# Final Project Report

# Distributed and scalable data engineering

Greeshma Bachala

# Netflix Movie Recommendation System:

## Motivation:

Now a days most of the people watch movies, TV shows and series anytime from anywhere. This is happening only because of digital environment. Other than the time the preferences that online TV brings to individuals, frequently decide for on the TV is the non-appearance of commercial breaks, the capacity to watch where they need, on which device they need, and the direct way to discover new content. Netflix is one of the parties that jumped into the world of online streaming services. Netflix is a company that handles a big collection of television programs and movies, by streaming it at any time via online. There is large number of users depending on online streaming services, by this the data also started growing in enormous amount. Netflix collects lots of data, which can be used in many ways. For instance, they can analyze data to increase the revenue, for marketing purpose, and to improve their customer satisfaction through feedback. Ratings is key for recommendations for movies to watch.

The data that is used to train the model will be preprocessed and analyzed. Therefore, the actual recommender systems will be trained. We will use customer ratings and predict the recommendations they would like to watch.

## Documentation of approach:

## Amazon EMR:

Amazon elastic MapReduce (EMR) is tool for bigdata processing and analysis. EMR is based on Apache Hadoop, a java-based programming framework that supports the processing the large datasets in distributed computing. It processes a Hadoop cluster of virtual servers on EC2 and S3. It is used for data analysis in log analysis, web indexing, data bioinformatics and more. It works on apache spark which integrates to hive and pig for more functions.

In this I created a cluster which runs on distributing compute power. It has 3 instance or workers in this. They are 1 master and 2 cores.

Cluster Auto scaling is disabled in this cluster as this project does not need auto scaling.

The version of spark is 2.4.4 on Hadoop 2.8.5 YARN with ganglia 3.7.2 and zeppelin 0.8.2.

**Configuration:** M5. X large with number of instances is 3.

As the spark the lazy evaluation it performs actions and transformations. For actions it shows up the spark job, which is interesting. For reading the data sources I used pyspark.

## Data source:

The dataset used is the Netflix prize, the dataset contains a total of 100,480,507 ratings, based on 17,700 movies which come from a total of 480,189 users from the United States. Of each movie, titles and corresponding year of release were available. Besides, every movie had a unique movie ID, which was a sequence from 1 to 17,700, There are ratings. Ratings are on a five-star scale from 1 to 5.

```
+-------+-------+------+----+--------------------+---------+|
|MovieID| UserID|Rating| YOR|          Movie_Desc|
+-------+-------+------+----+--------------------+----------+
|      8|1744889|   1.0|2004|What the #$*! Do ...|
|      8|1395430|   2.0|2004|What the #$*! Do ...|
|      8|1205593|   4.0|2004|What the #$*! Do ...|
|      8|1488844|   4.0|2004|What the #$*! Do ...|
|      8|1447354|   1.0|2004|What the #$*! Do ...|
|      8| 306466|   4.0|2004|What the #$*! Do ...|
|      8|1331154|   4.0|2004|What the #$*! Do ...|
|      8|1818178|   3.0|2004|What the #$*! Do ...|
|      8| 991725|   4.0|2004|What the #$*! Do ...|
|      8|1987434|   4.0|2004|What the #$*! Do ...|
|      8|1765381|   4.0|2004|What the #$*! Do ...|
|      8| 433803|   3.0|2004|What the #$*! Do ...|
|      8|1148143|   2.0|2004|What the #$*! Do ...|
|      8|1174811|   5.0|2004|What the #$*! Do ...|
|      8|1684516|   3.0|2004|What the #$*! Do ...|
|      8| 754781|   4.0|2004|What the #$*! Do ...|
|      8| 567025|   4.0|2004|What the #$*! Do ...|
|      8|1623132|   4.0|2004|What the #$*! Do ...|
|      8|1567095|   3.0|2004|What the #$*! Do ...|
|      8|1666394|   5.0|2004|What the #$*! Do ...|
+-------+-------+------+----+--------------------+
```

This is how the Data frame shows up after the preprocessing steps.

