# PRODUCT RECOMMENDATION SYSTEM

**A MINI PROJECT REPORT**

**18CSC305J - ARTIFICIAL INTELLIGENCE**

***Submitted by***

**HIMAL PANDEY(RA2111003011922)**
**MEGHANA RAJ(RA2111003011921)**
**ASHI GUPTA(RA2111003011925)**

***Under the guidance of***

**Dr.M.K.Vidhyalakshmi**

**Assistant Professor, Department of Computer Science and Engineering**

***in partial fulfillment for the award of the***

***degreeof***

**BACHELOR OF TECHNOLOGY**
**in**

**COMPUTER SCIENCE & ENGINEERING**
**of**

**FACULTY OF ENGINEERING AND TECHNOLOGY**

SRM

INSTITUTE OF SCIENCE & TECHNOLOGY
Deemed to be University u/s 3 of UGC Act, 1956

S.R.M. Nagar, Kattankulathur, Chengalpattu District

**MAY 2024**

# SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

**(Under Section 3 of UGC Act, 1956)**

## BONAFIDE CERTIFICATE

Certified that Mini project report titled **"PRODUCT RECOMMENDATION SYSTEM"** is the bonafide work of **HIMAL PANDEY(RA2111003011922), MEGHANA RAJ(RA2111003011921), ASHI GUPTA(RA2111003011925)** who carried out the minor project under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

**SIGNATURE**
Dr.M.K.Vidhyalakshmi
Assistant Professor
Department of Computing Technologies
SRM Institute of Science and Technology

## ABSTRACT

The cold start problem poses a significant challenge for recommender systems, particularly when dealing with new users or items lacking interaction data. This problem manifests in two main scenarios: user cold start and item cold start.
User cold start occurs when a new user joins a service without any past behavior data available to the recommender system. Conversely, item cold start arises when a new item is introduced, and there is no information on how users have interacted with it.
To mitigate the cold start problem, various strategies can be employed:

1. Rank-based recommendations: For new users, popular items can be recommended based on overall popularity or average ratings, allowing users to explore popular items on the platform irrespective of their preferences.
2. Content-based filtering: When new items are introduced, recommendations can be made based on the attributes or metadata of the items. This approach bypasses the need for interaction data and introduces new items to users with similar preferences.
3. Hybrid approaches: Combining multiple strategies, such as collaborative filtering and content-based filtering, offers a comprehensive solution to address the cold start problem. By leveraging both user similarity and item attributes, hybrid approaches can provide more accurate and diverse recommendations.

Specifically, collaborative filtering tackles the user cold start problem by identifying users with similar interests and recommending items that those users have rated highly. On the other hand, content-based filtering addresses the item cold start problem by analyzing item content and suggesting similar items based on their attributes.

# TABLE OF CONTENTS

| | |
|---|---|
| **ABSTRACT** | **iii** |
| **TABLE OF CONTENTS** | **iv** |
| **LIST OF FIGURES** | **v** |
| **ABBREVIATIONS** | **vi** |
| **1. INTRODUCTION** | **7** |
| **2. LITERATURE SURVEY** | **8** |
| **3. SYSTEM ARCHITECTURE AND DESIGN** | **9** |
| **4. METHODOLOGY** | **10** |
| 4.1. Methodological Steps | |
| **5. CODING AND TESTING** | **13** |
| **6. SREENSHOTS AND RESULTS.** | **14** |
| **7. CONCLUSION AND FUTURE ENHANCEMENT.** | **15** |

## LIST OF FIGURES

1. **User Cold Start Scenario Illustration**
2. **Item Cold Start Scenario Illustration**
3. **Rank-Based Recommendations**
4. **Content-Based Filtering**
5. **Collaborative Filtering Process**
6. **Content-Based Filtering Process**
7. **Hybrid Approaches Integration**
8. **User Engagement Over Time**

## ABBREVIATIONS

1. CF - Collaborative Filtering
2. CBF - Content-Based Filtering
3. ML - Machine Learning
4. CSV - Comma-Separated Values
5. PRS - Product Recommendation System
6. EDA- Exploratory Data Analysis

