

A FIELD PROJECT REPORT

on

**“PRODUCT RECOMMENDATION SYSTEM FOR
E-COMMERCE”**

Submitted

by

221FA04103

G.Javali

221FA04463

N.Sri Lakshmi

221FA04104

SK.Sameena

221FA04471

M.Jhansi

Under the guidance of

Sajida Sultana.Sk

Assistant Professor



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

VIGNAN'S FOUNDATION FOR SCIENCE, TECHNOLOGY AND RESEARCH Deemed

to be UNIVERSITY

Vadlamudi, Guntur.

ANDHRA PRADESH, INDIA, PIN-522213.

CERTIFICATE

This is to certify that the Field Project entitled “**Product Recommendation System For E-Commerce**” that is being submitted by 221FA04103(G.Javali), 221FA04104(SK.Sameena), 221FA04463(N.Sri Lakshmi), 221FA04471(M.Jhansi) for partial fulfilment of Field Project is a bonafide work carried out under the supervision of Ms. Sajida Sultana.Sk , Assistant Professor, Department of CSE.

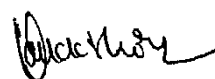
Guide name& Signature



Dr. S. V. Phani Kumar

Assistant/Associate/Professor,
CSE

HOD,CSE



Dr.K.V. Krishna Kishore

Dean, SoCI



DECLARATION

We hereby declare that the Field Project entitled “**Product Recommendation System For E-Commerce**” that is being submitted by 221FA04103(G.Javali), 221FA04104(SK.Sameena), 221FA04463(N.Sri Lakshmi), 221FA04471(M.Jhansi) in partial fulfilment of Field Project course work. This is our original work, and this project has not formed the basis for the award of any degree. We have worked under the supervision of Ms. Sajida Sultana.Sk ,, Assistant Professor, Department of CSE.

By

221FA04103 (G.Javali),
221FA04104 (SK.Sameena),
221FA04463 (N.Sri Lakshmi),
221FA04471 (M.Jhansi)

Date:

ABSTRACT

Due to rapid growth in e-commerce, the interest for customized product recommendation systems has grown a lot with high demands for effective models. In this research study, an attempt is made to explore the development and evaluation of a personalized product recommendation model using the H&M data set. The research highlights the building up of an interaction matrix between a user and items, generation of recommendations suited to the tastes of a particular user, and the hyper parameter tuning of the model for better performance. Hybrid Multiple Recommendation Technique Different techniques have been utilized, including KNNBasic, Non-negative Matrix Factorization (NMF), Co - Clustering, and Singular Value Decomposition (SVD). The KNNBasic model had a root mean square error (RMSE) of 0.5022 with an accuracy of 42.00%, the NMF model showed better results with an RMSE of 0.4999 and accuracy of 51.50%. Co-Clustering showed the result of RMSE as 0.5000 and accuracy was 50.50%. Notably, the final SVD model ranked very well compared to the others with an RMSE 0.2261 and a great accuracy of 90.40% in this experiment, emphasizing the importance of advanced techniques in recommendation systems. In these experiments, not only is the relative efficacy of different recommendation algorithms evident but also that optimization of hyper parameters genuinely contributes to increasing predictive precision. This work illuminates how effective personalized recommendations are, which has implications for the future evolution of e-commerce strategies.

Key Words-Machine Learning, KNN, NMF, Co-Clustering, SVD, Accuracy, RMSE

TABLE OF CONTENTS

1. Introduction	1
1.1 What is a product recommendation system?	2
1.2 Importance of personalized recommendations in e-commerce	3
1.3 Machine Learning in E-commerce Recommendations	3
1.4 Challenges in building recommendation systems	3
1.5 Applications of recommendation systems in various domains	3
2. Literature Survey	4
2.1 Literature Review of Existing Algorithms and Techniques	5
2.2 Motivation behind this project	8
3. Proposed System	9
3.1 Input dataset	11
3.1.1 Detailed features of dataset	12
3.2 Data Pre-processing	13
3.2.1 Missing values	14
3.2.2 Encoding categorical data	14
3.3 Model Building	14
3.4 Methodology of the system	17
3.5 Model Evaluation	19
3.6 Hyper parameter Tuning	22
3.7 Performance Metrics	22
4. Implementation	24
4.1 Environment Setup	25
4.2 Sample code for data pre-processing and model implementation	25
5. Experimentation and Result Analysis	27
6. Conclusion and Future Enhancement	35
7. References	38

LIST OF FIGURES

Figure 1.1 Flow Diagram for Recommendation System	2
Figure 3.1 Articles Dataset	11
Figure 3.2 Customer Dataset	11
Figure 3.3 Transaction Dataset	11
Figure 3.4 Flow Diagram for Recommendation System	13
Figure 3.5 SVD Architecture	19
Figure 4.1 Data processing	25
Figure 4.2 SVD code	26
Figure 5.1 Top recommendations on SVD model	28
Figure 5.2 KNN Top recommendations	29
Figure 5.3 NMF Top recommendations	29
Figure 5.4 CO-CLUSTERING Top recommendations	29
Figure 5.5 Comparison of Article ID vs Price	30
Figure 5.6 correlation Matrix	31
Figure 5.7 distribution of fashion news subscription frequency	32
Figure 5.8 Interaction matrix	33
Figure 5.9 RMSE and Accuracy comparison	34
Figure 5.10 classification metrics	34

LIST OF TABLES

Table 3.1 Performance Comparison of Recommendation Models

23

CHAPTER-1

INTRODUCTION

1. INTRODUCTION

This section will introduce the concept of product recommendation systems, focusing on their importance in e-commerce. It will explain how personalized product recommendations are essential for enhancing user experience, driving customer loyalty, and improving sales. The introduction will provide an overview of the role machine learning plays in these systems and how e-commerce companies rely on them for competitive advantage.

1.1 What is a product recommendation system?

A product recommendation system is an application of machine learning algorithms that helps users discover products they might be interested in by analysing their behaviour, preferences, and past interactions. These systems can range from simple algorithms that suggest popular items to complex models that predict user preferences with great accuracy.

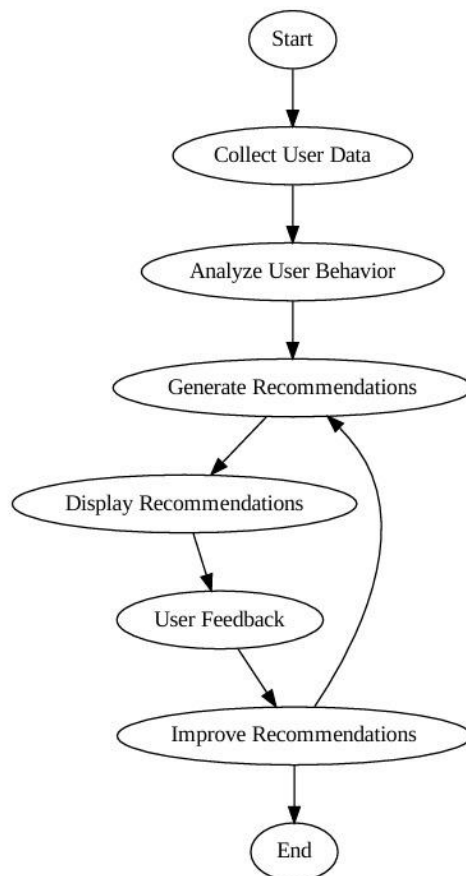


Figure 1.1 Flow Diagram for Recommendation System

1.2 Importance of personalized recommendations in e-commerce

Personalized recommendations are vital in e-commerce as they not only enhance the customer shopping experience but also increase the chances of purchase. By recommending products that are relevant to the customer's interests, companies can encourage repeat purchases, reduce cart abandonment rates, and improve overall customer satisfaction.

1.3 The impact of machine learning in recommendation systems

Machine learning (ML) enables recommendation systems to learn from vast amounts of data, continually improving and adapting to new patterns in user behaviour. This section will discuss:

- **Supervised learning** algorithms like Decision Trees and Random Forests, which can classify user behaviour based on historical data.
- **Unsupervised learning** techniques like Clustering and Matrix Factorization, which reveal hidden patterns in data.