**Preprocessing steps:**

1. Importing the data from EMR cluster.

2. Installing all the packages

3. Importing the required libraries

4. Reading the data sources by using the required delimiter to use.

5. Renaming the column names in both training and testing set.

6. Filtering and joining operations are performed for extraction of specific data.

7. Using the sql functions for data understanding.

8. Groupby and aggregate functions are used to set up the data.

9. Joining the training and movies data frame for understanding the data.

### System configuration:

The cluster is created in EMR by opening the amazon EMR console. Creating a cluster by choosing create cluster and it takes you to a configuration page where we have general configuration, software, and hardware configuration and the security and access.

Here, the cluster name should be given and check the logging means copying our files automatically to S3. Below that there is a S3 folder where all our files get uploaded. The s3 file path is given.

Launch mode is always the cluster because as the spark is not for step execution, we want it to execute group at a time as cluster, so we choose cluster.

Software configuration and the version emr-5.29.0 and the applications are spark with 2.4.4 on Hadoop 2.8.5 YARN with ganglia 3.7.2 and zeppelin 0.8.2.

Hardware configuration with instance type as M5 x large and number of instances are 3 (1 master and 2 core nodes).

Security and access are permissions as default, EMR role as default role and instance profile as EMR_EC2_default role.

The system configuration is M5. X large and no of instances are 3, (1 master and 2 core nodes).

## Collaborative filtering:

Collaborative filtering is the most common technique used when it comes to building intelligent recommender systems that can learn to give better recommendations as more information about users is collected.

Most of the websites like Amazon, YouTube, and Netflix use collaborative filtering as a part of their recommendation systems. we can use this technique to build the recommenders that give suggestions to a user on the basis o the likes and dislikes of similar users.

we will need the data with set of users and set of items, the rating is on a scale of 1 to 5, likes or dislikes.

In our case of movie recommendation system set of items are the list of movies and the data is the list of ratings given by users.

**Steps in collaborative filtering:**

- The users who are similar to one another k
- The rating that a user would give to a movie based on ratings of similar users
- Measuring the accuracy of the ratings.

## Problem 1:

### Collaborative filtering approach:

In this approach the we will be finding the similar users k. By performing the spark functions, I have chosen the top 20 users from user_DF who rated for most of the movies whose rating is more than 3. The same applies to the movies_DF to select the top movie whose rating are more than 3 and having rated by most of the users. This way to find the similarity of the users with the movies they rate.

```
+-------+-------------+------------------+
| UserID|count (MovieID)|      avg (Rating)|
+-------+-------------+------------------+
|1664010|         1535|4.2384364820846905|
|2118461|         1481| 4.088453747467927|
|1114324|          876|3.0079908675799087|
|1473980|          680|3.0808823529411766|
| 716173|          675|4.3822222222222225|
|1663888|          669| 3.478325859491779|
| 303948|          621|3.8003220611916264|
|1710658|          591|3.0761421319796955|
|1061195|          572|3.1223776223776225|
|2238060|          570| 4.101754385964912|
|1299887|          565|3.5805309734513275|
| 322009|          562| 3.311387900355872|
|1037245|          542|3.4649446494464944|
|2237185|          524|3.2938931297709924|
| 682963|          520|              3.25|
| 794999|          514| 4.996108949416342|
|1784150|          502| 3.254980079681275|
+-------+-------------+------------------+
```

Filtering for selecting the Users who have rated at least 500 movies and avg rating given is more than 3

