузі вище 36 В.

Так як при ураженні електричним струмом у людини може статися фібриляція шлуночків серця, в організації бажано мати дефібрилятор і підготовлений персонал для роботи з ним.

8.5 Розрахункова частина

Початкові дані для розрахунку захисного штучного заземлення: опір заземлювача, який нормується: $R_{ЗН} = 4$ *Ом*.

Для захисного штучного заземлення застосовуються ветикальні електроди з метелевого прутка діаметром 30 мм. (*D=30 мм.=0,03 м.)* довжиною *L*=2,5 *м.* та горизонтальний електрод — метелева полоса з перетином 40*4 мм. тип грунта — глина (питомий опір 40 Ом*м). Відстань між вертикальними заземлювачами (електродами) *A*=3 *м*.

Кафедра_КПЗ_2022рік

Глибина закладення горизонтального контура заземлення *t= 0,8 м*.

Умовна товщина верхнього шару грунта: *H=0,4 м*. Напруга — 220/380 В. Розрахункова схема розташування заземлюючих електродів — у ряд.

Розрахунок проводиться за допустимим опором розтіканню струму заземлювача.

Необхідно визначити необхідну кількість вертикальних заземлювачів та длину полоси (горизонтального заземлювача).

Розрахунок захисного заземлення можна автоматизувати за допомогою програми, сирцевий код якої опублікован на стр. 13-16 [56], або аналогічної.

Розрахунок

Відстань від центра вертикального заземлювача до поверхні землі:

T=t+L/2=0,8+2,5/2=2,05 м.

Розрахунковий питомий опір грунта (з врахуванням того, що фактично вся конструкція заземлювача розташовується у нижньому шарі грунта):

$\rho = \psi \rho_2 = 1,36*40=54,5$ *Ом*м.*

де $\psi = 1,36$ - табличне значення коефіцієнта сезонності для відповідної климатичной зони у багатошаровому грунті [58];

$\rho_1 = 50$ *Ом*м.* - табличне значення питомого опору верхнього шару грунта

$\rho_2 = 40$ *Ом*м*. - табличне значення питомого опору нижнього шару грунта

Опір розтіканню електричного струму одного електрода вертикального заземлювача [58]:

$$R_o = 0{,}366\frac{\rho}{L}\left(lg\frac{2L}{D}+\frac{1}{2}lg\frac{4T+L}{4T-L}\right) = 0{,}366\frac{54{,}5}{2{,}5}\left(lg\frac{2\cdot 2{,}5}{0{,}03}+\frac{1}{2}lg\frac{4\cdot 2{,}05+2{,}5}{4\cdot 2{,}05-2{,}5}\right) = 20{,}1 \text{ Ом}.$$

Відношення *A/L=3/2,5=1,2*.

Визначаємо коефіцієнт екранування ветикальньних електродів *Кев=0,8* при попередній (орієтовній) кількості ветикальньних електродів, яке дорівнює 4 [58].

Визначаємо необхідну кількість вертикальних заземлювачів (без вхарування горизонтального заземлювача), при $R_{ЗН} = 4$ *Ом* :

Кафедра_КОПЗ_2022рік

$$_{О}/(\text{Кев } R_{ЗН}) = 20,1 / (0,8*4) = 5,87 \approx 6 \text{ шт.}$$

Визначаємо довжину з'єднуючої полоси [58]:

$$_{П} = 1,05*A*N = 1,05*3*6 = 18,8 \approx 19 \text{ м.}$$

Опір розтіканню електричного струму з'єднуючої полоси [58]:

$$_{П} = 0,366(\rho_2 * K_П / L_П) lg(2(L_П * L_П)/(К*t)) =$$

$$l$$

де $K_П = 5$ - табличне значення коефіцієнта сезонності для відповідної климатичной зони з'єднуючої полоси: [58]. *0,04*0,8)]=16,5 Ом.*

Загальний опір розтіканню електричного струму заземлювача [58]:

$$_{О} * R_П)/(R_О * \eta_П + N * R_П * Кев) =$$

$$= (20,1 * 16,5)/(20,1 * 0,75 + 6 * 16,5 * 0,8) = 3,32 \text{ Ом.}$$

де $\eta_П = 0,75$ - табличне значення коефіцієнта екранування з'єднуючої полоси [58].

Умова $R \leq R_{ЗН}$ виконується ($3,32 \leq 4$).

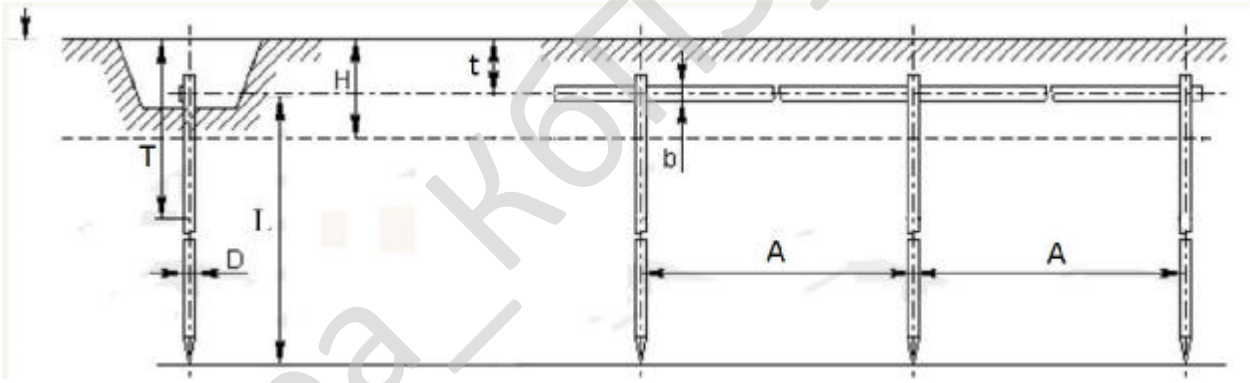

Рисунок 8.1 – Схема штучного заземлення

Висновки до розділу

Дотримання всіх необхідних умов праці не лише сприяє збереженню здоров'я працівників, а також підвищує ефективність виробництва вцілому.

