

INTERNSHIP REPORT

*A report submitted in partial fulfillment of the requirements for the Award of
Degree of*

**BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE AND ENGINEERING
by**

Name: Ameya Subhash
Roll. No.: BTECH/60082/20

**Under Supervision
of:
DR. BIDYUT PATRA**

**Assistant Professor, Dept. of CSE,
IIT-BHU, Varanasi**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
BIRLA INSTITUTE OF TECHNOLOGY, MESRA
OFF CAMPUS DEOGHAR
DEOGHAR, JHARKHAND-814142**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
BIRLA INSTITUTE OF TECHNOLOGY, MESRA
OFF CAMPUS DEOGHAR
DEOGHAR, JHARKHAND-814142**



CERTIFICATE

This is to certify that the “**Internship report**” submitted by **Ameya Subhash (Roll. No.: BTECH/60082/20)** is work done by her and submitted during **2020-2024** academic year, in partial fulfillment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE AND ENGINEERING**, at **Indian Institute of Technology – BHU, Varanasi**.

Prof. Balaram Mandal
Department of CSE
BIT Deoghar Campus

Dr. Sounak Paul
In-Charge, Dept. of CSE
BIT Deoghar Campus

TRAINING CERTIFICATE OF INTERNSHIP



भारतीय
प्रौद्योगिकी
संस्थान
काशी हिन्दू विश्वविद्यालय

IIT
Serving the nation since 1919

INDIAN
INSTITUTE OF
TECHNOLOGY
BANARAS HINDU UNIVERSITY

प्रशिक्षण एवं प्रस्थापना प्रकोष्ठ TRAINING AND PLACEMENT CELL
VARANASI-221 005

Phone: (0542) 7165958, 2369162 Website: iitbhu.ac.in/tpo E-mail: tpo@iitbhu.ac.in

No. : 3041

TRAINING CERTIFICATE

Name of the Student Ameya Subhash
Father's Name Subhash Choudhary Mother's Name Mani Choudhary
Name of the Institute/College Birla Institute of Technology, Mesra - off campus Deoghar
Course Bachelor of Technology (B.Tech) Branch Computer Science and Engineering (CSE) Semester 6th

Week / Month	Date		Actual working days put in	Remarks	Signature of the Supervisor
	From	To			
1 st	22.05.23	26.05.23	5	—	<u>[Signature]</u>
2 nd	29.05.23	02.06.23	5	—	<u>[Signature]</u>
3 rd	05.06.23	09.06.23	5	—	<u>[Signature]</u>
4 th	12.06.23	16.06.23	5	—	<u>[Signature]</u>
5 th	19.06.23	23.06.23	5	—	<u>[Signature]</u>
6 th	26.06.23	30.06.23	5	—	<u>[Signature]</u>
7 th	03.07.23	07.07.23	5	—	<u>[Signature]</u>
8 th	—	—	—	—	—

Conduct Good

Other remarks, if any Work progress is satisfactory

Signature of Head of the Department/
Coordinator of the School

(with Seal) Head
समन्वयक विज्ञान एवं अभियांत्रिकी विभाग
Department of Computer Sc. & Engg
भारतीय प्रौद्योगिकी संस्थान
Indian Institute of Technology
(बनारस हिन्दू यूनिवर्सिटी)
(Banaras Hindu University)
वाराणसी-221005 / Varanasi-221005

Training and Placement Officer
(with Seal)

समन्वयक/Coordinator
प्रशिक्षण एवं प्रस्थापना प्रकोष्ठ
Training and Placement Cell
भारतीय प्रौद्योगिकी संस्थान (का.हि.वि.)
Indian Institute of Technology (BHU)
वाराणसी/Varanasi -221005

ACKNOWLEDGEMENT

First, I would like to thank Dr. Bidyut Patra, Assistant professor at Dept. of CSE, IIT-BHU for giving me the opportunity to do an internship within the organization.

I also would like all the people that worked along with me **Indian Institute of Technology – BHU, Varanasi** with their patience and openness they created an enjoyable working environment.

It is indeed with a great sense of pleasure and immense sense of gratitude that I acknowledge the help of these individuals.

I am highly indebted to Director Dr. Aruna Jain for the facilities provided to accomplish this internship.

I would like to thank my Head of the Department **Mr. Balaram Mandal** for his constructive criticism throughout my internship.

I would like to thank Dr. Sounak Paul, internship coordinator Department of CSE for their support and advices to get and complete internship in above said organization.

I am extremely great full to my department staff members and friends who helped me in successful completion of this internship.

Ameya Subhash
(BTECH/60082/20)

ABSTRACT

This research internship, conducted under the guidance of Dr. Bidyut Patra, a professor at IIT-BHU, focused on the implementation of a recommender system based on his published paper. The paper explored the concept of item-based collaborative filtering and highlighted its superiority over user-based collaborative filtering.

The practical implementation of the paper's methodology involved the development of a recommender system using a dataset consisting of over one million movie ratings from the Movie Net website. The model not only recommended movies to users but also calculated the error rate in the prediction of movie ratings. Under Dr. Bidyut Patra's supervision, the internship successfully executed the model, demonstrating its effectiveness in making accurate movie recommendations. However, future work is required to further enhance the model by comparing the error rates between item-based and user-based collaborative filtering techniques in the context of the entire recommender system.

This research internship provided valuable hands-on experience in recommender systems, collaborative filtering algorithms, and data analysis. It also offered the opportunity to work closely with Dr. Bidyut Patra, a renowned expert in the field. The internship served as a foundation for potential future research and advancements in recommender systems and collaborative filtering methodologies.

Organisation Information:

IIT BHU: Prestigious institution, renowned faculty, cutting-edge research, diverse programs, innovation hub, state-of-the-art facilities, collaborative environment, industry partnerships, global recognition, academic excellence, empowering students, transformative education, shaping future leaders.

Programs and opportunities:

This research internship offered the opportunity to gain practical experience in implementing a recommender system based on Dr. Bidyut Patra's published paper. The program involved working with a dataset of over one million movie ratings, utilizing item-based collaborative filtering techniques. Under Dr. Patra's guidance, interns had the chance to learn about recommender systems, collaborative filtering algorithms, data analysis, and potentially contribute to advancements in the field.

Methodologies:

The methodologies involved in this research internship included implementing item-based collaborative filtering techniques based on Dr. Bidyut Patra's published paper. The interns worked with a dataset of over one million movie ratings from the website Movie Net. The internship focused on understanding and applying the concepts of recommender systems, collaborative filtering algorithms, data analysis, and error rate prediction. The interns had the opportunity to gain hands-on experience in developing and evaluating the performance of the implemented model under Dr. Patra's guidance.

Key parts of the report:

The key parts of the project implemented during the research internship included:

1. Literature Review: Conducting a thorough review of existing research and literature on recommender systems, specifically focusing on item-based collaborative filtering and its advantages over user-based collaborative filtering.

2. Dataset Collection and Preprocessing: Acquiring a dataset comprising over one million movie ratings from Movie Net and preprocessing the data to ensure its suitability for analysis and model implementation.