Extracted list of Top UserID

```
+-------+-------------+------------------+
| UserID|count (MovieID)|      avg (Rating)|
+-------+-------------+------------------+
|1664010|         1535|4.2384364820846905|
|2118461|         1481| 4.088453747467927|
|1114324|          876|3.0079908675799087|
|1473980|          680|3.0808823529411766|
| 716173|          675|4.3822222222222225|
|1663888|          669| 3.478325859491779|
| 303948|          621|3.8003220611916264|
|1710658|          591|3.0761421319796955|
|1061195|          572|3.1223776223776225|
|2238060|          570| 4.101754385964912|
|1299887|          565|3.5805309734513275|
| 322009|          562| 3.311387900355872|
|1037245|        542|3.4649446494464944|
|2237185|        524|3.2938931297709924|
| 682963|        520|              3.25|
| 794999|        514| 4.996108949416342|
|1784150|        502| 3.254980079681275|
+-------+-------------+------------------+
```

Movie rating with the movie description on an average rating based on UserID.

```
+-------+-------------+------------------+--------------------+
|MovieID|count (UserID)|      avg (Rating)|          Movie_Desc|
+-------+-------------+------------------+--------------------+
|   6971|        25468| 4.071815611748076|Ferris Bueller's ...|
|   6287|        24393|3.7261099495756977|        Pretty Woman|
|   4640|        23525| 4.047438894792774|            Rain Man|
|   9728|        23184|3.8246204278812974| As Good as It Gets|
|   8596|        23005| 4.103368832862421|               Seven|
|   4432|        22565|3.6721471305118545|     The Italian Job|
|  10947|        21209| 4.339148474704135|     The Incredibles|
|   6408|        21198|3.8013020096235492|        Good Morning|
|   1202|        20997| 3.757298661713578|National Lampoon'...|
|  13651|        20902|3.5951105157401204|       Air Force One|
|  12293|        20691| 4.464598134454594|       The Godfather|
|   8915|        20646|3.9663373050469826|Terminator 2: Ext... |
|   2660|        20153|3.9168362030466928|When Harry Met Sally|
|   1744|        20140| 3.802730883813307|    Beverly Hills Cop|
+-------+-------------+------------------+--------------------+
```

## Problem 2:

## Analyzing the Netflix Data:

In this problem you will analyze the input data to implementation of the approach. Using spark in jupyter notebook on amazon EMR, to predict the ratings for all the user-item pairs in the test set. To show the distinct users and distinct items in the test set.

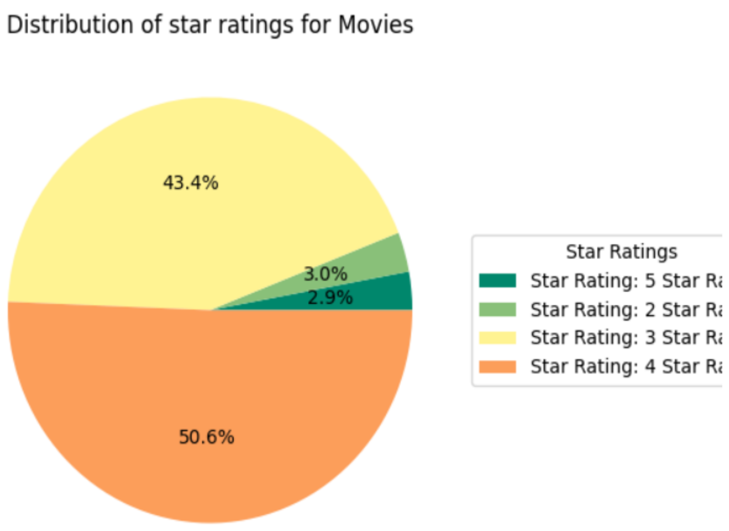
The collaborative filtering approach finding the many similar users by overlapping the movies. the prediction rating is better than the actuals.

Distribution of star rating for movies. Scaling from 1 to 5.

```
# Creating New Column for Star Rating

Rating_Composition = Rating_Composition.withColumn("Star
Rating",F.when(F.col("Rating(Avg)")>4,F.lit("5 Star
Rating")).otherwise(F.when((F.col("Rating(Avg)")<=4) & (F.col("Rating(Avg)")>3),F.lit("4 Star
Rating")).otherwise(F.when((F.col("Rating(Avg)")<=3) & (F.col("Rating(Avg)")>2),F.lit("3 Star
Rating")).otherwise(F.when((F.col("Rating(Avg)")<=2) & (F.col("Rating(Avg)")>1),F.lit("2 Star
Rating")).otherwise(F.lit("1 star Rating"))))))
```

Star rating for 5 rating is only 2.9% , for 4 rating is 3.0% close by, 3 star ratng is 43.4% and 2 star rating is 50.6%.