З цих міркувань було здійснено аналіз приміщення, призначеного для праці програмістів, проведено розгляд небезпечних та шкідливих факторів, що негативно впливають на програмістів під час роботи. Виконано розрахунок штучного

Кафедра_КОПЗ_2022рік

освітлення, як одного з ключових факторів впливу на працездатність та здоров’я програміста. Розроблено заходи з охорони праці.

Кафедра_КбПЗ_2022рік

9 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання магістерської роботи, призначено для системи створення персоналізованої реклами для користувачів веб-сайту.

У магістерській роботі наведені теоретичне узагальнення й рішення наукового завдання дослідження методів створення персоналізованої реклами для користувачів веб-сайту.

Рішення даного завдання полягало у вирішенні наступних задач:

– Було проведене дослідження існуючих системи створення персоналізованої реклами для користувачів веб-сайту на основі контентної та колаборативної фільтрації.

– На основі проведеного дослідження розроблено методи та алгоритми для системи створення персоналізованої реклами для користувачів веб-сайту на основі колаборативної фільтрації з застосуванням матричної факторизації.

– Використовуючи розроблені методи та алгоритми, створена програмна реалізація системи створення персоналізованої реклами для користувачів веб-сайту.

Розроблені під час виконання магістерської роботи алгоритми дозволяють успішно вирішувати завдання створення персоналізованої реклами для користувачів будь-якого веб-сайту, що збирає історію дій користувачів і дозволяє оцінювати контент, або будь-яким іншим чином фіксувати інтереси користувачів.

При створені програмного забезпечення було використано об’єктно-орієнтований підхід, а також застосовано методи машинного навчання.

Програма реалізована на мові високого рівня Python у середовищі розробки Anaconda. Дана мова програмування дозволяє максимально ефективно вирішувати задачі розробки системи створення персоналізованої реклами, оскільки заточена

Кафедра_КОПЗ_2022рік

під аналіз даних та веб-розробку. Це дозволило мінімізувати строк розробки програмного забезпечення, і, як наслідок, зменшити витрати на розробку.

Програма призначена для виконання під управлінням багатозадачної ОС Windows 10/11.

Даються необхідні рекомендації з установки та експлуатації розробленого програмного забезпечення.

Для захисту програмного забезпечення від несанкціонованого використання та поширення програмного коду був обраний метод псевдокоду та поліморфні технології.

В цілому результати тестування створеного програмного забезпечення підтверджують правильність проведених досліджень та використаних проектних рішень і повністю відповідають вимогам технічного завдання. Створене програмне забезпечення має потенційну можливість для подальшого вдосконалення і масштабування.

Розроблена програма має реальний економічний ефект від її впровадження у

в

и

р

о

б

н

и

ц

т

в

о

у

с

Кафедра_КбПЗ_2022рік

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Lutz M. Learning Python, 5th Edition Fifth Edition. - O'Reilly Media, 2016. - 1643 p.
2. Lutz M. Python: Pocket Reference Fourth Edition. - O'Reilly Media, 2016. - 210 p.
3. Grinberg M. Flask Web Development: Developing Web Applications with Python 2nd Edition - O'Reilly Media, 2018. - 312 p.
4. Weidman S. Deep Learning from Scratch: Building with Python from First Principles. – O'Reilly. – 252 p.
5. Hurbans R. Grokking Artificial Intelligence Algorithms. – Manning, 2020. – 631 p.
6. Teofili T. Deep Learning for Search. – Manning, 2019. – 695 p.
7. Kotu V., Deshpande B. Data Science: Concepts and Practice. – Elsevier Science, 2018. – 953 p.
8. Fenner M. Machine Learning with Python for Everyone (Addison-Wesley Data & Analytics Series) 1st Edition, Kindle Edition. - Addison-Wesley Professional, 2019. – 586 p.
9. Rajasekaran S., Vijayalakshmi Pai G.A. Neural networks, fuzzy logic, and genetic algorithms: synthesis and applications (with cd-rom) Kindle Edition. – PHI, 2013. – 628 p.
10. Aho A.V., Hopcroft J.E., Ullman J.D. Data Structures and Algorithms. – Pearson, 2001. – 620 p.13. Кормен Т., Лейзерсон Ч., Риверст Р., Штайн К. Алгоритмы: построение и анализ, 3-е издание – М.: Диалектика, 2019. – 1328 p.
11. Knuth D. The Art of Computer Programming, Vol. 1: Fundamental Algorithms, 3rd Edition 3rd Edition. – Addison-Wesley Professional, 2019. – 672 p.
12. Aggarwal C.C. Neural Networks and Deep Learning: A Textbook 1st ed. 2018 Edition. – Springer, 2018. – 520 p.

Кафедра_КбПЗ_2022рік

13. Gusfield D. Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology 1st Edition. – Cambridge University Press, 2008. – 556 p.

14. Дронов В. PHP, MySQL и Dreamweaver. Разработка интерактивных Web-сайтов; БХВ-Петербург - М., 2007. - 480 с.

15. Дунаев В.В. Web-программирование для всех; СПб. [и др.] : Питер - Москва, 2012. - 985 с.

16. Жадаев Александр PHP для начинающих; "Издательство "Питер" - М., 2014. - 288 с.

17. Зелковиц М. Принципы разработки программного обеспечения / М. Зелковиц, А. Шоу, Дж. Гэннон. - М.: Мир, 1982. - 364 с.

18. Игошин В.И. Теория алгоритмов: Учебное пособие / В.И. Игошин. - М.: ИНФРА-М, 2013. - 318 с.

19. Канцедал С.А. Алгоритмизация и программирование : Учебное пособие / С.А. Канцедал. - М.: ИД ФОРУМ, НИЦ ИНФРА-М, 2013. - 352 с.

20. Клейнберг Дж. Алгоритмы. Разработка и применение / Дж. Клейнберг, Е. Тардос. - М.: Питер, 2016. - 800 с.