# INTRODUCTION

In today's highly competitive business landscape, the ability to provide personalized and relevant product recommendations is essential for businesses looking to thrive and succeed. With the exponential growth of data and advancements in artificial intelligence (AI) and machine learning (ML) technologies, businesses have a unique opportunity to leverage these tools to enhance customer experiences and drive revenue growth.
The "Product Recommendation System" project aims to address this need by introducing a cutting-edge solution that harnesses AI and ML algorithms to deliver personalized product recommendations tailored to individual preferences. By analyzing vast amounts of customer data, including purchase history, browsing behavior, and demographic information, the system can uncover intricate patterns and correlations, enabling it to generate accurate and relevant recommendations.
This project encompasses a comprehensive methodology, including data preprocessing, feature engineering, model training, and deployment. By leveraging state-of-the-art algorithms such as collaborative filtering and content-based filtering, the system can efficiently identify products that align with each customer's interests and preferences.
The user interface of the system provides a seamless experience for both customers and businesses, allowing users to explore personalized recommendations, provide feedback, and refine preferences over time. Continuous monitoring and optimization of the recommendation algorithms ensure that the system adapts to evolving customer needs and market trends.
Overall, the "Product Recommendation System" empowers businesses to deliver targeted product suggestions, increase customer engagement, and drive conversion rates, ultimately fostering long-term customer loyalty and business success in today's dynamic marketplace.

## LITERATURE SURVEY

Customer sentiment analysis has garnered significant attention in both academic research and industry applications due to its potential to extract valuable insights from vast amounts of textual data generated by customers on online platforms. In this literature survey, we review key research papers, methodologies, and advancements in the field of customer sentiment analysis using artificial intelligence and natural language processing techniques.

1. **"Recommender Systems Handbook" by Francesco Ricci, Lior Rokach, Bracha Shapira, Paul B. Kantor**:This seminal paper introduces a pioneering approach to sentiment analysis by mining and summarizing customer reviews from online platforms. The authors propose a novel technique for sentiment classification based on a combination of supervised machine learning algorithms and lexicon-based sentiment analysis. The study demonstrates the effectiveness of the proposed methodology in extracting valuable sentiment information from large volumes of customer reviews.

**2. "Introduction to Information Retrieval" by Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze:**This textbook offers insights into information retrieval techniques, including content-based and collaborative filtering methods used in recommendation systems. It provides a solid theoretical foundation for understanding the underlying principles of recommendation algorithms.

**3."Mining of Massive Datasets" by Jure Leskovec, Anand Rajaraman, Jeff Ullman**: This book covers the fundamentals of data mining and machine learning techniques applicable to large-scale datasets, including those commonly used in recommendation systems. It offers practical insights and algorithms for building scalable and efficient recommendation systems.

**4."Hands-On Recommendation Systems with Python" by Rounak Banik**: This practical guide offers hands-on experience in building recommendation systems using Python. It covers various techniques, including collaborative filtering, content-based filtering, and hybrid approaches, with step-by-step tutorials and real-world examples.

**5. "A Survey of Recommender Systems in E-Commerce Contexts" by Mohsen Jamali, Martin Ester**: This survey paper provides an overview of recommender systems in e-commerce contexts, focusing on their applications, challenges, and future directions. It discusses the role of recommender systems in enhancing user experience and driving business success in online retail environments.

# SYSTEM ARCHITECTURE AND DESIGN

Architecture and System Design:

1. Data Collection: The system collects vast amounts of customer data from diverse sources, including purchase history, browsing behavior, and demographic information. This data serves as the foundation for generating personalized recommendations.

2. Data Preprocessing: Upon collection, the raw data undergoes preprocessing to clean, normalize, and transform it into a suitable format for analysis.

3.Feature Engineering: Feature engineering involves extracting relevant features from the preprocessed data that can be used to build recommendation models. This may include extracting product attributes, customer preferences, and behavioral patterns.

4.Model Training: The system employs machine learning algorithms, such as collaborative filtering and content-based filtering, to train recommendation models on the preprocessed data. These models learn intricate patterns and correlations within the data to generate accurate and relevant recommendations

5. Model Deployment: Once trained, the recommendation models are deployed within the system's infrastructure to generate real-time recommendations for users. This involves optimizing model performance, scalability, and reliability to handle large volumes of user requests.