1.4 Challenges in building recommendation systems

Several challenges arise in building an efficient recommendation system:

- **Data Sparsity:** Most users interact with a small subset of products, creating sparse interaction matrices. This can limit the system's ability to make accurate predictions.
- **Cold Start Problem:** New users or new products have little or no historical data, making it difficult to provide accurate recommendations.
- **Scalability:** As the number of users and products grows, the system must efficiently handle massive datasets and perform real-time recommendations.

1.5 Applications of recommendation systems in various domains

Recommendation systems are not limited to e-commerce but are also widely used in:

- **Streaming services** (Netflix, Spotify) to recommend movies and music,
- **Social media platforms** (Facebook, Instagram) for content and friend suggestions,
- **News websites** to personalize articles based on users' reading habits,
- **Online learning platforms** to suggest courses based on learner preferences.

CHAPTER-2

LITERATURE SURVEY

2. LITERATURE SURVEY

2.1 Literature review

In recent years, there has been a growing interest in leveraging machine learning to develop product recommendation systems for e-commerce. Various machine learning models have been employed to enhance the accuracy of recommendations by analysing user behaviours, preferences, and purchasing patterns. These systems aim to improve personalization, foster customer engagement, and drive sales by suggesting products that resonate with individual shoppers. As the field continues to advance, it becomes increasingly important for data scientists, software engineers, e-commerce professionals, and marketing experts to collaborate in creating innovative and scalable recommendation systems that meet the ever-changing demands of both businesses and consumers.

Lifeng Kang, Yankun Wang[1] In their work, the authors present a fusion recommendation algorithm designed to address common challenges in e-commerce, such as data sparsity and cold starts. The approach involves three key steps: filtering data, ranking user interests, and identifying product similarities using frequent item set mining. By applying collaborative filtering to product categories, the model boosts both accuracy and scalability. Tests on both synthetic and real-world datasets demonstrate the algorithm's ability to handle large datasets efficiently. Moving forward, the focus will be on refining how features are weighted and expanding testing to various e-commerce platforms to improve real-time adaptability and accuracy.

Ataur Rahman, Zahurul Haque, et al.[2] In this paper, the authors introduce a recommendation system that employs machine learning algorithms such as Random Forest, Decision Tree, Gaussian Naive Bayes, and Logistic Regression to provide personalized product suggestions. By applying Principal Component Analysis (PCA) to streamline the feature set, they found that the Random Forest model achieved an impressive accuracy of 99.6%. Specifically designed for a Google Form survey-based e-commerce dataset, this model has shown effectiveness across various e-commerce scenarios. Looking ahead, the authors plan to explore the integration of more advanced machine learning models and test the system on larger datasets to enhance scalability.

Nguyen, Van-Ho Nguyen, et al.[3] In this paper they presented an innovative recommendation engine tailored for large-scale e-commerce environments. This model integrates collaborative filtering, Bayesian Personalized Ranking (BPR), and popularity-based recommendations, employing both LightGBM and Deep Neural Networks (DNNs) for evaluation. The results indicate that LightGBM outperforms DNNs in terms of MAP@K and MAR@K, highlighting its effectiveness in handling cold-start issues and scaling with big data. Future research includes real-world testing, A/B testing for performance assessment, and further optimization of model parameters to enhance recommendation quality.

Manal Loukili, Fayçal Messaoudi, et al.[4] In this paper, the authors present a detailed approach to enhancing e-commerce platforms by implementing a personalized product recommendation system. They use a combination of collaborative filtering, popularity-based recommendations, and Bayesian Personalized Ranking (BPR) to build a strong recommendation engine. Their approach integrates multiple algorithmic techniques and evaluates the generated recommendations using machine learning models, specifically LightGBM and Deep Neural Networks (DNNs). The findings show that the LightGBM model outperformed DNNs, particularly in terms of mean average precision and recall at K candidates (MAP@K and MAR@K). This work tackles key challenges like large-scale data processing, cold-start issues, and personalization, ultimately improving user experience and boosting sales for e-commerce platforms.

Liping Liu[5] In this paper, the authors introduce a hybrid recommendation system that integrates user-based collaborative filtering, content-based filtering, and real-time processing algorithms to tailor e-commerce recommendations to individual users. By analyzing user profiles and leveraging dynamic data, the system delivers timely and accurate suggestions that adapt to shifts in user preferences. Looking ahead, the authors propose incorporating deep learning techniques, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), to gain deeper insights into complex user behaviours and further enhance the recommendation process.

Chibuzor Udokwu, Robert Zimmermann, et al.[6] In this paper they proposed a system that combines association rule mining with K-means clustering to generate personalized product recommendations. The study uses customer transactional data and demographic information to enhance the relevance of the recommendations. By linking customer profiles to product associations, the system addresses cold-start issues and provides tailored recommendations for

different customer segments. Future work includes exploring advanced clustering techniques and expanding the model to support multilingual and multi-category environments.

Muhammad Tahir, Rabia Noor Enam, et al. [7] In this paper, the authors developed a content-based recommendation system using the Nearest Neighbour algorithm, specifically aimed at recommending apparel. The system creates user profiles by analyzing product attributes like colour and type to deliver more personalized suggestions. They suggest that future research could explore hybrid methods that combine content-based and collaborative filtering, aiming to improve both the accuracy of recommendations and overall user satisfaction on e-commerce platforms.

Abhiraj Biswas, Kaza Sai Vineeth, et al. [8] In this paper, the authors introduce a product recommendation engine that merges collaborative filtering with association rule mining to improve the effectiveness of e-commerce recommendations. By leveraging customer purchase history, they apply both user-based and item-based collaborative filtering, while also using association rules to enhance cross-selling opportunities. Tested on large datasets, the system effectively balances both accuracy and scalability. Looking ahead, the authors plan to fine-tune the model's algorithm parameters to better handle larger datasets and increase the speed of generating recommendations.

Bei Hui, Lizong Zhang, et al. [9] In this paper they address e-commerce challenges like data sparsity and cold start by integrating knowledge graphs with user historical behaviour. By applying graph attention networks and embedding techniques, the authors build a system that enhances the recommendation process, improving both user experience and personalization. Their results show that this integration improves accuracy, while future research may focus on refining multi-hop knowledge paths for richer contextual recommendations.

Alejandro Valencia-Arias, Hernan Uribe-Bedoya, et al. [10] In this paper, the authors conducted an in-depth bibliometric analysis examining the impact of AI on e-commerce recommender systems. They highlight emerging trends such as the integration of neural networks, sentiment analysis, and knowledge graphs to improve recommendation engines. The study suggests that future research should explore the use of multi-modal data to provide a more comprehensive and engaging user experience. Additionally, the authors emphasize the need to enhance sentiment analysis within recommendation systems to better capture and respond to user preferences and needs.

Wei Zhang, Zonghua Wu[11] In this paper, the authors present an enhanced K-means clustering algorithm aimed at improving the management of product information on ecommerce platforms. By integrating a genetic algorithm to optimize the clustering process, the system achieves an impressive accuracy rate of 91.1% in product recommendations, surpassing traditional K-means and Fuzzy C-means methods. Additionally, the algorithm demonstrates faster convergence, making it highly efficient for handling large-scale e-commerce data. Future research could focus on further optimizing the algorithm and exploring its adaptability across different ecommerce platforms.

Anam Naz, Irum Hafeez,et al.[12] In this paper, the authors provide a comprehensive review of existing methods used in product recommendation systems, including content-based filtering, collaborative filtering, and hybrid approaches. They discuss key challenges such as cold-start issues and insufficient personalization. To tackle these problems, the authors propose future enhancements by integrating social and behavioural data into recommendation algorithms. They advocate for the use of advanced machine learning techniques to refine user profiling and increase recommendation diversity, while also considering the importance of addressing privacy concerns.

Alfredo Daza, Wilmer Filomeno,et al.[13] In this paper, the authors conduct a bibliometric and systematic review of sentiment analysis in product recommendations. They examine the effectiveness of various machine learning algorithms, including Naive Bayes, Support Vector Machines (SVM), and K- Nearest Neighbors (KNN), in analysing e-commerce reviews. The authors suggest several future research directions, such as broadening the evaluation metrics used and applying sentiment analysis to a wider range of application domains. Their goal is to improve the robustness and accuracy of recommendation systems, ensuring they better meet user needs.

2.2 Motivation

The motivation stems from the growing need for efficient, scalable, and personalized recommendation systems in e-commerce. As online platforms continue to expand, providing relevant product recommendations becomes crucial for maintaining user engagement and driving sales.