21. Котеров Д., Костарев А. PHP 5 В подлиннике. – СПб: БХВ-Петербург, 2016. – 1104.

22. Костин С.П. Самоучитель создания Web-сайтов. Триумф – Москва, 2009. – 176 с.

23. Кровчик Э., Кумар В. NET. Сетевое программирование для профессионалов – М.: «Лори», 2007. – 417 с.: ил.

24. Крупский В.Н. Математическая логика и теория алгоритмов: Учебное пособие для студентов учреждений высшего проф. образования / В.Н. Крупский, В.Е. Плиско. - М.: ИЦ Академия, 2013. - 416 с.

25. Кузнецов М., Симдянов И. PHP. Практика создания Web-сайтов; БХВ-Петербург - Москва, 2012. - 347 с.

26. Лафоре Р. Структуры данных и алгоритмы в Java / Роберт Лафоре. - М.: Питер, 2016. - 704 с.

КафедРа_КбПЗ_2022рік

27. Лю Б. Теория и практика неопределенного программирования / Б. Лю. - М.: Бином. Лаборатория знаний, 2015. - 416 с.

28. Магда Ю. Аппаратное обеспечение и эффективное программирование / Юрий Магда. - М.: Питер, 2007. - 352 с.

29. Макконнелл Дж. Анализ алгоритмов. Активный обучающий подход / Дж. Макконнелл. - М.: Техносфера, 2009. - 416 с.

30. Никсон Р. Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript и CSS - Питер, 2013. - 356 с.

31. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологии, протоколы: Учебник для вузов. 4-е изд. – СПб.: Питер, 2010. – 944 с.

32. Опалева Э.А. Языки программирования и методы трансляции / Э.А. Опалева. - М.: БХВ-Петербург, 2005. - 848 с.

33. Пароннджанов В. Учись писать, читать и понимать алгоритмы / В. Пароннджанов. - М.: ДМК Пресс, 2016. - 458 с.

34. Потопахин В. Искусство алгоритмизации / В. Потопахин. - М.: ДМК Пресс, 2011. - 320 с.

35. Пьюривал С. Основы разработки веб-приложений – Питер, 2014. - – 659 с.

36. Робертсон Л.А. Программирование - это просто. Пошаговый подход / Л.А. Робертсон. - М.: Бином. Лаборатория знаний, 2010. - 384 с.

37. Седерхольм Д. Пуленепробиваемый веб-дизайн. Библиотека специалиста / Д. Седерхольм. - СПб.: Питер, 2012. - 304 с.

38. Семакин И.Г. Основы алгоритмизации и программирования: Учебник для студ. учреждений сред. проф. образования / И.Г. Семакин, А.П. Шестаков. - М.: ИЦ Академия, 2013. - 304 с.

39. Скиена С. Алгоритмы. Руководство по разработке / Стивен Скиена. - М.: БХВ-Петербург, 2011. - 720 с.

40. Струченков В. И. Методы оптимизации в прикладных задачах / В.И. Струченков. - М.: Солон-Пресс, 2012. - 167 с.

Кафедра_КбПЗ_2022рік

41. Судоплатов С. В. Математика: математическая логика и теория алгоритмов : учебник и практикум для среднего профессионального образования / С. В. Судоплатов, Е. В. Овчинникова. — 5-е изд., стер. — Москва : Издательство Юрайт, 2019. — 255 с.

42. Сырых Ю. Современный веб-дизайн. Эпоха Веб 3.0 / Ю. Сырых. - М.: Вильямс И.Д., 2013. - 368 с.

43. Таненбаум Э. Компьютерные сети. 4-е изд. / СПб.: Питер, 2003. – 992с.: ил. – (Серия «Классика computer science»).

44. Ташков П.А. Веб-мастеринг: HTML, CSS, JavaScript, PHP, CMS, графика, раскрутка / П. А. Ташков. - СПб.: Питер, 2010.- 512 с.

45. Уайт Э. PHP 5 на практике - НТ Пресс, 2016. - 271 с.

46. Фельке-Моррис, Т. Большая книга веб-дизайна / Т. Фельке-Моррис. - М.: Эксмо, 2012. - 608 с.

47. Фрейен Б. HTML5 и CSS3.Разработка сайтов для любых браузеров и устройств; Питер - Москва, 2014. - 304 с.

48. Фримен Э. Изучаем программирование на HTML5 - Питер, 2013. – 592 с.

49. Хант Э. Программист-прагматик. Путь от подмастерья к мастеру / Э. Хант, Д. Томас. - М.: ЛОРИ, 2013. - 288 с.

50. Ховард Майкл , Лебланк Дэвид , Виега Джон Как написать безопасный код на C++, Java, Perl, PHP, ASP.NET; ДМК Пресс - М., 2014. - 288 с.

51. Державні будівельні норми України: ДБН В.2.5-28:2018. - Режим доступу до ресурсу: https://goo.su/9AkQ

52. Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин: ДСанПІН 3.3.2-007-98. - Режим доступу до ресурсу: https://zakon.rada.gov.ua/rada/show/v0007282-98

53. Закон України «Про охорону праці» від 14.10.1992 р. № 2694-XII. - Режим доступу до ресурсу: https://zakon.rada.gov.ua/laws/show/2694-12

Кафедра_КОПЗ_2022рік

54. Зеркалов Д. В. Охорона праці в Галузі: Загальні вимоги: навч. посіб. Київ: Основа. 2011. 551 с.

55. Наказ Міністерства соціальної політики України 14.02.2018 № 207 «Про затвердження Вимог щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями». - Режим доступу до ресурсу: https://zakon.rada.gov.ua/laws/show/z0508

56. Охорона праці. Ч. 1. Захисне заземлення: метод. вказ. до викон. розрахунків з викор. персон. ЕОМ ІВМ сумісного типу / Кіровоград. ін-т с.-г. машинобуд.; [укл. О. В. Оришака, Є. К. Солових, В. О. Оришака]. - Кіровоград: КІСМ, 1997. - 20 с. Режим доступу до ресурсу: http://dspace.kntu.kr.ua/jspui/handle/123456789/4358

57. Постанова № 42 від 01.12.1999 Головного державного санітарного лікаря України «Санітарні норми мікроклімату виробничих приміщень ДСН 3.3.6.042-99. - Режим доступу до ресурсу: https://zakon.rada.gov.ua/rada/show/va042282-99 (дата звернення 19.09.22).

