Collaborator-based Movie Recommendation System

Project of CSE 523: Machine Learning

Jainam Shah
School of Engineering and Applied Science,
Ahmedabad, Gujarat,
AU2040186
jainam.s4@ahduni.edu.in

Akshay Parmar
School of Engineering and Applied Science,
Ahmedabad, Gujarat,
AU2040199
akshay.p@ahduni.edu.in

Shubham Bhutt
School of Engineering and Applied Science,
Ahmedabad, Gujarat,
AU2040206
shubham.b1@ahduni.edu.in

Yash Chotaliya
School of Engineering and Applied Science,
Ahmedabad, Gujarat,
AU2040193
yashkumar.c@ahduni.edu.in

***Abstract*— Collaborator-based movie recommendation systems rely on the ratings and reviews of other users with similar tastes to generate recommendations for a given user. In this system, the user's preference is inferred based on the ratings of other users with similar profiles. The system then recommends movies that similar users have enjoyed but that the given user has not yet seen. Collaborator-based recommendation systems can be implemented using different algorithms, such as neighborhood-based and matrix factorization techniques. E-commerce platforms, social networks, and streaming services have widely adopted these systems. They are expected to continue to play a significant role in improving the accuracy and relevance of movie recommendations.**

Keywords—Collaborator, Cosine Similarity, Pearson Correlation, User to User

I. Introduction

In recent years, the increasing availability of data and advances in machine learning techniques have led to the development of highly accurate and effective movie recommendation systems. One popular approach to movie recommendation is collaborative filtering, where a user's preferences are inferred based on the preferences of other users with similar tastes. Collaborative-based movie recommendation systems leverage this approach to provide personalized recommendations to users based on their viewing history, ratings, and reviews. Machine learning algorithms play a critical role in implementing these recommendation systems, enabling the analysis of large datasets and the identification of hidden patterns in user behavior. In this report, we will explore the different techniques used in collaborator-based movie recommendation systems, including neighborhood-based and matrix factorization algorithms, and evaluate their performance in terms of accuracy and scalability. We will also discuss the challenges and limitations of these systems and potential avenues for future research.

One of the biggest advantages of using user-based Collaborative filtering is that the model doesn't want any information about the movie, like it's genre, plot, actors, and more. It relies completely on the rating of the user, which is often available in the online movie dataset.

II. Literature Survey

Several studies have explored using machine learning algorithms in collaborator-based movie recommendation systems. One of the most widely used approaches is the neighborhood-based algorithm, which identifies the most similar users or items based on their ratings and preferences. The k-nearest neighbors (k-NN) algorithm is a common implementation of this approach, which recommends movies based on the ratings of the k most similar users [1]. The performance of this algorithm can be improved by using more advanced techniques, such as weighted k-NN and clustering-based methods.

Another popular approach to collaborative filtering is matrix factorization, which decomposes the user-item rating matrix into two lower-dimensional matrices representing user and item features [2]. This approach has been shown to be highly effective in capturing latent features and making accurate recommendations. Singular value decomposition (SVD) and its variants, such as probabilistic matrix factorization (PMF) and non-negative matrix factorization (NMF), are commonly used algorithms in this category.

Recent studies have also explored deep learning techniques in collaborator-based recommendation systems, such as neural networks and autoencoders. These approaches have shown promising results in capturing complex patterns and interactions among users and items and handling sparse and noisy data.

III. Implementation

In the dataset, we have considered two files from the dataset [1]; in one file, the user's rating of the movies is mentioned, and in the other dataset, all the movie metadata is available. The data shows 670 unique users and around 40000 unique movie id and their titles, genre, actor, director, and many other details. In the dataset, there are around 1 lakh ratings for different movies and which are rated by 670 unique users.

The system uses collaborative filtering, which is a type of recommendation system that identifies similar users and recommends movies that these similar users have rated highly. Still, the current user has yet to see it. The system follows the following steps:

1. Import Python libraries for data processing, visualization, and similarity computation.
2. Download and read data using the Pandas library.
3. Conduct exploratory data analysis (EDA) to understand the data, including aggregating ratings by movie and visualizing the distribution of ratings.
4. Dividing the whole data into test and train datasets using train_test_split in the sklearn library
5. Create a user-movie matrix to represent users' ratings of movies.

