**CSE523 - Machine Learning**

**Movie Recommendation System using Machine Learning**

**Faculty - Prof. Mehul Raval**

**Weekly Report 8**

**Submission Date: 08-04-23**

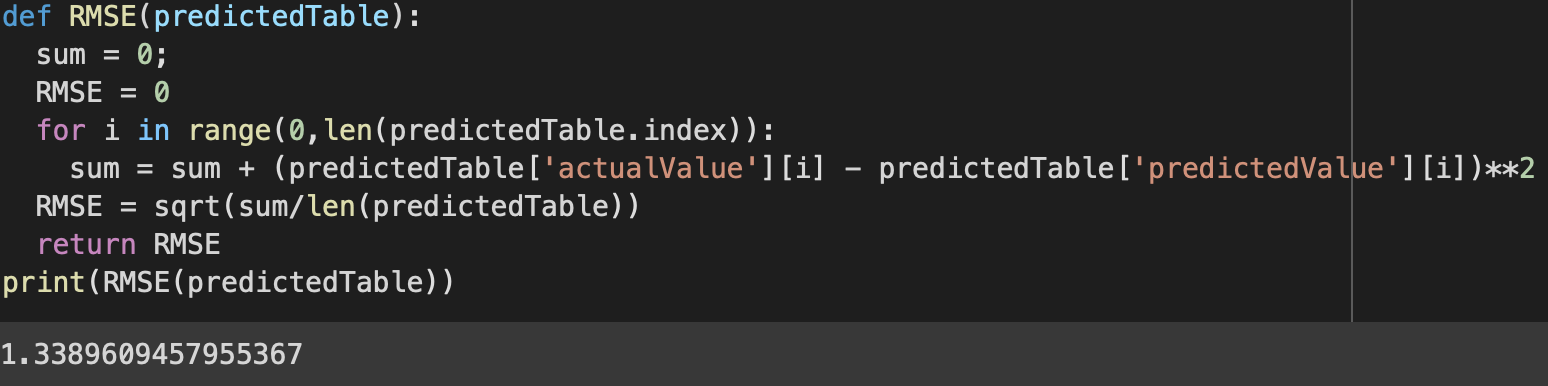
**Group: Tech Titans**

| **Name** | **Roll Number** | **Email** |
| --- | --- | --- |
| Jainam Shah | AU2040186 | jainam.s4@ahduni.edu.in |
| Yash Chotaliya | AU2040193 | yashkumar.c@ahduni.edu.in |
| Akshay Parmar | AU2040199 | akshay.p@ahduni.edu.in |
| Shubham Bhatt | AU2040206 | shubham.b1@ahduni.edu.in |

In the last week we had created a table to store the actual and predicted values of movie ratings for a specific user. Then the code loops through each movie in a data frame and checks if it exists in another. If the movie is present, a new row is added to the predictedTable with the movie name, actual rating value from the test dataset, and predicted rating value from the ranked\_item\_score data frame.