58. Сакулин В.П., Шептовицкий В.М. Безопасность труда при монтаже и эксплуатации электроустановок / В.П.Сакулин, В.М.Шептовицкий. – Л. : "Колос", 1973. – 238 с.

59. Оришака, О. В. Основи охорони праці: навч. посіб. / О. В. Оришака, Г. П. Горбачова, К. М. Марченко; М-во освіти і науки України, Центральноукраїн. нац. техн. ун-т. - Кропивницький : ЦНТУ, 2022. - 175 с. – Режим доступу до ресурсу: http://dspace.kntu.kr.ua/jspui/handle/123456789/12161 (дата звернення 19.09.22).

60. Методичні рекомендації до виконання розділу "Заходи з охорони праці та техніки безпеки" випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти для здобувачів вищої освіти спеціальностей 123 "Комп'ютерна інженерія" та 122 "Комп'ютерні науки" / М-во освіти і науки України, Центральноукраїн. нац. техн. ун-т, каф. кібербезпеки та програм. забезпечення; [укл. О.В. Оришака, К.М. Марченко]. - Кропивницький: ЦНТУ,

Кафедра_КБПЗ_2022рік

2022. — 19 с. [Електронний ресурс]. – Режим доступу : http://dspace.kntu.kr.ua/jspui/handle/123456789/12240

Кафедра_КбПЗ_2022рік

Додаток А
(обов'язковий)

Технічне завдання

Зміст

Кафедра_КОПЗ_2022рік

1 Найменування та область застосування

Це технічне завдання розповсюджується на дослідження та програмну реалізацію системи створення персоналізованої реклами для користувачів веб-сайту.

2 Підстава для розробки

Підставою для розробки служить завдання на магістерську роботу, видане на кафедрі кібербезпеки та програмного забезпечення (нак. №_____ від __.__.20__ року).

3 Мета та призначення розробки

Метою магістерської роботи є дослідження та програмна реалізація системи створення персоналізованої реклами для користувачів веб-сайту.

4 Джерела розробки

Джерелом цієї магістерської роботи є стосовна до теми література і існуючі аналоги.

5 Технічні вимоги

5.1 Склад продукції

Складниками розробки є:

– вибір і обґрунтування методів реалізації проекту;

– розробка програмної частини системи, а також розробка взаємодії системи з ОС та з користувачем;

Кафедра_КБПЗ_2022рік

– техніко-економічне обґрунтування доцільності прийнятого до розробки програмного забезпечення;

– аналіз умов праці;

– розробка програми, що реалізує спроектовані алгоритми роботи системи.

5.2 Показники призначення

Система повинна забезпечувати:

– створення персоналізованої реклами для користувачів веб-сайту;

– цілісність даних у процесі роботи та при зберіганні;

– простий, інтуїтивно зрозумілий інтерфейс.

5.3 Вимоги до функціональних характеристик

Розроблене програмне забезпечення не повинно мати обмежень на версію драйверів та операційної системи.

5.4 Вимоги до архітектури

Компонент, що розробляється повинен використовувати системні засоби та апаратні засоби, що на даному етапі розвитку обчислювальної техніки найбільше поширені.

5.5 Вимоги до надійності

Програмні модулі написані по всім правилам, які стосуються стандартних викликів процедур, функцій, методів і форм, визначених технічною документацією на середовище розробки.

Кафедра_КбПЗ_2022рік

5.6 Умови експлуатації

Робочі місця користувачів ПЗ повинні задовольняти наступним умовам експлуатації:

– температура повітря: 19-20 град. по Цельсію;

– відносна вологість повітря до 80%;

– атмосферний тиск 107 кПа.

5.7 Вимоги до складу та параметрів технічних засобів

Програмне забезпечення повинно бути реалізоване на ПЕОМ архітектури IBM PC, працювати в ОС Windows 10/11 і з сумісними з цією платформою пристроями і прикладним програмним забезпеченням.

5.8 Вимоги до інформаційної і програмної сумісності

Переносність програмного забезпечення повинна бути забезпечена за рахунок його реалізації стандартного інтерфейсу взаємодії з ОС, що працюють під управлінням ОС Windows 10/11.

5.8.1 Обладнання

Комп’ютер Intel® Celeron/8 Mb/1.2 Gb/SVGA 14” 1Mb або сумісні з ним.

5.8.2 Мова програмування

Мова програмування Python.

Кафедра_КбПЗ_2022рік

5.8.3 Вхідні дані

Опис алгоритму роботи запропонованої системи.

5.8.4 Вихідні дані

Робоча програма.

6 Вимоги до програмної документації

Програмна продукція повинна бути представлена у виді опису структури даних, схем та опису алгоритму, а також текстів вихідних модулів програмного забезпечення згідно ЄСПД .

7 Економічні вимоги

7.1 Для ПЗ необхідно виробити функціонально-вартісний аналіз варіантів розробки.

7.2 Виконати розрахунок витрат показників економічного ефекту з урахуванням цін на 3 вересня 2022 року.

8 Вимоги щодо охорони праці

В частині охорони праці випускної кваліфікаційної роботи повинна бути розглянуті умови праці програмістів під час розробки програмного забезпечення.

Кафедра_КбПЗ_2022рік

9 Перелік документів, що розробляються

– Наукова новизна – 1 аркуш.
– Структурна схема системи – 1 аркуш.
– Функціональна схема системи – 1 аркуш.
– Діаграма процесів – 1 аркуш.
– Блок-схема алгоритму роботи програми – 2 аркуша.
– Показники економічної ефективності – 1 аркуш.
– Пояснювальна записка – 86 аркушів.

10 Етапи розробки

10.1 Збір і обробка інформації по темі магістерської роботи. Постановка задачі на виконання магістерської роботи (складання ТЗ).