3. Model Implementation: Developing and implementing an item-based collaborative filtering model based on the techniques described in Dr. Patra's published paper. This involved designing algorithms and coding to create a functional recommender system.

4. Performance Evaluation: Assessing the performance of the implemented model by measuring its error rate in predicting movie ratings. Comparing the results with other collaborative filtering methods to determine the effectiveness and superiority of item-based filtering.

5. Analysis and Interpretation: Analyzing the findings, identifying patterns and trends in the data, and interpreting the implications of the results obtained. Drawing conclusions and discussing the significance of the research outcomes.

6. Future Work: Identifying potential areas for further improvement and future research, such as comparing item-based and user-based collaborative filtering techniques on a broader recommender system, exploring different datasets, or integrating additional features for enhanced recommendations.

Throughout the project, Dr. Patra's guidance and expertise played a vital role in shaping the direction and success of the research internship.

Benefits of the Company/Institution through your report:

Benefits for Dr. Bidyut Patra and IIT BHU through this research internship project can be outlined as follows:

1. Research Output: The successful implementation of the recommender system based on Dr. Patra's published paper adds to his research portfolio and contributes to his academic reputation. It showcases his expertise in the field of recommender systems and highlights his ability to translate theoretical concepts into practical applications.

2. Validation of Theory: The implementation and evaluation of the item-based collaborative filtering model provide empirical evidence supporting the effectiveness of the approach proposed in Dr. Patra's paper. The positive results validate his research hypothesis and strengthen the credibility of his theoretical framework.

3. Collaborative Effort: The research internship fosters a collaborative environment where Dr. Patra can work closely with a motivated student/researcher. This allows for knowledge exchange, brainstorming, and exploring different perspectives. It enhances Dr. Patra's own understanding of the subject matter and promotes academic growth.

4. Academic Reputation: The successful implementation of Dr. Patra's paper and the subsequent research outcomes enhance his academic reputation and standing in the field of recommender systems. It showcases his expertise and research capabilities, contributing to his professional profile and recognition within the academic community.

5. Research Collaboration: The research internship provides an opportunity for Dr. Patra to collaborate with aspiring researchers. By guiding and supervising the interns, he can foster a productive research environment and build a network of talented individuals who can contribute to future research projects.

6. Knowledge Transfer: Through the internship, Dr. Patra can share his knowledge and expertise with the interns, helping them gain practical experience in implementing complex algorithms and models. This knowledge transfer strengthens the research capabilities of the interns and equips them with valuable skills for their future careers.

Overall, this research internship project benefits Dr. Bidyut Patra by enhancing his academic reputation and facilitating research collaboration. Similarly, IIT BHU benefits through strengthened research capabilities, increased visibility in the academic community, and potential industry partnerships.

INDEX

S.no	CONTENTS	Page no
1.	INTRODUCTION	13
2.	SYSTEM ANALYSIS	14
3.	BRIEF REVIEW OF EXISTING WORK/MODEL	15
4.	EXPERIMENTAL SETUP.....	16
5.	MODEL/METHODOLOGY	17
6.	CODING AND SCREENSHOT... ..	18
7.	APPLICATION OF THE PROJECT WITH LIMITATION	33
8.	CONCLUSION.....	35
9.	REFERENCE.....	36

Learning Objectives/Internship Objectives

1. Gain practical experience: The internship provides hands-on experience in implementing a research paper on item-based collaborative filtering, enabling the intern to apply theoretical knowledge in a real-world setting and understand project complexities.

2. Understand recommender systems: The internship delves into the field of recommender systems, allowing the intern to develop a comprehensive understanding of concepts, algorithms, and the advantages of item-based collaborative filtering over user-based approaches.

3. Data analysis and modeling: Working with a large movie rating dataset, the intern learns data analysis and modeling skills, including preprocessing, deriving insights, and building predictive models for movie recommendations.

4. Error prediction and evaluation: The internship focuses on developing a model that predicts movie ratings and evaluates error rates. By comparing item-based and user-based collaborative filtering, the intern learns to assess recommender system performance and identify areas for improvement.

5. Mentorship and guidance: Under Dr. Bidyut Patra's mentorship, the intern receives valuable guidance, fostering research and analytical skills, problem-solving abilities, and a deeper understanding of the subject matter. This mentorship enhances the overall learning experience.

WEEKLY OVERVIEW OF INTERNSHIP ACTIVITIES

1st WEEK	DATE	DAY	NAME OF THE TOPIC/MODULE COMPLETED
	22/05/23	Monday	Started Andrew Ng Supervised Machine Learning (course 1)
	23/05/23	Tuesday	Completed Week 1: Introduction to ML
	24/05/23	Wednesday	Started Week 2 of the course
	25/05/23	Thursday	Completed Week 2: Regression
	26/05/23	Friday	Started Week 3 of the course

2nd WEEK	DATE	DAY	NAME OF THE TOPIC/MODULE COMPLETED
	29/05/23	Monday	Completed Week 3: Classification
	30/05/23	Tuesday	Started Andrew Ng Advanced Learning Algorithms (course 2)
	31/05/23	Wednesday	Started Week 1 of the course
	01/06/23	Thursday	Completed half of Week 1
	02/06/23	Friday	Completed Week 1: Neural Network

3rd WEEK	DATE	DAY	NAME OF THE TOPIC/MODULE COMPLETED
	05/06/23	Monday	Started Week 2 of the course
	06/06/23	Tuesday	Completed half of Week 2: Neural Network Training
	07/06/23	Wednesday	Completed Week 2: Neural Network Training
	08/06/23	Thursday	Started Week 3 of the course
	09/06/23	Friday	Completed Week 3: Bias and Variance

4 th WEEK	DATE	DAY	NAME OF THE TOPIC/MODULE COMPLETED
	12/06/23	Monday	Started Andrew Ng Recommender System (course 2)
	13/06/23	Tuesday	Started Week 2 of the course
	14/06/23	Wednesday	Completed Week 2: Recommender Systems
	15/06/23	Thursday	Started Reading about research paper on recommender System
	16/06/23	Friday	Started Reading about user-based collaborative filtering

5 th WEEK	DATE	DAY	NAME OF THE TOPIC/MODULE COMPLETED
	19/06/23	Monday	Started Reading the research paper of recommender system Published by Dr. Bidyut Patra
	20/06/23	Tuesday	Continue reading on paper.....
	21/06/23	Wednesday	Comparison on user-based and item-based collaborative filtering
	22/06/23	Thursday	Item-based collaborative filtering Algorithm
	23/06/23	Friday	Item similarity Computation

6 th WEEK	DATE	DAY	NAME OF THE TOPIC/MODULE COMPLETED
	26/06/23	Monday	Prediction Computation
	27/06/23	Tuesday	Performance Implications
	28/06/23	Wednesday	Experimental Evaluation and result
	29/06/23	Thursday	Implementation of the research paper by taking the dataset and giving it a formate of train and test dataset
	30/06/23	Friday	Implementing Adjusted Cosine similarity formula

7 th WEEK	DATE	DAY	NAME OF THE TOPIC/MODULE COMPLETED
	03/07/23	Monday	Implemented prediction formula for non-rated items
	04/07/23	Tuesday	Predicting rating for single movie at a time
	05/07/23	Wednesday	Modified the prediction formula for all the unrated items for a single user
	06/07/23	Thursday	Calculated the accuracy of the model
	07/07/23	Friday	Tested the model and calculated the Mean absolute error (MAE) for the model

1. INTRODUCTION

The project focuses on implementing an item-based collaborative filtering recommender system for movies. Collaborative filtering is a widely used technique to personalize recommendations based on users' past interactions. In this context, the system predicts movies that a user might be interested in, by analyzing their historical movie ratings and comparing them with other users' preferences. The emphasis lies in leveraging item-based collaborative filtering, which offers advantages over traditional user-based approaches.