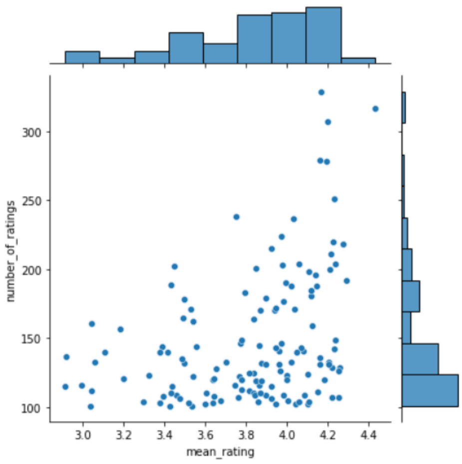


Figure 1: user-movie graph visualization

6. Normalize the user-movie(train dataset) matrix to center the ratings around zero and improve the accuracy of similarity measures.
7. Identify similar users based on their ratings of movies using either Pearson correlation and cosine similarity, and take the average of both.
8. Narrow down the pool of movies to recommend by selecting movies that similar users have rated higher but the current user has yet to see.
9. Recommend movies to the current user based on the average ratings of these movies among similar users.
10. After finding the average predicted rating, compare the actual and predicted values.
11. Using RMSE and MAE, find the difference between actual and predicted values.
12. Finding the precision of a recommendation system using precision and recall

We used the above steps to implement the program to recommend the user.

IV. Result

We try to recommend the movie for a particular mentioned user in a code; we recommended the other ten movies based on the similar user with similar ratings and predicted the user expected to rate for a specific movie.

	movie	movie_score	predicted_rating
659	The Elementary Particles	1.285829	4.813607
532	Rumble Fish	0.963688	4.491466
539	Saw	0.951462	4.479239
541	Saw IV	0.935241	4.463019
577	Star Trek II: The Wrath of Khan	0.878868	4.406646
177	Crank	0.877380	4.405158
710	The Matrix Revolutions	0.862554	4.390331
791	Three-Step Dance	0.831866	4.359644
53	Amélie	0.777724	4.305501
262	Frankenstein Created Woman	0.760744	4.288522

Figure 2: Predicted rating for a particular user for all movies

The data shows in Figure 2 that user one can watch “The Elementary Particles” as the movie score for that user for this movie is very high than other movies. So, it is more likely that user one will like the movie.

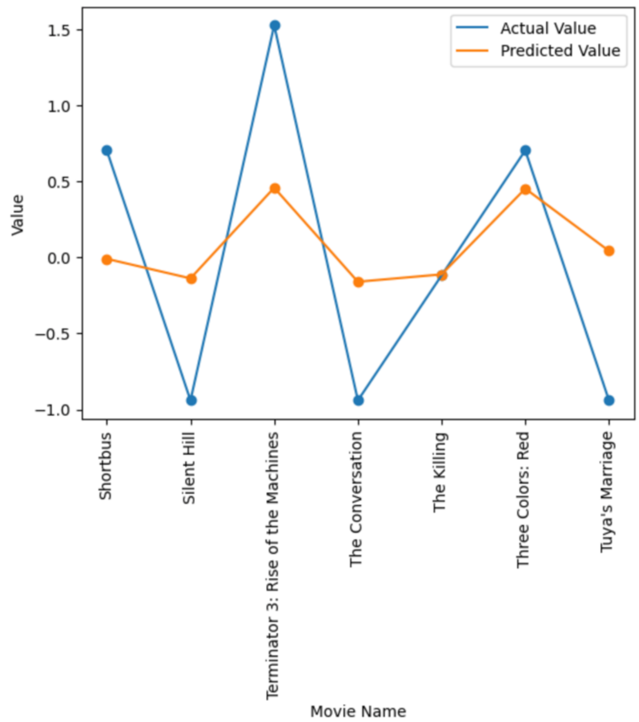
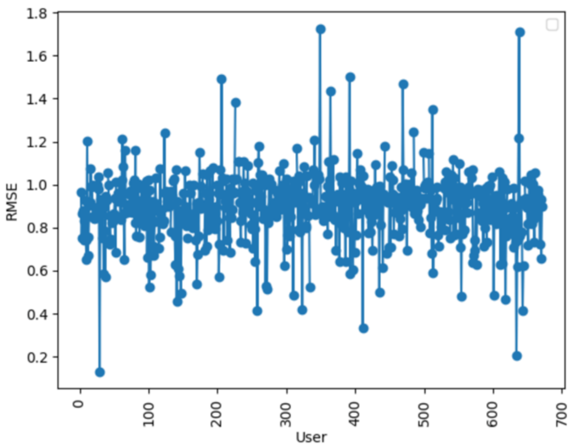


