НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Навчально-науковий інститут прикладного системного аналізу
Кафедра системного проектування

«На правах рукопису»
УДК 004.67

ДО ЗАХИСТУ ДОПУЩЕНО
Завідувач кафедри
_____ Вадим МУХІН
«___» _____ 2023 р.

Магістерська дисертація

на здобуття ступеня магістра

за освітньо-професійною програмою

“Інтелектуальні сервіс-орієнтовані розподілені обчислювання”

зі спеціальності 122 "Комп'ютерні науки"

на тему: «Методика редукції обсягу інформації в системах обробки великих даних**»**

Виконав:
студент ІІ курсу, групи ДА-21мп
Дзиговський Владислав Ігорович _____

Науковий керівник:
Професор, доктор технічних наук, професор,
Рогоза Валерій Станіславович _____

Рецензент:
Доцент, кандидат фізико-математичних наук, доцент,
Стусь Олександр Вікторович _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2023

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Навчально-науковий інститут прикладного системного аналізу

Кафедра системного проектування

Рівень вищої освіти – другий (магістерський)

Спеціальність – 122 "Комп'ютерні науки"

Освітньо-професійна програма – "Інтелектуальні сервіс-орієнтовані розподілені обчислювання"

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Вадим МУХІН

«___» _____ 2023 р.

ЗАВДАННЯ
на магістерську дисертацію студенту

Дзиговському Владиславу Ігоровичу

1. Тема дисертації «Методика редукції обсягу інформації в системах обробки великих даних», науковий керівник дисертації Рогоза Валерій Станіславович, доктор технічних наук, професор, затверджені наказом по університету від «_8_»__11_____ 2023 р., № 5200-с

2. Строк подання студентом дисертації – 05 січня 2024 р.

3. Об’єкт дослідження

Основним об'єктом дослідження є аналітичні методи обробки великих даних для їх застосування в побудові рекомендаційних систем, кластеризації текстових документів, кодуванні та декодуванні зображень.

4. Вихідні дані
- аналіз наукових та практичних статей, книг, звітів та інших джерел;
- створення математичної та комп'ютерної моделей системи обробки великих даних та її використання для дослідження різних методик та алгоритмів зменшення обсягу даних;
- результатом дослідження є знаходження статистично значущих залежностей між різними методами зменшення обсягу даних та їх ефективністю та точністю.

5. Перелік завдань, які потрібно розробити

1. Дослідження літератури.
2. Математичний аналіз.
3. Моделювання.
4. Комп'ютерне моделювання.

6. Перелік графічного (ілюстративного) матеріалу: структурні схеми алгоритмів, фрагменти розроблених програм, графічні представлення отриманих програмами даних.

7. Орієнтовний перелік публікацій:

1. Системні науки та інформатика: збірник доповідей II науково-практичної конференції «Системні науки та інформатика», 4–8 грудня 2023 року, Київ. – К., НН ІПСА КПІ ім. Ігоря Сікорського, 2023. – с. 273-278.

8. Дата видачі завдання – «_1__»_____09____ 2023 р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1.	Підготовка тестових прикладів для виконання комп’ютерних експериментів	5 листопада 2023	Виконано
2.	Створення алгоритмів обчислень для обраних методів обробки даних	12 листопада 2023	Виконано
3.	Розробка програм на підставі створених алгоритмів.	25 листопада 2023	Виконано
4.	Виконання комп’ютерних експериментів над обраними тестовими прикладами з використанням розроблених програм	15 грудня 2023	Виконано
5.	Розробка стратап-проекту	20 грудня 2023	Виконано
6.	Написання пояснювальної записки та підготовка до захисту дисертації	25 грудня 2023	Виконано
7.	Подання готової пояснювальної записки науковому керівникові та рецензентам для читання та написання відгуку та рецензій	05 січня 2024	Виконано

Студент Владислав ДЗИГОВСЬКИЙ

Науковий керівник дисертації Валерій РОГОЗА

РЕФЕРАТ

Магістерської дисертації Дзиговського Владислава Ігоровича

«Методика редукції обсягу інформації в системах обробки великих даних»

Робота виконана на 97 сторінках, містить 58 ілюстрацій, 22 таблиць. При підготовці використовувалась література з 28 джерел.

Актуальність теми. Актуальність запропонованих методів виникає із необхідності ефективної редукції обсягу інформації в системах обробки великих даних. В сучасному світі об'єми інформації швидко зростають, і важливо мати засоби для точного відбору, стиснення та аналізу цих даних. Ці методики мають значення як у сферах бізнесу, де необхідно оптимізувати ресурси та приймати рішення на основі обмеженого обсягу інформації, так і у наукових дослідженнях, де зменшення шуму та видалення надлишкових даних дозволяють виявити суттєві зв'язки. Ці методи можуть знайти своє застосування у багатьох галузях, сприяючи оптимізації ресурсів та поліпшенню аналізу даних.

Мета та задачі дослідження. Метою даної магістерської дисертації є дослідження методів редукції у задачах побудови рекомендаційних систем, кластеризації текстових документів, кодуванні та декодуванні зображень.

Об’єкт досліджень. Основним об'єктом дослідження є аналітичні методи обробки великих даних для їх застосування в побудові рекомендаційних систем, кластеризації текстових документів, кодуванні та декодуванні зображень.

Предмет досліджень. Предметом досліджень є методи зниження розмірностей інформації.

Методи досліджень. У роботі застосовувалися аналіз літературних джерел, порівняльний аналіз, моделювання, комп’ютерне моделювання.

Наукова новизна. Наукова новизна роботи полягає у тому, що було проведено дослідження та аналіз роботи методів редукції у задачах побудови рекомендаційних систем, кластеризації текстових документів, кодуванні та декодуванні зображень. Було розроблено програмні реалізації для проведення

дослідження. В результаті дослідження отримано графічні та чисельні дані робити методів редукції у кожній задачі. За результатами було проведено аналіз та побудована порівняльна характеристика роботи кожного методу у визначених задачах.

Потенційні застосування та практична цінність результатів магістерської дисертації:

1. Покращення алгоритмів рекомендаційних систем при роботі з великими даними.
2. Розвиток пошуку спільних тем у наборі текстових документів через методи редукції.
3. Пошук нових способів кодування та декодування з використанням методів редукції.

Публікації

1. Системні науки та інформатика: збірник доповідей II науково-практичної конференції «Системні науки та інформатика», 4–8 грудня 2023 року, Київ. – К., НН ІПСА КПІ ім. Ігоря Сікорського, 2023. – с. 273-278.

Ключові слова. Редукція обсягу даних, зниження розмірності, перетворення матриць, простори високої та низької розмірностей, обробка та аналіз даних.

ABSTRACT

Master's thesis of Dzyhovskyi Vladyslav on the topic
"Methodology for reducing the volume of information in big data processing systems"

The work is done on 97 pages, contains 58 illustrations, 22 tables. Literature from 28 sources was used in the preparation.

Actuality of theme. The relevance of the proposed methods arises from the need to effectively reduce the volume of information in big data processing systems. In today's world, volumes of information are growing rapidly, and it is important to have tools for accurate selection, compression and analysis of this data. These techniques are important both in business areas, where it is necessary to optimize resources and make decisions based on a limited amount of information, and in scientific research, where noise reduction and removal of redundant data allow to reveal significant relationships. These methods can find their application in many industries, helping to optimize resources and improve data analysis.

The purpose and objectives of the research. The purpose of this master's dissertation is to investigate methods of reduction in tasks related to building recommendation systems, clustering of text documents, and encoding and decoding of images.

Object of research. The primary object of investigation is analytical methods for processing big data and their application in the construction of recommendation systems, clustering of text documents, and encoding and decoding of images.

Subject of research. The subject of the research is methods for dimensionality reduction of information.

Research methods. The work involved the use of literary sources analysis, comparative analysis, modeling, and computer simulation.

Scientific novelty. The scientific novelty of the work lies in conducting research and analysis of methods for dimensionality reduction in tasks related to building recommendation systems, clustering of text documents, and encoding and decoding of images. Software implementations were developed to facilitate the

research process. The study resulted in graphical and numerical data depicting the performance of dimensionality reduction methods in each task. The findings were analyzed, and a comparative evaluation of each method's performance in the specified tasks was conducted.

Potential applications and practical value of the diploma thesis results include:

1. Improvement of recommendation system algorithms when dealing with large datasets.
2. Advancement in the identification of common themes in sets of text documents through dimensionality reduction methods.
3. Exploration of new approaches to encoding and decoding using dimensionality reduction methods.

Publications

1. System sciences and informatics: collection of reports of the II scientific and practical conference "System sciences and informatics", December 4–8, 2023, Kyiv. – K., Igor Sikorsky Kyiv Polytechnic Institute, 2023. - p. 273-278.

Keywords. Data reduction, dimensional reduction, matrix transformation, high and low dimensional spaces, data processing and analysis.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

- PCA (Principal Component Analysis) – це метод редукції, який дозволяє зменшити розмірність даних шляхом їх проекції на ортогональні головні компоненти, зберігаючи при цьому максимальну дисперсію.
- LDA (Linear Discriminant Analysis) - це метод редукції, що використовується для визначення лінійних комбінацій ознак, які найкраще розділяють класи в наборі даних.
- ICA (Independent Component Analysis) - це метод редукції, призначений для розкладу змішаних сигналів на незалежні компоненти шляхом максимізації їх статистичної незалежності.
- NMF (Non-negative Matrix Factorization) - метод редукції, який розкладає не від'ємні матриці на добуток двох також не від'ємних матриць.
- SVD (Singular Value Decomposition) - метод редукції, який розкладає матрицю на добуток трьох інших матриць, включаючи матриці з сингулярними значеннями.
- t-SNE (t-distributed Stochastic Neighbor Embedding) - це метод у машинному навчанні, призначений для візуалізації високорозмірних даних шляхом зображення їх точок у просторі.
- TF-IDF (Term Frequency-Inverse Document Frequency) - це метод в обробці природної мови, що оцінює важливість терміну в документі.
- GPU (Graphics Processing Unit) - це високоефективний процесор, спеціалізований на обробці графічних та паралельних обчислень, що знаходить застосування у машинному навчанні та наукових дослідженнях.
- TPU (Tensor Processing Unit) - це спеціалізований процесор, розроблений Google для прискорення обчислень, пов'язаних із штучним інтелектом та глибоким навчанням.

ВСТУП

У сучасну епоху обробка великих даних стала важливою складовою успішної роботи різноманітних галузей господарства та науки. Зростаюча різноманітність доступних даних призвела до виникнення нових завдань та їх можливих рішень в області обробки інформації. Проте з таким розвитком з'явилися труднощі, пов'язані з обробкою великих обсягів даних. Іноді найпотужніші обчислювальні ресурси можуть бути недостатніми для аналізу інформації в реальному часі.

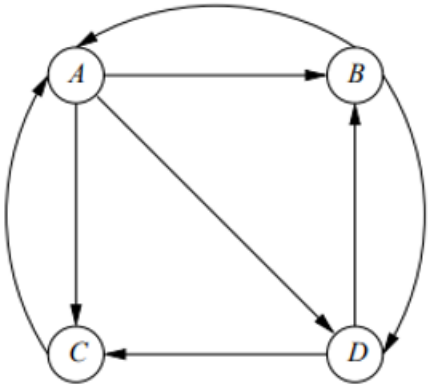
Одним із ключових завдань у сфері обробки великих даних є редукція обсягу інформації. Цей процес включає в себе відбір, фільтрацію та стиснення даних з метою зменшення їх обсягу, збереження лише найважливішої інформації та спрощення подальшого аналізу. Редукція обсягу інформації має на меті підвищення ефективності та швидкості обробки даних, та є ключовим етапом у розв'язанні проблем, пов'язаних з обробкою великих даних.

Магістерська дисертація присвячений дослідженню методів редукції обсягу інформації в системах обробки великих даних. Метою даного дослідження є визначення та оцінка існуючих підходів до зменшення обсягу інформації в обробці даних, що дозволить розпізнати їх роботу та ефективність.

1 ЗАГАЛЬНИЙ ОГЛЯД МЕТОДИК РЕДУКЦІЇ ОБСЯГУ ІНФОРМАЦІЇ

1.1 Поняття зниження розмірності

Багато джерел даних можуть бути представлені як велика матриця. Наприклад матриці переходу для веб-сторінок (рис. 1.1) або матриці переваг для рекомендаційних систем (рис. 1.2). В багатьох таких матрицях дані можна узагальнити, підібравши матрицю близьку до вихідної. Через те, що кількість рядків та стовпців у такій матриці буде менша, робота з нею буде набагато ефективніша та швидшою, ніж з початковою. Процес пошуком таких матриць і є зниженням розмірності.



$$M = \begin{bmatrix} 0 & 1/2 & 1 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix}$$

Рисунок 1.1 – Граф та відповідна матриця переходу веб-сторінок.

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Рисунок 1.2 – Матриця переваг для рекомендаційної системи фільмів.

Іншими словами можна сказати, що зниження розмірності це процес перетворення даних із простору високої розмірності в простір низької розмірності таким чином, щоб представлення низької розмірності зберігало деякі значущі властивості вихідних даних, які близькі до їх початкової розмірності (рис. 1.3).

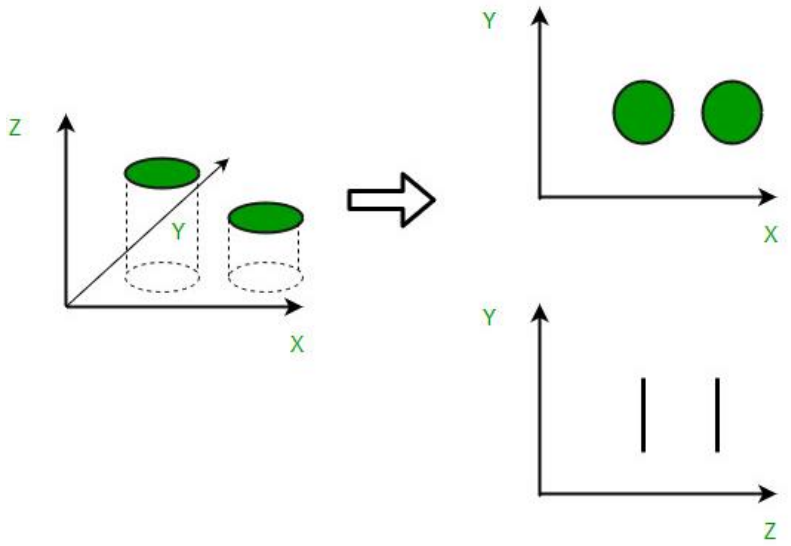


Рисунок 1.3 – Зниження розмірності даних тривимірного простору.

Одним із прикладів зниження розмірності є UV-декомпозиція матриці (рис. 1.4). У цій задачі необхідно розкласти велику матрицю M на дві матриці U та V, добуток яких приблизно дорівнює M. Таким чином у матриці U буде мало стовбців, а в матриці V – рядків, що менше ніж у M, але разом давало більшу частину інформації наявну в M. У випадку відсутності деяких значень у матриці M, такий метод дає можливість передбачення оцінок, наприклад для рекомендаційної системи.

$$\begin{bmatrix} 5 & 2 & 4 & 4 & 3 \\ 3 & 1 & 2 & 4 & 1 \\ 2 & & 3 & 1 & 4 \\ 2 & 5 & 4 & 3 & 5 \\ 4 & 4 & 5 & 4 & \end{bmatrix} = \begin{bmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \\ u_{31} & u_{32} \\ u_{41} & u_{42} \\ u_{51} & u_{52} \end{bmatrix} \times \begin{bmatrix} v_{11} & v_{12} & v_{13} & v_{14} & v_{15} \\ v_{21} & v_{22} & v_{23} & v_{24} & v_{25} \end{bmatrix}$$

Рисунок 1.4 – UV-декомпозиція матриці.

1.2 Метод головних компонент

Метод головних компонент (PCA) є одним із способів зменшити розмірність даних, втративши найменшу кількість інформації. Даний метод використовує ортогональне перетворення множини даних пов’язаних зі зміною у множину змінних без лінійної кореляції, які називаються головними компонентами. Зазвичай використовується два головних компоненти, щоб побудувати дані на площині, тобто двовимірному просторі, і візуально ідентифікувати кластери тісно пов’язаних точок даних (рис. 1.5).

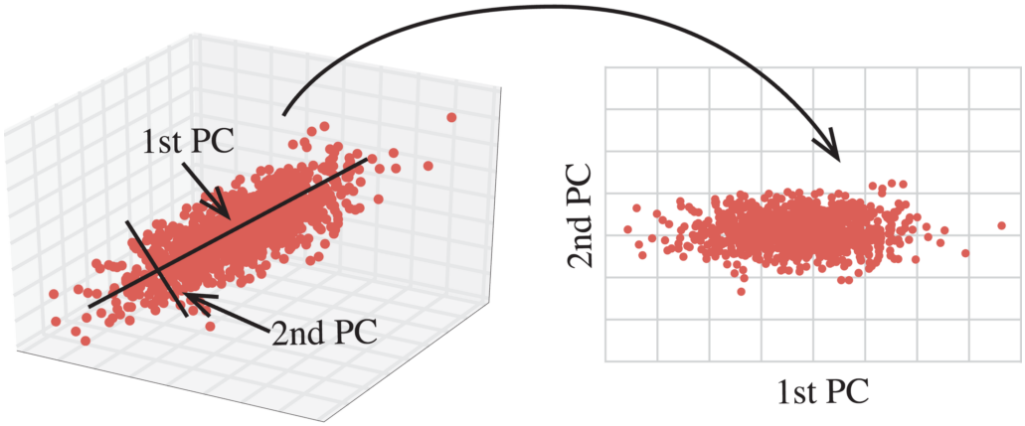


Рисунок 1.5 – Редукція даних методом головних компонент.

1.2.1 Основна концепція метода головних компонент

Ідея методу головних компонент полягає у тому, що для набору даних, що складаються з набору кортежів, представлених точками в багатовимірному просторі, знаходяться напрями, при яких кортежі розміщуються найкраще (рис. 1.6). Кортежі розглядаються як матриця, а матриця її власних векторів, як обертання багатовимірного простору.

Вісь, що відповідає головному власному вектору, є тією, уздовж якої найбільш розкидані точки, тобто уздовж такої осі досягається максимальна дисперсія. Так само вісь, що відповідає другому власному вектору, а саме відповідає другому за величиною власному значенню, є віссю, уздовж якої дисперсія відстаней від першої осі є найбільшою, і так далі (рис. 1.7).

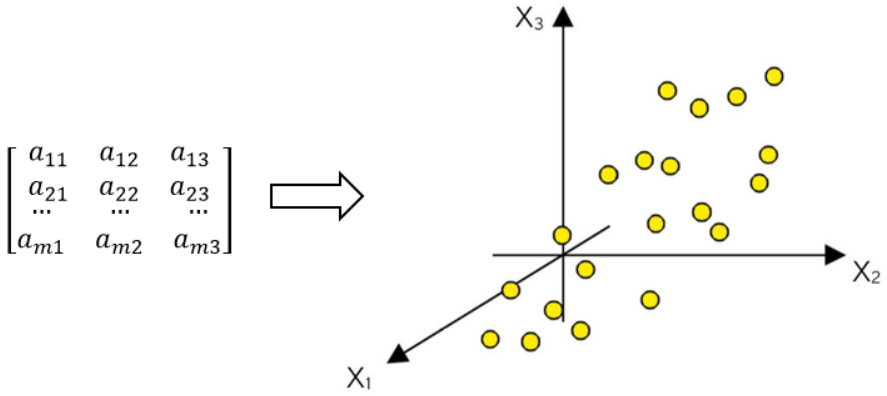


Рисунок 1.6 – Представлення матриці даних у багатовимірному просторі.

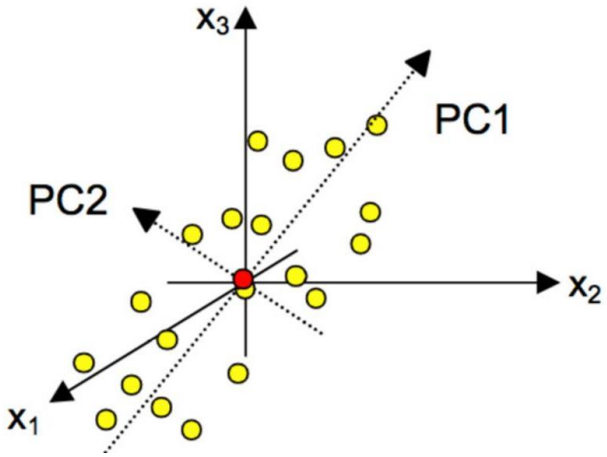


Рисунок 1.7 – Визначення головних компонент PC1 та PC2.

Метод головних компонент можна розглядати як спосіб видобутку даних. Дані, що мають високу розмірність, заміняються проекцією на осі, які відповідають найбільшим власним значенням. Таким чином, вихідні дані апроксимуються даними меншої розмірності та представляються узагальненими (рис. 1.8).

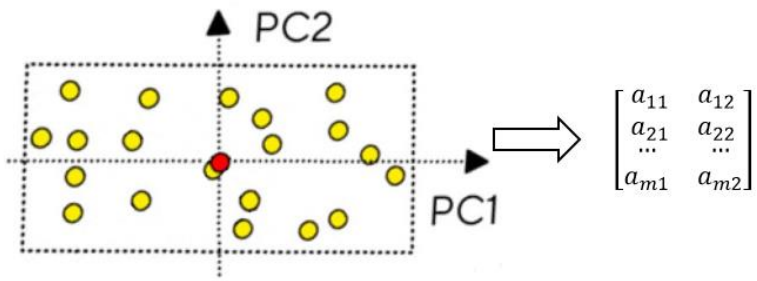


Рисунок 1.8 – Представлення даних у новому просторі методом PCA.