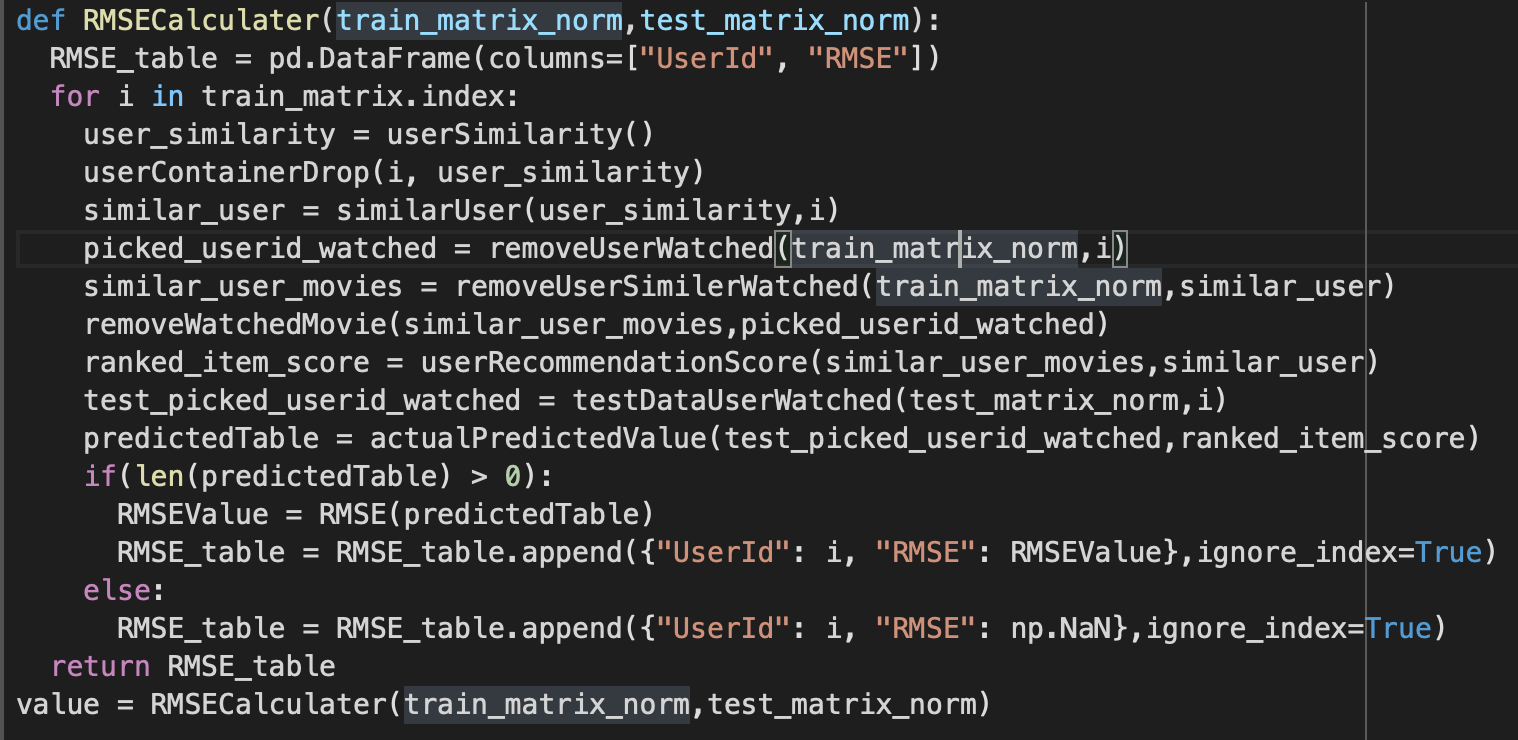
The predictedTable helps to evaluate the performance of the collaborative filtering model. By analyzing the differences between the actual and predicted values, we can visually understand how accurate the model is and identify areas where it may need improvement. The scatter plot and line chart were also created to visualize and compare each movie’s actual and predicted values. The scatter plot displays two sets of points, one for the actual values and one for the predicted values, while the line chart connects the points for each movie. By visually comparing the actual and predicted values, we can quickly identify trends or patterns in the data, such as movies with higher or lower prediction accuracy.

Then after that we had founded RMSE for a particular user. RMSE we had used because RMSE is a metric used to evaluate a predictive model's accuracy by measuring the difference between predicted and actual values. We get to see that it is calculated by taking the square root of the average of the squared differences between predicted and actual values. Also, as we had to found the RMSE we had made all the above parameters as functions and then after it we had founded RMSE.