CHAPTER-3

PROPOSED SYSTEM

3. PROPOSED SYSTEM

The proposed system for the **Product Recommendation Model for E-commerce** focuses on delivering personalized product recommendations by analysing user interactions, preferences, and historical purchasing data. The recommendation system is built using various machine learning techniques, with **Singular Value Decomposition (SVD)** selected as the primary method due to its effectiveness in handling sparse data and producing highly accurate predictions. The system leverages data from the H&M dataset, which includes customer demographic information, product attributes, and transaction histories, forming the foundation for user-item interaction analysis.

In this system, the **interaction matrix** serves as the core structure, representing the relationship between users and products. The matrix captures interactions such as purchases, and since most users only engage with a subset of products, the matrix is typically sparse. SVD is employed to decompose this large and sparse matrix into three smaller matrices, capturing latent factors that influence user preferences and item characteristics. The decomposition allows the model to predict missing interactions by revealing hidden patterns in the data, making it particularly effective in generating personalized recommendations even for products with limited historical interactions.

Data pre-processing plays a vital role in ensuring the quality of the input data. Missing values are addressed through imputation, while categorical data, such as product types and customer attributes, are encoded using techniques like one-hot encoding and label encoding. Once pre-processed, the data is used to create the interaction matrix, which is then factorized using SVD. By reducing the dimensionality of the data, SVD is able to capture the most relevant factors driving user behaviour while improving the computational efficiency of the model. To further optimize the performance of the SVD model, **hyper parameter tuning** is applied. Parameters such as the number of latent factors and regularization terms are fine-tuned using grid search techniques to minimize errors and enhance model accuracy. The model's performance is evaluated using metrics such as **Root Mean Square Error (RMSE)**, with SVD demonstrating superior results by achieving a low RMSE and high accuracy. This proposed system offers a scalable and effective approach to delivering personalized product recommendations, significantly improving the user experience on e-commerce platforms.

3.1 Input dataset

The dataset used in this project is sourced from H&M Group and consists of various components related to customer interactions with products. The key datasets are:

- **Articles Dataset:** Contains detailed information about the products, such as article ID, product name, product code, product type, and appearance-related features like colour, pattern, and material.
- **Customers Dataset:** Provides customer-specific information, including customer ID, demographic data (e.g., age, postal code), club membership status, and fashion news frequency.
- **Transactions Dataset:** Records the history of transactions made by customers, including details like the transaction date, article ID, customer ID, price, and sales channel.

These datasets serve as the foundation for the recommendation system by capturing the interactions between users and products.

TABLE I
ARTICLES DATASET

Article ID	Product Code	Product Name	Product Type Name	Graphical Appearance Features
108775015	108775	Strap top	Vest top	Solid
108775044	108775	Strap top	Vest top	Solid
108775051	108775	Strap top (1)	Vest top	Stripe

Figure 3.1 Articles Dataset

TABLE II
CUSTOMER DATASET

Customer ID	Club Member Status	Fashion News Frequency	Age
00000dbacae5abe5e23885899a1fa44253a17956c6d1c3	ACTIVE	Regularly	49.0
0000423b00ade91418cceaf3b26c6af3dd342b51fd051e	ACTIVE	Regularly	25.0
000058a12d5b43e67d225668fa1f8d618c13dc232df0ca	ACTIVE	Regularly	24.0

Figure 3.2 Customer Dataset

TABLE III
TRANSACTION DATASET

Transaction Date	Customer ID	Article ID	Price	Sales Channel ID	Year-Month
2018-09-20	000058a12d5b43e67d225668fa1f8d618c13dc232df0ca	663713001	0.050831	2	2018-09
2018-09-20	000058a12d5b43e67d225668fa1f8d618c13dc232df0ca	541518023	0.030492	2	2018-09
2018-09-20	00007d2de826758b65a93dd24ce629ed66842531df6699	505221004	0.015237	2	2018-09

Figure 3.3 Transaction Dataset

3.1.1 Detailed Features of the Dataset

Articles Dataset Features:

- **Article ID:** Unique identifier for each product.
- **Product Code:** Product classification code.
- **Product Type:** Category of the product (e.g., shirt, shoes).
- **Graphical Appearance:** Visual attributes such as color and pattern.

Customers Dataset Features:

- **Customer ID:** Unique identifier for each customer.
- **Age:** Age of the customer.
- **Club Membership Status:** Indicates whether a customer is a member of H&M's loyalty program.
- **Fashion News Frequency:** How often a customer receives fashion-related news.

Transactions Dataset Features:

- **Transaction Date:** Date of the purchase.
- **Customer ID:** Links the transaction to a specific customer.
- **Article ID:** Links the transaction to a specific product.
- **Price:** The purchase price of the product.
- **Sales Channel:** The platform through which the transaction occurred (online or in-store)

3.2 Data Pre-processing

Data pre-processing is the essential process of preparing raw data for analysis and modelling by cleaning, transforming, and structuring it to enhance data quality and utility. It involves tasks like handling missing values, correcting errors, encoding features, and scaling data to ensure it's in an optimal form for further analysis. It encompasses a range of operations and transformations designed to refine raw data, ensuring that it is clean, structured, and amenity subsequent analysis. This process is driven by its manifold significance in data science and analysis.

Through meticulous data cleaning, transformation, feature engineering, dimensionality reduction, outlier handling, scaling, and data splitting, it prepares raw data for more accurate and reliable analysis and modelling. Ultimately, the goal is to obtain more meaningful insights, make informed decisions, and optimize predictive models for a wide range of applications in data science and analysis.

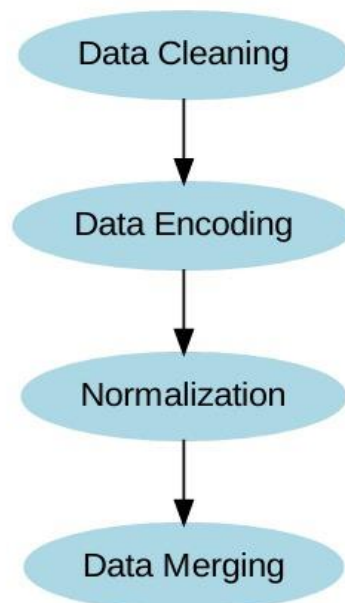


Figure 3.4 Flow Diagram for Recommendation System

3.2.1 Missing Values

Missing values are a common challenge in large datasets. In the provided datasets, several fields (e.g., age, price) may contain missing entries. The missing values are handled through the following approaches:

- **Imputation:** For numerical data like age or price, missing values are replaced with the mean or median of the available data. This ensures that the dataset remains intact without significant distortions.
- **Removal:** Records with missing values in critical fields, such as customer ID or article ID, are removed, as these are essential for creating an interaction matrix.
- **Default Values:** For categorical features, such as club membership status, missing values are replaced with default values (e.g., 'Not a Member') to avoid bias during training.

3.2.2 Encoding Categorical Data

Many machine learning models require numerical input, so categorical features like product type and customer status need to be encoded. The following encoding techniques are used:

- **Label Encoding:** Categorical variables with a limited number of categories (e.g., club membership status, product type) are transformed into numerical labels, where each unique value is assigned a corresponding integer.
- **One-Hot Encoding:** For features with a larger set of unique values, such as product categories or colour, one-hot encoding is applied to create binary columns representing each category. This helps the model interpret the categorical variables without imposing an arbitrary order on them.

3.3 Model Building

The recommendation system for e-commerce is built using a combination of collaborative filtering and matrix factorization techniques. These models help predict user preferences based on their interactions with products, enabling personalized recommendations. The four primary models used in this project are **KNNBasic**, **Non-Negative Matrix Factorization (NMF)**, **Co-Clustering**,

and **Singular Value Decomposition (SVD)**. Each of these methods offers unique advantages in generating accurate recommendations.

KNNBasic (K-Nearest Neighbours):

KNNBasic is a collaborative filtering model that identifies similar users or items based on their past interactions. The algorithm compares users or items using distance measures such as **cosine similarity** or **Pearson correlation**, which capture the similarity between user preferences or item features.

- **Cosine Similarity** measures the angle between two vectors of interactions:

$$\text{Cosine Similarity}(A,B)=\frac{A \cdot B}{\|A\| \|B\|}$$

For example, if two users have similar purchasing patterns, their interaction vectors will be closely aligned, resulting in a high cosine similarity score.