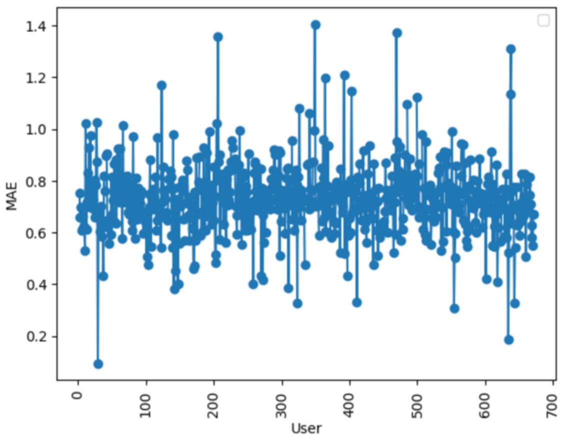
Figure 3: Graph of predicted and actual values for a specific user

In Figure 3, we can see that the value of the actual and predicted rating is not overfitting, and in underfitting, the model's prediction is getting nearly the same, with some errors.

So, after this, we need to check the model's performance using a particular movie's actual and predicted value movie rating prediction. We used RMSE and MAE to find the error between actual and predicted user values. This Graph of RMSE and MAE is given in Figure 4, with respect to a different user.



(a). RMSE value of each user



(b). MAE value of each user

Figure 4: graph of RMSE and MAE for different user

Now to conclude about the RMSE and MAE, we can use a histogram for how many users has the nearly same value of errors and what is the range of the common errors; in Figure 5, we can see it clearly:

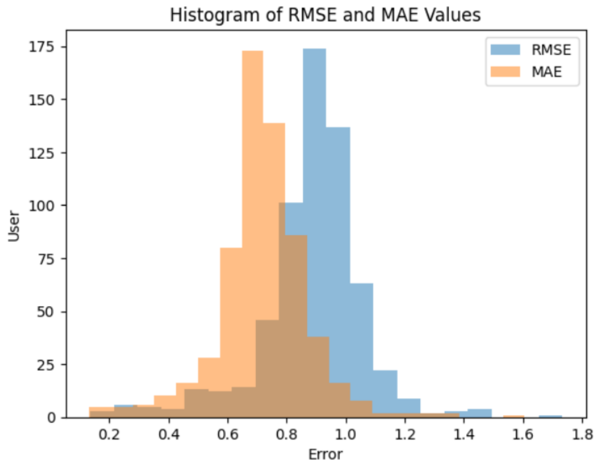


Figure 5: RMSE and MAE value distribution for finding average RMSE and MAE error

For, after using precision and recall, we can calculate the accuracy of a model, so after running the model multiple times, the accuracy of the model comes between 75% to 85%, so the average precision of a model comes to be 80%.

V. CONCLUSION

We have developed a movie recommendation system using a collaborative filtering technique. The plan recommends movies to the user based on the ratings given by other similar users. We have used the Pearson correlation and cosine similarity to calculate user similarity. We have also filtered the movies based on the number of ratings they received and kept the movies with over 100 ratings. Finally, we have used the item scores to recommend top movies to the user. So, we analyzed the model using RMSE, MAE, and precision and recall; we got an average of the model is 80%.

REFERENCES

Dataset Link:
[1]: https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset?select=movies_metadata.csv

[1] Wu, C. S. M., Garg, D., & Bhandary, U. (2018, November). Movie recommendation system using collaborative filtering. In 2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS) (pp. 11-15). IEEE.

[2] Lavanya, R., & Bharathi, B. (2021). Movie Recommendation System to Solve Data Sparsity Using Collaborative Filtering Approach. Transactions on Asian and Low-Resource Language Information Processing, 20(5), 1-14.

[3] Geetha, G., Safa, M., Fancy, C., & Saranya, D. (2018, April). A hybrid approach using collaborative filtering and content based filtering for the recommender system. In Journal of Physics: Conference Series (Vol. 1000, No. 1, p. 012101). IOP Publishing.

[4] Anwar, T., & Uma, V. (2021). Comparative study of recommender system approaches and movie recommendation using collaborative filtering. International Journal of System Assurance Engineering and Management, 12, 426-436.