6.User Interface: The system provides a user-friendly interface that allows users to interact with the recommendation system seamlessly. The interface enables users to explore personalized recommendations, provide feedback, and refine their preferences over time.

7. Continuous Monitoring and Optimization: The recommendation system undergoes continuous monitoring to track its performance and effectiveness in delivering recommendations. This involves monitoring key metrics such as click-through rates, conversion rates, and user engagement metrics.

## METHODOLOGY

Methodology:

1.Data Collection and Preparation:

- Gather data from diverse sources, including purchase history, browsing behavior, and demographic information.
- Preprocess the data to clean, normalize, and transform it into a suitable format for analysis.

2. Feature Engineering:

- Extract relevant features from the preprocessed data that can be used to build recommendation models.
- Include product attributes, customer preferences, behavioral patterns, and contextual information in the feature set.

3. Collaborative Filtering:

- Implement user-based collaborative filtering, where recommendations are based on the preferences of similar users.
- Calculate user similarity metrics, such as cosine similarity or Pearson correlation coefficient, to identify similar users.

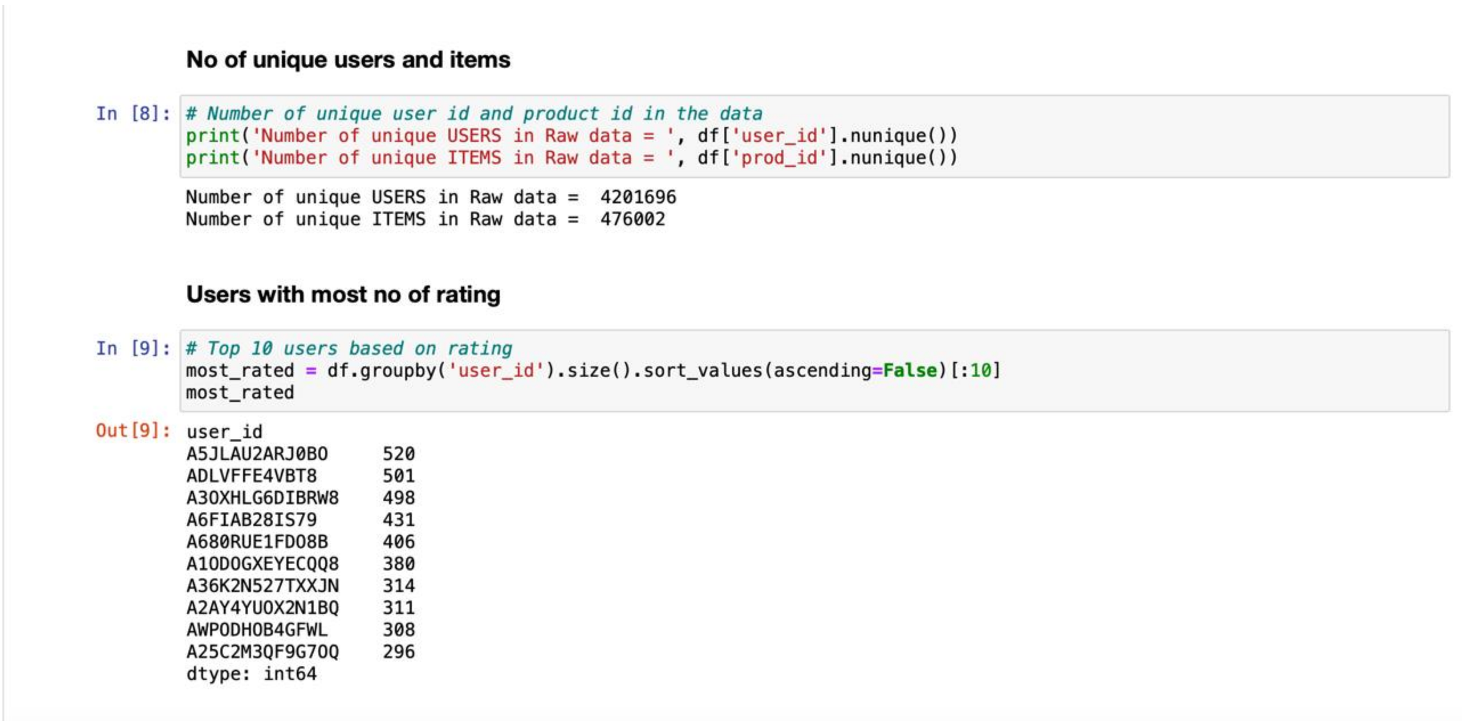
4.Content-Based Filtering:

- Develop content-based recommendation models that recommend items similar to those the user has interacted with in the past.
- Analyze product attributes and descriptions to identify items with similar characteristics.

5.Model Optimization:

- Fine-tune model hyperparameters using techniques like grid search or random search to optimize performance.
- Regularize models to prevent overfitting and improve generalization to unseen data
- Experiment with different algorithms and feature sets to identify the most effective approach for generating recommendations.

# CODING AND TESTING

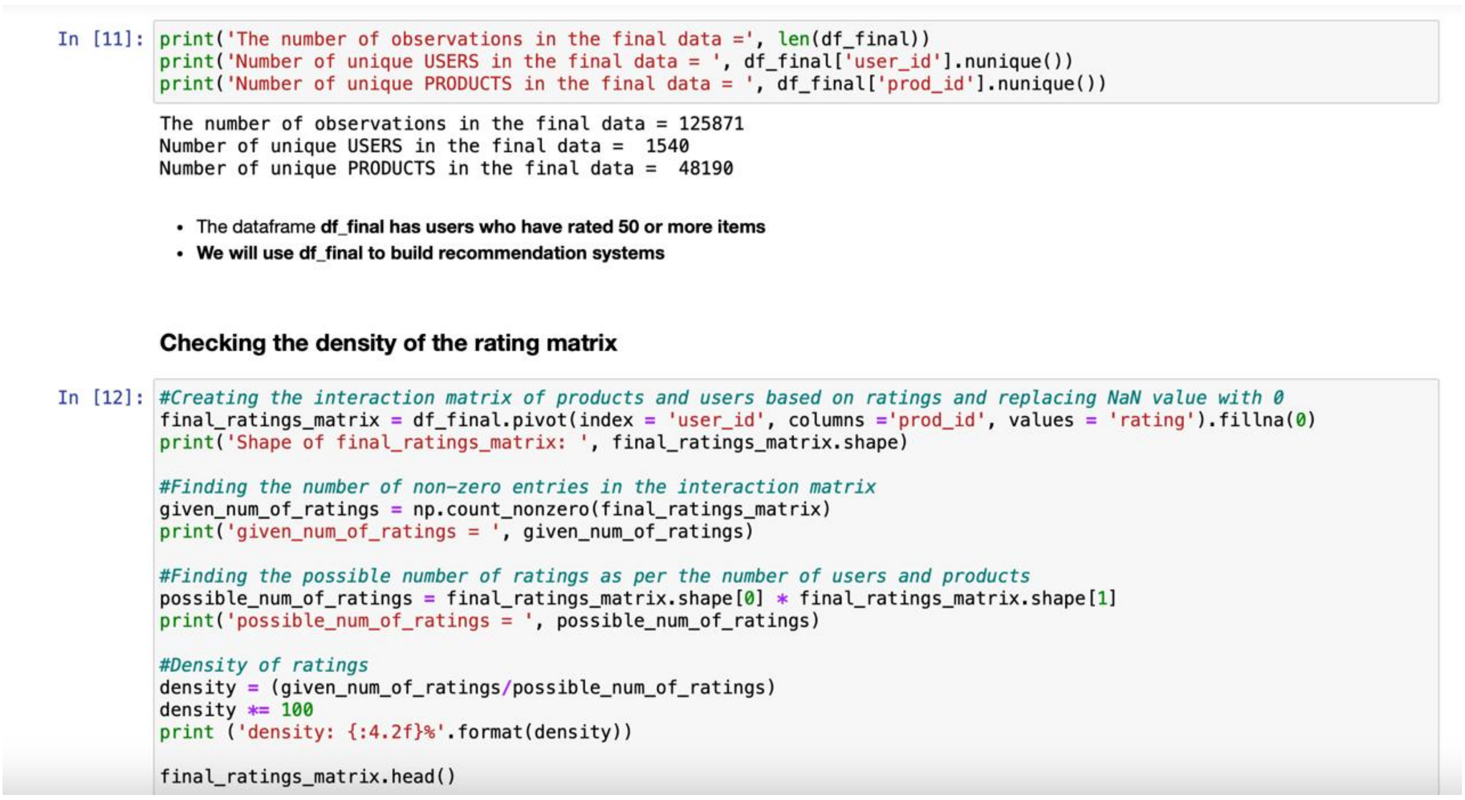


### No of unique users and items

```
In [8]: # Number of unique user id and product id in the data
        print('Number of unique USERS in Raw data = ', df['user_id'].nunique())
        print('Number of unique ITEMS in Raw data = ', df['prod_id'].nunique())

        Number of unique USERS in Raw data =  4201696
        Number of unique ITEMS in Raw data =  476002
```

### Users with most no of rating

```
In [9]: # Top 10 users based on rating