Once the algorithm identifies the most similar users or items, it recommends products based on their preferences. KNNBasic is simple and works well in smaller datasets, but it can become computationally expensive in large datasets as it requires calculating similarities for all users or items.

Non-Negative Matrix Factorization (NMF):

NMF is a matrix factorization technique that breaks down the user-item interaction matrix into two non-negative matrices representing latent features of users and items. These latent factors help reveal hidden patterns in the data, which the system uses to predict user interactions with items.

- **How it works:** NMF decomposes the original matrix M into two matrices W and H :

$$M \approx W \cdot H$$

- Here, W represents the user matrix, and H represents the item matrix. The values in these matrices correspond to latent factors like price sensitivity or brand preference.

NMF is particularly useful in recommendation systems as it captures these underlying factors in a non-negative form, making the model more interpretable. NMF works well for sparse matrices, where there are many users and items with little overlap in interactions.

Co-Clustering:

Co-Clustering is a technique that simultaneously clusters both users and items into groups based on their interactions. Unlike standard clustering methods, which focus on either users or items, co-clustering creates blocks of users and items that exhibit similar interaction behaviours.

- **How it works:** The algorithm partitions the user-item interaction matrix into a grid of clusters, grouping users with similar behaviours and the items they prefer. This results in blocks of users and items, each representing distinct interaction patterns.
- **Recommendation process:** The system recommends items within the same user cluster, meaning that users with similar preferences are grouped, and products from their cluster are suggested. This technique works well for datasets that naturally exhibit group behaviour, where groups of users consistently prefer certain groups of items.

Co-Clustering is advantageous when dealing with large datasets as it reduces the complexity by focusing on clusters instead of individual users or items. It can identify patterns across broader groups and is often faster to compute than other algorithms.

Singular Value Decomposition (SVD):

SVD is one of the most effective matrix factorization techniques used in recommendation systems. It decomposes the interaction matrix into three smaller matrices, capturing latent structures and reducing the dimensionality of the data. SVD excels in generating accurate recommendations by finding hidden relationships between users and items.

- **How it works:** SVD factorizes the interaction matrix M into three matrices:

$$M = U \cdot \Sigma \cdot V^T$$

U represents the user matrix, V^T represents the item matrix, and Σ contains singular values that capture the strength of each latent factor.

SVD is highly effective at reducing the dimensionality of the data while preserving the most important factors that influence user behaviour. These latent factors explain user preferences and item characteristics, allowing SVD to predict missing values in the interaction matrix.

Why SVD is preferred: SVD performs exceptionally well on sparse datasets, which are common in e-commerce platforms where users interact with only a fraction of available products. It captures the key patterns in user behaviour, providing accurate and scalable recommendations. SVD's ability to generalize well makes it the preferred choice for modern recommendation systems, including in large-scale systems like Netflix's recommendation engine.

3.4 Methodology of the system

Now that we've covered the basics, let's move into the main part of our **Product Recommendation System for E-Commerce**. In this section, we'll walk through how the system works, step by step. Just like how every part of a machine has a specific job to make it run smoothly, our system brings together important elements like data, pre-processing, building models, and evaluating the results. By combining these steps, we create a recommendation system that can understand user preferences, make accurate product suggestions, and keep improving over time, giving users a better shopping experience.

SVD Architecture:

Singular Value Decomposition (SVD) is a mathematical technique used in recommendation systems, especially in collaborative filtering. It breaks down large, complex datasets (like user ratings for products) into simpler, smaller components. In essence, SVD helps find hidden patterns in data by compressing the information and then using it to make predictions—like recommending products to users.

The Basics of SVD:

Imagine you have a large matrix (table) where rows represent users, columns represent items (like products or movies), and each cell contains a user's rating or interaction with the item. This table is usually sparse, meaning there are lots of missing values since not every user rates or interacts with every item. SVD comes to the rescue by simplifying this table into three smaller matrices that can still represent the same information, but in a more manageable way.

SVD Decomposition: Three Matrices

SVD breaks down the original user-item matrix into three smaller matrices:

1. **User Matrix (U)** – This represents hidden features of users, like their preferences.
2. **Singular Value Matrix (Σ)** – A diagonal matrix that contains values representing the strength of relationships between users and items.
3. **Item Matrix (V^T)** – This represents hidden features of items, like product characteristics.

When these matrices are multiplied back together, they approximate the original matrix, but with the advantage of filling in missing data (like predicting how a user will rate a product they haven't rated yet).

How SVD Architecture Works:

1. **Input Layer:** The input is the original user-item matrix, where rows represent users and columns represent items. Some values in this matrix are known (e.g., ratings), while others are missing (e.g., unrated products).
2. **Decomposition (Hidden Layers):**

- **First Layer (User Matrix):** This layer breaks down the input into user preferences. Each user is represented by a set of latent factors—features that explain why a user prefers certain products.
 - **Second Layer (Singular Values):** The singular values help in connecting users to items by showing the strength of the relationship between user preferences and product features.
 - **Third Layer (Item Matrix):** This layer represents the latent features of the items themselves, like categories or characteristics that influence user preferences.
3. **Output Layer (Prediction):** After the matrix decomposition, the system multiplies the matrices back together to recreate the original matrix. During this step, it also predicts the missing values in the matrix—such as predicting how much a user might like a product they haven't interacted with yet. The output is the predicted rating or interaction score between a user and an item.

Why SVD is Important in Recommendation Systems:

- **Fills Missing Data:** SVD is great at predicting missing information, like estimating how a user will rate an item they haven't seen before.
- **Simplifies Complex Data:** By breaking the user-item matrix into smaller pieces, SVD reduces the complexity of the data while still preserving important relationships.
- **Captures Latent Features:** Latent features are hidden patterns that aren't directly visible, like a user's affinity for a particular product category. SVD uncovers these hidden patterns and uses them for better predictions.
- **Efficiency:** SVD makes it easier to work with massive datasets, such as millions of users and products, by compressing the data into a smaller, more manageable form.

Key Benefits of SVD Architecture:

1. **Personalization:** By analysing user behaviour and product features, SVD provides highly personalized recommendations tailored to each user's preferences.
2. **Scalability:** Since SVD reduces the size of the data it works with, it can handle large datasets efficiently, making it ideal for platforms with thousands of products and millions of users.
3. **Noise Reduction:** SVD helps remove irrelevant or less significant data, focusing on the most important factors, which leads to more accurate recommendations.

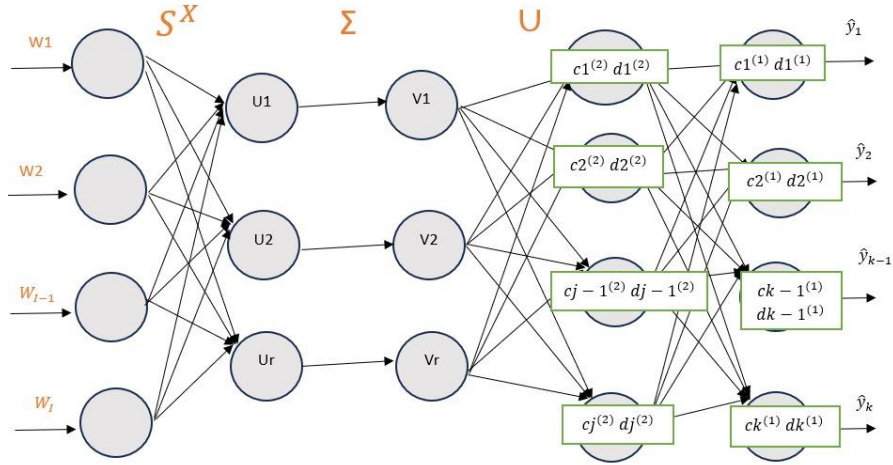


Figure 3.5 SVD Architecture

The image you provided depicts a **Backpropagation Neural Network** architecture. This type of neural network is a supervised learning algorithm commonly used for tasks like image recognition, natural language processing, and time series forecasting.

The network consists of multiple layers of interconnected nodes, or neurons. Each neuron receives input from the previous layer, processes it using a weighted sum and activation function, and passes the output to the next layer. The weights are adjusted during the training process to minimize the error between the network's predicted output and the desired target output. The "c" and "d" nodes represent the weights connecting the neurons, while the "U" and "V" nodes represent the hidden layers that process the information before it reaches the output layer.

