**Topic: Netflix Movie Recommendation**

**Team 5**

**Report By: Jeel Christian**

**Introduction:**

Netflix is which offers online streaming of a library of films and television programs. It connects people with movie & television programs they love.

To help people to find out answer to question “What should I watch this evening? Netflix movie recommendation system has been developed. This system will help user to find out movie based on how much they liked or disliked other movies, in simple words will recommend movies based on user interest, history and reviews for other movie. This system is used for proving recommendation to each user with personal movie recommendation based on user unique likes and choices.

**Existing method and our approach**:

Existing recommendation models consider what movies were watched by the user or the movies the users searched for and movies related to them. This approach is not efficient as it doesn’t consider the possibility that the user might not the like the movie that they watched.

Out approach is the focus the data on the rating provided by the user for the movie. It considers the likeliness about the movie and similar movies based on the rating by the user to know whether the user liked the movie or not.

There are many approaches based on which recommendation system can be developed. Few of which we have tried to used while making the recommendation system. Based on the predictions we have moved forward with the best approach which is explained later in the report.

**Approach:**

There are lot of anomality in movie rating data provided by Netflix and the prediction accuracy is just bit more as compared to other recommendation on the same training set. For recommendation system Accuracy is a measurement of how closely predicted ratings of movies match subsequent actual ratings.

**Objective:**

1. Learn from data and recommend best TV shows to users, based on self & others behavior
2. Predict what rating the user will give to movie which user has yet not rated.
3. Increase the prediction accuracy which means to decrease the difference between the predicted user rating and actual user rating.

**Data Overview:**

**Dataset downloaded:**  [https://www.kaggle.com/netflix-inc/netflix-prizedata/data](https://www.kaggle.com/netflix-inc/netflix-prize-data/data)

**Data Files:**

* movie\_titles.csv
* combined\_data\_1.txt • combined\_data\_2.txt • combined\_data\_3.txt
* combined\_data\_4.txt
* probe.txt
* qualifying.txt

**Data File description:**

There is data for 17770 movie ranging for movie ID 1 to 17770, 480189 unique user data such as user ID, movie rating ranging from 1 to 5( 5 being the best) and date on which user rated the movie, where the date is in format YYYY-MM-DD.

Main Data:

Combined\_data\_1.txt, combined\_data\_2.txt, combined\_data\_3.txt and combined\_data\_4.txt contains all the **data**.

The first lines of these files consist of the movie id containing userID, Rating of the user and Date of the rating.

In the Qualifying data, the first line of each file indicates a **movie id, followed by a colon, and then customer ids and rating dates, one per line for that movie id.** For example, in below format:

MovieID1:

CustomerID11, Date11

CustomerID12, Date12

...

MovieID2:

CustomerID21, Date21

CustomerID22, Date22

The Recommendation will be in the format:

MovieID1:

Rating11

Rating12

..

MovieID2:

Rating21

Rating22

…

Probe.txt data in same format and is used for testing the recommendation system.

**Machine Learning Model:**

This recommender system where for given movie and user we need to predict the user rating is a recommendation problem (also known as Regression Problem)

We have implemented various models and analyzed the performance of various model based on below two performance metrics:

* Root Mean Square Error: <https://en.wikipedia.org/wiki/Root-mean-square_deviation>- Mean Absolute Percentage Error:

<https://en.wikipedia.org/wiki/Mean_absolute_percentage_error>

Below are two main objectives:

* Try to provide some interpretability.
* Minimize Root Mean Square Error (RMSE)

Following **steps** were implemented for development of **movie recommendation system**:

**1. Related to reading and storing data**

**Data Pre-Processing**:

This was the getting started to understand the problem dataset step. We played around with the data for a bit understanding it fields and how they impact each other and how to convert them in a format so that we don’t get much noise in our model.