        most_rated = df.groupby('user_id').size().sort_values(ascending=False)[:10]
        most_rated

Out[9]: user_id
        A5JLAU2ARJ0BO     520
        ADLVFFE4VBT8      501
        A3OXHLG6DIBRW8    498
        A6FIAB28IS79      431
        A680RUE1FD08B     406
        A1ODOGXEYECQQ8    380
        A36K2N527TXXJN    314
        A2AY4YUOX2N1BQ    311
        AWPODHOB4GFWL     308
        A25C2M3QF9G7OQ    296
        dtype: int64
```



```
In [11]: print('The number of observations in the final data =', len(df_final))
         print('Number of unique USERS in the final data = ', df_final['user_id'].nunique())
         print('Number of unique PRODUCTS in the final data = ', df_final['prod_id'].nunique())

         The number of observations in the final data = 125871
         Number of unique USERS in the final data =  1540
         Number of unique PRODUCTS in the final data =  48190
```

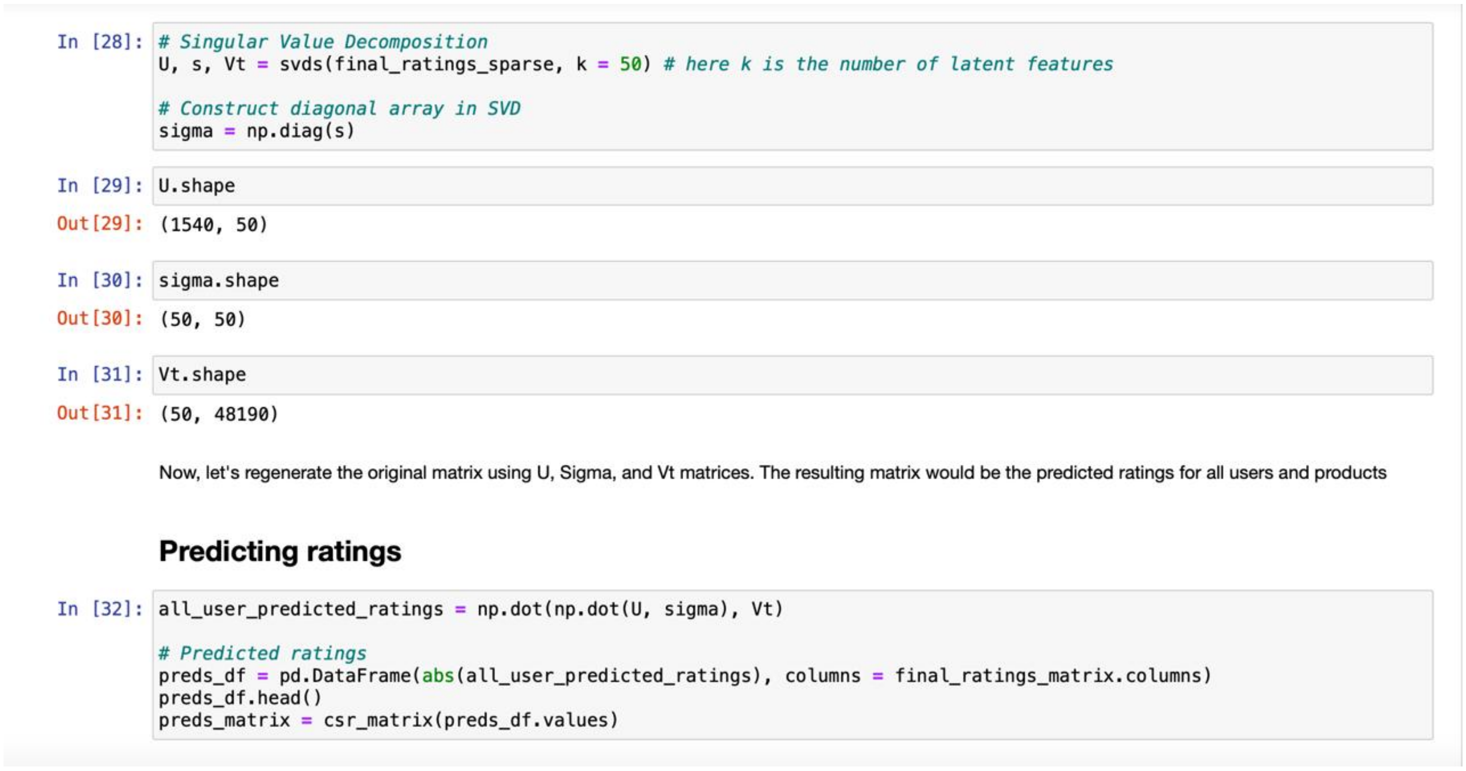
- The dataframe **df_final has users who have rated 50 or more items**
- **We will use df_final to build recommendation systems**