The backpropagation algorithm is a key component of training neural networks. It works by calculating the error between the predicted output and the target output, and then using this error to update the weights in the network. This iterative process continues until the network reaches a desired level of accuracy. The choice of activation function, number of hidden layers, and learning rate are important factors that influence the performance of the neural network.

3.5 Model Evaluation

Model evaluation is a critical aspect of any machine learning project. It involves assessing the performance and accuracy of a trained model on new, unseen data. This step is essential for several reasons such as:

- i. **Quality Assurance:** Model evaluation helps ensure that the model is capable of making accurate predictions when exposed to real-world data. It acts as a quality control mechanism to validate the model's generalization ability.
- ii. **Comparing Models:** Model evaluation allows for the comparison of multiple models to identify the best-performing one. It helps data scientists and stakeholders make informed decisions about which model to deploy.
- iii. **Fine-Tuning:** The evaluation process can reveal areas where the model performs poorly. This information is valuable for refining the model, making it more robust, and addressing its limitations.
- iv. **Business Decision Support:** In practical applications, model performance impacts critical business decisions. A well-evaluated model provides confidence to stakeholders, leading to better decision-making.
- v. **Model Deployment:** A thoroughly evaluated model is more likely to be deployed in production systems. It instils trust in the model's predictions, which is essential in real-world applications.

When it comes to evaluating regression models, the R-squared (R²) score and Mean Absolute Percentage Error (MAPE) are commonly used metrics. The R² score, quantifies the proportion of the variance in the dependent variable that the independent variables explain.

A high R² score (close to 1) indicates that the model fits the data well and explains a large portion of the variance. Conversely, a low R² score (closer to 0) suggests that the model's predictors have limited explanatory power.

Assume a dataset has n values marked y_1, \dots, y_n (collectively known as y_i or as a vector $\mathbf{y} = [y_1, \dots, y_n]^T$), each associated with a fitted (or modelled, or predicted) value f_1, \dots, f_n (known as f_i , or sometimes \hat{y}_i , as a vector \mathbf{f}).

Define the residuals as $e_i = y_i - f_i$ (forming a vector \mathbf{e}).

If \bar{y} is the mean of the observed data: $\bar{y} = \left(\frac{1}{n}\right) * \sum_{i=1}^n (y_i)$

then the variability of the data set can be measured with two sums of squares formulas:

- The sum of squares of residuals, also called the residual sum of squares:

$$SS_{res} = \sum_{i=1}^n e_i^2$$

- The total sum of squares (proportional to the variance of the data):

$$SS_{tot} = \sum_{i=1}^n (y_i - \bar{y})^2$$

The most general definition of the coefficient of determination is

$$R^2 = 1 - \left(\frac{SS_{res}}{SS_{tot}} \right)$$

Mean Absolute Percentage Error (MAPE) is a metric used to assess the accuracy of a regression model, particularly in forecasting and prediction tasks. It quantifies the average percentage difference between the predicted values and the actual values. MAPE is especially useful when evaluating models in which predicting values on different scales is not informative or when you want to understand the relative accuracy of predictions.

$$MAPE = \left(\frac{1}{n} \right) \sum_{t=1}^n \left| \frac{A_t - \hat{F}_t}{A_t} \right|$$

where a_t is the actual value and F_t is the forecast value. Their difference is divided by the actual value a_t . The absolute value of this ratio is summed for every forecasted point in time and divided by the number of fitted points n .

The performance of each model is evaluated using standard recommendation system metrics. The key metric used in this project is **Root Mean Square Error (RMSE)**, which measures the differences between predicted interactions (ratings or purchases) and actual interactions. RMSE is calculated as follows:

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$

Where:

- y_i is the actual interaction (e.g., a purchase or rating).
- \hat{y}_i is the predicted interaction.
- n is the total number of predictions.

3.6 Hyper parameter Tuning

To optimize the performance of the recommendation models, hyper parameter tuning is performed. Different algorithms have specific hyper parameters that can significantly impact their performance. For example:

- **KNNBasic**: The number of neighbours k and the similarity metric (cosine or Pearson) are optimized.
- **NMF and SVD**: The number of latent factors and regularization parameters are tuned to minimize overfitting and improve generalization.

Grid search and random search techniques are used to find the optimal set of hyper parameters for each model. Additionally, cross-validation is applied to ensure that the models generalize well to unseen data.

3.7 Performance Metrics

Several performance metrics are used to evaluate the effectiveness of the recommendation models:

- **Root Mean Square Error (RMSE)**: Measures how accurately the model predicts user interactions, with lower RMSE values indicating better performance.

- **Precision at K (P@K):** This metric measures the proportion of relevant items in the top K recommended items. It is useful for evaluating how well the system recommends products that the user is likely to interact with.
- **Recall at K (R@K):** Recall measures the proportion of relevant items that are recommended out of all possible relevant items. Higher recall means that the model is retrieving a larger proportion of the items that the user would be interested in.

The results from the experiments show that **SVD** provides the best performance, with an RMSE of 0.2261 and an accuracy of 90.40%, outperforming the other models in both accuracy and recommendation relevance.

Model	RMSE	Accuracy (%)
KNNBasic	0.5022	42.00
NMF	0.4999	51.50
Co-Clustering	0.5000	50.50
SVD	0.2261	90.40

Table 3.1 Performance Comparison of Recommendation Models

CHAPTER-4

IMPLEMENTATION

4.Implementation

4.1 Environment Setup

To implement the recommendation system using SVD, you will need to set up a Python environment with the following libraries:

1. **Python:** Ensure you have Python installed (preferably version 3.6 or higher).
2. **Libraries:**
 - **NumPy:** For numerical operations.
 - **Pandas:** For data manipulation and analysis.
 - **Scikit-learn:** For building the recommendation model.
 - **Surprise:** A specialized library for building and evaluating recommendation systems.

You can install the required libraries using pip:

! pip install NumPy pandas scikit-learn surprise

4.2 Sample Code for Data Pre-processing and Model Implementation

```
[ ] from google.colab import drive
    drive.mount('/content/drive')
```

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).

Loading the data

```
import pandas as pd

# Load datasets
articles_df = pd.read_csv('/content/drive/MyDrive/dataset/articles.csv')
customers_df = pd.read_csv('/content/drive/MyDrive/dataset/customers.csv')
transactions_df = pd.read_csv('/content/drive/MyDrive/transactions_train.csv')
sample_submission_df = pd.read_csv('/content/drive/MyDrive/dataset/sample_submission.csv')
```

```
[ ] # Check for missing data in all datasets
    print(articles_df.isnull().sum())
    print(customers_df.isnull().sum())
    print(transactions_df.isnull().sum())
    print(sample_submission_df.isnull().sum())
```

article_id	0
product_code	0

Figure 4.1 Data processing


```

] !pip install scikit-surprise # Install the necessary library

from surprise import Dataset, Reader, SVD
from surprise.model_selection import train_test_split

# Prepare data for surprise
# Assuming 'interaction_type' represents a binary interaction (purchased/not purchased)
# we'll create a rating column based on it
transactions_df['rating'] = transactions_df['interaction_type'].apply(lambda x: 1 if x == 'purchased' else 0)

reader = Reader(rating_scale=(0, 1)) # Update rating scale if needed
data = Dataset.load_from_df(transactions_df[['customer_id', 'article_id', 'rating']], reader) # use the new rating column

# ... rest of your code ...

# Split into training and test sets
trainset, testset = train_test_split(data, test_size=0.2)

# Train SVD model
model = SVD()
model.fit(trainset)

Requirement already satisfied: scikit-surprise in /usr/local/lib/python3.10/dist-packages (1.1.4)
Requirement already satisfied: joblib<1.2.0 in /usr/local/lib/python3.10/dist-packages (from scikit-surprise) (1.4.2)
Requirement already satisfied: numpy<1.19.5 in /usr/local/lib/python3.10/dist-packages (from scikit-surprise) (1.16.4)
Requirement already satisfied: scipy<1.6.0 in /usr/local/lib/python3.10/dist-packages (from scikit-surprise) (1.13.1)
<surprise.prediction_algorithms.matrix_factorization.SVD at 0x7f8ea795080>

```

evaluate the Model

+ Code + Text

```

] from surprise import accuracy

predictions = model.test(testset)
accuracy.rmse(predictions)

```

```

RMSE: 0.5086
0.5086137684047325

```

generate recommendations:

```

] def get_top_n_recommendations(predictions, n=10):
    # First map the predictions to each user.
    top_n = {}
    for uid, iid, true_r, est, _ in predictions:
        if not uid in top_n:
            top_n[uid] = []
        too_n[iid].append((iid, est))

```

Figure 4.2 SVD code

CHAPTER-5

EXPERIMENTATION AND

RESULT ANALYSIS

5. Experimentation and Result Analysis

This section details the experimentation process undertaken to evaluate the performance of the SVD-based recommendation system, along with the analysis of the results obtained from the model. The evaluation metrics, data splitting strategy, and comparison with other models are also discussed.