1.2.2 Переваги та недоліки метода головних компонент

Перевагами методу головних компонент є:

- зниження розмірності: допомагає зменшити розмірність даних, перетворюючи їх у новий набір змінних, які фіксують більшу частину відхилень вихідних даних, що робить обчислення більш керованими та покращує візуалізацію;
- виявлення важливих ознак: головні компоненти вказують на те, які змінні найбільше впливають на дані, що допомагає вибирати найбільш інформативні;
- низький вплив шуму: головні компоненти мають найбільшу дисперсію, при якій шум у даних є найменшим, що покращує точність алгоритмів;
- універсальність: даний метод може використовуватися майже у будь яких задачах зниження розмірності та візуалізації даних, а також бути основою інших методів зниження розмірності;
- простота реалізації: алгоритм даного метода є простим у реалізації, зберігаючи свою ефективність при будь якому розмірі матричних даних.

Недоліками методу головних компонент є:

- складна інтерпретація даних: головні компоненти є лінійними комбінаціями вихідних даних, тому не мають зв’язку з оригінальними змінними, що призводить до складної інтерпретації нових даних;
- потребує стандартизації даних: метод має високий вплив від масштабу змінних, тому перед застосуванням алгоритм повинен мати певну стандартизацію даних;
- відсутність зв’язків між даними: даний метод є лінійним, тому він не передбачає виявлення нелінійних залежностей між вихідними значеннями.

1.3 Лінійний розділювальний аналіз

Лінійний розділювальний аналіз (LDA) має на меті побудову та відокремлення одних класів від інших шляхом зниження розмірності. Результати можна використовувати як лінійний класифікатор, або щоб зменшити розміри даних перед їх класифікацією. LDA також дозволяє зручно візуалізувати дані, показуючи розміщення точок по класам та моделюючи різницю між ними (рис. 1.9).

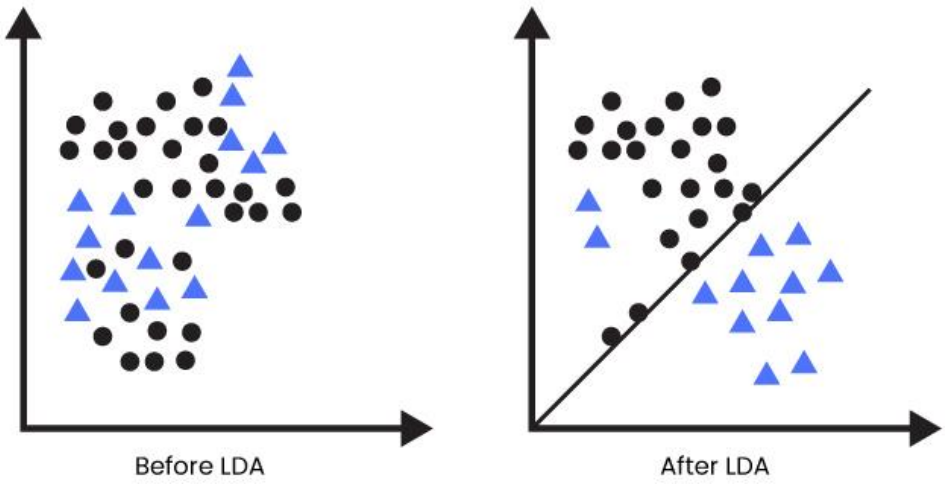


Рисунок 1.9 – Розподіл даних двох класів лінійним розділювальним аналізом.

1.3.1 Основна концепція лінійного розділювального аналізу

Ідея лінійного розділювального аналізу полягає в тому, щоб знайти оптимальний спосіб проектування даних на новий простір так, щоб класи були максимально розділені. Іншими словами, метод спрямований на збереження максимальної дисперсії між класами, одночасно зменшуючи дисперсію всередині класів.

Спочатку обчислюються середні значення та матриці розсіювання для кожного класу, а також матриці розсіювання між класами. Для кожної розрахованої матриці розсіювання розраховуються власні вектори та власні значення. Власні вектори представляють напрямки розподілу даних максимальної дисперсії, а власні значення – цінність напрямків. На обраній

кількості власних векторів будуються проекції даних, що створює новий набір ознак (рис. 1.10). Ці ознаки можна використовувати для аналізу, класифікації та візуалізації даних.

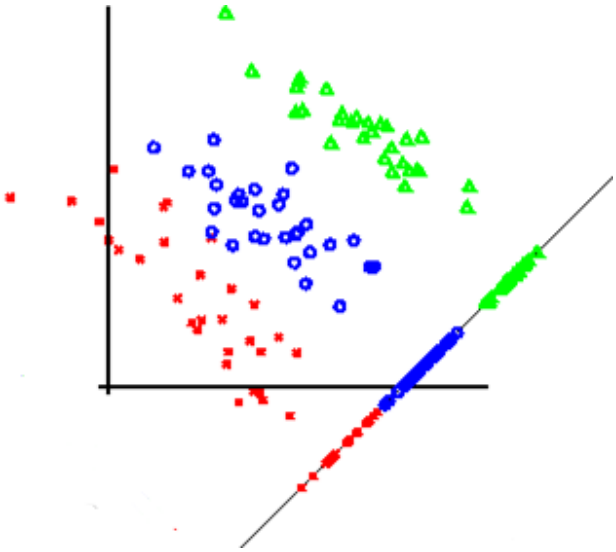


Рисунок 1.10 – Проекція даних на власний вектор у методі лінійного розділювального аналізу.

1.3.2 Переваги та недоліки лінійного розділювального аналізу

Перевагами лінійного розділювального аналізу є:

- ефективність при малих спостереженнях: у випадках, коли кількість ознак більша за кількість спостережень, метод не втрачає своєї ефективності;
- здатність розділу класів: шляхом максимізації дисперсії між класами та мінімізації всередині класу досягається максимальний розподіл між класами даних;
- спрощення класифікації даних: лінійний розділ підвищує точності класифікації даних, за рахунок перетворення даних у простір меншої розмірності;
- інтерпретованість: результати LDA можуть бути легко інтерпретовані, оскільки вони відображають просторове розташування класів.

Недоліками лінійного розділювального аналізу є:

- обмежена застосованість: даний метод розроблений для задач класифікації двох або більше класів, тому він може не підходити для більшості інших завдань аналізу даних, наприклад регресії;
- чутливість до викидів: спостереження, що сильно відрізняються від інших, можуть вплинути на точність результатів, оскільки метод вимагає задану наперед кількість класів;
- припущення щодо розподілу даних: лінійний розділювальний аналіз припускає, що дані визначені та розподілені між кожним класом наперед, інакше втрачається ефективність методу;
- обчислювальна складність: при великій кількість класів та унікальних змінних кожного спостереження метод потребує інтенсивних обчислень, через що ефективність методу зменшується.

1.4 Метод незалежних компонент

Метод незалежних компонент (ICA) використовується для виявлення незалежних компонентів даних у суміші багатьох різних даних. Даний метод є розширенням методу PCA, який поділяє вхідні дані на фактори, які спричиняють спостереження (рис. 1.11).

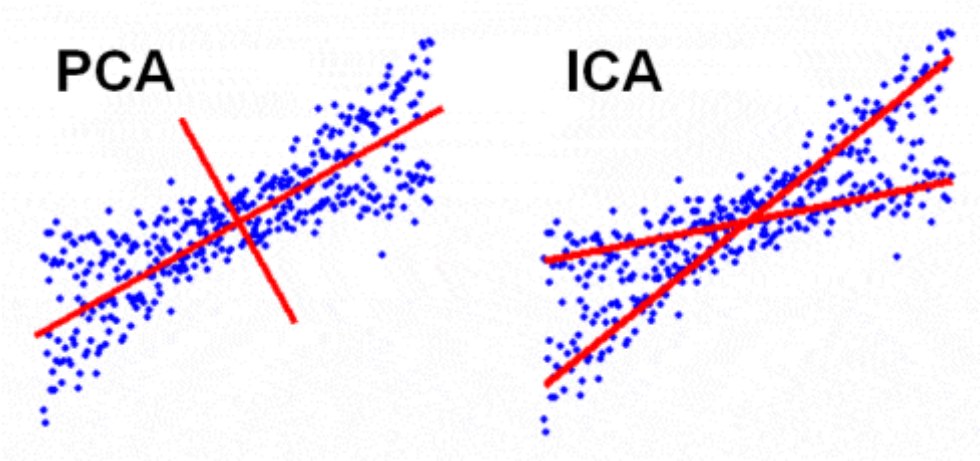


Рисунок 1.11 – Порівняння будови власних векторів методів PCA та ICA.

Метод знижує розмірність даних за рахунок виділення найважливіших незалежних спостережень з суміші даних, відкидаючи менш важливі. Цей метод часто використовується в задачах розкладу або знаходження сигналів з однієї великої суміші таких сигналів (рис. 1.12).

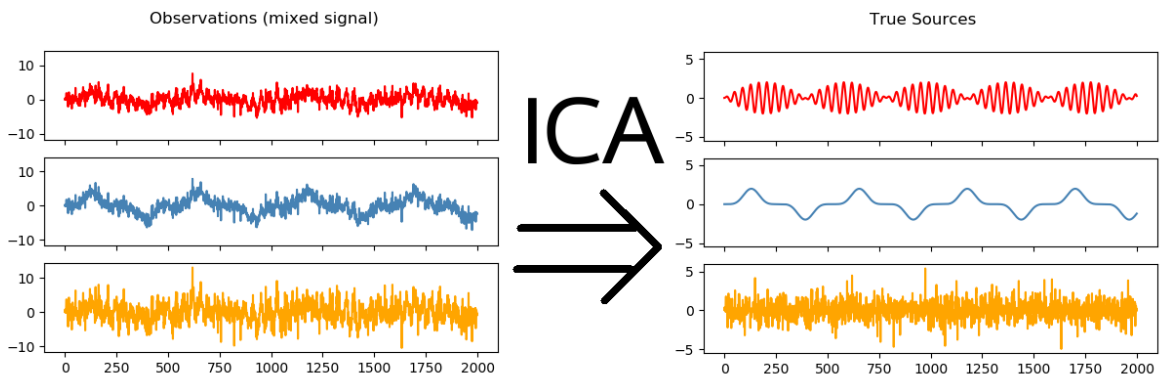


Рисунок 1.12 – Пошук сигналів із суміші сигналів методом ICA.

1.4.1 Основна концепція методу незалежних компонент

Ідея полягає в тому, що виходячи зі суміші сигналів, метод незалежних компонент намагається знайти матрицю, яка максимізує ступінь незалежності між рядками цієї матриці. Дані з різних сенсорів та приладів є стовпцями початкової матриці, а рядки – спостереженнями.

Після центрування, масштабування та нормалізації відбувається процес пошуку незалежних компонент – розрахунку матриці перетворення, яка розділить вихідні дані на незалежні компоненти. Для досягнення цієї мети можуть бути використаними різні алгоритми, а саме алгоритми незалежності на основі статистики.

Використовуючи матрицю перетворення, будуються незалежні компоненти розділом спостережень на окремі сигнали. Отримані незалежні компоненти можуть бути використані для подальшого аналізу або застосування в залежності від конкретної задачі, наприклад видалення шуму або розпізнавання образів.

1.4.2 Переваги та недоліки методу незалежних компонент

Перевагами методу незалежних компонент є:

- визначення незалежних компонент: дозволяє розкрити приховані незалежні компоненти в змішаних сигналах, навіть якщо джерела сигналів невідомі;
- застосування на практиці: метод є дуже ефективним в задачах, коли важлива незалежність даних, а саме в сигнальній обробці аудіосигналів, зображень, нейробіології, видобутку даних тощо;
- дуже низька чутливість до шуму: методу вдається виділяти незалежні джерела даних навіть якщо спостережувані сигнали спотворені шумом.

Недоліками методу незалежних компонент є:

- лінійність: даний метод очікує, що вихідний сигнал є лінійною комбінацією незалежних компонентів, інакше метод є неефективним для зниження розмірності даних;
- умовне зниження розмірності: головним завданням методу це пошук незалежних сигналів, зниження розмірності лише очевидний наслідок знайдених сигналів, тобто можливі випадки, коли зниження розмірності не відбудеться або в цьому не буде необхідність;
- чутливість до гіперпараметрів: вибір гіперпараметрів, таких як кількості незалежних компонент, впливає на продуктивність методу, і є нелегким завданням;
- обчислювальна складність: складність алгоритму зростає зі збільшенням кількості джерел та спостережень, що може потребувати інтенсивних обчислень при роботі з багатовимірними даними.

1.5 Розклад невід’ємних матриць

Розклад невід'ємних матриць (NMF) - це метод аналізу даних і розкладу матриць на декілька додатних складових матриць, який може бути використаний для зниження розмірності і виділення прихованих структур у вихідних даних (рис. 1.13).

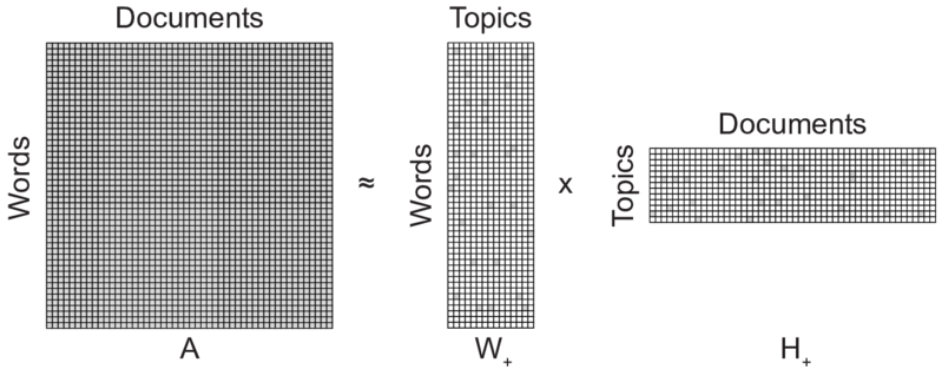


Рисунок 1.13 – Загальний вигляд розкладу невід'ємних матриць.

NMF допомагає знайти структури у вихідних даних шляхом розкладу вихідної матриці на добуток двох невід'ємних матриць меншої розмірності. Одна з цих матриць може розглядатися як набір базових шаблонів або тем, а інша - як коефіцієнти, що вказують на те, як ці шаблони комбінуються для створення вихідних даних.

1.5.1 Основна концепція розкладу невід’ємних матриць

Основна концепція NMF полягає в тому, що вихідні дані можуть бути представлені як комбінація декількох базових складових, де кожна з цих складових є невід'ємною і виражає якусь специфічну структуру або шаблон даних. Вихідна матриця містить рядки спостережень та стовпці ознак, а їх значення обов’язково невід’ємні.

Завданням методу є розклад вихідної матриці на добуток двох матриць, які зазвичай називаються матрицею базових складових та матрицею коефіцієнтів. Матриця базових складових містить базові шаблони, що

вказують на вихідні дані. Матриця коефіцієнтів містить елементи, які вказують, як кожен шаблон з матриці базових складових комбінується, щоб створити спостереження в початковій матриці.

Для пошуку найкращої апроксимації вихідної матриці з добутком матриці базових складових та матриці коефіцієнтів використовуються оптимізаційні методи, такі як градієнтний спуск.

1.5.2 Переваги та недоліки розкладу невід’ємних матриць

Перевагами розкладу невід’ємних матриць є:

- інтерпретованість даних: результати розкладу можуть мати фізичний зміст, що полегшує їх аналіз та подальше використання;
- обмеження негативних значень: метод ефективний в задачах, де від'ємні значення не мають сенсу в контексті самої задачі;
- Практичні застосування: має широкий спектр практичних застосувань, включаючи обробку зображень, аналіз даних, рекомендаційні системи, транспортні задачі тощо;
- генерація тем: метод часто використовується в аналізі тексту та обробці природніх мов для моделювання тем, які ідентифікуються в колекції документів.

Недоліками розкладу невід’ємних матриць є:

- важкість розрахунку: при великих розмірностях матриці збільшується обчислювальна складність розкладу;
- нестійкість: може привести до втрати точності та занадто оптимістичних результатів, що викликано нестійкістю самого методу;
- залежність від початкових шаблонів: має високу чутливість до вибору початкових значень для наступних розрахунків, що також впливає на точність результатів;

- обмеження на тип даних: розклад невід’ємних матриць обмежує тип даних, з якими він працює, виключаючи від'ємні значення та інші типи, які не можуть бути представлені як невід’ємні матриці.

1.6 Сингулярний розклад

Сингулярний розклад (SVD) дає чітке представлення матриці, і у той же час дозволяє відкинути менш важливі частини, залишивши близьке представлення бажаної розмірності. В основі метода лежить розклад вихідної матриці на добуток з трьох матриць, кожна з яких менша за розмірами від вихідної. Зменшення розмірності за допомогою SVD допомагає вирішувати проблеми з обробкою та аналізом великих обсягів даних, використанням менше пам'яті та обчислювальних ресурсів.

1.6.1 Основна концепція сингулярного розкладу

Нехай дано матрицю M, розмірності m x n рангу r. Рядки або стовпці матриці M є лінійно незалежними. Тоді матрицю M можна представити у вигляді добутку матриць U, Σ та V^T (рис. 1.14), з наступними властивостями:

1. U – ортогональна по стовпцях матриця m x r, тобто. всі її стовпці поодинокі вектори та їх попарні скалярні твори дорівнюють 0.
2. V^T – ортогональна по стовпцям транспонована матриця n x r.
3. Σ – діагональна матриця, елементи якої називаються сингулярними значеннями матриці M.

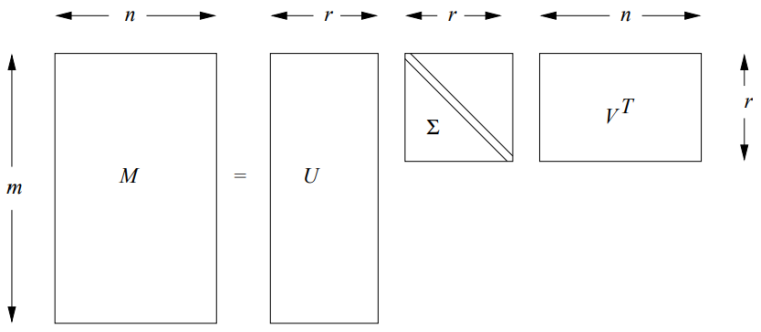


Рисунок 1.14 – Сингулярний розклад матриці M.

При сингулярному розкладі значення r необхідно розглядати, як концепти, в які об’єднуються стовпці або рядки вихідної матриці M. Таким чином матриці U та V^T створюють нові дані для аналізу, а рядки m та стовпці n показують як нові дані пов’язані з концептами r.

Діагональна матриця Σ відображає дисперсію кожного концепту, тобто її важливість у порядку спадання і відповідно до стовпчика матриці U або рядка матриці V^T. Зазвичай бувають дані які, не є корисними, що відображається дисперсією на матриці Σ. Якщо дисперсія дуже мала, то відповідний стовпчик матриці U або рядок матриці V^T можна прибрати, таким чином ще зменшуючи дані.

На рисунку 1.15 зображено сингулярний розклад, у якому останній елемент матриці Σ дуже малий, тому буде доречним прибрати останній стовпець матриці U та рядок матриці V^T, оскільки ці дані не є важливими.

$$\underset{M'}{\begin{bmatrix}1&1&1&0&0\\3&3&3&0&0\\4&4&4&0&0\\5&5&5&0&0\\0&2&0&4&4\\0&0&0&5&5\\0&1&0&2&2\end{bmatrix}} = \underset{U}{\begin{bmatrix}.13&.02&-.01\\.41&.07&-.03\\.55&.09&-.04\\.68&.11&-.05\\.15&-.59&.65\\.07&-.73&-.67\\.07&-.29&.32\end{bmatrix}} \underset{\Sigma}{\begin{bmatrix}12.4&0&0\\0&9.5&0\\0&0&1.3\end{bmatrix}} \underset{V^T}{\begin{bmatrix}.56&.59&.56&.09&.09\\.12&-.02&.12&-.69&-.69\\.40&-.80&.40&.09&.09\end{bmatrix}}$$

Рисунок 1.15 – Приклад сингулярного розкладу з різною дисперсією.

1.6.2 Переваги та недоліки сингулярного розкладу

Перевагами сингулярного розкладу є:

- зниження розмірності: визначення дисперсії кожного рядка та стовпця дозволяє виділяти більш важливу інформацію та прибрати менш важливу;
- сортування даних: значення дисперсії діагональної матриці розміщуються за спаданням, тому результати матриць є також відсортованими;

- спрощення аналізу: відокремлення та сортування важливої інформації робить даний метод дуже зручним при аналізи великих даних;
- зменшення шуму: сингулярний розклад здатен відокремлювати шум з вихідних даних;
- апроксимація матриць: розклад можна використовувати для апроксимації матриць, що допомагає у стисненні даних;
- інверсія матриці: сингулярний розклад може бути використаним у вирішені лінійних систем рівнянь та задачах обчислення псевдооберненої матриці;
- практичне застосування: метод використовується у стиснені комп'ютерних файлів та зображень, а також усунення шумів сигналів.