The primary objective of the project is twofold: first, to accurately predict movie ratings for users based on their previous behavior, and second, to assess the error rate in these predictions. By evaluating the error rate, the system can measure the effectiveness of its recommendations and identify areas for potential improvement.

To achieve these goals, the project utilizes a dataset comprising over one million movie ratings from the Movie Net website. The practical implementation of the research paper's concepts allows for a deeper understanding of recommender systems, data analysis, and predictive modeling. The project provides valuable hands-on experience and mentorship, enabling the intern to gain practical insights into the complexities of building an efficient item-based collaborative filtering recommender system.

2. SYSTEM ANALYSIS

The system analysis for this project involves the development of an item-based collaborative filtering recommender system that predicts movie preferences for users based on their historical ratings. The system utilizes a dataset of over one million movie ratings from the website "Movie Net." Through item-based collaborative filtering, the system identifies similar movies and recommends them to users with comparable interests.

The core functionality of the system includes:

1. Item-Based Collaborative Filtering: The system employs item-based collaborative filtering algorithms to identify movies that are similar to the ones a user has rated positively. By analyzing historical user ratings, the system recommends movies with similar attributes, enhancing personalized recommendations.

2. Prediction Generation: Based on the item-based collaborative filtering approach, the system generates predictions for how a user would rate specific movies. These predictions assist in recommending movies that align with the user's preferences.

3. Error Rate Calculation: The system evaluates the accuracy of its predictions by calculating the error rate for each user. By comparing predicted ratings with actual ratings, it measures the effectiveness of the item-based collaborative filtering method.

4. Dataset Handling: The system efficiently handles a large dataset of movie ratings, performing data preprocessing and analysis to derive meaningful insights for recommendation generation.

The ultimate goal of this system analysis is to create an effective and efficient item-based collaborative filtering recommender system that provides accurate movie predictions for users and helps identify areas for improving recommendation accuracy through error rate analysis.

3. BRIEF REVIEW OF EXISTING WORK/MODEL

The implemented model for the recommender system is based on item-based collaborative filtering and utilizes a dataset of over one million movie ratings. The dataset is split into an 80% training set and a 20% test set. The primary goal of the model is to predict the rating of a movie for a user and make personalized recommendations.

The model is using of adjusted cosine similarity to measure the similarity between rated items and the target item for which a prediction is required. This similarity calculation will help in identifying items that share similarities with the target, allowing for accurate predictions. In the prediction process, the model computes the similarity scores between the target movie and the other rated movies in the training dataset. Based on these similarity scores, the model determines the predicted rating for the target movies. Those movies are not rated by the user.

To evaluate the model's performance, the prediction phase is executed on the test dataset. The model's predicted ratings are then compared to the actual ratings in the test dataset. The mean absolute error (MAE) metric is used to quantify the accuracy of the model's predictions. The strength of this model lies in its utilization of item-based collaborative filtering, which has demonstrated advantages over user-based collaborative filtering in certain scenarios. By dividing the dataset into training and test sets, the model ensures its generalizability and robustness. The use of adjusted cosine similarity allows the model to capture more nuanced item relationships, further enhancing the accuracy of its recommendations.

Overall, this model provides an approach to building a recommender system based on item-based collaborative filtering. It successfully addresses the task of movie rating prediction and recommendation, making it a valuable tool for personalized user experiences in the context of large-scale movie databases.

4. EXPERIMENTAL SETUP

1. Data Collection:

Import the required datasets containing movie ratings, movie information, and user information from Movie Net.

2. Data Preprocessing:

Perform data preprocessing, handling missing values, duplicates, and ensuring data consistency.

Convert the data into a suitable format for further analysis.

3. Train-Test Split:

Split the dataset into an 80% training set and a 20% test set using `train_test_split` from `scikit-learn`.

Prepare the training and test data for building the item-based collaborative filtering model.

4. Evaluation using Mean Absolute Error (MAE):

Calculate the mean absolute error (MAE) between the predicted ratings and actual ratings in the test set.

MAE will measure the accuracy of the item-based collaborative filtering model.

5. MODEL/METHODOLOGY

1. Data Loading and Preprocessing:

- Mount the Google Drive and load the movie ratings data from CSV files (movies.csv, rating.csv, users.csv) into Pandas DataFrames (data_movies, data_ratings, data_users).
- Check the data and perform necessary preprocessing, such as handling missing values.

2. Train-Test Split:

- Split the movie ratings data into training (X_train) and test (X_test) sets using the ``train_test_split`` function from ``sklearn.model_selection``.

3. Data Transformation:

- Transform the data from the training and test sets into matrix form (data_ratings_array, dummy_train_array, dummy_test_array) suitable for collaborative filtering.
- Prepare the dummy_train and dummy_test matrices, where the missing ratings are marked as 0 for evaluation and 1 for prediction, respectively.

4. Item-Based Collaborative Filtering:

- Implement the ``adjusted_cosine_similarity`` function to calculate adjusted cosine similarity between items in the training set.
- Create a function ``predict_ratings`` to predict movie ratings for users in the test set based on item-based collaborative filtering.

5. Rating Prediction and Evaluation:

- For each user in the test set, predict the movie ratings using the ``predict_ratings`` function.
- Calculate the Mean Absolute Error (MAE) between the predicted ratings and the actual ratings from the test set using the ``mean_absolute_error`` function.

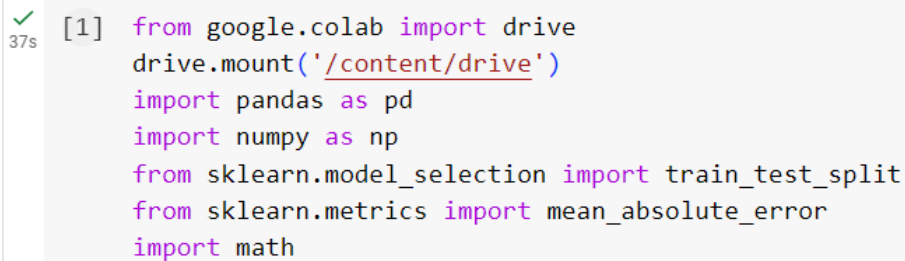
6. Results and Analysis:

- Print the MAE value as a measure of the model's performance in predicting movie ratings.
- Output the similarity values and predicted ratings for the specified user.
- Compare the predicted ratings with the actual ratings to gain insights into the model's accuracy and quality of recommendations.