Step 1 : Import the libraries

Step 2 : Import the data-set

Step 3 : Check out the missing values

Step 4 : Remove Duplicates

Step 4 : See the Categorical Values

Step 5 : Splitting the data-set into Training and Test Set

Step 6 : Feature Scaling

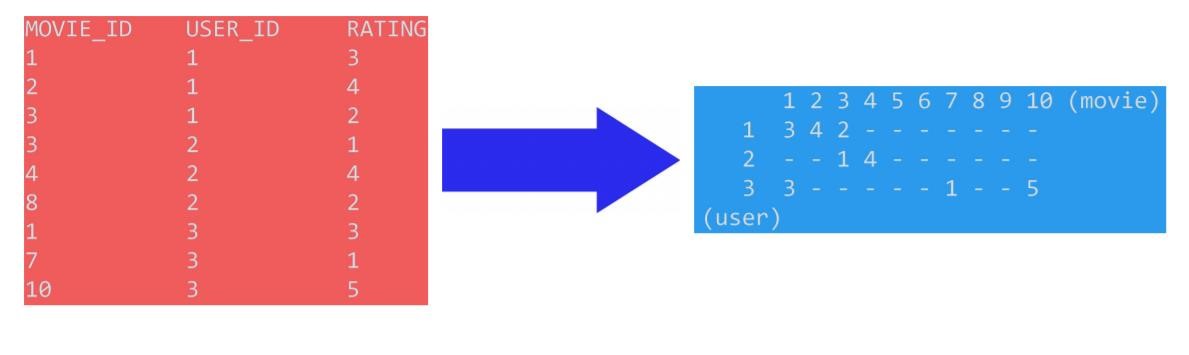
**2 . Exploratory Data Analysis on Train Data**:

Analyzing the data based on various parameters. We analyzed the model to understand the data and its features and performed various data analytics tasks.

* Distribution of Ratings in train data
* Number of Rating per Month (Millions)
* Analyzing Ratings given by User
* Number of ratings per Movie
* Day of Week VS Number Of Ratings

1. **Creating User-Item Spare matrix from data frame**

We had the data in a normal CSV format. The preprocessing step took care of the noise. However, the data is still not in a format that can be used to train a model. We converted researched on the solutions and came to the conclusion that sparse matrix can help us with it. We then created the User-Item Sparse matrix from the data frame.



1. **Computing Similarity Matrices**

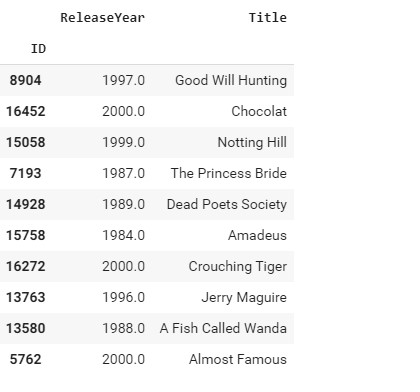
**Initial Approach:** Creating User-User Similarity Matrix.

We researched about mane recommendation models and one method that is highly experimental these days that we also studied In our Analyzing and Securing social media class was that- a FOAF model is more trustable that a random model i.e. considering user-user similarity and then recommending the movie based on their likings.

However, the downside of this was that the average time taken for computation of similarity matrix for one user was around 4 sec. This would result in nearly 10-20 days of computation time for the same.

**Final Approach:** Creating Movie-Movie Similarity Matrix.

This idea came to us from how amazon works. If I look for a watch. Then they automatically suggest me some more watches that I might like. We then tried this approach and looking at the outcome, we decided to move ahead with this one as it provided impressive results.



1. **Machine Learning Models**

Step 1: Creating sample Sparse Matrix for Train Data

Step 2: Creating sample Sparse Matrix for Test Data

Step 3: Finding Global Average of all movie ratings, Average Rating per User, and Average rating per Movie

Step 4: Featuring the data

Step 5: Transforming the data

**Evaluation Criteria:**

Error Metrics used in training the model:

* + - **RMSE**: Root Mean Square Error:

RMSE is the error of each point which is squared. Then mean is calculated. Finally root of that mean is taken as final value.