Недоліками сингулярного розкладу є:

- складна інтерпретація даних: пошук концептів в сингулярному розкладі може бути складною задачею особливо при великій кількості спостережень;
- необхідність в попередній обробці даних: для покращення результатів бажано виконувати попередню обробку вихідних даних, наприклад центрування середнього значення або масштабування;
- розрідженість даних: сингулярний розклад погано працює з розрідженими матрицями;
- складність реалізації: алгоритм працює повільно та потребує багато пам'яті, тому доречно використовувати методи, які беруть за основу сингулярний розклад, але модернізуються під конкретні задачі.

1.7 CUR-декомпозиція

Сингулярний розклад має недолік, коли вихідна матриця сильно розріджена при великій кількості даних. У такому випадку матриці

сингулярного розкладу будуть занадто щільними, а це призведе до того, що виділити концепти буде складно.

Даний недолік вирішує CUR-декомпозиція, яка при розрідженій вихідній матриці буде надавати розрідженими матриці розкладу. Хоча середня матриця Σ навпаки буде щільною, але сама щільність буде мала (рис. 1.16).

Нехай дано матрицю M з m рядками та n стовпцями. Обирається будь яке ціле додатне r – кількість концептів в розкладі. CUR-декомпозицією матриці M називається випадково обрані r стовбців матриці M, які утворюють матрицю C розмірності m x r, та випадково обрані r рядків матриці M, які утворюють матрицю R розмірів r x n.

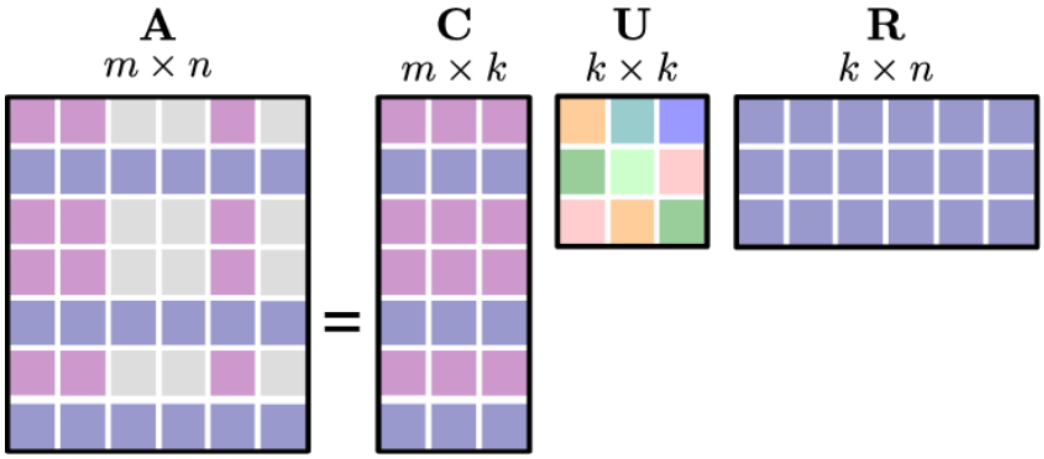


Рисунок 1.16 – CUR-декомпозиція.

1.7.1 Основна концепція CUR-декомпозиції

Хоча рядки та стовпці обираються випадковим способом, необхідно надавати пріоритет більш важливим рядкам та стовпцям. Для такої задачі додатково використовується норма Фробеніуса. Тоді вірогідність вибору рядку або стовпця з більш цінними значеннями більша.

Середня матриця U знаходиться шляхом псевдообернення Мура-Пенроуза матриці W, що формується з елементів матриці M, які знаходяться на перетині обраних стовпців та рядків для матриці C та R.

На рисунку 1.17 представлено приклад розкладу матриці CUR-декомпозицією. Оскільки обране r для декомпозиції невелике, то і точність отриманих даних зменшена. У випадку, коли необхідно отримати більш точні дані, значення r потрібно збільшувати.

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 0 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} 1.54 & 0 \\ 4.63 & 0 \\ 6.17 & 0 \\ 7.72 & 0 \\ 0 & 9.30 \\ 0 & 11.63 \\ 0 & 4.65 \end{bmatrix} \begin{bmatrix} 0 & 1/25 \\ 1/25 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 11.01 & 11.01 \\ 8.99 & 8.99 & 8.99 & 0 & 0 \end{bmatrix}$$

Рисунок 1.17 – Приклад CUR-декомпозиції при r = 2.

1.7.2 Переваги та недоліки CUR-декомпозиції

Перевагами CUR-декомпозиції є:

- вибір важливих рядків та стовпців: надає можливість самостійно визначати, які рядки та стовпці є більш важливими для дослідження, використовуючи різні алгоритми;
- ефективність обчислень: обчислення CUR-декомпозиції більш прості у реалізації;
- зменшене споживання пам'яті: алгоритми CUR-декомпозиції використовують менше пам’яті, оскільки зберігаються та оброблюються лише обрані стовпці та рядки, а не уся матриця;
- розрідженість даних: CUR-декомпозиції ефективно працює з розрідженими матрицями.

Недоліками CUR-декомпозиції є:

- втрата інформації: метод видаляє частину інформації з вихідної матриці до початку перетворень, що може вплинути на правдивість результатів;

- залежність від вибору: результати CUR-декомпозиції можуть змінюватися в залежності від обраних стовпців та рядків, що змушує визначати дійсність таких результатів;
- точність залежить від розмірності: чим менше стовпців та рядків буде обрано з вихідної матриці, тим менша точність буде в отриманих матрицях.

1.8 t-розподілене стохастичне вбудовування сусідів

t-розподілене стохастичне вбудовування сусідів (t-SNE) – це метод для візуалізації даних, у якому дані високої розмірності перетворюються у низькорозмірний простір, зазвичай у двовимірний. На меті даного метода є збереження та візуалізація взаємозв'язків між даними, тобто перетворення простору так, щоб схожі дані залишалися близькими, а відмінні дані відділялися.

Метод t-SNE широко використовується для візуалізації даних, а також для перегляду та аналізу складних високорозмірних даних, таких як зображення, текстові або генетичні дані. Візуалізація, отримана через t-SNE, допомагає визначити групи або структури в даних, які важко виявити в початковому високорозмірному просторі.

1.8.1 Основна концепція t-розподіленого стохастичного вбудовування сусідів

Суть t-SNE полягає в тому, щоб відобразити набір даних високої розмірності в низькорозмірний простір, зберігаючи при цьому структурні відносини між даними якнайточніше. Досягається це пошуком та мінімізацією розривів схожих точок низькорозмірного простору.

Спочатку для кожної точки високорозмірного простору визначається ймовірність того, що одна точка вибере іншу як сусіда. Для розрахунку такої ймовірності використовується гаусівське ядро, що враховує схожість між

точками. Далі за аналогією розраховуються ймовірності для точок низьковимірного простору.

Для мінімізації розривів між схожими точками необхідно знайти відображення низької розмірності, яке мінімізує розриви між ймовірностями схожості в обох просторах. Це досягається за допомогою оптимізаційного методу, зазвичай градієнтного спуску.

Використання t-розподілення дозволяє виміряти схожість між точками в низьковимірному просторі. t-розподілення є ширшим за нормальний розподіл і має тенденцію групувати точки в компактні кластери, що робить його більш чутливим до віддалених точок. За результатами t-розподілу отримується відображення низького виміру, де схожі дані залишаються близькими одна до одної, а відмінності відділяються. Це дозволяє зручно візуалізувати та дослідити структуру даних в низьковимірному просторі (рис. 1.18).

Рисунок 1.18 – Приклад візуалізації даних методом t-SNE.

1.8.2 Переваги та недоліки t-розподіленого стохастичного вбудовування сусідів

Перевагами t-SNE є:

- збереження локальної структури: головна мета t-SNE визначати подібні точки та розміщувати їх якнайближче одна до одної;
- нелінійність: метод може розпізнавати нелінійні зв’язки між даними;
- ефективна візуалізація: метод найчастіше використовують для візуалізації кластерів даних, які в подальшому зручно аналізувати;
- ефективне дослідження результатів: візуалізація даних за допомогою t-SNE надає чітке представлення про взаємозв’язки між даними, що дозволяє виконувати над ними подальші дослідження.

Недоліками t-SNE є:

- обчислювальна інтенсивність: алгоритми оптимізації в методі t-SNE є ітеративними та трудомісткими, що робить обчислення дуже повільними;
- стохастичний характер: наявність стохастичного елементу в процесах оптимізації може надавати різні результати при кожному запуску алгоритму, а також важко відтворювати, тому є потреба у повторних запусках;
- прокляття розмірності: в процесі зменшення розмірності загальна структура даних втрачається та не оброблюється, що може привести до переповненості в представленнях меншої розмірності та втрати даних;
- підбір гіперпараметрів: вибір гіперпараметрів сильно впливає на продуктивність методу та його результати, що потребує їх належного налаштування для отримання оптимальних результатів.

1.9 Висновки до розділу

У цьому розділі було розглянуто основні методики редукції обсягу інформації, а саме метод головних компонент, лінійний розділювальний аналіз, метод незалежних компонент, розклад невід'ємних матриць, сингулярний розклад, CUR-декомпозиція та t-розподілене стохастичне вбудовування сусідів. Було визначено їх принципи дії, основні ідеї, сильні та слабкі сторони їх використання, за якими можна визначити ефективність кожного методу в тому чи іншому завдані.

У випадку розподілу даних за класами або незалежними зв'язками та їх візуалізації доречно буде використовувати лінійний розділювальний аналіз, метод незалежних компонент та t-розподілене стохастичне вбудовування сусідів. Дані методи є ефективними у дослідженнях та аналізу взаємозв'язків між даними, проте їх ефективність в редукції даних невисока, через часткову або повну втрату значень загальної інформації про спостереження.

Розклад невід'ємних матриць є ефективним у знижені розмірності, але його реалізація на практиці занадто складна, а розрахунки є нестійкими, що впливає на точність досліджуваних даних.

Метод незалежних компонент, сингулярний розклад та CUR-декомпозиція є найкращим вибором для найбільш ефективної редукції обсягу інформації в системах обробки великих даних, оскільки їх технології направлені саме на зниження розмірності матриць зі збереженням загальної цінності отриманих даних відповідно до кожного спостереження. Реалізація та використання таких методик не є важкою та легко модернізувати під різні завдання, а відсутність пошуку зв'язків між даними дає найбільшу ефективність перетворення даних в багатовимірних просторах. Тому ці методи будуть використовуватися у подальших розділах даної роботи.

2 МАТЕМАТИЧНІ МОДЕЛІ МЕТОДУ ГОЛОВНИХ КОМПОНЕНТ, СИНГУЛЯРНОГО РОЗКЛАДУ ТА CUR-ДЕКОМПОЗИЦІЇ

2.1 Математична модель методу головних компонент

Для виконання редукції методом головних компонент необхідно виконати такі кроки: стандартизація даних, розрахунок коваріаційної матриці, обчислення власних векторів та власних значень, вибір головних компонент, побудова нової матриці на осях головних компонент.

Стандартизація даних необхідна для усунення масштабних різниць між ознаками, забезпечення ортогональності головних компонент та зменшення впливу шуму на данні. Розраховується стандартизована матриця за формулою:

$$X_{\text{станд}} = \frac{X - X_{\text{середнє}}}{X_{\text{відхилення}}} \tag{2.1}$$

де X – задана матриця;

$X_{\text{середнє}}$ – середнє значення матриці X;

$X_{\text{відхилення}}$ – середньоквадратичне відхилення матриці X.

Формули для розрахунку $X_{\text{середнє}}$ та $X_{\text{відхилення}}$:

$$X_{\text{середнє}} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{2.2}$$

$$X_{\text{відхилення}} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - X_{\text{середнє}})^2} \tag{2.3}$$

де n – кількість елементів матриці X;

x_i – відповідний елемент матриці X.

Коваріаційна матриця використовується для наступного визначення власних векторів і будується наступним чином:

$$X_{\text{ковар}} = \begin{bmatrix} cov(X_1, X_1) & cov(X_1, X_2) & \cdots & cov(X_1, X_m) \\ cov(X_2, X_1) & cov(X_2, X_2) & \dots & cov(X_2, X_m) \\ \vdots & \vdots & \ddots & \vdots \\ cov(X_m, X_1) & cov(X_m, X_2) & \dots & cov(X_m, X_m) \end{bmatrix} \tag{2.4}$$

$$cov(X_i, X_j) = \frac{1}{n}\sum_{k=1}^{n}((x_{ik} - X_{i\ \text{середнє}})(x_{jk} - X_{j\ \text{середнє}})) \quad (2.5)$$

де m – кількість вектор-стовпців матриці X;

X_i, X_j – відповідні вектор-стовпці матриці X;

n - кількість елементів вектор-стовпців X_i, X_j;

x_{ik}, x_{jk} – відповідні елементи вектор-стовпців X_i, X_j.

З коваріаційної матриці необхідно визначити власні значення та власні вектори. Власні значення отримуються як корені рівняння:

$$\Delta(X_{\text{ковар}} - \lambda I) = 0 \quad (2.6)$$

де λ – власні значення;

I – одинична матриця.

Власні вектори визначаються для кожного власного значення з рівняння:

$$(X_{\text{ковар}} - \lambda I)\upsilon = 0 \quad (2.7)$$

де υ – власний вектор відповідний власному значенню λ.

З власних векторів обираються головні компоненти. Головними компонентами стають вектори, які мають найбільші власні значення, оскільки вони є найважливішими.

Нова зменшена матриця будується з формули:

$$X_{\text{зменш}} = X_{\text{станд}}X_{\text{гк}} \quad (2.8)$$

де $X_{\text{гк}}$ – сформована матриця головних компонент.

Розмірність нової матриці буде залежати від кількості обраних головних компонент. Чим більша їх кількість, тим більша точність отриманих значень.

2.1.1 Приклад редукції методом PCA

Нехай дано матрицю X, яка представлена на рисунку 2.1.

Матриця містить чотири стовпця, які є особливостями, та п’ять рядків, які є змінними. Стандартизовані дані будуть мати вигляд представлений на рисунку 2.2.

F1	F2	F3	F4
1	5	3	1
4	2	6	3
1	4	3	2
4	4	1	1
5	5	2	3

Рисунок 2.1 – Вхідна матриця X.

F1	F2	F3	F4
-1.0695	0.8196	0	-1
0.5347	-1.6393	1.6042	1
-1.0695	0	0	0
0.5347	0	-1.0695	-1
1.0695	0.8196	-0.5347	1

Рисунок 2.2 – Стандартизована матриця X.

На рисунку 2.3 показано розраховану коваріаційну матрицю.

	F1	F2	F3	F4
F1	0.78	-0.7586	-0.055	0.424
F2	-0.7586	0.78	-0.607	-0.326
F3	-0.055	-0.607	0.78	0.426
F4	0.424	-0.326	0.426	0.78

Рисунок 2.3 – Коварційна матриця.

Далі розраховуються власні значення та власні вектори (рис. 2.4, рис. 2.5).

$$\lambda = 2.11691\,,\ 0.855413\,,\ 0.481689\,,\ 0.334007$$

Рисунок 2.4 – Власні значення розміщенні по спаданню.

E1	E2	E3	E4
0.515514	–0.623012	0.0349815	–0.587262
-0.616625	0.113105	0.452326	–0.634336
0.399314	0.744256	–0.280906	–0.455767
0.441098	0.212477	0.845736	0.212173

Рисунок 2.5 – Розраховані власні вектори відповідно до λ.

Обираються два головних компонента (рис. 2.6):.

E1	E2
0.515514	–0.623012
-0.616625	0.113105
0.399314	0.744256
0.441098	0.212477

Рисунок 2.6 – Обрані головні компоненти.

Та розраховується нова зменшена матриця з стандартизованих даних (рис. 2.7).

0.4268066978	0.00116114000000012
-0.4234920968	-0.39182856
-0.551342223	-0.7959219
0.4652040128	0.89996329
0.0827279480000002	0.28653037

Рисунок 2.7 – Результат редукції методом PCA.

Таким чином методом головних компонент дані четвертого виміру представлено в другій розмірності, які легше візуалізувати та можливо дослідити.

2.2 Математична модель сингулярного розкладу

Нехай дано матрицю X з m рядками та n стовпцями (m ≥ n). Результатом редукції сингулярним розкладом матриці X є матриці U, Σ та V^T такі, що X = $U\Sigma V^T$. Для цього необхідно: обчислити власні значення, розрахувати сингулярні значення та побудувати діагональну матрицю Σ, розрахувати власні вектори для ортогональних матриць U та V^T.

Власні значення та власні вектори розраховуються з матриці Y, яка є добутком XX^T.

Тоді власні значення λ_1, $\lambda_2 \ldots \lambda_p$ є розв'язками рівняння:

$$\Delta(Y - \lambda I) = 0 \tag{2.9}$$

де I – одинична матриця.

Власні значення розташовуються у порядку спадання для збереження найбільш важливих даних. З ненульових власних значень розраховуються сингулярні значення за формулою:

$$\sigma_i = \sqrt{\lambda_i}, \qquad i = 1, 2 \ldots p \tag{2.10}$$

З сингулярних значень формується діагональна матриця Σ, елементи якої розташовані у порядку спадання. Власні вектори, які є стовпцями матриці U, відповідно до кожного власного значення знаходяться з рівняння:

$$(Y - \lambda_i I)u_i = 0, \qquad i = 1, 2 \ldots p \tag{2.11}$$

Транспонована матриця V^T отримується з матриці V, стовпці якої є векторами υ_1, $\upsilon_2 \ldots \upsilon_p$ і розраховуються за формулою:

$$\upsilon_i = \frac{1}{\sigma_i} X^T u_i, \qquad i = 1, 2 \ldots p \tag{2.12}$$

У випадку, якщо у матриці X кількість рядків менша за кількість стовпців, то Y буде добутком X^TX, а отримані власні вектори з рівняння 10 будуть стовпцями матриці V. Вектор-стовпці матриці U будуть розраховуватися за формулою:

$$u_i = \frac{1}{\sigma_i} X v_i, \qquad i = 1, 2 \dots p \tag{2.13}$$

2.2.1 Приклад редукції SVD

Нехай дано матрицю X, яка представлена на рисунку 2.8 разом з добутком XX^T.

$$X = \begin{bmatrix} 3 & 2 \\ 2 & 3 \\ 2 & -2 \end{bmatrix}$$

$$Y = \begin{bmatrix} 3 & 2 \\ 2 & 3 \\ 2 & -2 \end{bmatrix} \cdot \begin{bmatrix} 3 & 2 & 2 \\ 2 & 3 & -2 \end{bmatrix} = \begin{bmatrix} 13 & 12 & 2 \\ 12 & 13 & -2 \\ 2 & -2 & 8 \end{bmatrix}$$

Рисунок 2.8 – Вхідна матриця X та добуток XX^T.

Власними значеннями матриці Y будуть $\lambda_1 = 25$, , $\lambda_2 = 9$ та $\lambda_3 = 0$. А сингулярними значеннями $\sigma_1 = 5$ та $\sigma_2 = 3$. Будується матриця Σ (рис. 2.9).

$$\Sigma = \begin{bmatrix} 5 & 0 \\ 0 & 3 \\ 0 & 0 \end{bmatrix}$$

Рисунок 2.9 – Знайдена матриця Σ.

Після розрахунку власних векторів, будується матриця U (рис. 2.10).

$$U = \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{6} & -\frac{2}{3} \\ \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{6} & \frac{2}{3} \\ 0 & \frac{2\sqrt{2}}{3} & \frac{1}{3} \end{bmatrix}$$

Рисунок 2.10 – Знайдена матриця U.

Далі розраховуються вектор-стовпці для матриці V з її транспонуванням (рис. 2.11).

$$v_1 = \frac{1}{\sigma_1} \cdot \begin{bmatrix} 3 & 2 \\ 2 & 3 \\ 2 & -2 \end{bmatrix}^T \cdot u_1 = \frac{1}{5} \cdot \begin{bmatrix} 3 & 2 & 2 \\ 2 & 3 & -2 \end{bmatrix} \cdot \begin{bmatrix} \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{bmatrix}$$

$$v_2 = \frac{1}{\sigma_2} \cdot \begin{bmatrix} 3 & 2 \\ 2 & 3 \\ 2 & -2 \end{bmatrix}^T \cdot u_2 = \frac{1}{3} \cdot \begin{bmatrix} 3 & 2 & 2 \\ 2 & 3 & -2 \end{bmatrix} \cdot \begin{bmatrix} \frac{\sqrt{2}}{6} \\ -\frac{\sqrt{2}}{6} \\ \frac{2\sqrt{2}}{3} \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} \end{bmatrix}$$

$$V = \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \end{bmatrix} \qquad V^T = \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \end{bmatrix}$$

Рисунок 2.11 – Розрахунок вектор-стовпці матриці V та побудова матриці V^T.

Таким чином за допомогою сингулярного розкладу матриці X отримано матриці U, Σ та V^T, які тепер можна дослідити для визначення нових властивостей.

2.3 Математична модель CUR-декомпозиції

Нехай дано матрицю X з m рядками та n стовпцями. Результатом CUR-декомпозиції матриці X є матриці менших розмірності C, U та R такі, що X = CUR. Для цього необхідно: сформулювати матриці C та R, визначити матрицю перетину стовпців C з рядками R з наступним сингулярним розкладом та розрахувати матрицю U.