10.2 Проведення досліджень або експериментальних робіт для уточнення основних положень магістерської роботи.

10.3 Розробка функціональних схем, блок схем алгоритмів роботи програмного забезпечення.

10.4 Побудова схем взаємодії даних.

10.5 Створення прототипу ПЗ.

10.6 Віднаходження ПЗ, аналіз отриманих результатів.

10.7 Робота над питанням охорони праці і техніки безпеки.

10.8 Розрахунок з техніко-економічного обґрунтування.

10.9 Оформлення пояснювальної записки і виконання робіт по графічній частині.

11 Порядок контролю та приймання

11.1 Подання магістерської роботи на попередній захист 10.12.2022 р.

11.2 Подання магістерської роботи на захист .12.2022 р.

Кафедра_КбПЗ_2022рік

Додаток Б

(обов’язковий)

Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

ЗАТВЕРДЖУЮ
Керівник випускної кваліфікаційної роботи
за другим (магістерським) рівнем вищої освіти
_____ Р.М. Минайленко

Дослідження та програмна реалізація системи створення персоналізованої реклами для користувачів веб-сайту

Лістинг програми

Код документу 12

Носій: DVD-диск

Загальна кількість аркушів: 14

Літера: РП

Кропивницький – 2022 року

Кафедра_КОПЗ_2022рік

Main.py – файл основної програми

```
sDirProg = "c:/test/"
sDirData =  "c:/data/"
import numpy
import sys
import csv
sys.path.append(sDirProg + "RecommendationSystem/")
sys.path.append(sDirProg + "RecommendationSystem/Graph")
sys.path.append(sDirProg + "RecommendationSystem/Table")
sys.path.append(sDirProg + "RecommendationSystem/Time")

from RecommendationSystem import *
import time
import math


driver = DatabaseConnection("1000", "user", "1")
session = driver.session()


n = 10 # кількість користувачів для формування рекомендацій

nn = "1" # номер списку рекомендацій
FileЛог = open(sDirData + "ratings"+str(n)+"_"+str(nn)+"_log"+".txt", "w",
newline="", encoding='utf-8-sig')



ListNode = SelectNUsersFromTable(sDirData, n, nn, sDirData + "ratings.csv") #
ListNode - Список користувачів, для яких формувати рекомендації
ListNode = ReadyListOfUsers(sDirData, n, nn)
WriteToDBGraph(session, sDirData + "ratings"+str(n)+"_"+str(nn)+".csv")
SplitDataIntoTrainingAndTest(session)

DeleteSimilarityRelationshipsOfAllTypes(session)
DeleteRelationshipsRecommendations(session)


def AlgSVD(IdUsers, IdItems, RatingMatrix):

    MinRating = 0.5
    MaxRating = 5

    def ForecastingRating(BiasUser,
                                BiasItem,
                                VectorHiddenUserFactors,
                                VectorHiddenItemFactors,
                                GlobalAverageRating):

        Prediction = GlobalAverageRating + BiasUser + BiasItem
        Prediction += numpy.dot(VectorHiddenUserFactors,
VectorHiddenItemFactors)

        return Prediction



    def RMSECalc(list1, list2):
        n = len(list1)
        rmse = 0
        for i in range(n):
            rmse += math.pow((list1[i] - list2[i]), 2)


        rmse = rmse / n
        rmse = math.sqrt(rmse)
```

КафеДра_КОПЗ_2022рік

```
        return rmse


    def TrainingSystem(RatingMatrix,
                       NumberOfHiddenFactors = 70,
                        LearningRate = 0.005,
                        RegularizationFactor = 0.02,
                        NumberOfEpochsOfTraining = 100,
                        StopCondition = 0.001):

        # Ініціалізація

        NumUsers = len(RatingMatrix)
        NumItems = len(RatingMatrix[0])

        k = 0
        GlobalAverageRating = 0

        for user in range(len(RatingMatrix)):
            for item in range(len(RatingMatrix[0])):
                if RatingMatrix[user][item] != 0:
                    k += 1
                    GlobalAverageRating += RatingMatrix[user][item]

        GlobalAverageRating = GlobalAverageRating / k

        print("GlobalAverageRating", GlobalAverageRating)

        UserBiasVector = [0.01] * NumUsers
        ItemBiasVector= [0.01] * NumItems
        for ii in range(NumUsers):
            UserBiasVector[ii] * ii
        for ii in range(NumItems):
            ItemBiasVector[ii] * ii


# зміщення об’єктів
        for item in range(len(RatingMatrix[0])):
            sumDRi = 0
            ki = 0
            for user in range(len(RatingMatrix)):
                if RatingMatrix[user][item] != 0:
                    ki += 1
                    sumDRi += RatingMatrix[user][item] - GlobalAverageRating
            if ki != 0:
                ItemBiasVector[item] = (sumDRi / ki ) # * 1.5

# зміщення користувачів
        for user in range(len(RatingMatrix)):
            sumDRu = 0
            ku = 0
            for item in range(len(RatingMatrix[0])):
                if RatingMatrix[user][item] != 0:
                    ku += 1
                    sumDRu += RatingMatrix[user][item] - GlobalAverageRating -
ItemBiasVector[item]
            if ku != 0:
                UserBiasVector[user] = (sumDRu / ku) # * 1.5



        MatrixHiddenFactorsUsers = numpy.random.normal(0, 0.1,
                                                       (NumUsers,
NumberOfHiddenFactors))

        MatrixHiddenFactorsItems = numpy.random.normal(0, 0.1,
                                                       (NumItems,
```