* + - **MAPE**: Mean Absolute Percentage Error:

The mean absolute percentage error (MAPE), also known as mean absolute percentage deviation (MAPD), is a measure of prediction accuracy of forecasting method.

We implemented different training models and compare them all to find out which one works best for the development of Netflix movie recommendation system for the provided data. We generate a user-friendly bar chart to give a proper understanding of the performance of every model.

* 1. **Feature Extraction:**

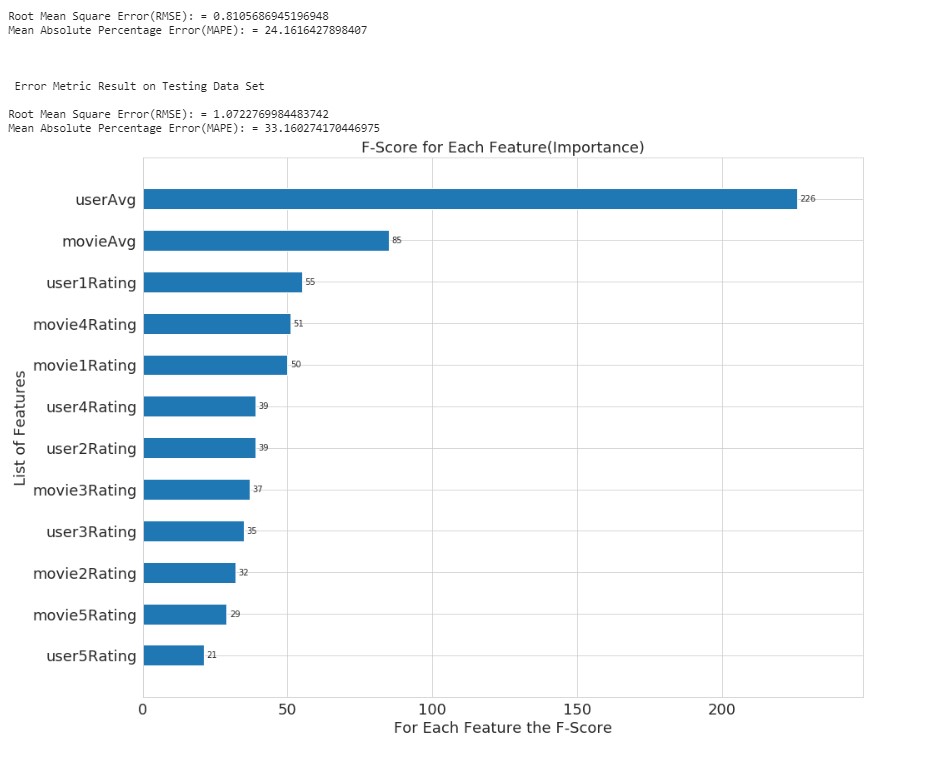
We considered the data those contributed the most to the user pattern and behavior. We concluded using the below mentioned features by the steps that we followed in the previous sections and recognized the best fitted features.

The first one is User ID and Movie ID. This is important as the output of out model is in this form and, they uniquely identify a user and the movie respectively. We also considered the ratings given to the movie by top 5 similar users to that of the current user, also, the ratings given by the particular user to top 5 similar movies with this movie. These were obtained from the similarity matrix that we computed in the previous steps. Amongst this, Total average rating, User average rating, Movie average rating and the specific rating are also the features that are considered and provide us the efficiency that we need.

* 1. **Implementation:**

Below are the list of different training models and their train and test RMSE and MAPE.