На початку обирається ранг r матриці U. Тоді матриця C формується з випадково обраної множини r стовпців X, а R – випадково обраної множини r рядків X. Для того, щоб підвищити вірогідність вибору важливих рядків та стовпців необхідно використовувати норму Фробеніуса. Для цього розраховується квадрат норми Фробеніуса матриці:

$$f = \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij}^2 \qquad (2.14)$$

Таким чином вірогідність вибору рядку буде:

$$p_i = \frac{1}{f}\sum_{j=1}^{n} x_{ij}^2, \qquad i = 1, 2 \dots m \tag{2.15}$$

А вірогідність вибору стовпця буде:

$$q_j = \frac{1}{f}\sum_{i=1}^{m} x_{ij}^2, \qquad j = 1, 2 \dots n \tag{2.16}$$

Обрані стовпці та рядки необхідно масштабувати. Для матриці C:

$$c_{ij} = \frac{x_{ij}}{\sqrt{rq_j}}, \qquad j = 1, 2 \dots r, \qquad i = 1, 2 \dots m \tag{2.17}$$

Для матриці R:

$$r_{ij} = \frac{x_{ij}}{\sqrt{rp_i}}, \qquad j = 1, 2 \dots p, \qquad i = 1, 2 \dots r \tag{2.18}$$

Нехай W - матриця розмірності r x r, сформована перетином обраних стовпців і рядків для матриць C та R, тобто елемент на перетині рядка i і стовпця j в матриці W дорівнює елементу X, стовпець якого співпадає з j-м стовпцем C, а рядок - з i-м рядком R.

Далі розраховується сингулярний розклад матриці W, таким чином отримавши матриці Y, Σ та X^T, та зберігаючи умову $W = Y\Sigma X^T$

Обчислюється матриця Σ^+, яка є псевдооберненою матрицею Мура-Пенроуза для діагональної матриці Σ. Іншими словами, якщо i-й діагональний елемент Σ є ненульовим, то він замінюється на оберенний елемент. Якщо ж i-й елемент дорівнює 0, то він залишається без змін.

Тоді матриця U розраховується за формулою:

$$U = Y(\Sigma^+)^2 X^T \tag{2.19}$$

2.3.1 Приклад CUR-декомпозиції

Нехай дано матрицю M зображену на рисунку 2.12.

1	1	1	0	0
3	3	3	0	0
4	4	4	0	0
5	5	5	0	0
0	0	0	4	4
0	0	0	5	5
0	0	0	2	2

Рисунок 2.12 – Матриця X для CUR-декомпозиції.

Нехай для CUR-декомпозиції r = 2. За нормою Фробеніуса вірогідність вибору перших трьох стовпців буде 0.210, а двох останніх 0.185. Вірогідності вибору рядків дорівнюватимуть: 0.012, 0.111, 0.198, 0.309, 0.132, 0.206, 0.033.

Запропонуємо, що навмання для матриці C було обрано другий та четвертий стовпці, а для R – четвертий та шостий рядки. Тоді в результаті масштабування будуть отримані матриці зображені на рисунку 2.13.

$$C = \begin{bmatrix} 1.54 & 0 \\ 4.63 & 0 \\ 6.17 & 0 \\ 7.72 & 0 \\ 0 & 9.30 \\ 0 & 11.63 \\ 0 & 4.65 \end{bmatrix} \quad R = \begin{bmatrix} 0 & 0 & 0 & 11.01 & 11.01 \\ 8.99 & 8.99 & 8.99 & 0 & 0 \end{bmatrix}$$

Рисунок 2.13 – Знайдені матриці C та R.

Перетином обраних стовпців і рядків для матриць C та R будується матриця W та виконується її сингулярний розклад (рис. 2.14).

$$W = \begin{bmatrix} 0 & 5 \\ 5 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 5 & 0 \\ 0 & 5 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

Рисунок 2.14 – Розрахунок матриці W.

Далі розраховується матриця Σ^+ та матриця U (рис. 2.15).

$$U = Y(\Sigma^+)^2 X^T = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1/5 & 0 \\ 0 & 1/5 \end{bmatrix}^2 \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1/25 \\ 1/25 & 0 \end{bmatrix}$$

Рисунок 2.15 – Розрахунок матриці U.

Таким чином за допомогою CUR-декомпозиції матриці M отримано матриці C, U та R, які тепер можна дослідити для визначення нових властивостей.

2.4 Висновки до розділу

В цьому розділі були детально розглянуті математичні моделі обраних для дослідження методів редукції, зокрема методу головних компонент, сингулярного розкладу та CUR-декомпозиції. Було подано опис та наведено формули для розрахунків зменшених матриць для кожного з вищезазначених методів. Крім того, наведено приклади використання кожного методу на невеликих матрицях. Це дає змогу точно зрозуміти принцип розрахунку кожного з методів.

3 ОПИС ЗАДАЧ ДОСЛІДЖЕННЯ МЕТОДІВ РЕДУКЦІЇ ДАНИХ

3.1 Задача 1: редукція даних в рекомендаційних системах

Рекомендаційні системи - це тип програм, алгоритмів або систем, які надають користувачам персоналізовані рекомендації щодо об'єктів або послуг. Ці об'єкти можуть включати товари, фільми, музику, книги, статті, соціальні контенти та багато іншого. Рекомендаційні системи спрощують вибір для користувачів, допомагаючи їм знаходити та вибирати об'єкти, які ймовірно їх зацікавлять, на основі їхніх попередніх дій, вподобань або характеристик.

Основні цілі рекомендаційних систем включають:

1. Підвищення рівня залучення користувачів. Забезпечення персоналізованих рекомендацій сприяє активнішій участі користувачів, оскільки вони знаходять цікаві об'єкти, які вони можуть спробувати або придбати.
2. Збільшення задоволення користувачів. Рекомендаційні системи допомагають знаходити об'єкти, які відповідають особистим вподобанням і інтересам користувача. Це призводить до покращення загального задоволення користувачів від використання платформи або послуги.
3. Збільшення конверсії та продажів. У випадку електронної комерції, рекомендаційні системи можуть сприяти збільшенню конверсії, оскільки вони допомагають користувачам знаходити товари, які відповідають їхнім потребам, що може призвести до зростання продажів.
4. Зменшення інформаційного перевантаження. Велика кількість доступної інформації може призвести до інформаційного перевантаження. Рекомендаційні системи допомагають користувачам зорієнтуватися в об'ємах даних і швидко знайти те, що їх цікавить.
5. Покращення участі в спільноті. В соціальних мережах та форумах рекомендаційні системи можуть сприяти формуванню більш активних

та зацікавлених спільнот, оскільки користувачі знаходять спільні інтереси та контент, що їх цікавить.

6. Збільшення часу проведеного на платформі. Рекомендаційні системи можуть підтримати збільшення тривалості сесій користувачів, оскільки вони допомагають уникнути затримок у пошуку і забезпечують швидкий доступ до релевантного контенту.

3.1.1 Опис задачі та діаграма діяльності рекомендаційної системи

Дослідити роботу PCA, SVD та CUR-декомпозиції у рекомендаційній системі. Для цього необхідно розробити алгоритм рекомендаційної системи для готового набору даних.

Алгоритм включатиме:

1. Перетворення вхідних даних у зручний формат для роботи з методами редукції та рекомендаційною системою.
2. Використання методу редукції на перетворених вхідних даних.
3. Побудова матриці кореляції для визначення найбільш підходящих товарів для кожного товару в наборі даних.

В результаті дослідження буде отримано графіки залежності кількості рекомендованих товарів для одного товару від розмірності матриці розкладу: для PCA – це кількість компонент, для SVD – кількість сингулярних значень, для CUR-декомпозиції – ранг матриці U.

На рисунку 3.1 зображено UML діаграму діяльності, яка коротко демонструє порядок дій для виконання поставленої задачі, а саме визначення списку рекомендованих товарів для конкретного товару, використовуючи методи редукції.

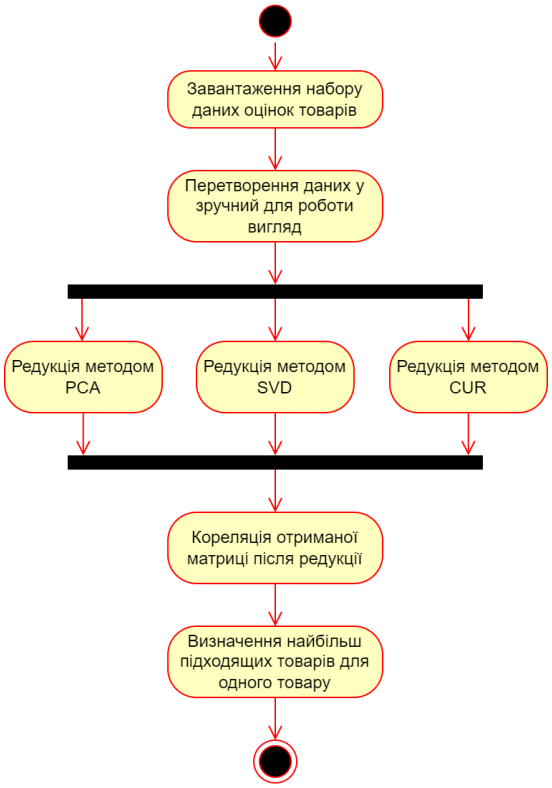


Рисунок 3.1 – Діаграма діяльності рекомендаційної системи з редукцією даних.

3.2 Задача 2: редукція даних при кластеризації текстових документів

Кластеризація текстових документів - це процес групування схожих текстів у спільні категорії чи кластери на основі їхнього змісту. Цей метод дозволяє автоматично організувати велику кількість текстів за схожістю тем, термінів чи інших характеристик.

Основними цілями кластеризація текстових документів є:

1. Організація інформації. Кластеризація дозволяє організувати великий об'єм текстової інформації в логічно зв'язані групи, що полегшує навігацію і пошук інформації.
2. Аналіз тематики. Визначення тематичних груп текстів допомагає розуміти, які теми або концепції представлені в документах.
3. Компресія інформації. Кластеризація може слугувати методом компресії інформації, оскільки вона замінює групою представників

кожен кластер, що дозволяє зменшити об'єм даних для подальшого аналізу.

4. Виявлення зв'язків і залежностей. Допомагає виявляти зв'язки та залежності між текстами, що може бути корисним для виявлення нових інсайтів або аналізу зв'язків у великих колекціях документів.
5. Пошук та рекомендації. Використовуючи результати кластеризації, можна покращити системи пошуку та рекомендацій, дозволяючи користувачам знаходити та отримувати більш специфічні та цікаві матеріали.
6. Виявлення аномалій. Знаходження кластерів, які містять аномалії або виокремлені текстові документи, може вказувати на невизначені аспекти або важливі відхилення в даних.

3.2.1 Опис задачі та діаграма діяльності кластеризації текстових документів

Дослідити роботу PCA, SVD та CUR-декомпозиції при кластеризації текстових документів. Для цього необхідно розробити алгоритм кластеризації для готового набору текстових документів.

Алгоритм включатиме:

1. Векторизація вхідного набору текстових документів методом TF-IDF для роботи з методами редукції та кластеризації.
2. Використання методу редукції на векторизованому наборі текстових даних.
3. Кластеризація сформованих даних методом k-середніх.

В результаті дослідження буде отримано графіки залежностей кількості статей в кожному кластерів від розмірності матриці розкладу: для PCA – це кількість компонент, для SVD – кількість сингулярних значень, для CUR-декомпозиції – ранг матриці U. Додатково визначається середнє абсолютне відхилення отриманого графіку для кожного методу. Також буде отримано візуалізацію самих кластерів для визначення якості пошуку спільних тем.

На рисунку 3.2 зображено UML діаграму діяльності, яка коротко демонструє порядок дій для виконання поставленої задачі, а саме побудову кластерів текстових документів, накладаючи на них методи редукції.

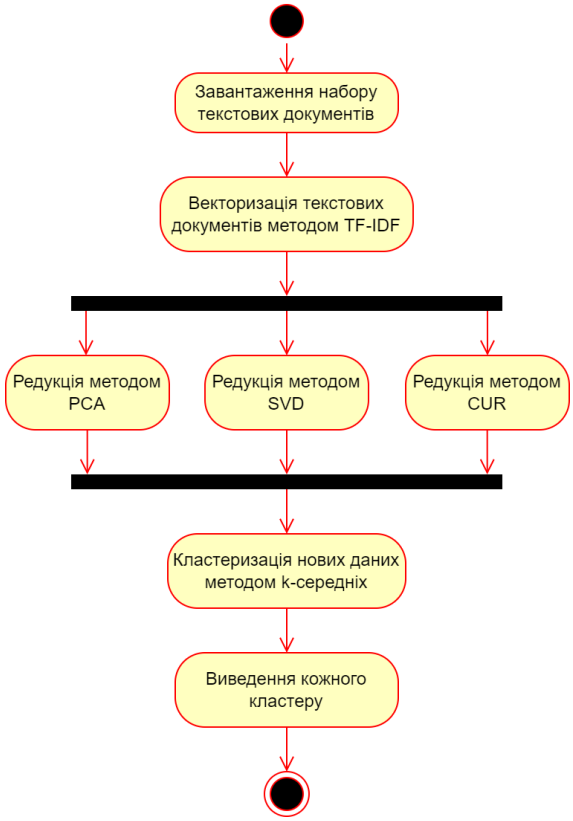


Рисунок 3.2 – Діаграма діяльності кластеризації з редукцією даних.

3.3 Задача 3: редукція даних при кодуванні та декодуванні зображень

Кодування та декодування зображень - це процеси перетворення візуальної інформації з формату, зрозумілого для людей, у формат, придатний для зберігання, передавання або обробки за допомогою комп'ютерних систем.

Основними цілями кодування та декодування зображень є:

1. Збереження інформації. Кодування зображень дозволяє зберегти велику кількість візуальної інформації у компактному форматі для подальшого зберігання або передавання.
2. Стиснення даних. Кодування може включати методи стиснення для зменшення розміру файлу без втрати важливої інформації. Це

особливо важливо при передачі зображень через мережу або зберіганні на пристроях з обмеженим обсягом пам'яті.

3. Захист від втрат. Деякі методи кодування можуть дозволяти зберегти зображення без втрати якості, що важливо, наприклад, при архівуванні важливих зображень.
4. Відтворення оригіналу. Основною метою декодування є відтворення оригінального зображення з закодованої форми зберігання або передавання.
5. Відновлення якості. При втратному стисненні декодування повинно відновлювати якість зображення настільки, наскільки це можливо, з урахуванням втрат, які відбулися під час кодування.
6. Обробка та аналіз. Декодування може бути використано для подальшої обробки та аналізу зображення, такої як витягання різних ознак, виявлення об'єктів або виконання інших завдань комп'ютерного зору.

3.3.1 Опис задачі та діаграма діяльності кодування та декодування зображень

Дослідити роботу PCA, SVD та CUR-декомпозиції у кодуванні та декодуванні зображень. Для цього необхідно розробити алгоритм кодування та декодування матриць каналів кольорового зображення.

Алгоритм включатиме:

1. Розклад кольорового зображення на три матриці.
2. Виконання методу редукції на кожній матриці.
3. Відновлення матриць після їх редукції та об’єднання для порівняння з вихідним зображенням.

В результаті дослідження буде отримано графіки залежності схожості відновленого зображення з оригінальним від розмірності матриці розкладу: для PCA – це кількість компонент, для SVD – кількість сингулярних значень, для CUR-декомпозиції – ранг матриці U.

На рисунку 3.3 зображено UML діаграму діяльності, яка коротко демонструє порядок дій для виконання поставленої задачі, а саме отримання відсоткову схожість відновленого зображення після накладання редукції з оригінальним.

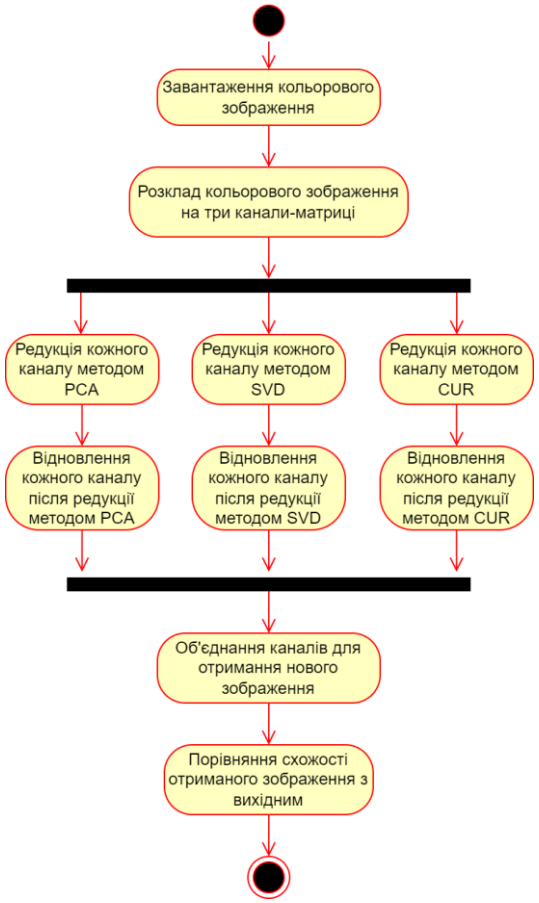


Рисунок 3.3 – Діаграма діяльності кодування та декодування з редукцією даних.

3.4 Визначення технічних засобів та реалізація задач

Для розробки програмного коду та виконання на ньому досліджень було вирішено використовувати платформу Kaggle (рис. 3.4). Цей сервіс дозволяє розробникам співпрацювати з іншими розробниками, знаходити та публікувати набори даних, використовувати блокнот з інтегрованим графічним процесором. Мета цієї онлайн-платформи полягає в тому, щоб

допомогти розробникам досягти своїх цілей у дослідженні даних за допомогою потужних інструментів і ресурсів, які вона надає.

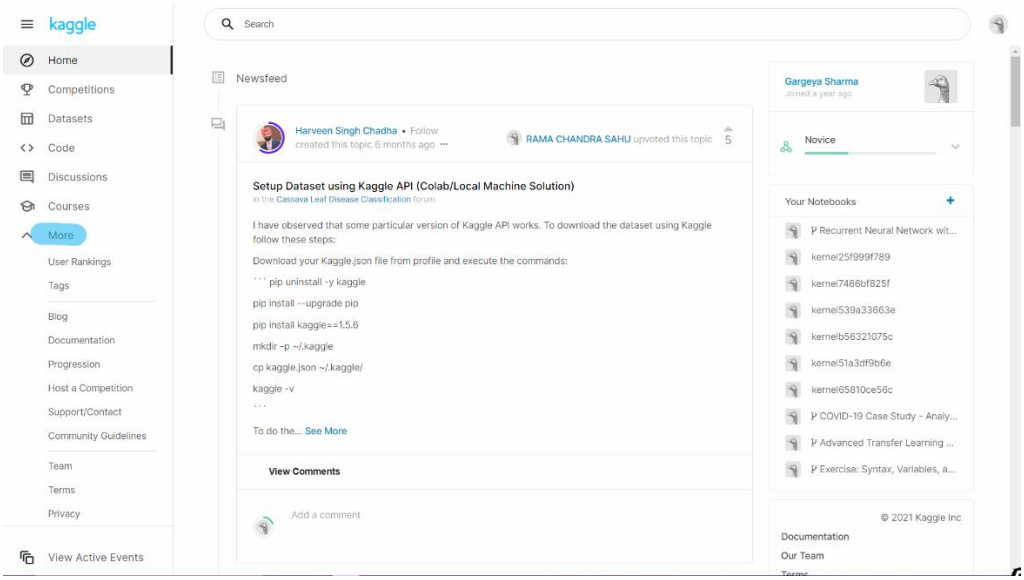


Рисунок 3.4 – Користувацький інтерфейс платформи Kaggle.

Kaggle надає потужні ресурси в хмарі та дозволяє використовувати до 30 годин GPU та 20 годин TPU на тиждень. Можливо завантажувати свої набори даних, а також використовувати дані інших розробників. Крім того, є можливість переглядати та обговорювати дослідження інших розробників.

Kaggle підтримує дві мови програмування для аналізу даних: R та Python. Для дослідження було обрано мову Python через її наступні переваги:

- універсальність: Python є універсальною мовою програмування, яка використовується у різних галузях, включаючи веб-розробку, штучний інтелект, машинне навчання і багато іншого. Це надає можливість інтегрувати дослідження у різні проекти;
- машинне навчання та глибинне навчання: бібліотеки, як TensorFlow, PyTorch і Scikit-Learn, роблять Python популярним вибором для розробки моделей машинного навчання і глибинного навчання;
- синтаксис: Python вважається більш читабельним і легким для вивчення;

- підтримка мови: спільнота Python достатньо велика, що робить легшим знаходження рішень на форумах та отримання підтримки.

Для програмної реалізації будуть використані вбудовані у Kaggle бібліотеки Python, такі як:

- pandas – бібліотека для обробки великих наборів даних;
- numpy – для роботи з матрицями;
- scikit-learn – надає класи для реалізації методів головних компонент та сингулярного розкладу;
- cur – надає клас для реалізації CUR-декомпозиції;
- matplotlib – дозволяє будувати графіки та працювати з зображеннями;

Програмна реалізація кожної з задач представлено у Додатку, а саме:

- редукція даних в рекомендаційних системах: Додаток А1;
- редукція даних при кластеризації текстових документів: Додаток А2;
- редукція даних при кодуванні та декодуванні зображень: Додаток А3.