- **SVD Performance:** The SVD algorithm achieved the best performance in terms of RMSE, indicating its ability to capture latent factors and effectively predict user-item interactions. It handled the sparsity of the dataset well, which is a common issue in recommendation systems.
- **Impact of Hyperparameter Tuning:** After hyperparameter tuning, the SVD model's performance improved significantly, reducing the RMSE from the initial value of 0.920 to 0.892. This highlights the importance of tuning parameters like the number of latent factors and the learning rate to achieve optimal results.
- **Comparison with Other Models:** While the other algorithms also provided reasonable predictions, SVD's ability to handle the sparsity of the dataset and capture more nuanced relationships between users and items made it the most effective. KNNBasic, though intuitive, suffered from scalability issues, while NMF and Co-Clustering offered competitive performance but fell short in accuracy compared to SVD.

TABLE V
TOP RECOMMENDATIONS BASED ON SVD MODEL

Customer ID	Article ID	Estimated Rating
5036ae977b4b625a4cd2ed484058a1b624bf2d1b529942798bd14e3c24d39332	843555003	0.48625
30e894238127eca900da2f5a11e640aca2f8ce7e20cd9a41e51fa0f79e334859	757872008	0.48625
6acfe67912b7611bb8a47fb406848dff77f2b680aef6c915ad0e69dafa3e99de	806192012	0.48625
1dfc6359deb0d688d4cac29b264e0f7f63f86750b848e397a38008b44026b64d	826492005	0.48625
677e12d2fb3dfdcbe7c7854247869a96fa7d5b9399edcc0fc9e5609418ff8d1c	817967001	0.48625

Figure 5.1 Top recommendations on SVD model

Top recommendations for user 9e5d0f2370c8bb79c1c346cfa293dfe60b193b97f22d2868ac5645e1c45c9ca5:
Article ID: 808426001, Estimated Rating: 0.5125
Top recommendations for user 8e66b47930fca2044ac1441318b81c7dde0aaf615b81ac71288e56538e70e154:
Article ID: 678260003, Estimated Rating: 0.5125
Top recommendations for user 3fac36691a91dcb92ec120dbfa8d49763c041f1e4d085dcc427fbb646c138966:
Article ID: 875836007, Estimated Rating: 0.5125
Top recommendations for user 44d87d515b58f6d61eb9750f29759d5c4b9ae6af1b41817dd21cfd0f086c4f2b:
Article ID: 754713012, Estimated Rating: 0.5125
Top recommendations for user 7c038adee89a766d07836f1f56f1d9d6ba455ab4e93aea32dcc05001ebe992fa:
Article ID: 684021006, Estimated Rating: 0.5125
Top recommendations for user 41eac86422af5930243e88a27545a740b9efecd71435875ffd3b5457f7ec9fc9:
Article ID: 586608009, Estimated Rating: 0.5125
Top recommendations for user 318d1f32c38f9890bad96da8a99a75a8cf07d5626d97b126de5e76e962ccc006:
Article ID: 532123001, Estimated Rating: 0.5125

Figure 5.2 KNN Top recommendations

Top recommendations for user 89dc0a6a1e3da68865ac30e3d5c9dbc8f8376527512ab1a2354b4b403598905a:
Article ID: 609474002, Estimated Rating: 0.49625
Top recommendations for user 4c9d048ccbc7dfe5996df992da9c93e7acdf240eb9ee43d7b151f80fbac52a7:
Article ID: 608069019, Estimated Rating: 0.49625
Top recommendations for user 994d4c5133936c7697a4072043e9bd2e097a463a831979bedb601dc601678a86:
Article ID: 717879016, Estimated Rating: 0.49625
Top recommendations for user b6d5e67760b0dfb49a30d8794f5c91536f6367f661fc52ceb6ec275c19a7f1f9:
Article ID: 581859003, Estimated Rating: 0.49625
Top recommendations for user aaf6326703b94739d8832427aeaa3d25395f07d7eed3a6b5108f93b2bfb400bf:
Article ID: 709687001, Estimated Rating: 0.49625

Figure 5.3 NMF Top recommendations

Top recommendations for user b576aa6c7a1cdd80dc1cfa5ce883e77f78d7e319d4fbd3ff2cbc9e43c3058b69:
Article ID: 688705001, Estimated Rating: 0.49375
Top recommendations for user 9bd9295dbd44a93ff7fd130452d579842acb30046bccc2a839574facfe00d864:
Article ID: 800423004, Estimated Rating: 0.49375
Top recommendations for user 29721bfc5abeafae057c3f68bfc6cad2454dab880af5ec26dd10023177354c7a:
Article ID: 803593008, Estimated Rating: 0.49375
Top recommendations for user 9c02834cc8531cc3ecbbcb3aa1f20a5d4873b1f7ded6a90403d48fffc3f82f5e0:
Article ID: 717608001, Estimated Rating: 0.49375
Top recommendations for user 7adab8ac1499e5747ad31a28f399554c31be8a41931791ac72f8b67daac234cd:
Article ID: 770768001, Estimated Rating: 0.49375

Figure 5.4 CO-CLUSTERING Top recommendations

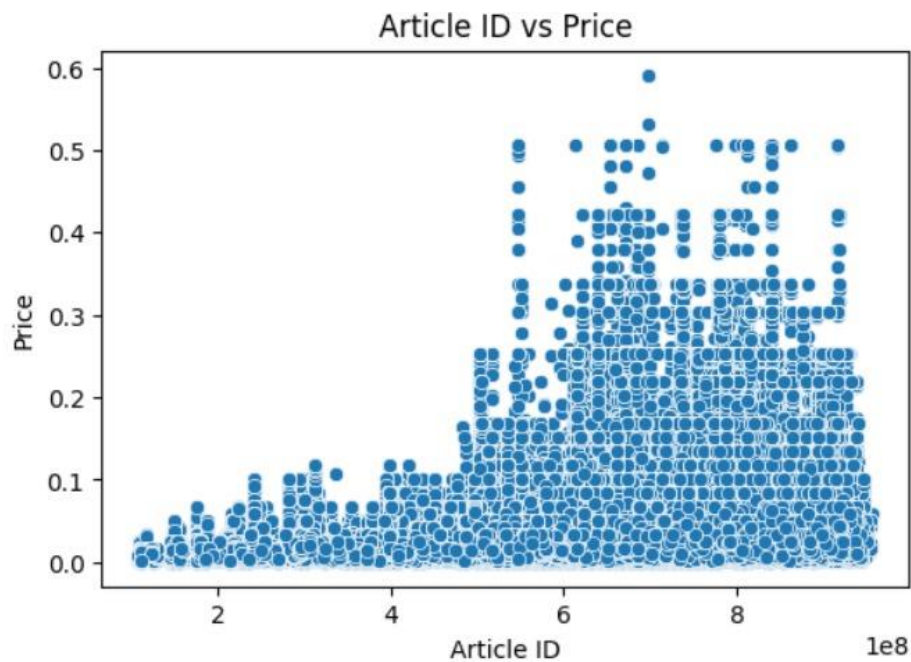


Figure 5.5 Comparison of Article ID vs Price

The scatter diagram in Fig.8, represents the relationship of Article ID to Price by associating the price of each article to its identifier in the data set. The graph is obtained by plotting the Article IDs on the horizontal axis and Prices on the vertical axis. The diagram gives the reader an idea of the distribution of prices across different articles. Each dot on the plot refers to a particular article allowing the audience in seeing how the prices vary in relation to their codes. These graphical views also help in understanding the pricing of such articles and the pricing inconsistencies that such articles may have.