**1. XGBoost\_13**

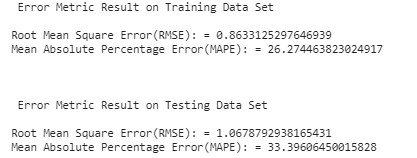


**Surp**

**rise Baseline Only**

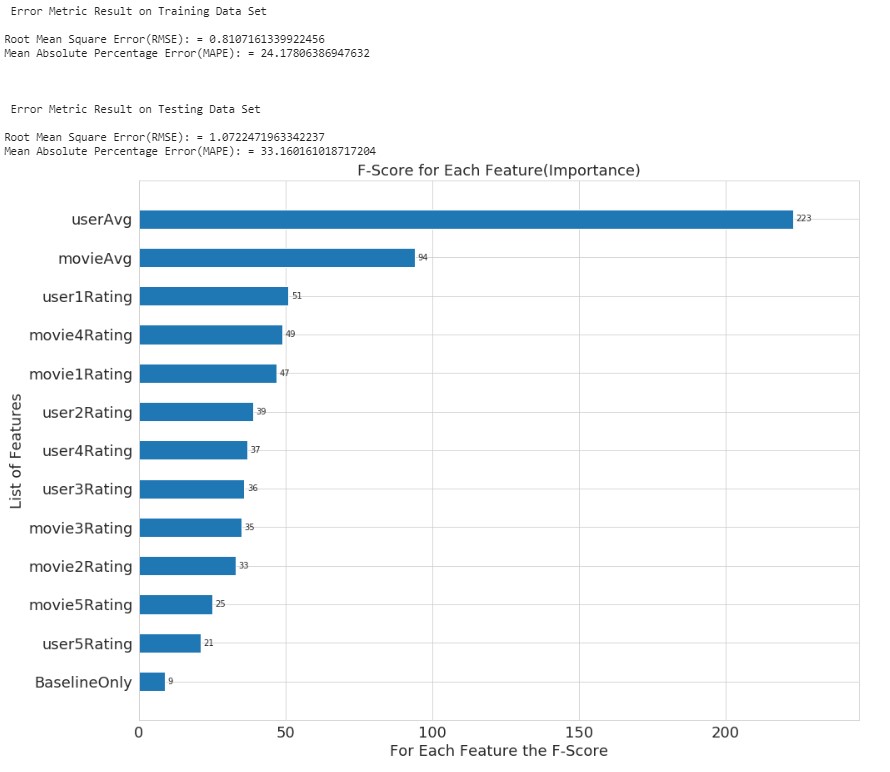
**Model**

Algorithm predicting the baseline estimate for given user and item.



**XGBoost \_13 & Surprise Baseline Only Model**

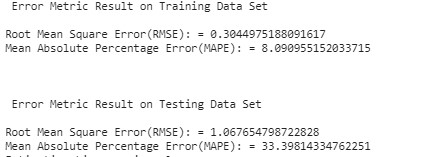
Adding predicted ratings from Surprise BaselineOnly model to train and test dataframe.



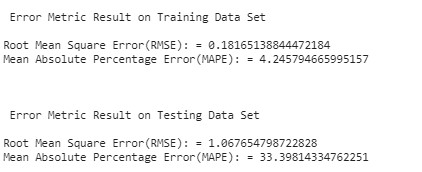
**Surprise KNN-Baseline with User-User and Item-Item Similarity**

These are algorithms that are directly derived from a basic nearest neighbors’ approach.