3.5 Висновки до розділу

В цьому розділі було визначено завдання та технічні засоби для дослідження роботи методів редукції, а саме методу головних компонент, сингулярного розкладу та CUR-декомпозиції. Для кожної задачі, а саме редукція даних в рекомендаційних системах, при кластеризації текстових документів та при кодуванні та декодуванні зображень, сформовано алгоритми, очікувані об’єкти результатів, діаграми діяльності. Було розглянуто платформу для реалізації задач Kaggle, на якій, як результат, створено програмні реалізації для проведення дослідження методик редукції, використовуючи мову програмування Python.

4 ТЕСТУВАННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ МЕТОДИК РЕДУКЦІЇ

4.1 Тестування роботи програмних реалізації.

Перед початком досліджень по кожній задачі необхідно переконатися у роботі кожного з реалізованих додатків, провівши серію тестувань на них.

4.1.1 Тестування рекомендаційної системи

Тестування та дослідження рекомендаційної системи відбувається на наборі даних оглядів товарів Amazon (рис. 4.1). З даного набору було взято 886 товарів-рядків, та 9697 оцінок-стовпців та побудовано матрицю, у якій стовпці є покупцями, рядки – товарами, а їх значення – виставлена покупцем оцінка відповідному товару.

	UserId	ProductId	Rating	Timestamp
0	A39HTATAQ9V7YF	0205616461	5.0	1369699200
1	A3JM6GV9MNOF9X	0558925278	3.0	1355443200
2	A1Z513UWSAAO0F	0558925278	5.0	1404691200
3	A1WMRR494NWEWV	0733001998	4.0	1382572800
4	A3IAAVS479H7M7	0737104473	1.0	1274227200
5	AKJHHD5VEH7VG	0762451459	5.0	1404518400
6	A1BG8QW55XHN6U	1304139212	5.0	1371945600
7	A22VW0P4VZHDE3	1304139220	5.0	1373068800
8	A3V3RE4132GKRO	130414089X	5.0	1401840000
9	A327B0I7CYTEJC	130414643X	4.0	1389052800

Рисунок 4.1 – Набір даних оглядів товарів Amazon.

Після виконання редукції будь яким з методів та побудови матриці кореляції, було визначено рекомендовані товари для товару з номером 130414089X (рис. 4.2). Кореляція товару має бути більше 0.9, щоб можна було вважати його рекомендованим.

PCA

```
['0733001998',
 '0737104473',
 '1304196070',
 '1412759676',
 '1906987238',
 '3292000971',
 '535795545X',
 '5357955867',
 '6022600804',
 '6152000416',
 '6165290582',
 '6169040335',
 '7899123011',
 '951311290X',
 '9571044822',
 '9744287233',
```

SVD

```
['5357955786',
 '7883633309',
 '9790770758',
 '9790771983',
 '979077365X',
 '9790787049',
 'B000052XYH']
```

CUR

```
['1304196070',
 '1879187396',
 '5357956227',
 '9788078893',
 '9790779968',
 '9790785356',
 '9790787286',
 '9790794258',
 '9790798962',
 '9790801106']
```

Рисунок 4.2 – Результати роботи рекомендаційної системи для кожного методу редукції.

Перевірки були пройдені усіма трьома методами успішно. Таким чином було перевірено коректну роботу програмної реалізації рекомендаційної системи.

4.1.2 Тестування кластеризації текстових документів

Тестування та дослідження кластеризації текстових документів відбувається на наборі текстових документів, який з складається з 1000 текстових невеликих статей, розподілених між 10 темами (рис. 4.3).

	File Name	Text
0	space_73.txt	Archive-name: space/intro\nLast-modified: $Dat...
1	space_33.txt	shag@aero.org (Rob Unverzagt) writes:\n>In art...
2	space_5.txt	Does anyone know how to size cold gas roll con...
3	space_8.txt	nanderso@Endor.sim.es.com (Norman Anderson) wr...
4	space_66.txt	Here are some recent observations taken by the...
..	...	...
995	business_79.txt	US interest rate rise expected\n\nUS interest ...
996	business_61.txt	House prices drop as sales slow\n\nHouse price...
997	business_52.txt	UK economy facing 'major risks'\n\nThe UK manu...
998	business_13.txt	Industrial revival hope for Japan\n\nJapanese ...
999	business_5.txt	Peugeot deal boosts Mitsubishi\n\nStruggling J...

Рисунок 4.3 – Набір текстових документів.

Спочатку кластеризацію було проведено без використання методів редукції. Текстовий набір було векторизовано методом TF-IDF та

кластеризовано методом k-середніх. Для спрощення відображення кількість документів в кожному кластері була підрахована (рис. 4.4).

Cluster	
0	46
1	15
2	346
3	79
4	166
5	63
6	10
7	48
8	155
9	72

Рисунок 4.4 – Результат кластеризації без редукції.

Наступні тестування проводилися з методами редукції (рис. 4.5). Перевірки були пройдені усіма трьома методами успішно. Таким чином було перевірено коректну роботу програмної реалізації кластеризації текстових документів.

PCA

Cluster	
0	68
1	258
2	59
3	88
4	48
5	71
6	30
7	89
8	14
9	275

SVD

Cluster	
0	93
1	28
2	87
3	33
4	66
5	88
6	14
7	236
8	57
9	298

CUR

Cluster	
0	83
1	56
2	51
3	289
4	28
5	92
6	67
7	48
8	216
9	70

Рисунок 4.5 – Результат кластеризації з кожним методом редукції.

4.1.3 Тестування кодування та декодування зображень

Тестування та дослідження кодування та декодування зображень відбувалося на кольоровому зображені з трьома каналами розмірами 1600x1200 пікселів (рис. 4.6).

Рисунок 4.6 – Вихідне зображення для досліджень.

Оскільки зображення містить три кольорових канали, представлених матрицями відтінку, то їх необхідно було розкласти перед редукцією (рис. 4.7).

Рисунок 4.7 – Розклад зображення на три канали-матриці.

Далі з кожною матрицею виконувалась редукція та інверсія редукції для відновлення матриць, з наступним об’єднанням каналів у зображення. Також порівнювалося їх схожість з оригінальним зображенням. Таким чином було перевірено роботу для кожного з методів редукції. Для PCA та SVD отримані результати були схожі, а на відновленому зображені можна було побачити втрату даних, у вигляді випадкових кольорів у певних частинах зображення (рис. 4.8, рис. 4.9).

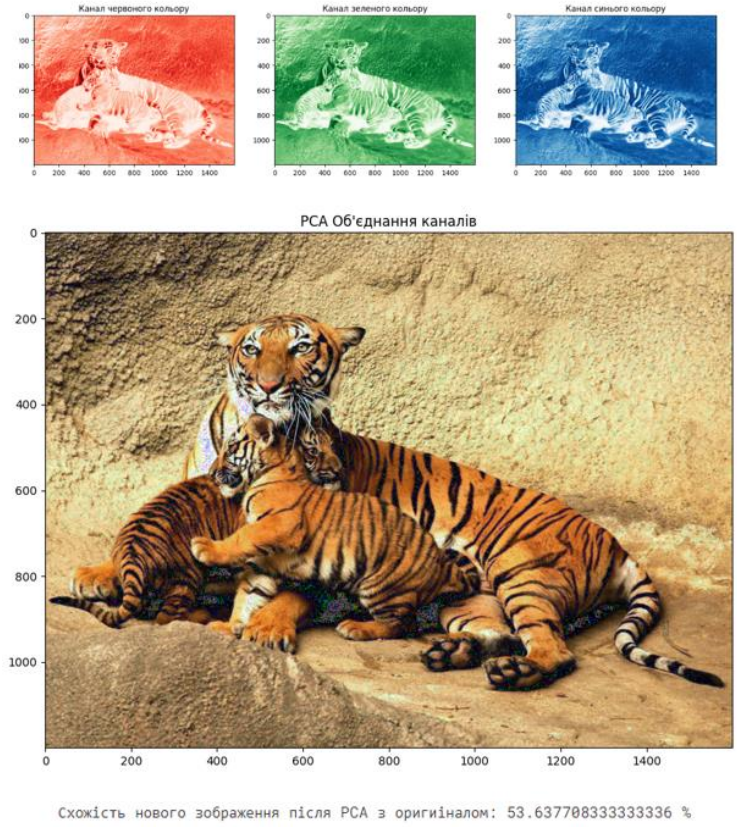


Рисунок 4.8 – Відновлення зображення після PCA.

Рисунок 4.9 – Відновлення зображення після SVD.

Проте CUR-декомпозиція навіть при збільшені точності за рахунок збільшення рангу матриці U дає погані результати, з дуже низьким відсотком схожості (рис. 4.10). Що вже дає розуміння, що CUR-декомпозиція не є ефективною для даного завдання через свою низьку точність відновлювання даних.

Таким чином було перевірено роботу програмної реалізації кодування та декодування зображень.

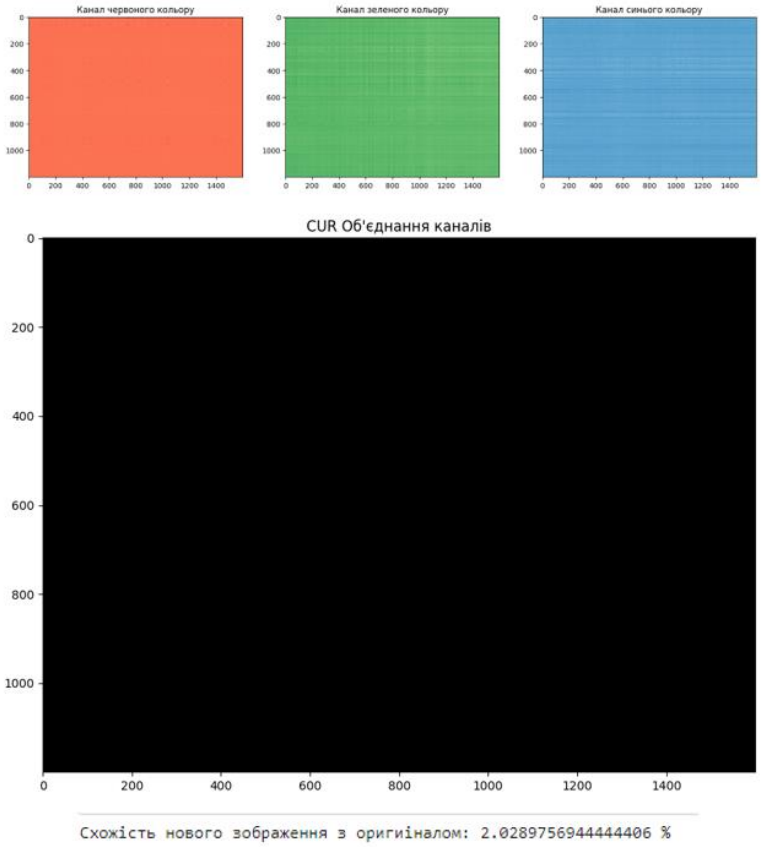


Рисунок 4.10 – Відновлення зображення після CUR-декомпозиції.

4.2 Результати дослідження редукції даних в рекомендаційних системах

Для PCA було отримано графік зображений на рисунку 4.11. З графіку можна визначити, що найкращі результати знаходяться при кількості головних компонентів у межах 40%-50% від кількості товарів-рядків. Більше 50% -

рекомендації будуть відсутні, менше 40% - висока кількість рекомендованих товарів, що вказує на втрату точності та правдивості таких рекомендацій.

Для SVD було отримано графік зображений на рисунку 4.12. З графіку можна визначити, що найкращі результати знаходяться при кількості сингулярних значень у межах 3%-5% від кількості товарів-рядків, більше 5% - рекомендації будуть відсутні, менше 3% - висока кількість рекомендованих товарів, що вказує на втрату точності та правдивості таких рекомендацій.

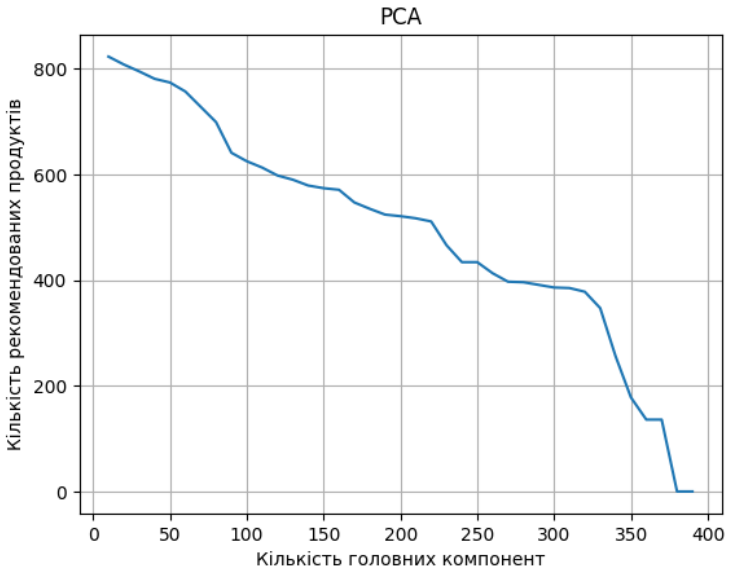


Рисунок 4.11 – Графік кількості рекомендованих товарів для PCA.

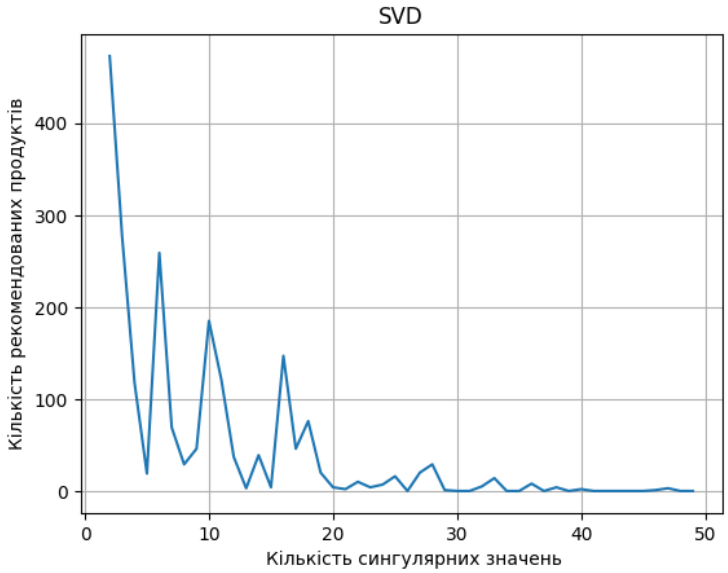


Рисунок 4.12 – Графік кількості рекомендованих товарів для SVD.

Для CUR-декомпозиції було отримано графік зображений на рисунку 4.13. З графіку можна визначити, що найкращі результати знаходяться при рангу U матриці у межах 0%-2% від кількості товарів-рядків. Але враховуючи відомості, що CUR-декомпозиція має високу втрату даних та низьку точність, при низьких рангах, то правдивість таких результатів не є гарантованою, а рекомендовані товари можна вважати випадковими.

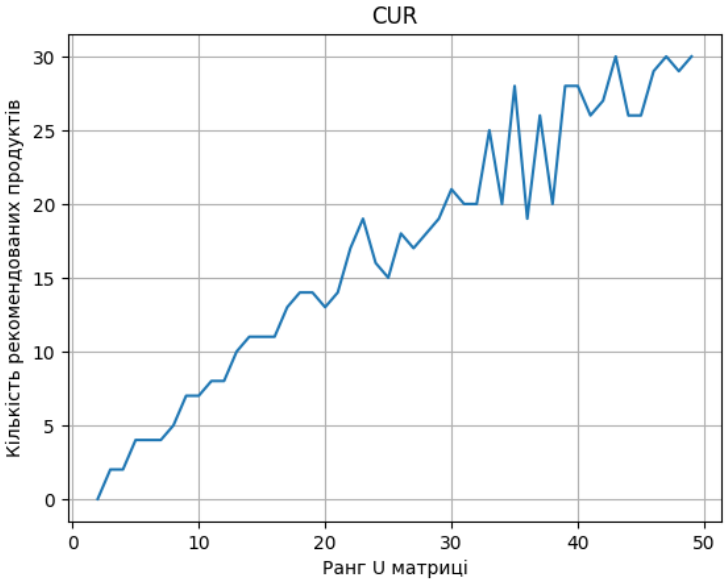


Рисунок 4.13 – Графік кількості рекомендованих товарів для CUR-декомпозиції.

4.3 Результати дослідження редукції даних при кластеризації текстових документів

Для кожного методу редукції було отримано графіки залежностей кількості статей у кожному кластері від розмірності редукції кожного методу (рис. 4.14).

За цими графіками можна ствердити, що кількість статей у кластері не залежить від обраної розмірності матриць розкладу, оскільки графіки мають стохастичний характер, та побачити якусь закономірність дуже важко або потребує більш глибокого дослідження.

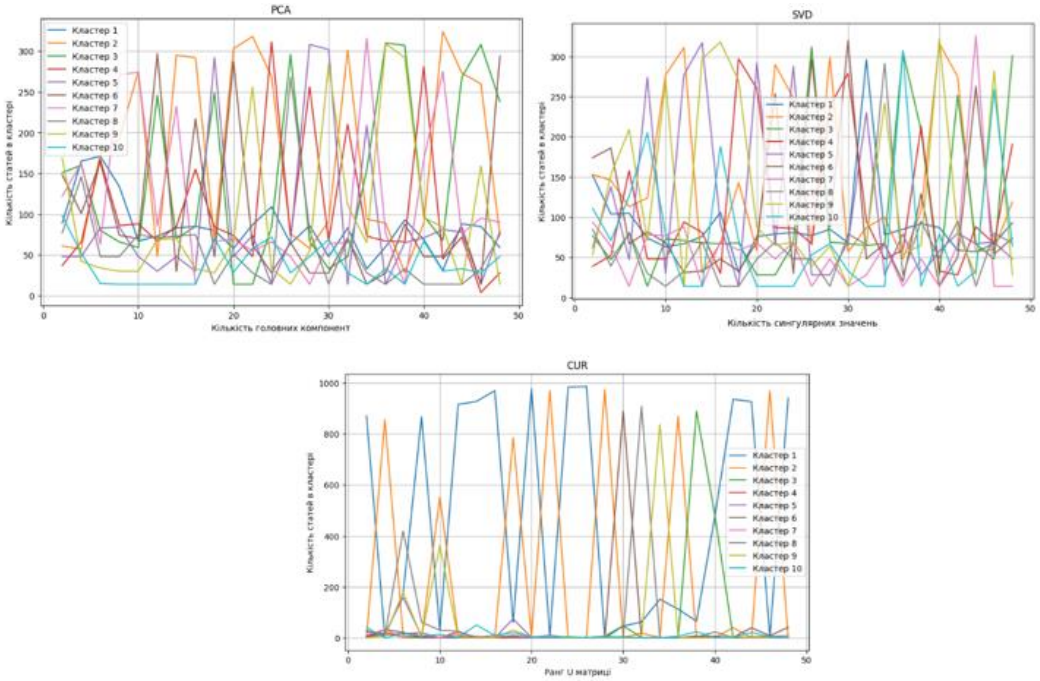


Рисунок 4.14 – Графіки кластерів отриманих методами редукції.

Тому було вирішено розглянути кожен кластер окремо та визначити середню абсолютну похибку кожного з методів редукції від кластеру, що не містить редукції. Таким чином побудовано графіки для кожного кластеру, один з яких зображений на рисунку 4.15.

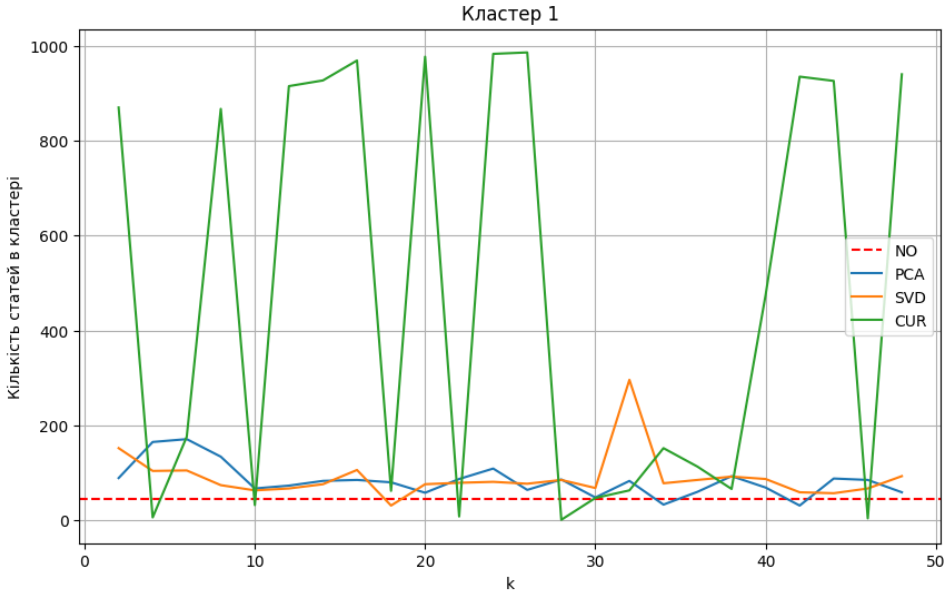


Рисунок 4.14 – Графік першого кластера.