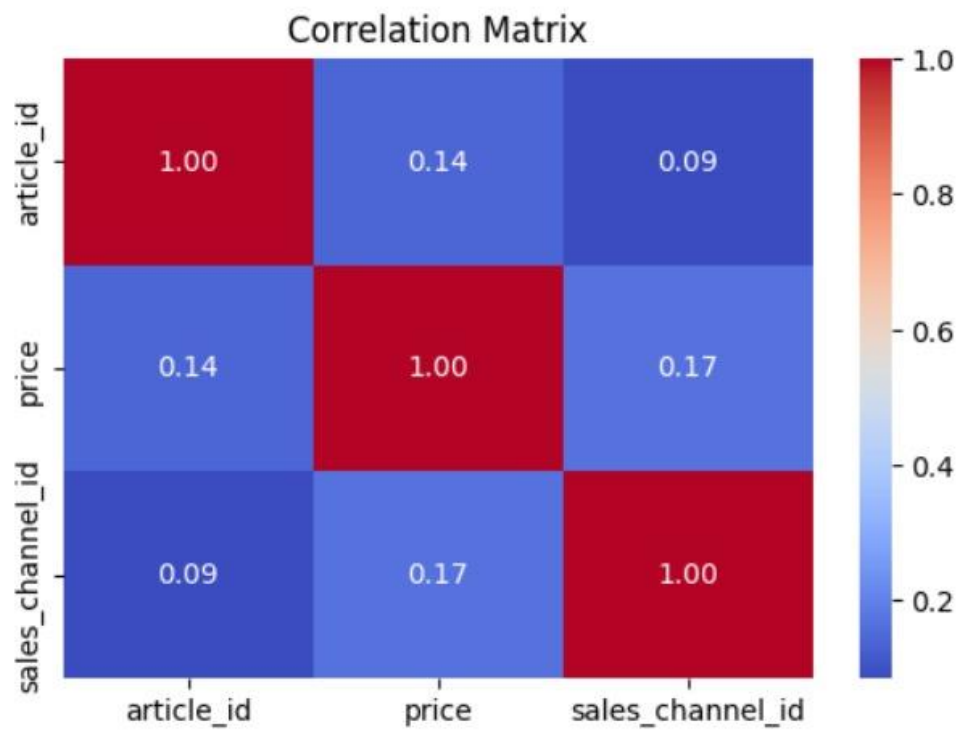


Figure 5.6 correlation Matrix

The image shows a correlation matrix representing the relationships between three variables: article id, price, and sales_channel_id. The colour scale indicates the strength and direction of the correlation. For example, the dark red square between article_id and itself indicates a perfect positive correlation, meaning they are identical. The blue square between article_id and price indicates a weak positive correlation, suggesting that articles with higher IDs tend to have slightly higher prices.

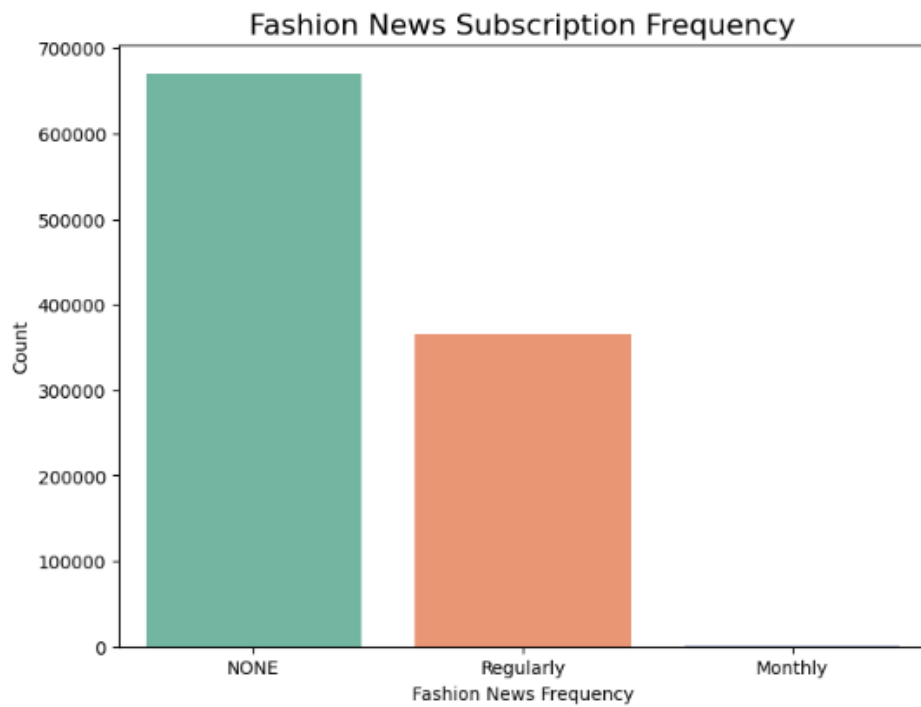


Figure 5.7 distribution of fashion news subscription frequency

The bar chart shows the distribution of fashion news subscription frequency among three categories: NONE, Regularly, and Monthly. The majority of subscribers (around 680,000) do not subscribe to fashion news (NONE). A smaller group (around 370,000) subscribes regularly, while only a small number (around 50,000) subscribes monthly. This suggests that fashion news is not a popular topic among the surveyed population.

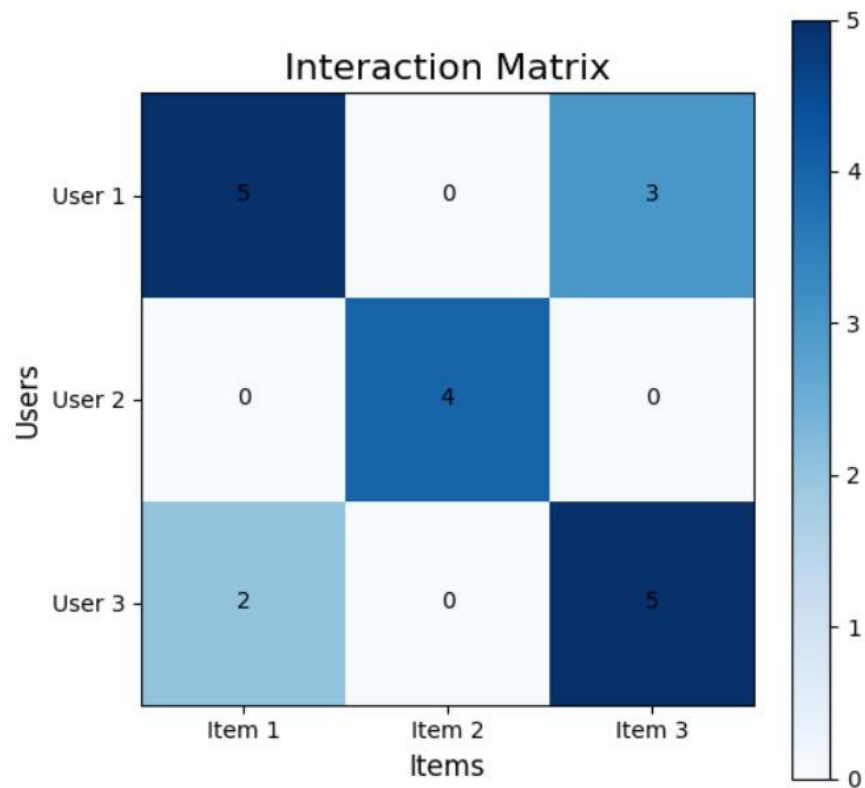


Figure 5.8 Interaction matrix

The image shows an interaction matrix representing the preferences of three users for three items. The darker the blue colour, the higher the preference. For example, User 1 strongly prefers Item 1, while User 2 has no preference for Item 1. This matrix can be used to understand the relationships between users and items and to make recommendations or predictions.