The model uses item-based collaborative filtering with adjusted cosine similarity to make personalized movie recommendations for users based on their ratings. The Python code leverages machine learning libraries such as `pandas`, `numpy`, and `scikit-learn` to facilitate data manipulation, matrix operations, and the evaluation of the model's performance.

6. CODING AND SCREENSHOT

```
from google.colab import drive
drive.mount('/content/drive')
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error
import math
```



```
[1] from google.colab import drive
    drive.mount('/content/drive')
    import pandas as pd
    import numpy as np
    from sklearn.model_selection import train_test_split
    from sklearn.metrics import mean_absolute_error
    import math
```

Mounted at /content/drive

```
data_movies = pd.read_csv('/content/drive/MyDrive/movies rating/movies.csv')
data_ratings = pd.read_csv('/content/drive/MyDrive/movies rating/rating.csv')
```

```
data_users = pd.read_csv('/content/drive/MyDrive/movies rating/users.csv')
```

```
✓ [2] data_movies = pd.read_csv('/content/drive/MyDrive/movies rating/movies.csv');  
3s data_ratings = pd.read_csv('/content/drive/MyDrive/movies rating/rating.csv')  
data_users = pd.read_csv('/content/drive/MyDrive/movies rating/users.csv')
```

```
data_ratings.head()
```

```
✓ [3] data_ratings.head()  
0s
```

	UserID	MovieID	Rating
0	1	1193	5
1	1	661	3
2	1	914	3
3	1	3408	4
4	1	2355	5

```
data_ratings.tail()
```

```
✓ [4] data_ratings.tail()  
0s
```

	UserID	MovieID	Rating
1000204	6040	1091	1
1000205	6040	1094	5
1000206	6040	562	5
1000207	6040	1096	4
1000208	6040	1097	4

```
data_ratings.shape
```

```
data_ratings.shape
```

```
(1000209, 3)
```

data_ratings.info()

```
[6] data_ratings.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 1000209 entries, 0 to 1000208  
Data columns (total 3 columns):  
#   Column      Non-Null Count  Dtype  
---  ---  
0   UserID      1000209 non-null  int64  
1   MovieID     1000209 non-null  int64  
2   Rating      1000209 non-null  int64  
dtypes: int64(3)  
memory usage: 22.9 MB
```

data_ratings= pd.DataFrame(data_ratings)

print(data_ratings)

```
[7] data_ratings= pd.DataFrame(data_ratings)  
print(data_ratings)
```

	UserID	MovieID	Rating
0	1	1193	5
1	1	661	3
2	1	914	3
3	1	3408	4
4	1	2355	5
...
1000204	6040	1091	1
1000205	6040	1094	5
1000206	6040	562	5
1000207	6040	1096	4
1000208	6040	1097	4

```
[1000209 rows x 3 columns]
```

data_ratings_array=data_ratings_array = data_ratings.pivot(index='UserID',


```
columns='MovieID', values='Rating').fillna(0).values  
print(data_ratings_array)
```

```
✓ [8] data_ratings_array=data_ratings_array = data_ratings.pivot(index='UserID',  
0s  
✓ [9] print(data_ratings_array)  
0s  
[[5. 0. 0. ... 0. 0. 0.]  
 [0. 0. 0. ... 0. 0. 0.]  
 [0. 0. 0. ... 0. 0. 0.]  
 ...  
 [0. 0. 0. ... 0. 0. 0.]  
 [0. 0. 0. ... 0. 0. 0.]  
 [3. 0. 0. ... 0. 0. 0.]
```

```
X_train, X_test = train_test_split(data_ratings, test_size = 0.20, random_state = 42)
```

```
print(X_train.shape)  
print(X_test.shape)
```

```
✓ [10] X_train, X_test = train_test_split(data_ratings, test_size = 0.20, random_s  
0s  
      print(X_train.shape)  
      print(X_test.shape)  
      (800167, 3)  
      (200042, 3)
```

```
X_train.shape
```

```
✓ [11] X_train.shape  
0s  
      (800167, 3)
```

```
dummy_train = X_train.copy()  
dummy_test = X_test.copy()
```

```
# The movies not rated by user is marked as 1 for prediction
```