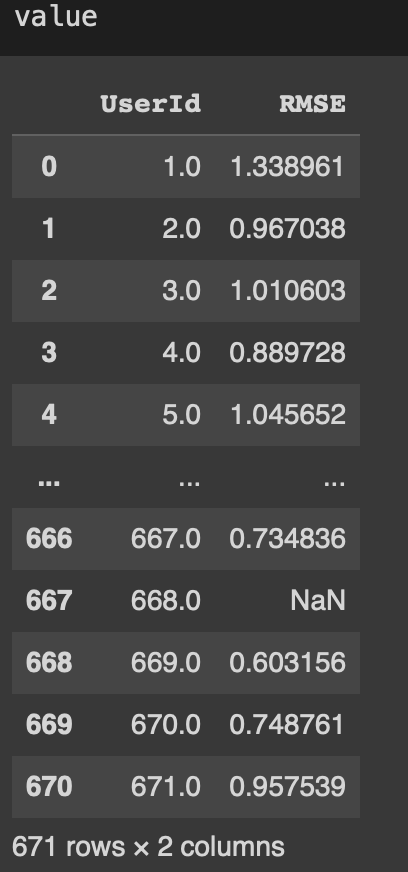
Here above we see that the RMSE function takes the predictedTable as input and initializes sum and RMSE variables to 0. It then iterates over each row of the predictedTable and calculates the squared difference between the actual and predicted values for each row. It accumulates the sum of squared differences in the sum variable. After iterating over all rows, it calculates the RMSE using the formula, RMSE = sqrt(sum/len(predictedTable)). Finally, we get the RMSE value.

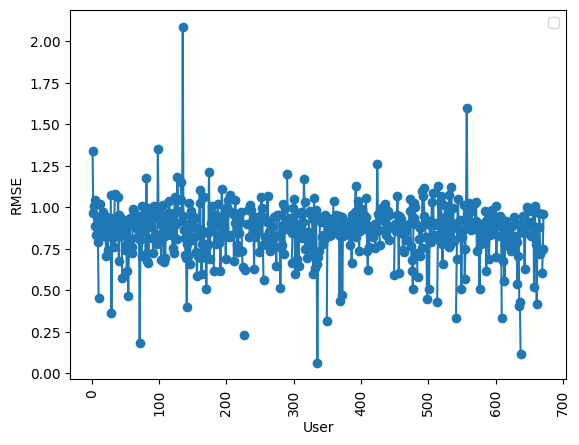
Then after we get a RMSE value for a user then we had RMSE\_table with two columns named UserId and RMSE. The UserId column will be used to store the user IDs for which the RMSE was calculated, and the RMSE column will store the RMSE values calculated for each user. The RMSE\_table we had made because in further process for finding RMSE for all the users it would be useful and it can be also used to compare the performance of different recommendation systems and choose the one that performs the best.



This above code we had written for finding RMSE for all the users . So , in short explaining that what is the code about. So, first of all we had made a function RMSECalculater. This function is used to calculate the Root Mean Squared Error (RMSE) between the actual and predicted values of the ratings given by users for different movies. The function RMSE\_table that we had made earlier then iterates through each user in the train\_matrix and performs the following steps: Calculates the similarity between the current user and all other users using the userSimilarity() function, then it drops the current user from the similarity matrix using the userContainerDrop() function, then it finds the most similar users to the current user using the similarUser() function, then it extracts the movies watched by the current user from the train\_matrix\_norm using the picked\_userid\_watched() function, then it extracts the movies watched by similar users from the train\_matrix\_norm DataFrame using the similarUserMovies() function, then it removes movies watched by the current user from the movies watched by similar users using the removeWatchedMovie() function, then it calculates the recommendation score for each movie using the userRecommendationScore() function, then it extracts the actual ratings for movies watched by the current user from the test\_matrix\_norm DataFrame using the testDataUserWatched() function, then it calculates the predicted ratings for movies watched by the current user using the actualPredictedValue() function and then calculates the RMSE between the actual and predicted ratings using the RMSE() function and at last stores the user ID and RMSE value in the RMSE\_table using the append() function.

The output that we see below is the RMSE\_table with all the RMSE values for all the users.





After finding RMSE for all the users we use the scatter() method to create the scatter plot, passing in the UserId and RMSE columns from the value data frame. We also add a label to the scatter plot using the label parameter. Next, we use the plot() method to create a line plot, passing in the same UserId and RMSE columns. We also add a label to the line plot using the label parameter. We then rotate the x-axis labels using the xticks() method with the rotation parameter set to 90. This helps prevent the labels from overlapping.

Here the above scatter plot graph represents the Root means square error (RMSE) of each user for the given rating by each user. Where the x axis represents user and the y axis represents root means square error (RMSE) of the each user. In this we can see that most off the users have root means square error (RMSE) around between 0.75 to 1. In this some of the user RMSE value is NaN which means that the users data are go to trainining set so for that there no data for testing.