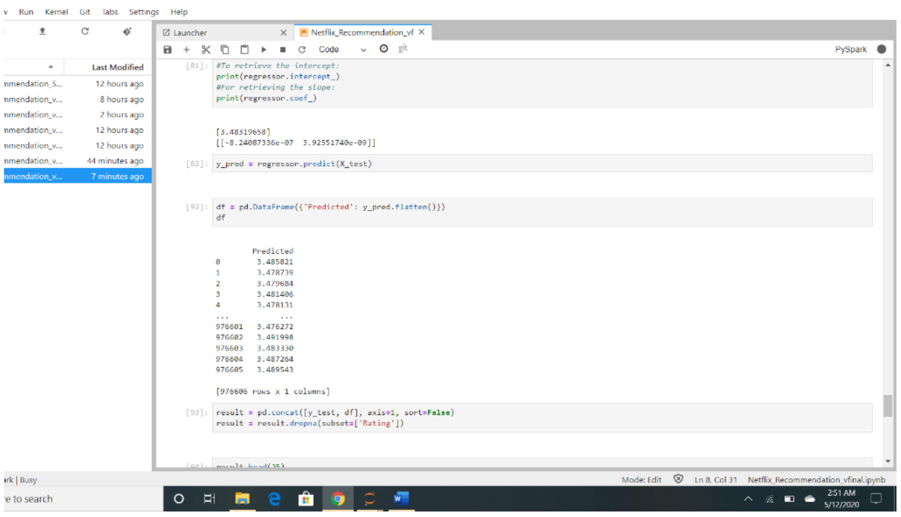
```
%matplot plt
```



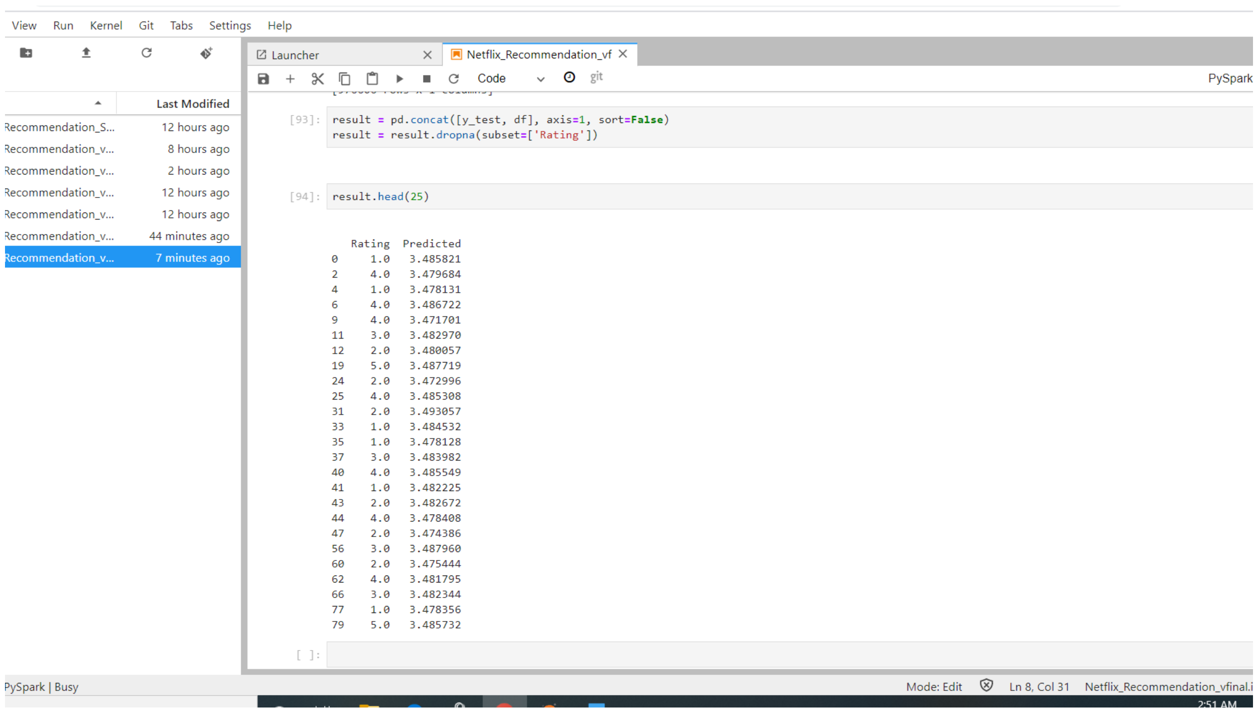
## Problem 3:

### Collaborative filtering implementation:

In this implementation I have predicted the ratings pairs in test ratings data frame. I chosen a regression models to predict my ratings. The predictions are better than the actuals. I tried using surprise to evaluate my model but did not work well. So, I tried using machine learning technique regression model.



```
[81]: #To retrieve the intercept:
      print(regressor.intercept_)
      #For retrieving the slope:
      print(regressor.coef_)

[3.48319658]
[[-8.24087336e-07  3.92551740e-09]]

[82]: y_pred = regressor.predict(X_test)

[92]: df = pd.DataFrame({'Predicted': y_pred.flatten()})
      df

        Predicted
0        3.485821
1        3.478739
2        3.479684
3        3.481406
4        3.478131
...           ...
976601   3.476272
976602   3.491998
976603   3.483330
976604   3.487264
976605   3.489543

[976606 rows x 1 columns]

[93]: result = pd.concat([y_test, df], axis=1, sort=False)
      result = result.dropna(subset=['Rating'])

[94]: result.head(25)
```


View Run Kernel Git Tabs Settings Help
Last Modified
Recommendation_S... 12 hours ago
Recommendation_v... 8 hours ago
Recommendation_v... 2 hours ago
Recommendation_v... 12 hours ago
Recommendation_v... 12 hours ago
Recommendation_v... 44 minutes ago
Recommendation_v... 7 minutes ago
Launcher
Netflix_Recommendation_vf
Code
git
PySpark
PySpark | Busy
Mode: Edit
Ln 8, Col 31
Netflix_Recommendation_vfinal.i

```
[93]: result = pd.concat([y_test, df], axis=1, sort=False)
      result = result.dropna(subset=['Rating'])

[94]: result.head(25)
```

```
    Rating  Predicted
0      1.0   3.485821
2      4.0   3.479684
4      1.0   3.478131
6      4.0   3.486722
9      4.0   3.471701
11     3.0   3.482970
12     2.0   3.480057
19     5.0   3.487719
24     2.0   3.472996
25     4.0   3.485308
31     2.0   3.493057
33     1.0   3.484532
35     1.0   3.478128
37     3.0   3.483982
40     4.0   3.485549
41     1.0   3.482225
43     2.0   3.482672
44     4.0   3.478408
47     2.0   3.474386
56     3.0   3.487960
60     2.0   3.475444
62     4.0   3.481795
66     3.0   3.482344
77     1.0   3.478356
79     5.0   3.485732
```

## Conclusion:

Netflix movie recommendation system project is very vast in details and prediction is not just the ratings, but it also includes the movies, and users. The surprise model helps us to evaluate the performance of the algorithm. The RMSE and MAE are used to evaluate and compute the model. There was a problem during evaluate function and I tried using cross validate but no use, then I went to machine learning technique to predict the ratings. SVD should have performed better than the Item-based approach, because the Low-dimensional recommenders are trying to capture the preferences of the users, and it is known that if we want to provide recommendations based on people's preferences then SVD is a good approach.