### Checking the density of the rating matrix

```
In [12]: #Creating the interaction matrix of products and users based on ratings and replacing NaN value with 0
         final_ratings_matrix = df_final.pivot(index = 'user_id', columns ='prod_id', values = 'rating').fillna(0)
         print('Shape of final_ratings_matrix: ', final_ratings_matrix.shape)

         #Finding the number of non-zero entries in the interaction matrix
         given_num_of_ratings = np.count_nonzero(final_ratings_matrix)
         print('given_num_of_ratings = ', given_num_of_ratings)

         #Finding the possible number of ratings as per the number of users and products
         possible_num_of_ratings = final_ratings_matrix.shape[0] * final_ratings_matrix.shape[1]
         print('possible_num_of_ratings = ', possible_num_of_ratings)

         #Density of ratings
         density = (given_num_of_ratings/possible_num_of_ratings)
         density *= 100
         print ('density: {:4.2f}%'.format(density))

         final_ratings_matrix.head()
```

## Function to recommend products

```
In [33]: import numpy as np

         def recommend_items(user_index, interactions_matrix, preds_matrix, num_recommendations):

             # Get the user's ratings from the actual and predicted interaction matrices
             user_ratings = interactions_matrix[user_index,:].toarray().reshape(-1)
             user_predictions = preds_matrix[user_index,:].toarray().reshape(-1)

             #Creating a dataframe with actual and predicted ratings columns
             temp = pd.DataFrame({'user_ratings': user_ratings, 'user_predictions': user_predictions})
             temp['Recommended Products'] = np.arange(len(user_ratings))
             temp = temp.set_index('Recommended Products')

             #Filtering the dataframe where actual ratings are 0 which implies that the user has not interacted with that pro
             temp = temp.loc[temp.user_ratings == 0]

             #Recommending products with top predicted ratings
             temp = temp.sort_values('user_predictions',ascending=False)#Sort the dataframe by user_predictions in descending
             print('\nBelow are the recommended products for user(user_id = {}):\n'.format(user_index))
             print(temp['user_predictions'].head(num_recommendations))
```



```
In [28]: # Singular Value Decomposition
         U, s, Vt = svds(final_ratings_sparse, k = 50) # here k is the number of latent features

         # Construct diagonal array in SVD
         sigma = np.diag(s)

In [29]: U.shape

Out[29]: (1540, 50)

In [30]: sigma.shape

Out[30]: (50, 50)

In [31]: Vt.shape

Out[31]: (50, 48190)
```