**Surprise KNN-Baseline with User-User**



**Surprise KNN-Baseline with Item-Item**

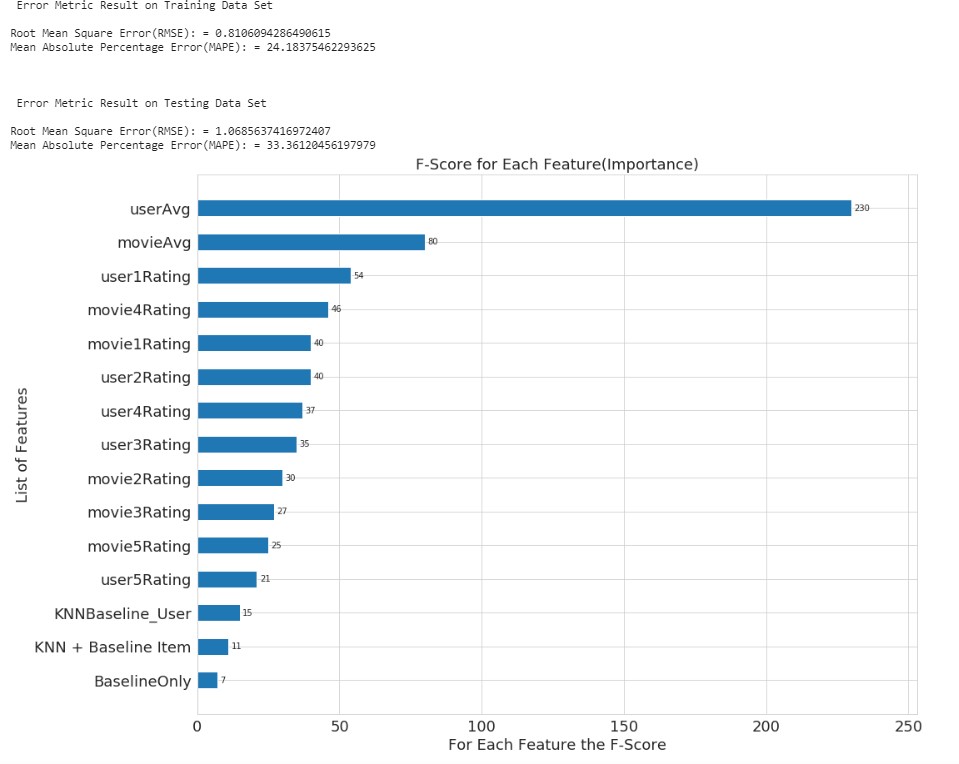


**XGBoost\_13 Features , Surprise Baseline Only and Surprise KNN**

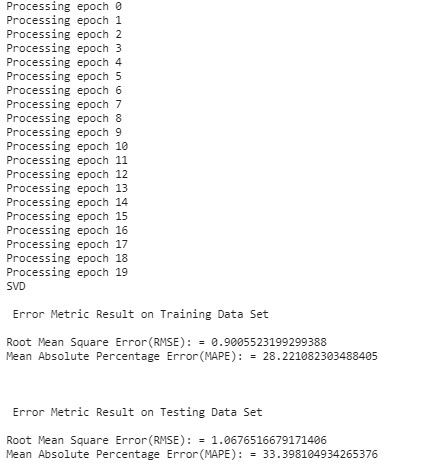
**Baseline**

Adding Predicted ratings from Surprise KNN Baseline model to our

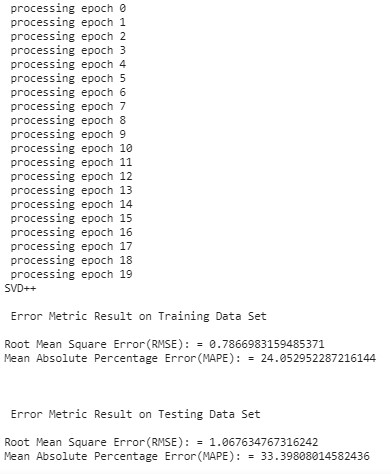
Train and Test Dataframe



**Matrix Factorization SVD**



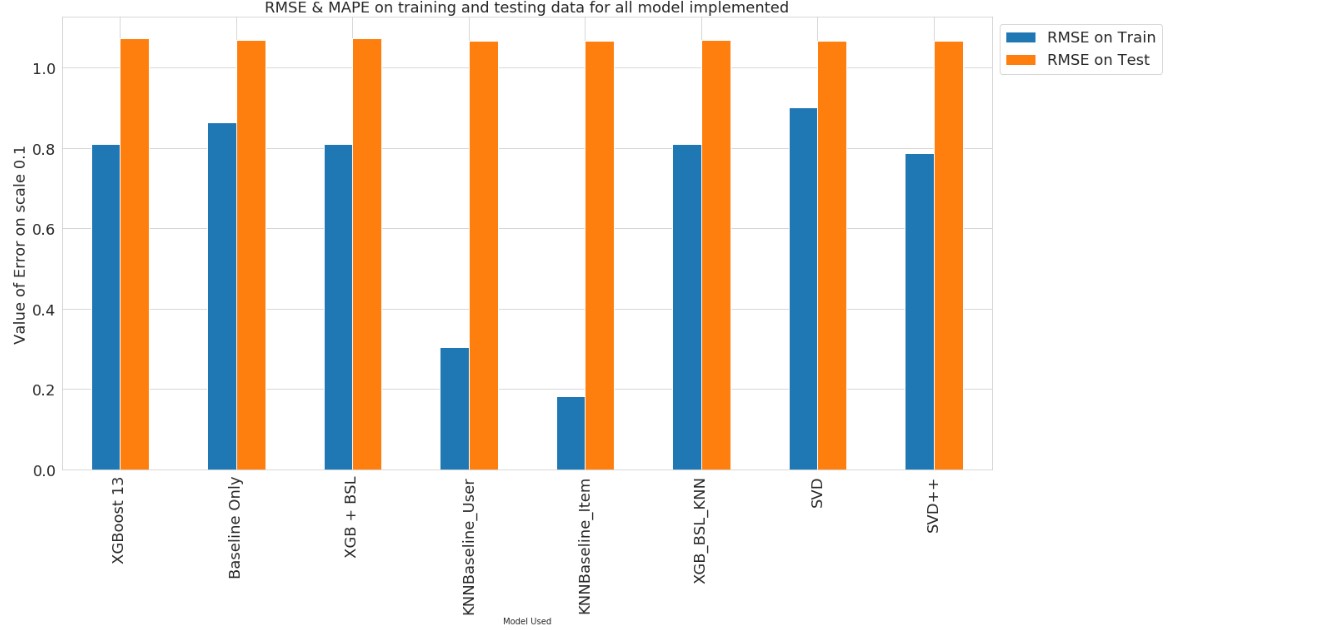
**Matrix Factorization SVD++ with Implicit feedback**



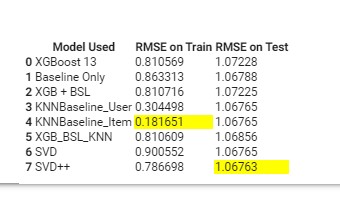
**Results:**

**Train and Test result on all the above models:**

In Bar Chart Form:



In Tabular Form:



**Conclusion**: Best fit for our Data is SVDpp having test RMSE 1.06763

**Work Distribution:**

We worked in a divide and conquer manner to reach the end point of this project. We realized that our goal was ambitious and if we wanted to complete it in time, we needed to follow a proper schedule. As the project progressed, we divided the tasks accordingly.

Initially, we were doing the data processing techniques and trying to figure out the important features. We divided the tasks of evaluating different features and handling different data preprocessing techniques.

Next, we divided the machine learning model to be implemented to follow as proper parallel working system. However, we made sure that no one gets stuck at any problem and keep rotating our tasks so that everyone gets to know about every aspect of different models.

It was divide and conquer when needed and united work when needed without wasting much time.

The most interesting part that I felt while working on this model was how we researched about different models and got to learn which one to use I what case. We tried various models as we all had different ideas. The end result of the project was not just the result that we got as the output but the actual result for me was that how we worked on the problem.

**Summary:**

We implemented some models that we were very excited about. Some of which were something that we did work on previously as well and some that we worked on for the first time and it was a very exciting experience. The best of which was SVDpp when evaluated based on RMSE.