Значення k вказує на розмірність редукції для кожного з методів редукції. Пунктирна лінія - це стале значення без редукції. З цього графіка можна визначити середню абсолютну похибку кожного методу редукції.

```
PCA
Середня абсолютна похибка: 66.025
SVD
Середня абсолютна похибка: 64.95
CUR
Середня абсолютна похибка: 120.88333333333334
```

Рисунок 4.15 – Середня абсолютна похибка першого кластеру.

Таким чином на рисунку 4.16 зображені отримані похибки для кожного кластеру:

	Кластер 1	Кластер 2	Кластер 3	Кластер 4	Кластер 5	Кластер 6	Кластер 7	Кластер 8	Кластер 9	Кластер 10
Метод головних компонент	66.025	85.325	246	60.766	104.158	60.55	90.05	65.058	97.675	59.741
Сингулярний розклад	64.95	85.225	246	59.45	104.808	58.9	90	63.9	97.958	58.325
CUR-декомпозиція	120.883	102.5	350.425	144.475	209.616	132.733	100.341	122.241	201.216	139.308

Рисунок 4.15 – Середні абсолютні похибки кожного кластеру

Метод головних компонентів та сингулярний розклад мають схожі результати у пошуках тем, їх середнє абсолютне відхилення майже однакове. CUR-декомпозиція має більше середнє абсолютне відхилення, але у деяких випадках схожу з методом головних компонентів та сингулярним розкладом.

Додатково для порівняння роботи методів побудовано візуалізацію кластеризації кожним методом редукції, з одною і тою ж розмірністю редукції.

За візуалізацією можна ствердити, що PCA спрямований на кластеризацію по більш загальним або вже відомим темам, оскільки кластери не пересікаються один з одним і мають чітке групування (рис. 4.16). SVD надає нові спільні теми, тому що тепер кластери пересікаються, а елементи можуть віддалятися від центру кластеру (рис. 4.17).

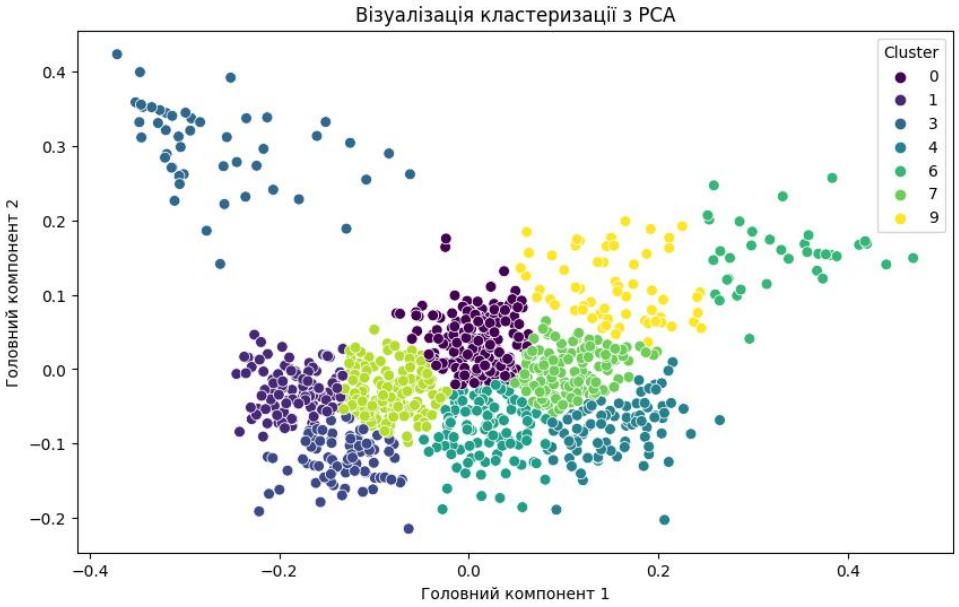


Рисунок 4.16 – Візуалізація кластеризації методом PCA.

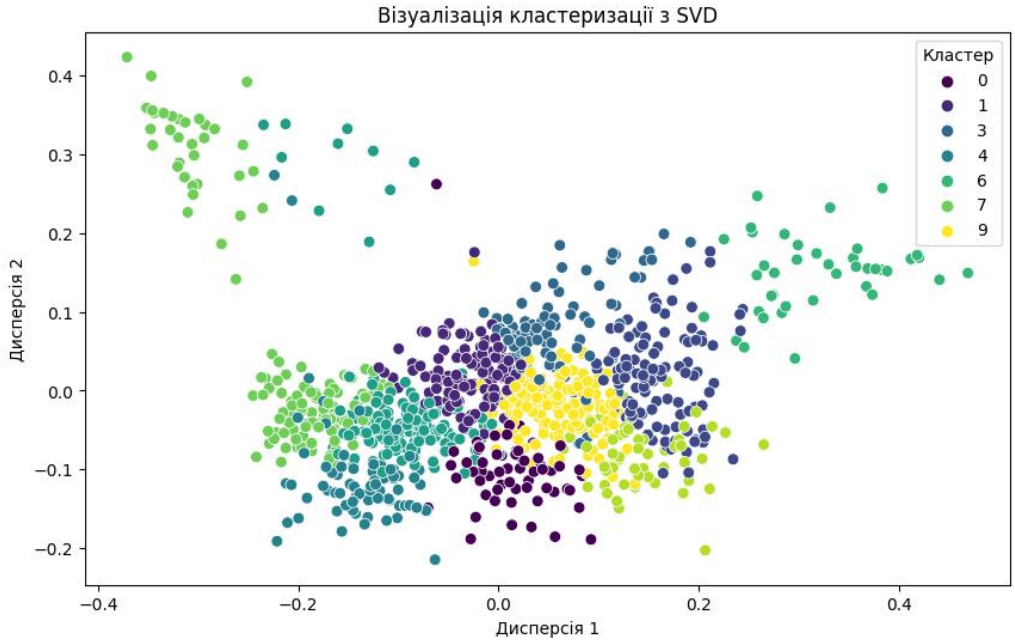


Рисунок 4.17 – Візуалізація кластеризації методом SVD.

На візуалізації CUR-декомпозиції видно, що частина кластерів відсутня, а також більшість елементів відноситься до одного кластеру (рис. 4.18). Можна припустити, що CUR шукає спільну тему по конкретному однаковому слові в текстах.

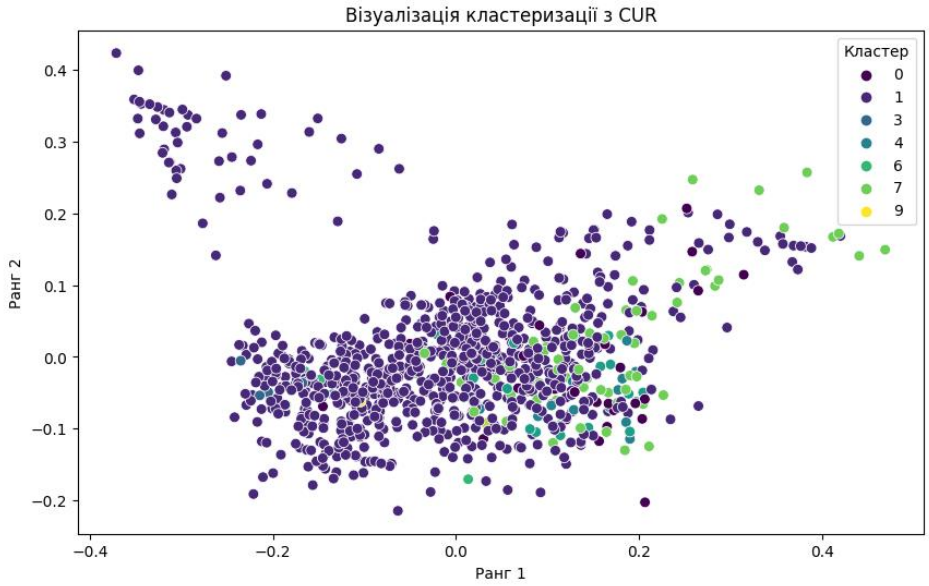


Рисунок 4.18 – Візуалізація кластеризації CUR-декомпозицією.

4.4 Результати дослідження редукції даних при кодуванні та декодуванні зображень

Для методу головних компонентів було отримано графік зображений на рисунку 4.19. За графіком можна побачити, що найбільший приріст схожості є при кількості головних компонентів у межах 50%-80% від ширини зображення. Максимальна схожість, яку можна отримати 95%.

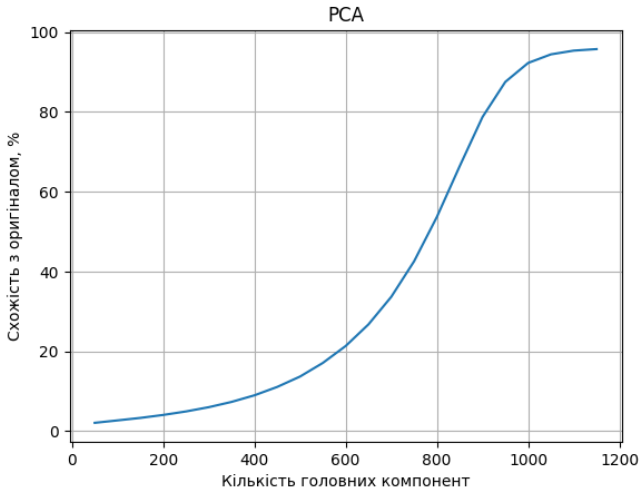


Рисунок 4.19 – Графік схожості відновленого зображення для PCA.

Для сингулярного розкладу було отримано графік зображений на рисунку 4.20. За графіком можна визначити, що найбільший приріст схожості є при кількості головних компонентів у межах 40%-65% від ширини зображення. Максимальна схожість, яку можна отримати 52%.

Для CUR-декомпозиції було отримано графік зображений на рисунку 4.21. За графіком можна визначити, що метод має настільки погане відновлення, що зберігає лише 2% зображення при будь якому рангу матриці U. Метод повністю відкидає більшу частину інформації при редукції. А 2% це лише чорна частина зображення, з яка співпадає з оригіналом.

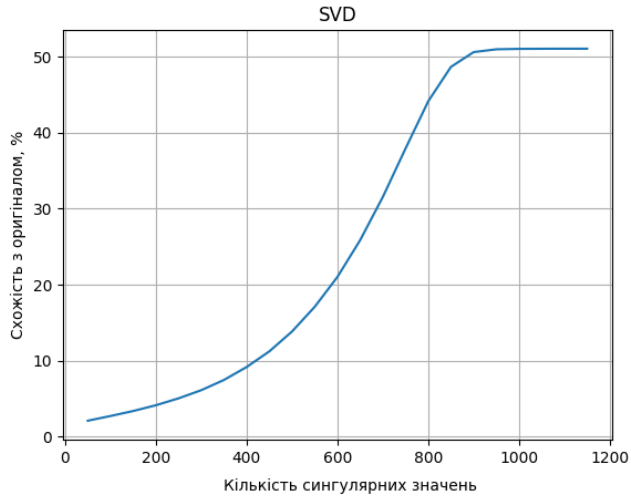


Рисунок 4.20 – Графік схожості відновленого зображення для SVD.

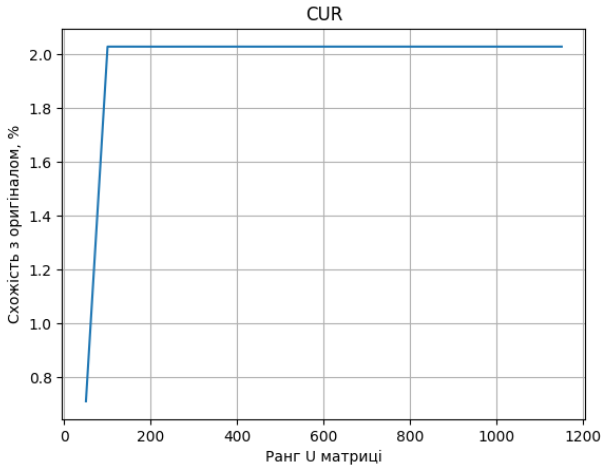


Рисунок 4.21 – Графік схожості відновленого зображення для CUR-декомпозиції.

4.5 Висновки до розділу

В цьому розділі було проведено дослідження та аналіз роботи методів редукції у задачах рекомендаційної системи, кластеризації текстових документів, кодування та декодування зображень. Було проведено тестування розроблених програмних реалізацій, отримано графіки та числові дані, за якими можна порівняти кожен метод редукції відповідно до кожної задачі.

Найкращим методом для рекомендаційних систем є сингулярний розклад, оскільки він має найкращу точність при низькій розмірності матриць розкладу. Про те метод головних компонентів також можна використовувати, але при більшій кількості даних його швидкодія може різко впасти, через велику кількості головних компонентів. Результати CUR-декомпозиції важко вважати точними через елемент випадковості, тому цей метод можна використовувати у випадках, коли рекомендації повинні бути випадковими або кількість оцінок у новому сервісі є недостатньою для надання об’єктивної рекомендації.

Вибір методу редукції для кластеризації текстових документів залежить від цілей самої кластеризації. У випадку кластеризації по відомим спільним темам найкраще буде використати метод головних компонентів. Для пошуку нових спільних тем – сингулярний розклад. У випадку, коли необхідно відокремити документи за певним словом можна використати CUR-декомпозицію.

Для кодування та декодування найкращу відновлювальність даних має метод головних компонентів, зберігаючи до 95% інформації, і можна вважати найкращим варіантом для таких і подібних задач. Сингулярний розклад також може бути використаний, якщо точність відновлених даних не є важливою. CUR-декомпозиція не є рішенням для даної задачі.

5 РОЗРОБКА СТАРТАП-ПРОЕКТУ

5.1 Опис ідеї проекту

Описані дослідження можуть бути реалізовані в якості бібліотеки для мови програмування для Python. Цей розділ ставить перед собою мету створення бібліотеки підпрограм для спрощення інтеграції методів редукції у рекомендаційних системах, кластеризації документів, кодуванні та декодуванні зображень із використанням методів редукції. Цей процес включає в себе:

- реалізацію бібліотеки підпрограм з використанням підходів та технологій, що були описані в попередніх розділах даної роботи;
- розробку стратегії виводу конкурентоздатного продукту на ринок та подальший розвиток стартапу.

Для отримання повної картини про зміст ідеї та можливих базових ринків, на яких можна знайти групи потенційних клієнтів, важливо створити таблицю, яка включає опис ідеї, можливі напрямки застосування та основні переваги, які можуть мати користувачі продукту.

Таблиця 5.1 - Опис ідеї стартап-проекту.

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Створення підпрограм із використанням методів редукції у вигляді бібліотеки для Python	1. Розробка рекомендаційних систем 2. Розробка систем кластеризаці текстових документів 3. Додатки, що включають кодування та декодування зображень	Спрощена інтеграція методів редукції у різноманітних завданнях розробки програмного забезпечення Користувач буде мати змогу використовувати та модернізовувати підпрограми з бібліотеки для власних задач

Отже, ідея проекту полягає в тому, що за допомогою даної бібліотеки користувачі зможуть спростити інтеграцію методів редукції у

рекомендаційних системах, кластеризації документів, кодуванні та декодуванні зображень. Також, можливість розвитку бібліотеки може значно збільшити варіативність задач, у яких можуть бути використані методи редукції.

Нижче наведено список переваг продукту в порівнянні з конкурентами, який базується на техніко-економічних характеристиках та властивостях ідеї. Цей перелік наведено в таблиці 5.2 та може бути використаний для формування конкурентних переваг товару, які нададуть йому більшу привабливість для потенційних клієнтів.

Таблиця 5.2 - Визначення сильних, слабких та нейтральних характеристик ідеї проекту.

№ п/п	Техніко-економічні характеристики ідеї	Потенційні товари/концепції конкурентів				W	N	S
		Власний проект	Конкурент 1	Конкурент 2	Конкурент 3			
1.	Форма виконання	Бібліотека підпрограм	Розширення до додатку	Десктопний додаток	Бібліотека			+
2.	Собівартість	Висока	Висока	Висока	Висока		+	
3.	Емоційність	Контрольована	Контрольована	Контрольована	Контрольована		+	
4.	Кросплатформність	так	ні	ні	так			+
5.	Потреба в інтернеті	ні	ні	ні	ні		+	

Перевагами даного проекту є те, що форма виконання продукту – бібліотека підпрограм, що є більш зручним у при розробці додатків у різних сферах, а, також, кросплатформність, що забезпечується технологією, яку було

обрано для реалізації. Адже здатність використовувати програмний продукт на різних платформах напряму збільшує кількість потенційних клієнтів. Всі інші характеристики є нейтральними. Тому даний проект можна вважати конкурентоздатним.

5.2 Технологічний аудит ідеї проекту

У даному розділі необхідно здійснити аудит технології для реалізації ідеї проекту в межах даного підрозділу (технології створення продукту).

Таблиця 5.3 - Технологічна здійсненність ідеї проекту.

п/п	Ідея проекту	Технології та її реалізації	Наявність технологій	Доступність технологій
1	Розробка бібліотеки підпрограм для різних задач із використанням методів редукції	Python	Наявна	Безкоштовна, доступна
Обрана технологія реалізації ідеї проекту: для створення бібліотеки підпрограм для інтеграції методів редукції у рекомендаційних системах, кластеризації документів, кодуванні та декодуванні зображень була обрана мова програмування Python, яка є зручною у використанні та безкоштовною, а також користується великою популярністю серед спільноти розробників.				

Отже, проект буде реалізовано за допомогою мови програмування Python, що використовується у вирішення широкого спектру задач. Дана мова програмування є легкою для розуміння та вивчення, є кросплатформеною мовою програмування та отримує підтримку спільноти розробників. Для створення бібліотеки підпрограм на Python можна використовувати фреймворки, такі як Tkinter або PyQt.

5.3 Аналіз ринкових можливостей

Дослідження ринкових можливостей та загроз може допомогти у складанні стратегії розвитку проекту, зважаючи на поточний стан ринкового середовища, потреби майбутніх клієнтів та конкурентні пропозиції.

Важливим етапом такого аналізу є визначення потреб ринку, зокрема наявність попиту, його обсяг та динаміку розвитку. Тому, для початку розглянемо їх.

Таблиця 5.4 - Попередня характеристика потенційного ринку стартап-проекту.

№ п/п	Показники стану ринку (найменування)	Характеристика
1	Кількість головних гравців, од	1000
2	Загальний обсяг продаж, грн/ум.од	10000
3	Динаміка ринку (якісна оцінка)	Зростає
4	Наявність обмежень для входу	Висока вартість початкового капіталу
5	Специфічні вимоги до стандартизації та сертифікації	-
6	Середня норма рентабельності в галузі (або по ринку), %	R = (3000000 * 1000) / (10000000 * 12) = 25%

Таким чином, створення проекту має більшу середню норму рентабельності, ніж банківський процент за вкладення коштів. Отже, інвестування в цей проект має потенційний вигляд.

Далі проводиться ідентифікація можливих сегментів аудиторії, їх особливості та складається приблизний перелік вимог до продукту для кожного з них (табл. 5.5).

Таблиця 5.5 - Характеристика потенційних клієнтів стартап-проекту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1	Рекомендаційні системи, кластеризація текстових додатків, кодування та декодування зображень методами редукції	а) Компанії розробники б) Навчальна сфера в) Наукова сфера г) Незалежні програмісти-фрілансери	а) оцінка фінансових показників та фінансового ринку б) оцінка ефективності навчання користувачів в) пошук нових технологій г) доступність засобів розробки	Бібліотека підпрограм: надійна, функціонально розвинена, зрозуміла користувачу Постачальник сервісу: надає підтримку, постійно покращує рішення

Визначено характеристики стартап-проекту: основну потребу, що формує ринок – використання методів редукції у різних сферах, а саме у науково-навчальній та сфері інформаційних технологій. Також, було визначено відмінності та цілі у поведінці зазначених цільових груп. Було визначено та затверджено основні вимоги споживачів до розробленого додатку – надійна система з ефективним функціоналом редукції даних.

Потім формуються таблиці позитивних факторів, що сприяють впровадженню проекту на ринку, та негативних факторів, що можуть ускладнити його реалізацію (табл. 5.6 - 5.7).

Таблиця 5.6 - Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1	Зростаюча вимогливість покупців	Клієнти очікують, що алгоритми будуть працювати точніше та буде забезпечуватись підтримка	Створення науково-дослідної лабораторії з метою вдосконалення роботи алгоритмів
2.	Збільшення витрат на технічну підтримку	Недостатньо швидка реакція на потреби сучасного ринку користувачів	Вчасно виконувати оновлення програмного продукту
3.	Конкуренція	Вихід нових конкуруючих проектів	Оновлення програмного продукту; Закриття проекту; Додати переваги в проект для користувачів
4.	Зменшення кількості замовників	Зменьшення клієнтської бази або зменшення попиту на продукт	Постійна підтримка та оновлення продукту; Додавання переваг для користувачів

В таблиці 5.6 було наведено основні чинники, які можуть загрожувати стартап-проекту. Найбільшою загрозою для проекту є зміна потреб користувачів, які вимагають нового функціоналу та більш точної роботи системи. Для зменшення цих ризиків необхідне створення дослідницької лабораторії, що буде зосереджена на покращенні роботи алгоритмів, а також своєчасне оновлення програмного забезпечення та користувацького інтерфейсу. Менші загрози становлять збільшення витрат на технічну підтримку та зменшення кількості замовників.