Now, let's regenerate the original matrix using U, Sigma, and Vt matrices. The resulting matrix would be the predicted ratings for all users and products

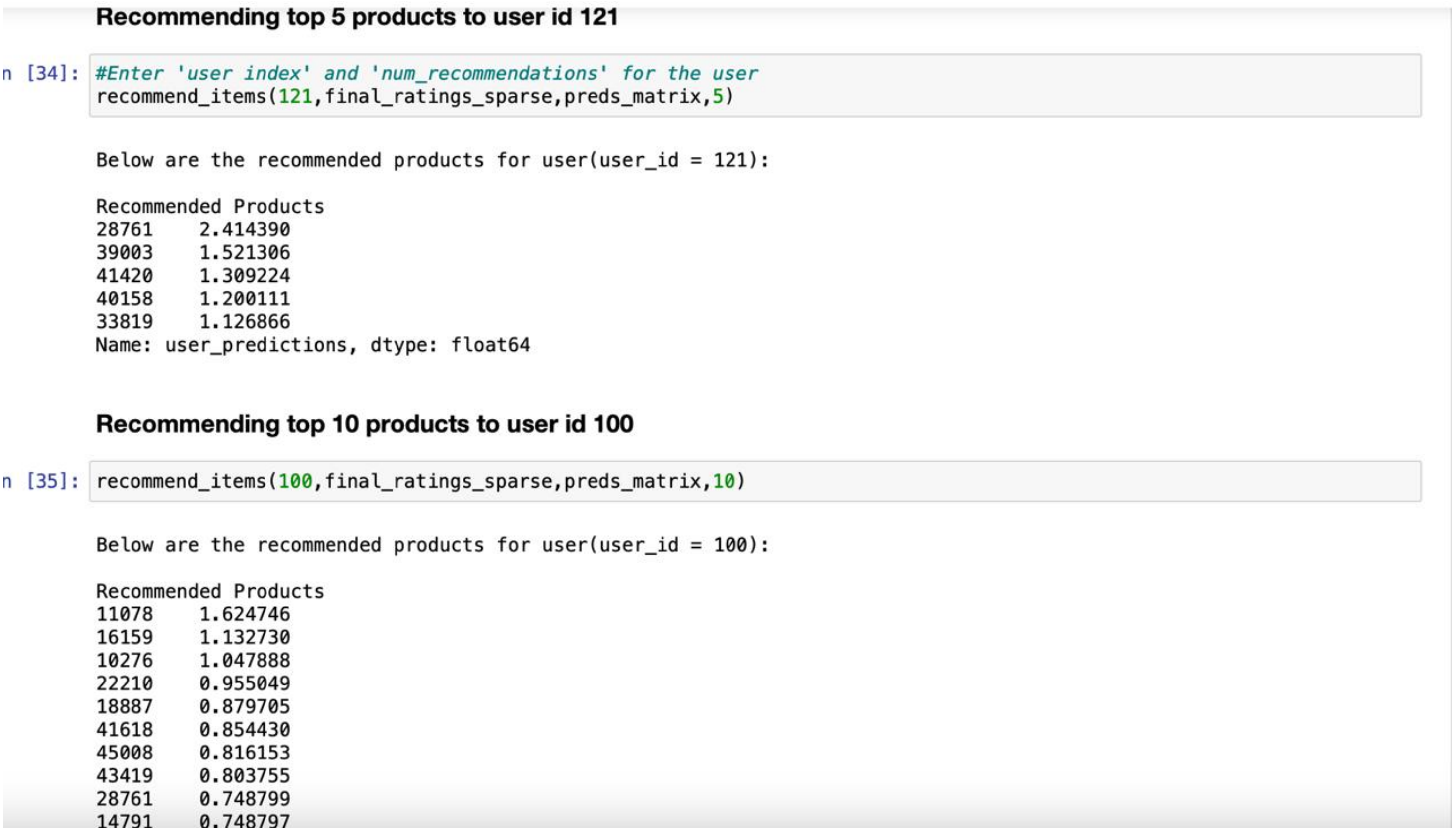
## Predicting ratings

```
In [32]: all_user_predicted_ratings = np.dot(np.dot(U, sigma), Vt)

         # Predicted ratings
         preds_df = pd.DataFrame(abs(all_user_predicted_ratings), columns = final_ratings_matrix.columns)
         preds_df.head()
         preds_matrix = csr_matrix(preds_df.values)
```