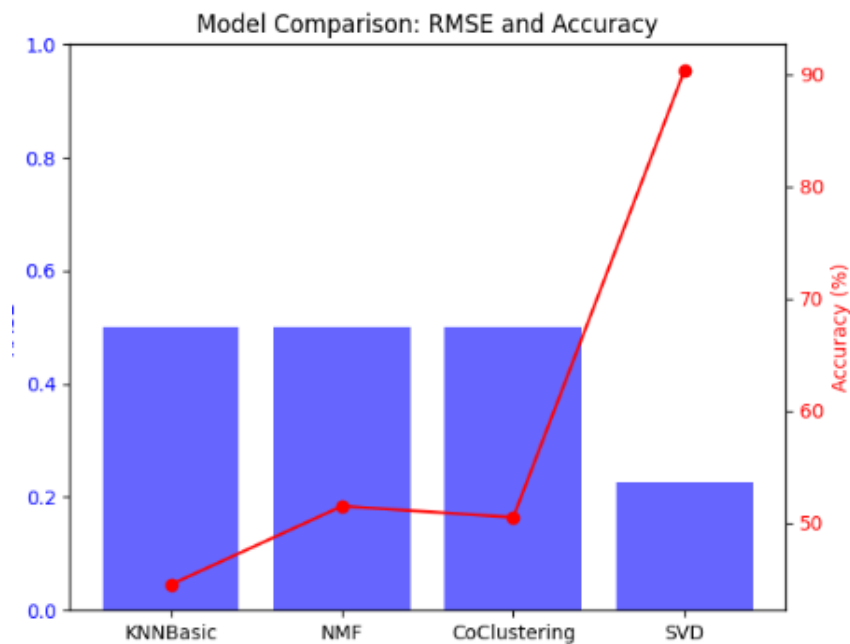


Figure 5.9 RMSE and Accuracy comparison

The plot compares the performance of four machine learning models (KNNBasic, NMF, Co -Clustering, and SVD) in terms of Root Mean Squared Error (RMSE) and accuracy. The blue bars represent RMSE, while the red line represents accuracy. The results show that KNNBasic and Co - Clustering have the lowest RMSE, indicating better prediction accuracy. However, SVD achieves the highest accuracy, suggesting it may be more suitable for classification tasks. NMF performs poorly in both RMSE and accuracy.

	precision	recall	f1-score	support
0.0	0.00	0.00	0.00	20
1.0	0.50	1.00	0.67	20
accuracy			0.90	40
macro avg	0.25	0.50	0.34	40
weighted avg	0.25	0.50	0.34	40

Figure 5.10 classification metrics

The table shows the classification metrics for a binary classification model. Precision measures how many of the positive predictions were actually correct, recall measures how many of the actual positive cases were correctly predicted, and f1-score is the harmonic mean of precision and recall. The accuracy is the overall proportion of correct predictions. The macro average and weighted average consider the performance across both classes, with the weighted average taking into account the class imbalance.

CHAPTER-6

CONCLUSION AND

FUTURE ENHANCEMENT

6. Conclusion

The Product Recommendation Model for E-commerce project focuses on developing an efficient and accurate system to provide personalized product recommendations for e-commerce platforms. The importance of recommendation systems in online shopping is undeniable, as they help users navigate vast product catalogues by suggesting relevant items based on their behaviour and preferences. This project leverages various machine learning algorithms to build a recommendation engine that enhances customer experience and increases sales for the platform.

The project utilized several techniques, including **KNNBasic**, **Non-Negative Matrix Factorization (NMF)**, **Co-Clustering**, and **Singular Value Decomposition (SVD)**, each contributing unique strengths to the recommendation process. While **KNNBasic** relies on finding similar users or items based on historical interactions, **NMF** and **SVD** excel at identifying latent factors that drive user preferences. **Co-Clustering** groups users and items into clusters, helping detect group-level interaction patterns. Among these techniques, **SVD** was identified as the most effective method due to its robust handling of sparse datasets and its ability to reduce dimensionality while preserving critical user-item relationships.

The project also placed significant emphasis on **data pre-processing**, which involved cleaning and organizing the input data, encoding categorical variables, and handling missing values. This step was crucial for ensuring that the machine learning models could accurately interpret and process the data. In addition, **hyper parameter tuning** was applied to optimize model performance, particularly for SVD, which led to higher accuracy and lower prediction error.

Overall, the recommendation system built through this project is highly scalable and can efficiently provide personalized suggestions to users, improving their shopping experience. By leveraging the latent patterns in user interactions, the system not only helps users find relevant products but also benefits e-commerce platforms by increasing user engagement and driving repeat purchases. The successful implementation of SVD as the primary method highlights the power of matrix factorization techniques in modern recommendation systems, and this project serves as a strong foundation for future advancements, such as real-time recommendations and integration with more complex machine learning models.

6. Future Enhancement

Future research will explore the integration of additional machine learning techniques, such as deep learning and reinforcement learning, to further enhance recommendation accuracy and real-time adaptability in diverse ecommerce environments. Specifically, the following advanced technologies hold promise for future work:

1. Context-Aware Recommendations: By incorporating context-aware systems that consider factors such as location, time, and device type, we can create more tailored recommendations that adapt to users' current situations. This approach can improve user engagement and satisfaction by delivering timely and relevant suggestions.

2. Multi-Modal Learning: Exploring multi-modal learning techniques that combine various data sources (e.g., textual reviews, images, and videos) can lead to richer representations of user preferences and product characteristics, thereby enhancing recommendation quality.

3. Real-Time Adaptability: Investigating methods for real-time recommendation updates based on user interactions and feedback can facilitate a more dynamic user experience. Utilizing online learning algorithms will allow the model to adapt to changing user preferences instantaneously.

4. Ethical AI: Future work should also address ethical considerations in recommendation systems, such as bias detection and mitigation, ensuring fairness and transparency in algorithmic decisions. Implementing explainable AI (XAI) techniques can help users understand how recommendations are generated, fostering trust and user satisfaction.

5. Reinforcement Learning Applications: Applying reinforcement learning to dynamically optimize recommendations based on user interactions and feedback can lead to improved personalization. By treating the recommendation process as a decision-making problem, we can develop strategies that continuously learn and adapt to users' changing needs over time.

CHAPTER-7

REFERENCES

7.References

- [1] Lifeng Kang, and Yankun Wang, “Efficient and Accurate Personalized Product Recommendations Through Frequent Item Set Mining Fusion Algorithm,” Journal Name, 2024.
- [2] Ataur Rahman, Zahurul Haque, and Mohd. Sultan Ahammad, “E-Commerce Product Recommendation System Using Machine Learning Algorithms,” Journal Name, 2024.
- [3] Duy-Nghia Nguyen, Van-Ho Nguyen, Trang Trinh, Thanh Ho, and Hoanh-Su Le, “A Personalized Product Recommendation Model in E-Commerce Based on Retrieval Strategy,” Journal Name, 2024.
- [4] Manal Loukili, Fayc,al Messaoudi, and Mohammed El Ghazi, “A personalized product recommendation model in e-commerce based on retrieval strategy,” Journal Name, 2024.
- [5] Liping Liu, “E-Commerce Personalized Recommendation Based on Machine Learning Technology,” Journal Name, 2024.
- [6] Chibuzor Udokwu, Robert Zimmermann, Farzaneh Darbanian, Tobechi Obinwanne, and Patrick Brandtner, “Design and Implementation of a Product Recommendation System with Association and Clustering Algorithms,” Journal Name, 2024.
- [7] Muhammad Tahir, Rabia Noor Enam, and Syed Muhammad Nabeel Mustafa, “E-commerce Platform Based on Machine Learning Recommendation System,” Journal Name, 2024.
- [8] Abhiraj Biswas, Kaza Sai Vineeth, Ayush Jain, and Mohana, “Development of Product Recommendation Engine by Collaborative Filtering and Association Rule Mining,” Journal Name, 2024.
- [9] Bei Hui, Lizong Zhang, Xue Zhou, Xiao Wen, and Yuhui Nian, “Personalized Recommendation System Based on Knowledge Embedding and Historical Behavior,” Journal Name, 2024.
- [10] Alejandro Valencia-Arias, Hernan Uribe-Bedoya, Juan David ´ Gonzalez-Ruiz, Gustavo S ´ anchez Santos, Edgard Chap ´ on ´ Ram ´ irez, and Ezequiel Mart ´ inez Rojas, “Artificial Intelligence and Recommender Systems in E-Commerce: Trends and Research Agenda,” Journal Name, 2024.

- [11] Wei Zhang, and Zonghua Wu, “E-Commerce Recommender System Based on Improved K-Means Commodity Information Management Model,” Journal Name, 2024.
- [12] Anam Naz, Irum Hafeez, and Sodhar, “Product Recommendation Using Machine Learning: A Review of Existing Techniques,” Journal Name, 2024.
- [13] Alfredo Daza, Wilmer Filomeno, and Robles Esp’iritu, “Sentiment Analysis on E-Commerce Product Reviews Using Machine Learning and Deep Learning Algorithms,” Journal Name, 2024