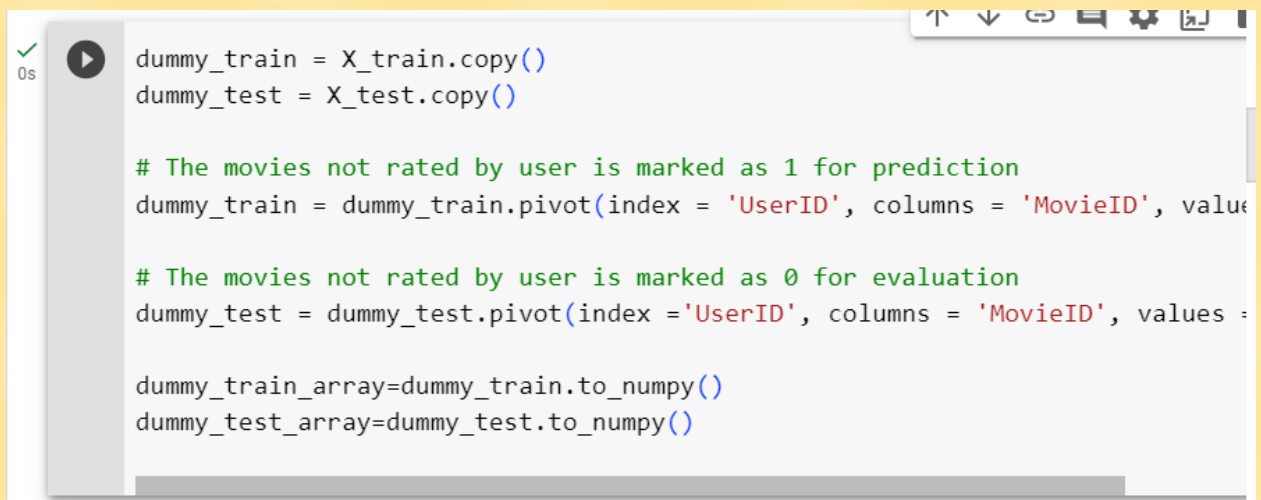
```
dummy_train = dummy_train.pivot(index = 'UserID', columns = 'MovieID', values = 'Rating')
```

```
# The movies not rated by user is marked as 0 for evaluation
```

```
dummy_test = dummy_test.pivot(index = 'UserID', columns = 'MovieID', values = 'Rating')
```

```
dummy_train_array=dummy_train.to_numpy()
```

```
dummy_test_array=dummy_test.to_numpy()
```



```
0s dummy_train = X_train.copy()
dummy_test = X_test.copy()

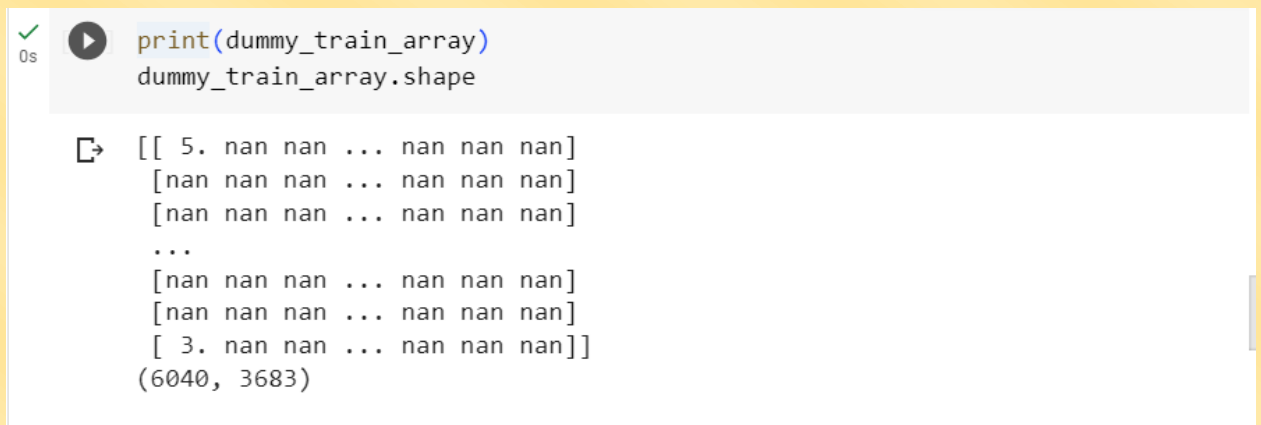
# The movies not rated by user is marked as 1 for prediction
dummy_train = dummy_train.pivot(index = 'UserID', columns = 'MovieID', values = 'Rating')

# The movies not rated by user is marked as 0 for evaluation
dummy_test = dummy_test.pivot(index = 'UserID', columns = 'MovieID', values = 'Rating')

dummy_train_array=dummy_train.to_numpy()
dummy_test_array=dummy_test.to_numpy()
```

```
print(dummy_train_array)
```

```
dummy_train_array.shape
```



```
0s print(dummy_train_array)
dummy_train_array.shape
```

```
[[ 5. nan nan ... nan nan nan]
 [nan nan nan ... nan nan nan]
 [nan nan nan ... nan nan nan]
 ...
 [nan nan nan ... nan nan nan]
 [nan nan nan ... nan nan nan]
 [ 3. nan nan ... nan nan nan]]
(6040, 3683)
```

```
print(dummy_test_array)
```

```
dummy_test_array.shape
```

```
✓ [14] print(dummy_test_array)
0s      dummy_test_array.shape

[[nan nan nan ... nan nan nan]
 [nan nan nan ... nan nan nan]
 [nan nan nan ... nan nan nan]
 ...
 [nan nan nan ... nan nan nan]
 [nan nan nan ... nan nan nan]
 [nan nan nan ... nan nan nan]]
(6038, 3444)
```

```
dummy_train_array.shape[1]
```

```
user_id = int(input("Enter UserID: "))
```

```
✓ [15] dummy_train_array.shape[1]
0s      3683

✓ [16] user_id = int(input("Enter UserID: "))
6s

Enter UserID: 987
```

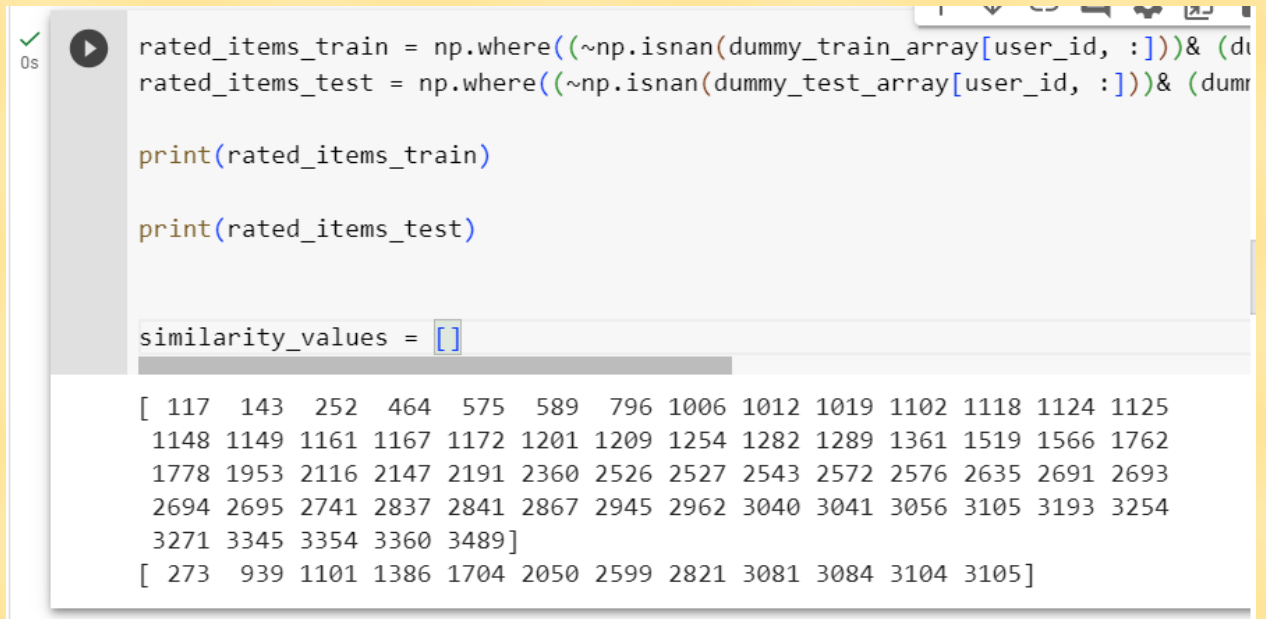
```
rated_items_train = np.where((~np.isnan(dummy_train_array[user_id, :]))&
(dummy_train_array[user_id, :] >= 1) & (dummy_train_array[user_id, :] <= 5))[0]
```

```
rated_items_test = np.where((~np.isnan(dummy_test_array[user_id, :]))&
(dummy_test_array[user_id, :] >= 1) & (dummy_test_array[user_id, :] <= 5))[0]
```

```
print(rated_items_train)
```

```
print(rated_items_test)
```

```
similarity_values = []
```