### Function to recommend products

```
In [24]: # defining the recommendations function to get recommendations by using the similar users' preferences
def recommendations(user_index, num_of_products, interactions_matrix):

    #Saving similar users using the function similar_users defined above
    most_similar_users = similar_users(user_index, interactions_matrix)[0]

    #Finding product IDs with which the user_id has interacted
    prod_ids = set(list(interactions_matrix.columns[np.where(interactions_matrix.loc[user_index] > 0)]))
    recommendations = []

    observed_interactions = prod_ids.copy()
    for similar_user in most_similar_users:
        if len(recommendations) < num_of_products:

            #Finding 'n' products which have been rated by similar users but not by the user_id
            similar_user_prod_ids = set(list(interactions_matrix.columns[np.where(interactions_matrix.loc[similar_us
            recommendations.extend(list(similar_user_prod_ids.difference(observed_interactions)))
            observed_interactions = observed_interactions.union(similar_user_prod_ids)
        else:
            break

    return recommendations[:num_of_products]
```

# SCREENSHOTS AND RESULTS



**Recommending top 5 products to user id 121**

```
n [34]: #Enter 'user index' and 'num_recommendations' for the user
        recommend_items(121,final_ratings_sparse,preds_matrix,5)

        Below are the recommended products for user(user_id = 121):

        Recommended Products
        28761    2.414390
        39003    1.521306
        41420    1.309224
        40158    1.200111
        33819    1.126866
        Name: user_predictions, dtype: float64
```

**Recommending top 10 products to user id 100**

```
n [35]: recommend_items(100,final_ratings_sparse,preds_matrix,10)

        Below are the recommended products for user(user_id = 100):

        Recommended Products
        11078    1.624746
        16159    1.132730
        10276    1.047888
        22210    0.955049
        18887    0.879705
        41618    0.854430
        45008    0.816153
        43419    0.803755
        28761    0.748799
        14791    0.748797
```

## CONCLUSION AND FUTURE ENHANCEMENTS

The "Product Recommendation System" project presents a comprehensive solution for businesses to deliver personalized product recommendations, leveraging advanced artificial intelligence and machine learning techniques. By analyzing vast amounts of customer data, including purchase history, browsing behavior, and demographic information, the system generates accurate and relevant recommendations tailored to individual preferences.
Through collaborative filtering, content-based filtering, and hybrid approaches, the system identifies products that align with each customer's interests and preferences. The user-friendly interface allows users to explore recommendations, provide feedback, and refine preferences over time, enhancing user engagement and satisfaction.
Continuous monitoring and optimization ensure that the recommendation system adapts to evolving customer needs and market trends, driving business success and fostering long-term customer loyalty.

Upcoming Improvements:

1. **Personalization:** Enhance personalization by incorporating additional user attributes such as location, browsing history, and social media interactions to further refine recommendations.
2. **Context-Aware Recommendations:** Incorporate contextual information such as time of day, weather, and user activity to provide more relevant recommendations based on the user's current context.
3. **Dynamic Pricing:** Introduce dynamic pricing strategies based on demand, user behavior, and competitor pricing to optimize revenue and increase customer satisfaction.
4. **Multimodal Recommendations:** Incorporate multimedia content such as images and videos into the recommendation process to provide more immersive and engaging recommendations.
5. **Integration with External Data Sources:** Integrate data from external sources such as product reviews, social media mentions, and industry trends to enrich the recommendation process and provide more comprehensive insights.
6. **Predictive Analytics:** Utilize predictive analytics techniques to anticipate user preferences and behavior, enabling proactive recommendations and personalized marketing campaigns.

By incorporating these future enhancements, the "Product Recommendation System" can further elevate user experience, drive business growth, and maintain a competitive edge in the ever-evolving e-commerce landscape.