Кафедра_КОПЗ_2022рік

```
NumberOfHiddenFactors))

        RMSEDic = []


        Set1 = [i for i in range(len(RatingMatrix))]
        random.shuffle(Set1)


        for epoch in range(NumberOfEpochsOfTraining):
            errList = []
            ratingList = []
            predList = []

            for user in Set1:




                Set2 = [i for i in range(len(RatingMatrix[0]))]
                random.shuffle(Set2)

                for item in Set2:
                    if True:
                        BiasUser = UserBiasVector[user]
                        BiasItem = ItemBiasVector[item]
                        VectorHiddenUserFactors = MatrixHiddenFactorsUsers[user]
                        VectorHiddenItemFactors = MatrixHiddenFactorsItems[item]

                        # прогнозування вподобань
                        Prediction = ForecastingRating(BiasUser,
                                                       BiasItem,
VectorHiddenUserFactors,
VectorHiddenItemFactors,
                                                       GlobalAverageRating)



                        # помилка прогнозування
                        ErrorPrediction = RatingMatrix[user][item] - Prediction

                        errList.append(ErrorPrediction)

                        predList.append(Prediction)
                        ratingList.append(RatingMatrix[user][item])


                        # оновлення прихованих факторів
                        for factor in range(NumberOfHiddenFactors):
                            MSFP_f = MatrixHiddenFactorsUsers[user][factor]
                            MSFPr_f = MatrixHiddenFactorsItems[item][factor]


                            MatrixHiddenFactorsUsers[user][factor] +=
LearningRate * (ErrorPrediction * MSFPr_f - RegularizationFactor * MSFP_f)
                            MatrixHiddenFactorsItems[item][factor] +=
LearningRate * (ErrorPrediction * MSFP_f - RegularizationFactor * MSFPr_f)
                            GSP = UserBiasVector[user]
                            GSPr = ItemBiasVector[item]
                            UserBiasVector[user] += LearningRate *
(ErrorPrediction - RegularizationFactor * GSP)
                            ItemBiasVector[item] += LearningRate *
(ErrorPrediction - RegularizationFactor * GSPr)
```

Кафедра_КСПЗ_2022рік

```
            RMSEDic.append( RMSECalc(predList, ratingList))

        print(RMSEDic)
        print(RMSECalc(predList, ratingList) )



        return (UserBiasVector,
                ItemBiasVector,
                MatrixHiddenFactorsUsers,
                MatrixHiddenFactorsItems,
                GlobalAverageRating,
                NumUsers,
                NumItems,
                RMSEDic)


    (UserBiasVector,
     ItemBiasVector,
     MatrixHiddenFactorsUsers,
     MatrixHiddenFactorsItems,
     GlobalAverageRating,
     NumUsers,
     NumItems,
     listRMSE) = TrainingSystem(RatingMatrix,
                                            NumberOfHiddenFactors = 70,
                                             LearningRate = 0.005,
                                             RegularizationFactor = 0.02,
                                             NumberOfEpochsOfTraining = 100,
                                             StopCondition = 0.0001)

    MatrixPredictions = []
    for user in range(NumUsers):
        MatrixPredictions.append([0] * NumItems)


    for user in range(NumUsers):
        for item in range(NumItems):
            if RatingMatrix[user][item] == 0:
                BiasUser = UserBiasVector[user]
                BiasItem = ItemBiasVector[item]
                VectorHiddenUserFactors = MatrixHiddenFactorsUsers[user]
                VectorHiddenItemFactors = MatrixHiddenFactorsItems[item]

                # prediction рейтинга
                Prediction = ForecastingRating(BiasUser,
                                               BiasItem,
                                               VectorHiddenUserFactors,
                                               VectorHiddenItemFactors,
                                               GlobalAverageRating)

                MatrixPredictions[user][item] = Prediction

    return (MatrixPredictions,
            listRMSE,
            MatrixHiddenFactorsUsers,
            MatrixHiddenFactorsItems,
            UserBiasVector,
            ItemBiasVector)

IdUsers, IdItems, RatingMatrix = GetRatingMatrix(session)



(MatrixPredictions,
 listRMSE,
```

Кафедра КбПЗ 2022рік

```
 MatrixHiddenFactorsUsers,
 MatrixHiddenFactorsItems,
 UserBiasVector,
 ItemBiasVector) = AlgSVD(IdUsers, IdItems, RatingMatrix)

DeleteRelationshipsRecommendations(session)

k1 = 0.2
k2 = 0.3
def Rounding(x, k1, k2):

    if x <= (0.5 - k1):
        x = 0
    elif x >= (0.5 - k1) and x < (0.5 + k2):
        x = 0.5
    elif x >= (1.0 - k1) and x < (1.0 + k2):
        x = 1.0
    elif x >= (1.5 - k1) and x < (1.5 + k2):
        x = 1.5
    elif x >= (2.0 - k1) and x < (2.0 + k2):
        x = 2.0
    elif x >= (2.5 - k1) and x < (2.5 + k2):
        x = 2.5
    elif x >= (3.0 - k1) and x < (3.0 + k2):
        x = 3.0
    elif x >= (3.5 - k1) and x < (3.5 + k2):
        x = 3.5
    elif x >= (4.0 - k1) and x < (4.0 + k2):
        x = 4.0
    elif x >= (4.5 - k1) and x < (4.5 + k2):
        x = 4.5
    elif x >= (5.0 - k1):
        x = 5.0

    return x

for i in range(len(MatrixPredictions)):
    for j in range(len(MatrixPredictions[0])):
        if MatrixPredictions[i][j] > 0:
            uId = IdUsers[i]
            mId = IdItems[j]
            rt = MatrixPredictions[i][j]

            if rt < 0.5:
                rt = 0.5
            if rt > 5:
                rt = 5


    xlist1 = []

    for record in gen :
        for k,v in record.items() :
            xlist1.append(v)


countValueRating = [0]*11
for i in range(len(MatrixPredictions)):
    for j in range(len(MatrixPredictions[0])):
        if MatrixPredictions[i][j] < 0.5:
            countValueRating[0] += 1
        if MatrixPredictions[i][j] >= 0.5 and MatrixPredictions[i][j] < 1.0:
            countValueRating[1] += 1
        if MatrixPredictions[i][j] >= 1.0 and MatrixPredictions[i][j] < 1.5:
            countValueRating[2] += 1
        if MatrixPredictions[i][j] >= 1.5 and MatrixPredictions[i][j] < 2.0:
            countValueRating[3] += 1
        if MatrixPredictions[i][j] >= 2.0 and MatrixPredictions[i][j] < 2.5:
            countValueRating[4] += 1
```