A screenshot of a Jupyter Notebook cell. The cell contains Python code that filters out NaN values from two dummy arrays (dummy_train_array and dummy_test_array) to create rated_items_train and rated_items_test. It then prints these arrays. Below the code, the output of the print statements is shown, displaying two 1D arrays of item IDs. The first array has 15 elements, and the second array has 10 elements.

```
0s ✓ [ 117  143  252  464  575  589  796 1006 1012 1019 1102 1118 1124 1125
rated_items_train = np.where((~np.isnan(dummy_train_array[user_id, :])) & (dummy_train_array[user_id, :] > 0))
rated_items_test = np.where((~np.isnan(dummy_test_array[user_id, :])) & (dummy_test_array[user_id, :] > 0))

print(rated_items_train)

print(rated_items_test)

similarity_values = []
```

```
[ 117  143  252  464  575  589  796 1006 1012 1019 1102 1118 1124 1125
 1148 1149 1161 1167 1172 1201 1209 1254 1282 1289 1361 1519 1566 1762
 1778 1953 2116 2147 2191 2360 2526 2527 2543 2572 2576 2635 2691 2693
 2694 2695 2741 2837 2841 2867 2945 2962 3040 3041 3056 3105 3193 3254
 3271 3345 3354 3360 3489]
[ 273  939 1101 1386 1704 2050 2599 2821 3081 3084 3104 3105]
```

```
def adjusted_cosine_similarity(ratings_matrix, u, v):
```

```
    ratings_u = ratings_matrix[:, u]
```

```
    ratings_v = ratings_matrix[:, v]
```

```
    bu = np.nanmean(ratings_u)
```

```
    print
```

```
    bv = np.nanmean(ratings_v)
```

```
    centered_ratings_u = ratings_u - bu
```

```
    centered_ratings_v = ratings_v - bv
```

```
    numerator = np.nansum(centered_ratings_u * centered_ratings_v)
```

```
    denominator = np.sqrt(np.nansum(centered_ratings_u**2)) *
```

```
    np.sqrt(np.nansum(centered_ratings_v**2))
```



```
if denominator == 0.0 or np.isnan(denominator):
```

```
    similarity = 0.0
```

```
else:
```

```
    similarity = numerator / denominator
```

```
return similarity
```

```
✓ [18] def adjusted_cosine_similarity(ratings_matrix, u, v):  
0s     ratings_u = ratings_matrix[:, u]  
     ratings_v = ratings_matrix[:, v]  
  
     bu = np.nanmean(ratings_u)  
     print  
     bv = np.nanmean(ratings_v)  
  
     centered_ratings_u = ratings_u - bu  
     centered_ratings_v = ratings_v - bv  
  
     numerator = np.nansum(centered_ratings_u * centered_ratings_v)  
     denominator = np.sqrt(np.nansum(centered_ratings_u**2)) * np.sqrt(np.nansum(centered_ratings_v**2))  
  
     if denominator == 0.0 or np.isnan(denominator):  
         similarity = 0.0  
     else:  
         similarity = numerator / denominator  
     return similarity
```

```
predicted_ratings = {}
```

```
similarity_values = []
```

```
similarity_dict = {}
```

```
sorted_similarity_dict={}
```

```
pos_similarity_dict={}
```

✓
0s

```
[19] predicted_ratings = {}  
      similarity_values = []  
      similarity_dict = {}  
      sorted_similarity_dict={}  
      pos_similarity_dict={}
```

```
def predict_ratings(user_id, rated_items_test):  
    global predicted_ratings  
    global similarity_values  
    global similarity_dict  
    global sorted_similarity_dict  
    global pos_similarity_dict  
    for item_id in rated_items_test:  
  
        for item in rated_items_train:  
            similarity = adjusted_cosine_similarity(dummy_train_array, item_id, item)  
            similarity_values.append(similarity)  
  
        similarity_dict = dict(zip(rated_items_train, similarity_values))  
        sorted_similarity_dict = dict(sorted(similarity_dict.items(), key=lambda x: x[1],  
reverse=True))  
        pos_similarity_dict = {k: v for k, v in sorted_similarity_dict.items() if v > 0}  
  
        weighted_sum = 0.0  
        similarity_sum = 0.0  
  
        for rated_item, similarity_value in pos_similarity_dict.items():  
            rating = dummy_train_array[user_id, rated_item]  
  
            weighted_sum += similarity_value * rating
```

```
similarity_sum += similarity_value
```

```
if similarity_sum == 0.0 or np.isnan(similarity_sum):
```

```
    predicted_rating = 0.0
```

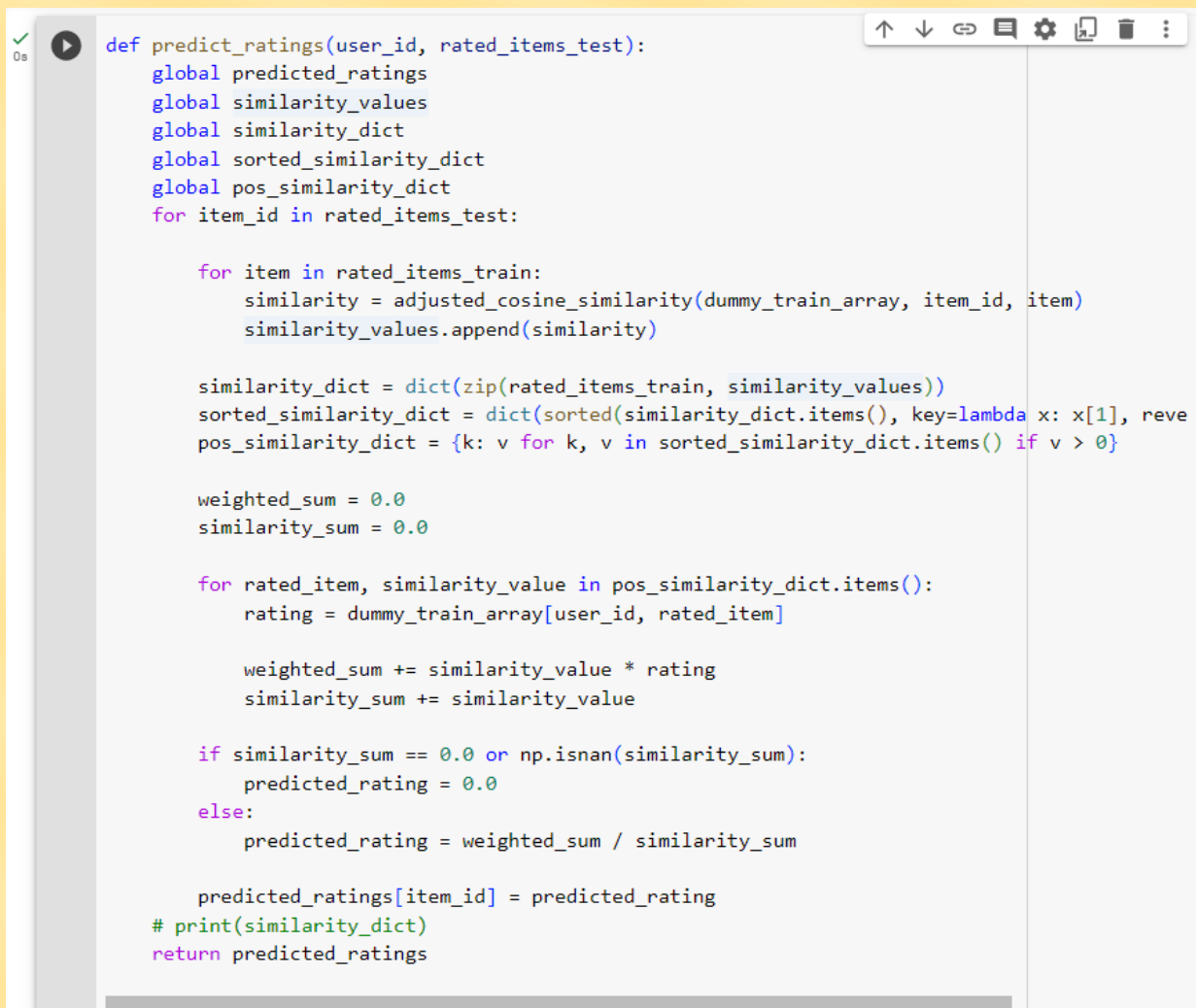
```
else:
```

```
    predicted_rating = weighted_sum / similarity_sum
```

```
predicted_ratings[item_id] = predicted_rating
```

```
# print(similarity_dict)
```

```
return predicted_ratings
```