Таблиця 5.7 - Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1	Підвищились технологічні бар’єри входу на ринок: оновлення алгоритмів - важкий дослідницький проект	Для створення задовільної для користувачів функціональності необхідні висококваліфіковані розробники та аналітики	Залучення та підготовка фахівців з потенціалом підвищення власних знань
2	Ймовірність швидкого зростання завдяки раптовому збільшенню попиту на ринку	Методи редукції є актуальними та використовуваними у різних галузях	Максимальне охоплення ринку завдяки вагомих клієнтів
3	Зростання можливостей потенційних покупців	Зростання фінансування досліджень у галузі нечіткої логіки та прогнозування	Запропонувати свої послуги зацікавленим підприємствам
4	Зниження довіри до конкурентів	Зменшення якості роботи конкурентних продуктів	Під час виходу на ринок звертати увагу на якість розробленої бібліотеки
5	Мінімізація витрат на технічну підтримку	Компанія прагне збільшити продуктивність свого штату шляхом підвищення їхньої кваліфікації	Забезпечувати професійний розвиток співробітників

Було наведено основні фактори, які допоможуть успішно впровадити проект на ринок: підвищення технологічних бар'єрів для конкурентів, оновлення програмного забезпечення, що потребує значних досліджень, можливість швидкого зростання завдяки підвищенню попиту на ринку та зростання можливостей потенційних клієнтів. Компанія реагує на ці фактори шляхом надання послуг зацікавленим підприємствам, забезпечення безпеки алгоритмів, підвищення кваліфікації своїх співробітників, найму та підготовки

перспективних фахівців і максимального захоплення ринку через привабливих клієнтів.

Наступним кроком є визначення загальних рис конкуренції на ринку (табл. 5.8).

Таблиця 5.8 - Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1. Вказати тип конкуренції - олігополія	Існування невеликої кількості конкурентно спроможних компаній	Створення рішень з більш високим рівнем технологічної досконалості, ніж ті, що пропонують конкуренти
2. За рівнем конкурентної - інтернаціональна	Всі конкуренти - закордонні	Можна знайти дослідників за вигіднішу ціну
3. За галузевою ознакою - внутрішньогалузева	Використання продукту в даній галузі	Добре описані рамки галузі та способи впливу на неї
4. Конкуренція за видами товарів - товарно-видова	Вид послуг є однаковим чи схожим, а реалізація продукту є різною	Конкруенція між схожими продуктами
5. За характером конкурентних переваг - нецінова	Оптимізація технології розробки додатку для зниження його вартості	Покращення якості продукції
6. За інтенсивністю - не марочна	Бренди не мають відчутного впливу на галузь або їх взагалі немає	Легше вийти на ринок початковій компанії

Було проведено аналіз конкуренції на ринку, а саме визначено: тип конкуренції - олігополія; конкуренція за рівнем конкурентної боротьби - міжнародна; конкуренція за галузевою ознакою - внутрішньогалузева; конкуренція за видами товарів - товарно-видова; конкуренція за характером конкурентних переваг - нецінова; конкуренція за інтенсивністю - не марочна. Також було наведено можливі дії компанії, щоб бути конкурентоспроможною: створення рішень з більш високим рівнем технологічної досконалості, ніж ті, що пропонують конкуренти; пошук дослідників за вигіднішу ціну; добре описані рамки галузі та способи впливу на неї; покращення якості продукції.

Далі розробляється перелік факторів конкурентоспроможності для ринку на основі аналізу складових моделі п’яти сил М. Портера (табл. 5.9).

Таблиця 5.9 - Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
	CubiCalc	HyperLogic Corporation	Amazon Google Сильні сторони: стабільні, мають підтримку	Побутові користувачі та інформація відсутня	-
Висновки	Дуже інтенсивна боротьба за першість	Потенційно може вийти будь-яка компанія з необхідними ресурсами	Нічого не диктують, є можливість забезпечення власної інфраструктури	Диктують умови	-

Отже, можна стверджувати, що проект має можливість успішного виходу на ринок, оскільки серед конкурентів, наведених у даній галузі, немає жодного, який здатен конкурувати з ним, адже розроблене рішення значно полегшує та прискорює роботу фахівця.

На основі аналізу висновків, наведених вище, визначається та обґрунтовується перелік факторів конкурентоспроможності (табл. 5.10).

Таблиця 5.10 - Обґрунтування факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)
1	Науково-технічний потенціал	Кращі дослідження, а отже вища якість послуг
2	Кадровий потенціал	Кращі кадри, а отже кращі дослідження

Основними факторами конкурентоспроможності, які будуть забезпечені на ринку, є науково-технічний потенціал для надання високоякісних послуг та кадровий потенціал для проведення більш швидких та якісних досліджень у нечіткої логіки.

За визначеними факторами конкурентоспроможності проводиться аналіз сильних та слабких сторін стартап-проекту (табл. 5.11).

Таблиця 5.11 - Порівняльний аналіз сильних та слабких сторін проекту

№ п/п	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні зі стартап-проектом						
			-3	-2	-1	0	1	2	3
1	Науково-технічний потенціал	19			+				
2	Кадровий потенціал	19			+				

Був проведений порівняльний аналіз між проектом нашої компанії та конкуруючими продуктами, де були визначені найсильніші сторони. Науково-технічний та кадровий потенціали отримали найвищу оцінку серед факторів конкурентоспроможності.

Далі проводимо SWOT-аналіз (табл. 5.12).

Таблиця 5.12 - SWOT- аналіз стартап-проекту

Сильні сторони: висококваліфіковані фахівці в області	Слабкі сторони: маркетинговий відділ потребує покращень, компанія ще не встигла зарекомендувати себе на ринку, оскільки є молодою
Можливості: стрімкий ріст ринку та технологічності рішень у сфері прогнозування із нечіткими даними	Загрози: конкуренти володіють значними ресурсами та відомостями, що робить їх потужнішими в порівнянні з нашою компанією. З моменту введення нашого рішення на ринок, конкуренти можуть знайти спосіб покращити їх програмний продукт.

Отже, внутрішні можливості компанії і спроможності щодо виведення продукту на ринок характеризуються такими сильними і слабкими сторонами: сильні - висококваліфіковані фахівці в області; слабкі - маркетинговий відділ потребує покращень, компанія ще не встигла зарекомендувати себе на ринку, оскільки є молодою. Ринкові та можливості компанії щодо зовнішнього оточення характеризуються можливостями і загрозами: можливості - стрімкий ріст ринку та технологічності рішень; загрози - конкуренти володіють значними ресурсами та відомостями, що робить їх потужнішими в порівнянні з нашою компанією, можливість конкурентів знайти спосіб покращити їх програмний продукт.

На основі SWOT-аналізу складаються альтернативи ринкового впровадження стартап-проекту (табл. 5.13).

Таким чином, було визначено, що найбільш оптимальним варіантом введення стартап-проекту на ринок є покращення додатку та алгоритмів. Це пов'язано з тим, що покращення алгоритмів має більший успіх ніж запуск обмеженого продукту. Однак, покращення додатку та алгоритмів потребує більше часу.

Таблиця 5.13 - Альтернативи ринкового впровадження стартап-проекту

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	Покращення додатку та алгоритмів	75%	1 рік
2	Запуск на ринок продукту з обмеженими можливостями і зосереджуємося на швидкому зростанні ринку	25%	6 місяців

5.4 Розробка ринкової стратегії проекту

Розроблення ринкової стратегії першим кроком передбачає визначення стратегії охоплення ринку: опис цільових груп потенційних споживачів (табл. 5.14).

Таблиця 5.14 - Вибір цільових груп потенційних споживачів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Сфери, які працюють з рекомендаційними системами	Готові, як найшвидше	Попит - високий	Достатньо інтенсивна	Середня
2	Сфери, які працюють з великикми наборами документів	Готові, як найшвидше	Попит - середній	Достатньо інтенсивна	Середня
3	Сфери інформаційної безпеки	Готові, як найшвидше	Попит - високий	Достатньо інтенсивна	Середня
Які цільові групи обрано: обрано усі групи, так як вони майже не відрізняються, а проект їх може задовольнити					

Отже, було вибрано основні цільові групи: сфери, які працюють з рекомендаційними системами, сфери, які працюють з великими наборами документів, сфери інформаційної безпеки.

Далі визначається базова стратегія розвитку (табл. 5.15).

Таблиця 5.15 - Визначення базової стратегії розвитку

№ п/п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
1	Створення бібліотеки підпрограм	Створення технологічних інновацій	Основна позиція - новизна, адже є ключовою технологією і перевагою проекту	Стратегія диференціації

Наступним кроком є вибір стратегії конкурентної поведінки (табл. 5.16).

Таблиця 5.16 - Визначення базової стратегії конкурентної поведінки

№ п/п	Чи є проект «першопрохідцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки*
1	Ні	Шукати нових і забирати існуючих	Стабільність/надійн ість алгоритмів, підтримка, розвиток	Оборонна

Таким чином, була обрана оборонна стратегія конкурентної поведінки, яка буде підтримуватись технічною підтримкою та забезпечить розвиток, що допоможе підвищити довіру і лояльність споживачів.

Далі визначається стратегія позиціонування проекту, яка допоможе користувачам ідентифікувати продукт (табл. 5.17).

Таблиця 5.17 - Визначення стратегії позиціонування

№ п/п	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)
1	Коректна робота, надійність, підтримка	Стратегія диференціації	Основна позиція - емоційність, адже є ключовою технологією і перевагою нашого проекту	Швидкодія, безпека, простота

Отже, була розроблена стратегія позиціонування, яка включає основні вимоги цільової аудиторії до товару, такі як коректна робота, надійність та підтримка. Базова стратегія розвитку обрана як диференціація. Ключові конкурентні переваги стартап-проекту полягають у його емоційному аспекті, оскільки він є ключовою технологією і перевагою нашого проекту. Крім того, була розроблена комплексна позиція проекту, яка включає швидкодію, безпеку та простоту.

5.5 Розробка маркетингової програми

Для початку рекомендовано сформувати маркетингову концепцію товару, який буде залучати споживачів. Для цього можна скористатися результатами аналізу конкурентоспроможності товару, який можна зібрати в таблиці 5.18.

Таблиця 5.18 - Визначення ключових переваг концепції потенційного товару

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити)
1	Автоматизація певних програмних систем: зменшення розмірів інформації перед їх обробкою	Проста інтеграції та ефективність роботи алгоритмів	Більший набір функціоналу, який буде розширюватися

Надалі розробляється трирівнева маркетингова модель товару (табл. 5.19).

Таблиця 5.19 - Опис трьох рівнів моделі товару

<table>
<tr><td>Рівні товару</td><td colspan="3">Сутність та складові</td></tr>
<tr><td>I. Товар за задумом</td><td colspan="3">Бібліотека підпрограм для різних задач із використанням методів редукції. Внаслідок цього не потрібно наймати додаткових фахівців для консультацій користувачів.</td></tr>
<tr><td rowspan="4">II. Товар у реальному виконанні</td><td>Властивості/характер истики</td><td>М/Нм</td><td>Вр/Тх /Тл/Е/Ор</td></tr>
<tr><td>Надійність
Емоційність
Найдовший час відповіді</td><td>М
М
М</td><td>Тл
Тл
Тл</td></tr>
<tr><td colspan="3">Надає кращу точність прогнозування</td></tr>
<tr><td colspan="3">Бібліотека підпрограм</td></tr>
<tr><td rowspan="2">III. Товар із підкріпленням</td><td colspan="3">Акції, знижки</td></tr>
<tr><td colspan="3">Підтримка</td></tr>
<tr><td colspan="4">За рахунок чого потенційний товар буде захищено від копіювання: патент.</td></tr>
</table>

Наступним кроком є визначення цінових меж (табл. 5.20). Аналіз проводиться експертним методом.

Таблиця 5.20 - Визначення меж встановлення ціни

№ п/п	Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
1	5000 грн. / місяць	2500 грн. / місяць	>500000 грн. / місяць	2500 грн. – 5000 грн. /місяць

Визначено межі встановлення ціни на бібліотеку підпрограм, а саме рівень цін на товари-замінники – 5000 грн на місяць використання ліцензії, рівень цін на товари-аналоги 2500 грн., рівень доходів цільової групи споживачів – 50000 грн, верхня та нижня межі встановлення ціни на товар – 2500 – 5000 грн. Аналіз був проведений експертним методом.

Наступним кроком є визначення оптимальної системи збуту, в межах якого приймається рішення (табл. 5.21).

Таблиця 5.21 - Формування системи збуту

№ п/п	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
1	Клієнти зазвичай оцінюють можливості платформи через безкоштовну пробну версію, а потім почнуть платити за кожен місяць ліцензії протягом довгого часу	Фінансування витрат на функціонування каналу збуту, фінансування збутових операцій	Канал нульового рівня	Проводити збут власними силами

Отже, було сформовано систему збуту у вигляді щомісячної оплати за певну кількість ліцензій та буде включати користувацьку підтримку. Збут буде проводитися власними силами.

Далі розробляється концепція маркетингових комунікацій (табл. 5.22).

Отже, була розроблена концепція маркетингових комунікацій, в якій основними позиціями для позиціонування є емоційність чат-ботів під час спілкування з ними. Головним завданням рекламного повідомлення є привернення максимальної уваги до продукту та залучення компаній до спроби продукту, з фокусуванням на компромісі між потребами компаній та їх користувачів.

Таблиця 5.22 - Концепція маркетингових комунікацій

№ п/п	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
1	Це компанії, які надають перевагу новітнім технологіям	E-mail, письмові звернення до компаній, конференції	Унікальна особливість чат-ботів - емоційність, покращує задоволення клієнтів	Привернути максимум уваги до продукту, змусити компанію спробувати продукт	Користувачі хочуть спілкуватись з живими людьми, компанії хочуть наймати менше людей - ми компроміс.

5.6 Висновки до розділу

У даному розділі були досліджені основні аспекти виходу на ринок бібліотеки підпрограм для спрощення інтеграції методів редукції у рекомендаційних системах, кластеризації документів, кодуванні та декодуванні зображень. Розроблений продукт може використовуватись у різних сферах з метою покращення їх ефективності та якості роботи.

Визначено сильні, слабкі та нейтральні характеристики та властивості потенційного товару, які утворюють основу його конкурентоспроможності. Для створення бібліотеки була обрана мова Python - безкоштовна, зручна та гнучка технологія. Крім того, був проведений ступневий аналіз конкуренції на ринку, SWOT аналіз та були обґрунтовані фактори конкурентоспроможності. Також були визначені основні цільові групи: сфери, де оброблюються великі обсяги інформації. Головними характеристиками проекту є новизна, надійність та широкий спектр послуг у аналізі рішень. Крім того, технологія буде захищена від копіювання завдяки патенту.

ВИСНОВКИ

В рамках даної магістерської дисертації було проведено широкий аналіз та дослідження задач, пов’язаних з редукцією обсягу інформації в системах обробки великих даних. Дослідження було спрямоване на аналіз існуючих методів редукції, їх особливості, математичні моделі та застосування у практичних задачах.

З початку було розглянуто існуючі методи редукції даних, серед яких було обрано три найефективніші у зменшені розмірностей матриць, а саме: метод головних компонент, сингулярний розклад та CUR-декомпозиція. Ці методи є універсальними під різні типи задач, а також мають нескладну реалізацію.

Для обраних методів було розглянуто їх математичні моделі та наведено приклади розрахунків на невеликих матрицях. Це дало змогу зрозуміти принципи роботи кожного з них, що дозволило аналізувати результати досліджень з врахуванням особливостей кожного з методів.

Для дослідження методів редукції було визначено три задачі, які мають різні напрямки досліджень, а саме: рекомендаційні системи, кластеризація текстів та кодування і декодування інформації. Таким чином визначено задачі та розроблено програмні реалізації для рекомендаційної системи товарів Amazon, кластеризація текстових статей та кодування і декодування зображень з трьома каналами.

За проведеними дослідженнями було визначено найкращі методики редукції для кожного напрямку досліджень. Так для рекомендаційної системи найкращі результати дає метод SVD. Для кластеризації, в залежності від цілей кластеризації, може підійти кожен з трьох методів. Для кодування та декодування інформації найкращу відновленість даних має метод PCA.

На основі проведених досліджень можна зробити висновок, що методи редукції можуть стати ефективними засобами та отримати розвиток у сферах

комерції, аналізу текстів та захисту інформації, де використовуються великі масиви даних.

Окрім цього, проведені дослідження можуть мати продовження. Методи редукції можуть бути інтегрованими та дослідженими у різних типах рекомендаційних систем. Для кластеризованих за темами текстах можна розробити алгоритми на основі машинного навчання для визначення цих тем. Сингулярний розклад та метод головних компонент можуть бути застосовані для кодування та декодування інших типів файлів, які можна представити у вигляді матриць.

Також було розроблено та досліджено стартап-проект бібліотеки підпрограм для спрощення інтеграції методів редукції у рекомендаційних системах, кластеризації документів, кодуванні та декодуванні зображень. Розроблений продукт може використовуватись у різних сферах з метою покращення їх ефективності та якості роботи.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Leskovec J., Rajaraman A., Ullman J. Mining of Massive Datasets. 2014. P. 429–460.

2. Marz N., Warren J. Big Data: Principles and best practices of scalable real-time data systems. 2015. P. 179–219.

3. Zheng A., Casari A. Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists. 2018. P. 128–145.

4. Principal Component Analyses (PCA)-based findings in population genetic studies are highly biased and must be reevaluated. Scientific reports : веб-сайт. URL: https://www.nature.com/articles/s41598-022-14395-4 (дата звернення: 16.10.2023).

5. What Is Principal Component Analysis (PCA) and How It Is Used? Sartorius : веб-сайт. URL: https://www.sartorius.com/en/knowledge/science-snippets/what-is-principal-component-analysis-pca-and-how-it-is-used-507186 (дата звернення: 16.10.2023).

6. PCA | What Is Principal Component Analysis & How It Works? Analytics Vidhya : веб-сайт. URL: https://www.analyticsvidhya.com/blog/2016/03/pca-practical-guide-principal-component-analysis-python (дата звернення: 16.10.2023).

7. Linear Discriminant Analysis, Explained. Towards Data Science : веб-сайт. URL: https://towardsdatascience.com/linear-discriminant-analysis-explained-f88be6c1e00b (дата звернення: 17.10.2023).

8. Is LDA a dimensionality reduction technique or a classifier algorithm? Towards Data Science : веб-сайт. URL: https://towardsdatascience.com/is-lda-a-dimensionality-reduction-technique-or-a-classifier-algorithm-eeed4de9953a (дата звернення: 17.10.2023).

9. A Brief Introduction to Linear Discriminant Analysis. Analytics Vidhya : веб-сайт. URL: https://www.analyticsvidhya.com/blog/2021/08/a-brief-introduction-to-linear-discriminant-analysis (дата звернення: 17.10.2023).

10. Introduction to ICA: Independent Component Analysis. Towards Data Science : веб-сайт. URL: https://towardsdatascience.com/introduction-to-ica-independent-component-analysis-b2c3c4720cd9 (дата звернення: 17.10.2023).

11. Independent component analysis: An introduction. Emerald Insight : веб-сайт. URL: https://www.emerald.com/insight/content/doi/10.1016/j.aci.2018.08.006/full/html (дата звернення: 17.10.2023).

12. Diving Deeper into Dimension Reduction with Independent Components Analysis (ICA). Paperspace : веб-сайт. URL: https://blog.paperspace.com/dimension-reduction-with-independent-components-analysis (дата звернення: 17.10.2023).

13. Non-Negative Matrix Factorization (NMF) for Dimensionality Reduction in Image Data. Towards Data Science : веб-сайт. URL: https://towards datascience.com/non-negative-matrix-factorization-nmf-for-dimensionality-reduction-in-image-data-8450f4cae8fa (дата звернення: 18.10.2023).

14. Optimization and expansion of non-negative matrix factorization. BMC : веб-сайт. URL: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-019-3312-5 (дата звернення: 18.10.2023).

15. Non-Negative Matrix Factorization. Geeks for Geeks : веб-сайт. URL: https://www.geeksforgeeks.org/non-negative-matrix-factorization/ (дата звернення: 18.10.2023).

16. Understanding Singular Value Decomposition and its Application in Data Science. Towards Data Science : веб-сайт. URL: https://towardsdatascience.com/understanding-singular-value-decomposition-and-its-application-in-data-science-388a54be95d (дата звернення: 19.10.2023).

17. Using SVD for Dimensionality Reduction. Oracle Blogs : веб-сайт. URL: https://blogs.oracle.com/machinelearning/post/using-svd-for-dimensionality-reduction (дата звернення: 19.10.2023).

18. Singular Value Decomposition. Geeks for Geeks : веб-сайт. URL: https://www.geeksforgeeks.org/singular-value-decomposition-svd/ (дата звернення: 19.10.2023).

19. CUR Decompositions, Similarity Matrices, and Subspace Clustering. Frontiers : веб-сайт. URL: https://www.frontiersin.org/articles/10.3389/fams.2018.00065/full (дата звернення: 19.10.2023).

20. CUR matrix decompositions for improved data analysis. PNAS : веб-сайт. URL: https://www.pnas.org/doi/10.1073/pnas.0803205106 (дата звернення: 19.10.2023).

21. Perspectives on CUR decompositions. ScienceDirect : веб-сайт. URL: https://www.sciencedirect.com/science/article/abs/pii/S1063520319300867 (дата звернення: 19.10.2023).

22. What, Why and How of t-SNE. Towards Data Science : веб-сайт. URL: https://towardsdatascience.com/what-why-and-how-of-t-sne-1f78d13e224d (дата звернення: 20.10.2023).