КафедраКбПЗ_2022рік

```
        if MatrixPredictions[i][j] >= 2.5 and MatrixPredictions[i][j] < 3.0:
            countValueRating[5] += 1
        if MatrixPredictions[i][j] >= 3.0 and MatrixPredictions[i][j] < 3.5:
            countValueRating[6] += 1
        if MatrixPredictions[i][j] >= 3.5 and MatrixPredictions[i][j] < 4.0:
            countValueRating[7] += 1
        if MatrixPredictions[i][j] >= 4.0 and MatrixPredictions[i][j] < 4.5:
            countValueRating[8] += 1
        if MatrixPredictions[i][j] >= 4.5 and MatrixPredictions[i][j] < 5.0:
            countValueRating[9] += 1
        if MatrixPredictions[i][j] >= 5.0:
            countValueRating[10] += 1
print("countValueRating", countValueRating)

countValueRating2 = [0]*3
for i in range(len(MatrixPredictions)):
    for j in range(len(MatrixPredictions[0])):
        if MatrixPredictions[i][j] < 0.5:
            countValueRating2[0] += 1
        elif MatrixPredictions[i][j] > 0.5 and MatrixPredictions[i][j] < 3.5:
            countValueRating2[1] += 1
        elif MatrixPredictions[i][j] >= 3.5:
            countValueRating2[2] += 1


print("countValueRating2", countValueRating2)

print("max([max(x) for x in MatrixPredictions])",max([max(x) for x in
MatrixPredictions]))
print("min([min(x) for x in MatrixPredictions])",min([min(x) for x in
MatrixPredictions]))

print("len(listRMSE)",len(listRMSE))

import matplotlib.pyplot as plt

xx = [0.0, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0]
xx2 = [0.5, 3.5, 5.0]

plt.plot(xx, countValueRating)

plt.plot(xx2, countValueRating2)

session.close
FileJor.close()


f1 = open("usersFactors.csv", "w", newline="")
writer1 = csv.writer(f1)
for i in range(len(MatrixHiddenFactorsUsers)):
    s = ""
    for j in range(len(MatrixHiddenFactorsUsers[0])):
        s += str(MatrixHiddenFactorsUsers[i][j]) + ";"
    s += str(MatrixHiddenFactorsUsers[i][j]) + "\n"
    writer1.writerows(s)

f1.close()



f2 = open("moviesFactors.csv", "w", newline="")
writer2 = csv.writer(f2)
for i in range(len(MatrixHiddenFactorsItems)):
    s = ""
    for j in range(len(MatrixHiddenFactorsItems[0])):
        s += str(MatrixHiddenFactorsItems[i][j]) + ";"
    s += str(MatrixHiddenFactorsItems[i][j]) + "\n"
    writer2.writerows(s)
f2.close()
```

Кафедра_КбПЗ_2022рік

```
plt.plot(MatrixHiddenFactorsUsers)

plt.show()


plt.plot(sort(MatrixHiddenFactorsUsers[0]))
plt.show()

l = []
for i in range(len(MatrixHiddenFactorsUsers)):
  l.append(sort(MatrixHiddenFactorsUsers[i]))
  plt.plot(l[i])
plt.show()

for i in range(len(MatrixHiddenFactorsItems)):

  plt.plot(MatrixHiddenFactorsItems[i])
                                    plt.show()
```

Кафедра_КбПЗ_2022рік

table_work.py – файл для роботи з електронною таблицею

```
from numpy import *
import matplotlib.pyplot as plt
import csv
import random
import re


def SelectNUsersFromTable(sDirData, NumUsers, NumSeries, FileNameFrom, Scope =
500):


    n = NumUsers
    nn = NumSeries
    p = []*n
    l = []
    x = 0
    for i in range(n):
        r = x + random.randint(1, Scope)
        p.append(r)
        x = r

    FileToData = open(sDirData + "ratings"+str(n)+"_"+str(nn)+".csv", "w",
newline="", encoding='utf-8-sig')
    FileToListID = open(sDirData + "ratings"+str(n)+"_users"+str(nn)+".csv",
"w", newline="", encoding='utf-8-sig')


    with open(FileNameFrom + "ratings.csv", "r", newline="", encoding='utf-8-
sig') as FileFrom:
        reader1 = csv.reader(FileFrom, delimiter=',')

        i = p[0]
        j = 0

        next(reader1)
        for row in reader1:

            i = int(row[0])

            if (i != j) and (i in p):


                l.append(row[0])

            if i > p[n-1]:
                break

            if i in p:
                FileToData.write(row[0] + ';' + row[1] + ';' + row[2] + ';' +
row[3] + '\n')

            j = i


    FileToListID.write(str(l))
    FileToListID.write(';' + str(len(l)))



    FileToData.close()
    FileToListID.close()
    FileFrom.close()

    print("З датасету вибрано користувачів: ", len(l))
```

Кафедра_КБПЗ_2022рік

```
    return l

def SelectNUsersFromTablePlus(sDirData, NumUsers, NumSeries, FileNameFrom):


    n = NumUsers
    nn = NumSeries
    p = []*n
    l = []
    lm = []
    x = 0
    for i in range(n):
        r = x + random.randint(1, 500)
        p.append(r)
        x = r

    FileToData = open(sDirData + "ratings"+str(n)+"_"+str(nn)+".csv", "w",
newline="", encoding='utf-8-sig')
    FileToListID = open(sDirData + "ratings"+str(n)+"_users"+str(nn)+".csv",
"w", newline="", encoding='utf-8-sig')


    with open(sDirData + "ratings.csv", "r", newline="", encoding='utf-8-sig')
as FileFrom:
        reader1 = csv.reader(FileFrom, delimiter=',')

        i = p[0]
        j = 0

        next(reader1)
        for row in reader1:

            i = int(row[0])

            if (i != j) and (i in p):


                l.append(row[0])

            if i > p[n-1]:
                break

            if i in p:
                FileToData.write(row[0] + ';' + row[1] + ';' + row[2] + ';' +
row[3] + '\n')

                if row[1] not in lm:
                    lm.append(row[1])

            j = i


    FileToListID.write(str(l))
    FileToListID.write(';' + str(len(l)))

    FileFrom.close()
    with open(sDirData + "ratings.csv", "r", newline="", encoding='utf-8-sig')
as FileFrom:
        reader1 = csv.reader(FileFrom, delimiter=',')
        next(reader1)
        for row in reader1:
            x = random.randint(0,100)
            if (row[1] in lm) and (row[0] not in l) and x <= 2:
                FileToData.write(row[0] + ';' + row[1] + ';' + row[2] + ';' +
row[3] + '\n')
```