A screenshot of a code editor with a light blue background. The code is written in Python and implements a function named `predict_ratings`. The function takes `user_id` and `rated_items_test` as arguments. It uses global variables `predicted_ratings`, `similarity_values`, `similarity_dict`, `sorted_similarity_dict`, and `pos_similarity_dict`. The function iterates over `rated_items_test` and for each `item_id`, it iterates over `rated_items_train` to calculate the adjusted cosine similarity. It then builds a `similarity_dict`, sorts it to create `sorted_similarity_dict`, and filters for positive similarities to create `pos_similarity_dict`. It calculates a weighted sum and a similarity sum for each item. If the similarity sum is zero or NaN, it sets the predicted rating to 0.0. Otherwise, it calculates the predicted rating as the weighted sum divided by the similarity sum. Finally, it updates `predicted_ratings` for the current `item_id`, prints the `similarity_dict`, and returns `predicted_ratings`.

```
def predict_ratings(user_id, rated_items_test):  
    global predicted_ratings  
    global similarity_values  
    global similarity_dict  
    global sorted_similarity_dict  
    global pos_similarity_dict  
    for item_id in rated_items_test:  
        for item in rated_items_train:  
            similarity = adjusted_cosine_similarity(dummy_train_array, item_id, item)  
            similarity_values.append(similarity)  
  
        similarity_dict = dict(zip(rated_items_train, similarity_values))  
        sorted_similarity_dict = dict(sorted(similarity_dict.items(), key=lambda x: x[1], reverse=True))  
        pos_similarity_dict = {k: v for k, v in sorted_similarity_dict.items() if v > 0}  
  
        weighted_sum = 0.0  
        similarity_sum = 0.0  
  
        for rated_item, similarity_value in pos_similarity_dict.items():  
            rating = dummy_train_array[user_id, rated_item]  
  
            weighted_sum += similarity_value * rating  
            similarity_sum += similarity_value  
  
        if similarity_sum == 0.0 or np.isnan(similarity_sum):  
            predicted_rating = 0.0  
        else:  
            predicted_rating = weighted_sum / similarity_sum  
  
        predicted_ratings[item_id] = predicted_rating  
    # print(similarity_dict)  
    return predicted_ratings
```

```

def mean_absolute_error(predicted_ratings, actual_ratings):
    abs_diff = 0.0
    count = 0

    for i in range(len(predicted_ratings)):
        if i < len(actual_ratings):
            abs_diff += abs(actual_ratings[i] - predicted_ratings[i])
            count += 1

    if count > 0:
        mae = abs_diff / count
    else:
        mae = 0.0

    return mae

actual_ratings = dummy_test_array[user_id, rated_items_test]
actual_ratings_dict = dict(zip(rated_items_test, actual_ratings))

predicted_ratings = predict_ratings(user_id, rated_items_test)
predicted_ratings_values = list(predicted_ratings.values())
mae = mean_absolute_error(predicted_ratings_values, actual_ratings)
print("\nMean Absolute Error: ", mae)

```



```
0s ✓ def mean_absolute_error(predicted_ratings, actual_ratings):
    abs_diff = 0.0
    count = 0

    for i in range(len(predicted_ratings)):
        if i < len(actual_ratings):
            abs_diff += abs(actual_ratings[i] - predicted_ratings[i])
            count += 1

    if count > 0:
        mae = abs_diff / count
    else:
        mae = 0.0

    return mae

0s ✓ [22] actual_ratings = dummy_test_array[user_id, rated_items_test]
    actual_ratings_dict = dict(zip(rated_items_test, actual_ratings))

0s ✓ [23] predicted_ratings = predict_ratings(user_id, rated_items_test)
    predicted_ratings_values = list(predicted_ratings.values())
    mae = mean_absolute_error(predicted_ratings_values, actual_ratings)
    print("\nMean Absolute Error: ", mae)

Mean Absolute Error: 1.1081428515856586
```

```
print(rated_items_train)
```

```
print(rated_items_test)
```

```
print("Similarity values of userID: ", user_id)
```

```
print(similarity_dict)
```

```
print("\nPredicted rating for userID: ", user_id)
```

```
print(predicted_ratings)
```

```
print("\nActual rating: ", actual_ratings_dict)
```

```
print("\nMean Absolute Error for userID: ", user_id)
```

```
print(mae)
```

```
# print("\n", rated_items_train)
```

```
# print(rated_items_test)
```

✓
0s

```
▶ print(rated_items_train)
print(rated_items_test)

print("Similarity values of userID: ",user_id)
print(similarity_dict)
print("\nPredicted rating for userID: ",user_id)
print(predicted_ratings)
print("\nActual rating: ",actual_ratings_dict)
print("\nMean Absolute Error for userID: ",user_id)
print(mae)
# print("\n",rated_items_train)
# print(rated_items_test)
```

```
↳ [ 117  143  252  464  575  589  796 1006 1012 1019 1102 1118 1124 1125
    1148 1149 1161 1167 1172 1201 1209 1254 1282 1289 1361 1519 1566 1762
    1778 1953 2116 2147 2191 2360 2526 2527 2543 2572 2576 2635 2691 2693
    2694 2695 2741 2837 2841 2867 2945 2962 3040 3041 3056 3105 3193 3254
    3271 3345 3354 3360 3489]
[ 273  939 1101 1386 1704 2050 2599 2821 3081 3084 3104 3105]
Similarity values of userID:  987
{117: 0.02250338622638183, 143: 0.007944249467961394, 252: 0.017433731741886865, 464
Predicted rating for userID:  987
{273: 4.162214277378488, 939: 4.162214277378488, 1101: 4.162214277378488, 1386: 4.16
Actual rating:  {273: 4.0, 939: 4.0, 1101: 4.0, 1386: 1.0, 1704: 5.0, 2050: 4.0, 259
Mean Absolute Error for userID:  987
1.1081428515856586
```