23. Introduction to t-SNE. DataCamp : веб-сайт. URL: https://www.datacamp.com/tutorial/introduction-t-sne (дата звернення: 20.10.2023).

24. How to use t-SNE for dimensionality reduction? AIM : веб-сайт. URL: https://analyticsindiamag.com/how-to-use-t-sne-for-dimensionality-reduction/ (дата звернення: 20.10.2023).

25. A Step-By-Step Complete Guide to Principal Component Analysis. Turing : веб-сайт. URL: https://www.turing.com/kb/guide-to-principal-component-analysis (дата звернення: 10.11.2023).

26. Recommendation System. Nvidia : веб-сайт. URL: https://www.nvidia.com/en-us/glossary/data-science/recommendation-system/ (дата звернення: 15.11.2023).

27. Understanding K-means Clustering in Machine Learning. Medium : веб-сайт. URL: https://towardsdatascience.com/understanding-k-means-clustering-in-machine-learning-6a6e67336aa1/ (дата звернення: 15.11.2023).

28. Image decoding on the web. Dexecure : веб-сайт. URL: https://dexecure.com/blog/image-decoding/ (дата звернення: 15.11.2023).

ДОДАТКИ

Додаток А

Програмний код, який використовувався для дослідження методів редукції

1. Рекомендаційна система.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
from sklearn.decomposition import TruncatedSVD
from cur import cur_decomposition

df = pd.read_csv('/kaggle/input/product-dataset/ratings_Beauty.csv')
df = df.sort_values(by='ProductId')
df.head(10)

ratings_utility_matrix = df.head(10000).pivot_table(values='Rating', index='UserId', columns='ProductId', fill_value=0)
X = ratings_utility_matrix.T
X.head(10)

i = "0205616461"
k = 100

SVD = TruncatedSVD(n_components=k)
decomposed_matrix = SVD.fit_transform(X)
correlation_matrix = np.corrcoef(decomposed_matrix)

product_names = list(X.index)
product_ID = product_names.index(i)
correlation_product_ID = correlation_matrix[product_ID]
Recommend = list(X.index[correlation_product_ID > 0.99])
Recommend.remove(i)
Recommend

def recommend_products(k):
    pca = PCA(n_components=k)
    decomposed_matrix = pca.fit_transform(X)
    correlation_matrix = np.corrcoef(decomposed_matrix)
    product_names = list(X.index)
    product_ID = product_names.index(i)
    correlation_product_ID = correlation_matrix[product_ID]
    recommend_products = list(X.index[correlation_product_ID > 0.9])
    recommend_products.remove(i)
    return len(recommend_products)

k_values = list(range(10, 400, 10))
recommend_counts = []
for k_val in k_values:
    result = recommend_products(k_val)
    recommend_counts.append(result)
plt.plot(k_values, recommend_counts)
plt.title('PCA')
plt.xlabel('Кількість головних компонент')
plt.ylabel('Кількість рекомендованих продуктів')
plt.grid(True)
```

```
plt.show()

def recommend_products(k):
    SVD = TruncatedSVD(n_components=k)
    decomposed_matrix = SVD.fit_transform(X)
    correlation_matrix = np.corrcoef(decomposed_matrix)
    product_names = list(X.index)
    product_ID = product_names.index(i)
    correlation_product_ID = correlation_matrix[product_ID]
    recommend_products = list(X.index[correlation_product_ID > 0.9])
    recommend_products.remove(i)
    return len(recommend_products)

k_values = list(range(2, 50, 1))
recommend_counts = []
for k_val in k_values:
    result = recommend_products(k_val)
    recommend_counts.append(result)
plt.plot(k_values, recommend_counts)
plt.title('SVD')
plt.xlabel('Кількість дисперсій')
plt.ylabel('Кількість рекомендованих продуктів')
plt.grid(True)
plt.show()

def recommend_products(k):
    C, U, R = cur_decomposition(np.array(X+1), k)
    decomposed_matrix = C
    correlation_matrix = 1 - (np.corrcoef(decomposed_matrix))
    product_names = list(X.index)
    product_ID = product_names.index(i)
    correlation_product_ID = correlation_matrix[product_ID]
    recommend_products = list(X.index[correlation_product_ID > 0.9])
    return len(recommend_products)

k_values = list(range(2, 50, 1))
recommend_counts = []
for k_val in k_values:
    result = recommend_products(k_val)
    recommend_counts.append(result)
plt.plot(k_values, recommend_counts)
plt.title('CUR')
plt.xlabel('Ранг U матриці')
plt.ylabel('Кількість рекомендованих продуктів')
plt.grid(True)
plt.show()
```

2. Кластеризація текстових документів.

```
import os
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.decomposition import TruncatedSVD


def create_dataframe(folder_path):
    data = []
    for root, dirs, files in os.walk(folder_path):
        for file in files:
            if file.endswith('.txt'):
                file_path = os.path.join(root, file)
                with open(file_path, 'r', encoding='utf-8') as f:
                    text = f.read()
                data.append({'File Name': file, 'Text': text})
    df = pd.DataFrame(data)
    return df
folder_path = '/kaggle/input/klaster-dataset'
dataframe = create_dataframe(folder_path)
print(dataframe)

vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(dataframe['Text'])
pca = PCA(n_components=100)
X_pca = pca.fit_transform(X.toarray())
kmeans = KMeans(n_clusters=10, random_state=42, n_init=10)
dataframe['Cluster'] = kmeans.fit_predict(X_pca)
cluster_counts_original = dataframe['Cluster'].value_counts().sort_index()
print(cluster_counts_original)

def PCA_kluster(k):
    pca = PCA(n_components=k)
    X_pca = pca.fit_transform(X.toarray())

    kmeans = KMeans(n_clusters=10, random_state=42, n_init=10)
    dataframe['Cluster'] = kmeans.fit_predict(X_pca)

    cluster_counts = dataframe['Cluster'].value_counts().sort_index()
    cluster_counts_array = cluster_counts.values
    return cluster_counts_array

k_values = list(range(2, 50, 2))
result_arrays_pda = []
for k in k_values:
    result_arrays_pda.append(PCA_kluster(k))

plt.figure(figsize=(10, 6))
for i in range(10):
    plt.plot(k_values, [arr[i] for arr in result_arrays_pda], label=f'Кластер {i+1}')
plt.xlabel('Кількість головних компонент')
plt.ylabel('Кількість статей в кластері')
plt.title('PCA')
plt.legend()
plt.grid(True)
plt.show()
```

```
def SVD_kluster(k):
    svd = TruncatedSVD(n_components=k)
    X_svd = svd.fit_transform(X.toarray())

    kmeans = KMeans(n_clusters=10, random_state=42, n_init=10)
    dataframe['Cluster'] = kmeans.fit_predict(X_svd)

    cluster_counts = dataframe['Cluster'].value_counts().sort_index()
    cluster_counts_array = cluster_counts.values
    return cluster_counts_array

k_values = list(range(2, 50, 2))
result_arrays_svd = []
for k in k_values:
    result_arrays_svd.append(SVD_kluster(k))

plt.figure(figsize=(10, 6))
for i in range(10):
    plt.plot(k_values, [arr[i] for arr in result_arrays_svd], label=f'Кластер {i+1}')
plt.xlabel('Кількість дисперсій')
plt.ylabel('Кількість статей в кластері')
plt.title('SVD')
plt.legend()
plt.grid(True)
plt.show()

from cur import cur_decomposition

def CUR_kluster(k):
    C, U, R = cur_decomposition(X.toarray(), k)
    X_cur = C

    kmeans = KMeans(n_clusters=10, random_state=42, n_init=10)
    dataframe['Cluster'] = kmeans.fit_predict(X_cur)

    cluster_counts = dataframe['Cluster'].value_counts().sort_index()
    cluster_counts_array = cluster_counts.values
    return cluster_counts_array

k_values = list(range(2, 50, 2))
result_arrays_cur = []
for k in k_values:
    result_arrays_cur.append(CUR_kluster(k))

plt.figure(figsize=(10, 6))
for i in range(10):
    plt.plot(k_values, [arr[i] for arr in result_arrays_cur], label=f'Кластер {i+1}')
plt.xlabel('Ранг U матриці')
plt.ylabel('Кількість статей в кластері')
plt.title('CUR')
plt.legend()
plt.grid(True)
plt.show()

for i in range(10):
    plt.figure(figsize=(10, 6))
    plt.axhline(y=cluster_counts_original[i], color='r', linestyle='--', label='NO')
    plt.plot(k_values, [arr[i] for arr in result_arrays_pda], label='PCA')
    plt.plot(k_values, [arr[i] for arr in result_arrays_svd], label='SVD')
    plt.plot(k_values, [arr[i] for arr in result_arrays_cur], label='CUR')
    plt.xlabel('k')
    plt.ylabel('Кількість статей в кластері')
```

```
    plt.title(f'Кластер {i+1}')
    plt.legend()
    plt.grid(True)
    plt.show()

    print('PCA')
    mae = np.mean(np.abs(result_arrays_pda - cluster_counts_original[i]))
    print(f"Середня абсолютна похибка: {mae}")

    print('SVD')
    mae = np.mean(np.abs(result_arrays_svd - cluster_counts_original[i]))
    print(f"Середня абсолютна похибка: {mae}")

    print('CUR')
    mae = np.mean(np.abs(result_arrays_cur - cluster_counts_original[i]))
    print(f"Середня абсолютна похибка: {mae}")

import seaborn as sns
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X.toarray())

kmeans = KMeans(n_clusters=10, random_state=42, n_init=10)
dataframe['Cluster'] = kmeans.fit_predict(X_pca)

plt.figure(figsize=(10, 6))
sns.scatterplot(x=X_pca[:, 0], y=X_pca[:, 1], hue=dataframe['Cluster'], palette='viridis', s=50)
plt.title('Візуалізація кластеризації з PCA')
plt.xlabel('Головний компонент 1')
plt.ylabel('Головний компонент 2')
plt.legend(title='Кластер')
plt.show()

svd = TruncatedSVD(n_components=2)
X_svd = svd.fit_transform(X.toarray())

kmeans = KMeans(n_clusters=10, random_state=42, n_init=10)
dataframe['Cluster'] = kmeans.fit_predict(X_svd)

plt.figure(figsize=(10, 6))
sns.scatterplot(x=X_pca[:, 0], y=X_pca[:, 1], hue=dataframe['Cluster'], palette='viridis', s=50)
plt.title('Візуалізація кластеризації з SVD')
plt.xlabel('Дисперсія 1')
plt.ylabel('Дисперсія 2')
plt.legend(title='Кластер')
plt.show()

C, U, R = cur_decomposition(X.toarray(), 2)
X_cur = C

kmeans = KMeans(n_clusters=10, random_state=42, n_init=10)
dataframe['Cluster'] = kmeans.fit_predict(X_cur)
print(dataframe['Cluster'])
plt.figure(figsize=(10, 6))
sns.scatterplot(x=X_pca[:, 0], y=X_pca[:, 1], hue=dataframe['Cluster'], palette='viridis', s=50)
plt.title('Візуалізація кластеризації з CUR')
plt.xlabel('Ранг 1')
plt.ylabel('Ранг 2')
plt.legend(title='Кластер')
plt.show()
```

3. Кодування та декодування зображень.

```
import os
import cv2
import time
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.image import imread
from sklearn.decomposition import PCA
from cur import cur_decomposition

path = "/kaggle/input/photo-test/test.jpg"
image = cv2.imread(path)
print("Розмір матриці фото з трьома каналами: " + str(image.shape))
plt.figure(figsize=[12,8])
plt.title("Оригінальне зображення")
plt.imshow(cv2.cvtColor(image, cv2.COLOR_BGR2RGB))
plt.show()

red_channel = image[:, :, 0]
green_channel = image[:, :, 1]
blue_channel = image[:, :, 2]
plt.figure(figsize=[18,8])
plt.subplot(1, 3, 1)
plt.imshow(red_channel, cmap='Reds')
plt.title('Червоний канал')
plt.subplot(1, 3, 2)
plt.imshow(green_channel, cmap='Greens')
plt.title('Зелений канал')
plt.subplot(1, 3, 3)
plt.imshow(blue_channel, cmap='Blues')
plt.title('Синій канал')
plt.show()

pca = PCA(n_components=k)
pca_red_channel = np.round(pca.fit_transform(red_channel))
pca_inverse_red_channel = np.round(pca.inverse_transform(pca_red_channel))
pca_green_channel = np.round(pca.fit_transform(green_channel))
pca_inverse_green_channel = np.round(pca.inverse_transform(pca_green_channel))
pca_blue_channel = np.round(pca.fit_transform(blue_channel))
pca_inverse_blue_channel = np.round(pca.inverse_transform(pca_blue_channel))

plt.figure(figsize=[18,8])
plt.subplot(1, 3, 1)
plt.imshow(pca_inverse_red_channel, cmap='Reds')
plt.title('Канал червоного кольору')
plt.subplot(1, 3, 2)
plt.imshow(pca_inverse_green_channel, cmap='Greens')
plt.title('Канал зеленого кольору')
plt.subplot(1, 3, 3)
plt.imshow(pca_inverse_blue_channel, cmap='Blues')
plt.title('Канал синього кольору')
plt.show()

pca_merged_image = cv2.merge([pca_inverse_red_channel.astype(np.uint8),
pca_inverse_green_channel.astype(np.uint8), pca_inverse_blue_channel.astype(np.uint8)])
pca_merged_image = np.round(pca_merged_image)
save_merged_file = '/kaggle/working/mergedPCAimage.jpg'
cv2.imwrite(save_merged_file, pca_merged_image)
print("Розмір матриці фото з трьома каналами: " + str(pca_merged_image.shape))
plt.figure(figsize=[12,8])
plt.title("PCA Об\'єднання каналів")
```

```
plt.imshow(cv2.cvtColor(pca_merged_image, cv2.COLOR_BGR2RGB))
plt.show()

res = cv2.absdiff(image, pca_merged_image)
res = res.astype(np.uint8)
percentage = 100 - (np.count_nonzero(res) * 100)/ res.size
print("Схожість нового зображення після PCA з оригиіналом: " + str(percentage) + " %")

U, sigma, V = np.linalg.svd(red_channel, k)
svd_inverse_red_channel = np.matrix(U[:, :k]) * np.diag(sigma[:k]) * np.matrix(V[:k, :])
U, sigma, V = np.linalg.svd(green_channel, k)
svd_inverse_green_channel = np.matrix(U[:, :k]) * np.diag(sigma[:k]) * np.matrix(V[:k, :])
U, sigma, V = np.linalg.svd(blue_channel, k)
svd_inverse_blue_channel = np.matrix(U[:, :k]) * np.diag(sigma[:k]) * np.matrix(V[:k, :])

plt.figure(figsize=[18,8])
plt.subplot(1, 3, 1)
plt.imshow(svd_inverse_red_channel, cmap='Reds')
plt.title('Канал червоного кольору')
plt.subplot(1, 3, 2)
plt.imshow(svd_inverse_green_channel, cmap='Greens')
plt.title('Канал зеленого кольору')
plt.subplot(1, 3, 3)
plt.imshow(svd_inverse_blue_channel, cmap='Blues')
plt.title('Канал синього кольору')
plt.show()

svd_merged_image = cv2.merge([svd_inverse_red_channel.astype(np.uint8),
svd_inverse_green_channel.astype(np.uint8), svd_inverse_blue_channel.astype(np.uint8)])
save_merged_file = '/kaggle/working/mergedSVDimage.jpg'
cv2.imwrite(save_merged_file, svd_merged_image)

print("Розмір матриці фото з трьома каналами: " + str(svd_merged_image.shape))
plt.figure(figsize=[12,8])
plt.title("SVD Об\'єднання каналів")
plt.imshow(cv2.cvtColor(svd_merged_image, cv2.COLOR_BGR2RGB))
plt.show()

res = cv2.absdiff(image, svd_merged_image)
res = res.astype(np.uint8)
percentage = 100 - (np.count_nonzero(res) * 100)/ res.size
print("Схожість нового зображення з оригиіналом: " + str(percentage) + " %")

C, U, R = cur_decomposition(red_channel, k)
cur_inverse_red_channel = np.matrix(C[:, :k]) * np.matrix(U) * np.matrix(R[:k, :])
C, U, R = cur_decomposition(green_channel, k)
cur_inverse_green_channel = np.matrix(C[:, :k]) * np.matrix(U) * np.matrix(R[:k, :])
C, U, R = cur_decomposition(blue_channel, k)
cur_inverse_blue_channel = np.matrix(C[:, :k]) * np.matrix(U) * np.matrix(R[:k, :])

plt.figure(figsize=[18,8])

plt.subplot(1, 3, 1)
plt.imshow(cur_inverse_red_channel, cmap='Reds')
plt.title('Канал червоного кольору')
plt.subplot(1, 3, 2)
plt.imshow(cur_inverse_green_channel, cmap='Greens')
plt.title('Канал зеленого кольору')
plt.subplot(1, 3, 3)
plt.imshow(cur_inverse_blue_channel, cmap='Blues')
plt.title('Канал синього кольору')
plt.show()
```

```
merged_image = cv2.merge([cur_inverse_red_channel.astype(np.uint8), cur_inverse_green_channel.astype(np.uint8), cur_inverse_blue_channel.astype(np.uint8)])
save_merged_file = '/kaggle/working/mergedCURimage.jpg'
cv2.imwrite(save_merged_file, merged_image)
print("Розмір матриці фото з трьома каналами: " + str(merged_image.shape))
plt.figure(figsize=[12,8])
plt.title("CUR Об\'єднання каналів")
plt.imshow(cv2.cvtColor(merged_image, cv2.COLOR_BGR2RGB))
plt.show()

res = cv2.absdiff(image, merged_image)
res = res.astype(np.uint8)
percentage = 100 - (np.count_nonzero(res) * 100)/ res.size
print("Схожість нового зображення з оригиіналом: " + str(percentage) + " %")

def compare(img1, img2):
    res = cv2.absdiff(img1, img2)
    res = res.astype(np.uint8)
    percentage = 100 - (np.count_nonzero(res) * 100)/ res.size
    return percentage

def reducing(k):
    pca = PCA(n_components=k)
    pca_red_channel = np.round(pca.fit_transform(red_channel))
    pca_inverse_red_channel = np.round(pca.inverse_transform(pca_red_channel))
    pca_green_channel = np.round(pca.fit_transform(green_channel))
    pca_inverse_green_channel = np.round(pca.inverse_transform(pca_green_channel))
    pca_blue_channel = np.round(pca.fit_transform(blue_channel))
    pca_inverse_blue_channel = np.round(pca.inverse_transform(pca_blue_channel))
    merged_image = cv2.merge([pca_inverse_red_channel.astype(np.uint8), pca_inverse_green_channel.astype(np.uint8), pca_inverse_blue_channel.astype(np.uint8)])
    return compare(image, merged_image)

k_values = list(range(50, 1200, 50))
image_quality = []

for k_val in k_values:
    image_quality.append(reducing(k_val))

plt.plot(k_values, image_quality)
plt.title('PCA')
plt.xlabel('Кількість головних компонент')
plt.ylabel('Схожість з оригіналом, %')
plt.grid(True)
plt.show()

def reducing(k):
    U, sigma, V = np.linalg.svd(red_channel, k)
    svd_inverse_red_channel = np.matrix(U[:, :k]) * np.diag(sigma[:k]) * np.matrix(V[:k, :])
    U, sigma, V = np.linalg.svd(green_channel, k)
    svd_inverse_green_channel = np.matrix(U[:, :k]) * np.diag(sigma[:k]) * np.matrix(V[:k, :])
    U, sigma, V = np.linalg.svd(blue_channel, k)
    svd_inverse_blue_channel = np.matrix(U[:, :k]) * np.diag(sigma[:k]) * np.matrix(V[:k, :])
    merged_image = cv2.merge([svd_inverse_red_channel.astype(np.uint8), svd_inverse_green_channel.astype(np.uint8), svd_inverse_blue_channel.astype(np.uint8)])
    return compare(image, merged_image)

k_values = list(range(50, 1200, 50))
image_quality = []

for k_val in k_values:
```

```
    image_quality.append(reducing(k_val))

plt.plot(k_values, image_quality)
plt.title('SVD')
plt.xlabel('Кількість дисперсій')
plt.ylabel('Схожість з оригіналом, %')
plt.grid(True)
plt.show()

def reducing(k):
    C, U, R = cur_decomposition(red_channel, k)
    cur_inverse_red_channel = np.matrix(C[:, :k]) * np.matrix(U) * np.matrix(R[:k, :])
    C, U, R = cur_decomposition(green_channel, k)
    cur_inverse_green_channel = np.matrix(C[:, :k]) * np.matrix(U) * np.matrix(R[:k, :])
    C, U, R = cur_decomposition(blue_channel, k)
    cur_inverse_blue_channel = np.matrix(C[:, :k]) * np.matrix(U) * np.matrix(R[:k, :])
    merged_image = cv2.merge([cur_inverse_red_channel.astype(np.uint8),
cur_inverse_green_channel.astype(np.uint8), cur_inverse_blue_channel.astype(np.uint8)])
    return compare(image, merged_image)

k_values = list(range(50, 1200, 50))
image_quality = []

for k_val in k_values:
    image_quality.append(reducing(k_val))

plt.plot(k_values, image_quality)
plt.title('CUR')
plt.xlabel('Ранг U матриці')
plt.ylabel('Схожість з оригіналом, %')
plt.grid(True)
plt.show()
```