Кафедра_КбПЗ_2022рік

```
    FileToData.close()
    FileToListID.close()
    FileFrom.close()

    print("З датасету вибрано користувачів: ", len(l))

    return l

def SelectNUsersFromTableConsistently(sDirData, NumUsers, NumSeries,
FileNameFrom):


    n = NumUsers
    nn = NumSeries
    p = []*n
    l = []
    x = 0
    for i in range(n):
        r = i
        p.append(r)
        x = r

    FileToData = open(sDirData + "ratings"+str(n)+"_"+str(nn)+".csv", "w",
newline="", encoding='utf-8-sig')
    FileToListID = open(sDirData + "ratings"+str(n)+"_users"+str(nn)+".csv",
"w", newline="", encoding='utf-8-sig')


    with open(sDirData + "ratings.csv", "r", newline="", encoding='utf-8-sig')
as FileFrom:
        reader1 = csv.reader(FileFrom, delimiter=',')

        i = p[0]
        j = 0

        next(reader1)
        for row in reader1:

            i = int(row[0])

            if (i != j) and (i in p):


                l.append(row[0])

            if i > p[n-1]:
                break

            if i in p:
                FileToData.write(row[0] + ';' + row[1] + ';' + row[2] + ';' +
row[3] + '\n')

            j = i


    FileToListID.write(str(l))
    FileToListID.write(';' + str(len(l)))



    FileToData.close()
    FileToListID.close()
    FileFrom.close()

    print("З датасету вибрано користувачів: ", len(l))

    return l
```

Кафедра_КбПЗ_2022рік

```
def SelectNUsersFromTableВДиапазоне(sDirData, First, Last, NumSeries,
FileNameFrom):


    n = Last - First + 1
    nn = NumSeries
    p = []*n
    l = []
    x = 0
    for i in range(First, Last+1):
        r = i
        p.append(r)
        x = r

    FileToData = open(sDirData + "ratings"+str(n)+"_"+str(nn)+".csv", "w",
newline="", encoding='utf-8-sig')
    FileToListID = open(sDirData + "ratings"+str(n)+"_users"+str(nn)+".csv",
"w", newline="", encoding='utf-8-sig')


    with open(FileNameFrom + "ratings.csv", "r", newline="", encoding='utf-8-
sig') as FileFrom:
        reader1 = csv.reader(FileFrom, delimiter=',')

        i = p[0]
        j = 0

        next(reader1)
        for row in reader1:

            i = int(row[0])

            if (i != j) and (i in p):


                l.append(row[0])

            if i > p[n-1]:
                break

            if i in p:
                FileToData.write(row[0] + ';' + row[1] + ';' + row[2] + ';' +
row[3] + '\n')

            j = i


    FileToListID.write(str(l))
    FileToListID.write(';' + str(len(l)))



    FileToData.close()
    FileToListID.close()
    FileFrom.close()

    print("З датасету вибрано користувачів: ", len(l))

    return l



def ReadyListOfUsers(sDirData, NumUsers, NumSeries):


    n = NumUsers
    nn = NumSeries
    l = [0]
```

Кафедра_КбПЗ_2022рік

```
    FileToListID = open(sDirData + "ratings"+str(n)+"_users"+str(nn)+".csv",
"r", newline="", encoding='utf-8-sig')

    reader1 = csv.reader(FileToListID, delimiter=';')

    for row in reader1:
        l = row[0]

    FileToListID.close()

    if "#" in l:
        l = re.findall(r'(\#\w\d+)\,*', l)
    else:
        l = re.findall(r'(\d+)\,*', l)

    return l

def ReadyMovieList(NumMovies, NumSeries):


    n = NumMovies
    nn = NumSeries
    l = [0]


    FileListId = open(sDirData + "movies/movies"+str(n)+"_id"+str(nn)+".csv",
"r", newline="", encoding='utf-8-sig')

    reader1 = csv.reader(FileListId, delimiter=';')

    for row in reader1:
        l = row[0]

    FileListId.close()

    l = re.findall(r'(\d+)\,*', l)

    return l


def SelectNMoviesFromTable(NumMovies, NumSeries, Directory, FileName):

    n = NumMovies
    nn = NumSeries
    p = []*n
    l = []
    x = 0

    for i in range(n):
        r = x + random.randint(1, 100)
        p.append(r)
        x = r


    FileToData = open(Directory + "movies" + str(n)+"_"+str(nn)+".csv", "w",
newline="", encoding='utf-8-sig')
    FileToListID = open(Directory + "movies" + str(n)+"_id"+str(nn)+".csv", "w",
newline="", encoding='utf-8-sig')


    with open(Directory + FileName, "r", newline="", encoding="ISO-8859-1") as
FileFrom:
        reader1 = csv.reader(FileFrom, delimiter=';')

        t = 0
```

Кафедра_КбПЗ_2022рік

```
        next(reader1) # пропустити заголовок таблиці
        for row in reader1:

            if t in p:
                try:

                    FileToData.write(row[0] + ';' + row[1] + ';' + row[2] + ';'
+ row[3] + '\n')
                    l.append(row[0])

                except:
                    print("помилка")

            t += 1


    FileToListID.write(str(l))
    FileToListID.write(';' + str(len(l)))



    FileToData.close()
    FileToListID.close()
    FileFrom.close()

    print("З датасету вибрано фільмів: ", len(l))

    return l
```

Кафедра_КбПЗ_2022рік