Final Output:

[117 143 252 464 575 589 796 1006 1012 1019 1102 1118 1124 1125
1148 1149 1161 1167 1172 1201 1209 1254 1282 1289 1361 1519 1566 1762
1778 1953 2116 2147 2191 2360 2526 2527 2543 2572 2576 2635 2691 2693
2694 2695 2741 2837 2841 2867 2945 2962 3040 3041 3056 3105 3193 3254
3271 3345 3354 3360 3489]

[273 939 1101 1386 1704 2050 2599 2821 3081 3084 3104 3105]

Similarity values of userID: 987{117: 0.02250338622638183, 143: 0.007944249467961394,
252: 0.017433731741886865, 464: -0.005260908669116286, 575: 0.030974947207173913,
589: 0.03804110277328473, 796: 0.03320321735619562, 1006: 0.04973136256785386,
1012: 0.038393732329919036, 1019: 0.03207613713353227, 1102: 0.015747313070699938,
1118: 0.019911885529747084, 1124: 0.007314503668197317, 1125: 0.01056027359174633,
1148: 0.013245909948208172, 1149: 0.03435428141202326, 1161: 0.044434552674177966,
1167: -0.0013672007679487073, 1172: 0.018617001733228042, 1201:
0.025166500896265775, 1209: 0.030969079765496656, 1254: 0.030565509278072726,
1282: -0.003414029743450797, 1289: 0.05928782870347655, 1361:
0.013450031659983677, 1519: -0.007004430410303464, 1566: 0.0017287052587902656,
1762: 0.012173912287386963, 1778: 0.04117099109597159, 1953: 0.0535117384305543,
2116: 0.038691571449644024, 2147: 0.054159082096822825, 2191:
0.056777796902737425, 2360: 0.018760165434597045, 2526: 0.01408369354636499, 2527:
0.030326330341724272, 2543: 0.032884212500080785, 2572: 0.05266077562009939, 2576:
0.04861062491493559, 2635: 0.029332408814125768, 2691: 0.05813140817103815, 2693:
0.09208897304044077, 2694: 0.045320551868821675, 2695: 0.018260047777835506, 2741:
0.059947588197347725, 2837: 0.1087690314183129, 2841: 0.0504430897409682, 2867:
0.06403783351914807, 2945: -0.0014488286489708377, 2962: 0.11609386428900141,
3040: 0.02564565175506007, 3041: 0.006249296145486994, 3056: 0.037827654630244824,
3105: 0.01702248959931926, 3193: 0.03343487285402384, 3254: 0.038409770664613235,

3271: 0.02570513599141206, 3345: 0.024508571425940363, 3354: -
0.021861301093482596, 3360: 0.005678169521009291, 3489: -0.008457635434200984}

Predicted rating for userID: 987

{273: 4.162214277378488, 939: 4.162214277378488, 1101: 4.162214277378488, 1386:
4.162214277378488, 1704: 4.162214277378488, 2050: 4.162214277378488, 2599:
4.162214277378488, 2821: 4.162214277378488, 3081: 4.162214277378488, 3084:
4.162214277378488, 3104: 4.162214277378488, 3105: 4.162214277378488}

Actual rating: {273: 4.0, 939: 4.0, 1101: 4.0, 1386: 1.0, 1704: 5.0, 2050: 4.0, 2599: 4.0, 2821:
4.0, 3081: 5.0, 3084: 2.0, 3104: 1.0, 3105: 2.0}

Mean Absolute Error for userID: 987

1.1081428515856586

7. APPLICATION OF THE PROJECT WITH LIMITATIONS

Application of the Model:

The implemented item-based collaborative filtering model with adjusted cosine similarity has various practical applications in the domain of personalized recommendation systems for movies or any other items. It enables the system to predict movie ratings and generate personalized movie recommendations for individual users. This application can be utilized in online movie platforms, streaming services, or e-commerce platforms, where providing personalized recommendations is crucial for enhancing user engagement and satisfaction.

Limitations of the Model:

The current limitation of the model is that it can only predict ratings for a single user at a time. While the model can generate accurate predictions for individual users, it lacks the ability to scale and predict ratings for all users simultaneously. This constraint hinders its efficiency in large-scale applications with a substantial user base.

Future Scope of the Model:

To overcome the limitation and enhance the model's scalability and usability, the future scope of this model involves the following advancements:

1. Parallel Processing:

Implement parallel processing techniques to predict ratings and generate recommendations for multiple users simultaneously. This optimization can significantly speed up the process and improve the overall performance of the recommender system.

2. Accuracy Rate for All Users:

Develop a mechanism to calculate the accuracy rate for predictions made for all users in the dataset. This will enable comprehensive evaluation and comparison of the model's performance for different users and highlight areas for further improvement.

By addressing the limitations and incorporating these future enhancements, the item-based collaborative filtering model can evolve into a powerful and scalable recommender system, providing highly personalized movie recommendations for a broad user base. This would significantly enhance user satisfaction and engagement, making the platform more appealing and competitive in the movie recommendation market.

8. CONCLUSION

In conclusion, this project on building an item-based collaborative filtering recommender system has been a rewarding journey in the realm of machine learning and personalized recommendations. Starting my machine learning journey with Andrew Ng's Coursera course provided a solid foundation, allowing me to grasp the fundamental concepts and techniques. Delving into recommender systems and their applications further piqued my interest, and I began exploring various resources such as YouTube, Stack Overflow, GitHub, and even ChatGPT to augment my learning.

As a Java developer, transitioning to Python for this project presented challenges, but the perseverance and dedication paid off. I successfully implemented the item-based collaborative filtering model using Python and libraries like pandas, numpy, and scikit-learn. This project offered a hands-on experience in building and evaluating a recommendation system, enriching my practical understanding of machine learning. Throughout the process, I received invaluable guidance from Dr. Patra, whose expertise and mentorship were instrumental in the successful implementation of the model. I am truly grateful for his support and encouragement.

The model's application in predicting movie ratings and generating personalized recommendations showcases its potential in enhancing user engagement and satisfaction on various platforms. While the model's current limitation lies in its ability to predict ratings for a single user at a time, the future scope of implementing parallel processing and evaluating accuracy rates for all users holds promise for scalability and wider usability.

In conclusion, this project has been a significant learning experience, not only in machine learning and Python but also in problem-solving, adaptability, and perseverance. It highlights the power of continuous learning, the significance of mentorship, and the joy of overcoming challenges. I take immense pride in successfully implementing this model and look forward to exploring further advancements in the field of machine learning and recommender systems.

9. REFERENCES

1. Dr. Bidyut Patra recommender system research paper:
http://files.grouplens.org/papers/www10_sarwar.pdf
2. Deployed Code of the model:
https://github.com/ameyasubhash/recommender_system/tree/main
3. Andrew Ng machine learning and deep learning course:
<https://www.coursera.org/specializations/machine-learning-introduction>