PREDICTION OF CUSTOMER RATINGS ON NETFLIX DATA

Purdue School of Computer Science,
IUPUI

Instructor: Yuni Xia

Sai Pavan Choudary
Sai Pramod Yerubandi
Sri Navya Paruchuri

Table of Contents

ABSTRACT

Over the past decade, the usage of Internet has increased tremendously and almost every individual can easily access it from any part of the world. This gave rise to the development of many technologies in the field of cyberspace thus introducing people to the virtual world. In today's fast-paced world numerous online activities are taking place and the one that grabbed a lot of attention is watching movies online. People prefer online movies over going to theatres since it helps save time and watch movies from their own comfort zone. There are many companies like **NETFLIX, STREAMLORD, HOTSTAR** that serve customers. Customer ratings are vital for such companies and hence they are providing platforms for their customers to share their views about the movies watched. Depending on the user's activity data is collected to scrutinize customer behavior. Like this concept, we developed a prediction algorithm in this project by collecting data from NETFLIX website. Using this prediction algorithm, we generate the ratings for the customers and recommend some movies to the customers.

INTRODUCTION

Watching Online movies has become an essential part of everyone's lifestyle. As the people don't find time to go the theatres or multiplexes to watch the movies many of them now prefer watching their desired movie online in their home with their family. As this trend is rapidly increasing many of the small movies and also TV-Series are now casting their shows directly online in spite of releasing them in theatres. These websites also recommend the movies to the customers taking their interests and ratings they have given to the previous movies into consideration.

Star rating which is a quantitative review enables the users to choose their product rating between 1 to 5 where 5 stars are the highest rating and 1 star being the lowest rating. Hence, relying on star ratings help the consumers and even going through all the descriptive reviews helps the consumers to watch the movie or not.

The Ranking algorithm implemented in this project acts as a perfect scenario to rate the movies using the concept of prediction. In this algorithm, based on the ratings of train data given by the consumers a movie rank is computed.

MOTIVATION

An important aspect of all the online entertainment providing companies is customer reviews for the movies and recommend movies to a customer based on his desire i.e. every individual has their own taste for movies, some like action, animation etc. and may not like sci-fi, horror type of movies, this lead us to implement existing or a new algorithm that recommends movies of any genre with better ratings to the user. This platform provides a better way of communication for both the consumers and the companies providing online entertainment.

From the average star ratings given by the customers for a particular movie, data analytics tools were used to evaluate the reviews, however, such tools generate ranks or ratings depending on the total number of views for the movie. Techniques such as Most Popular Movie, Best Movie function depending only on the movie details and not on customer's response.

Taking these flaws into consideration, we implemented few algorithms that can predict movie ratings. The generated movie rating signifies the customer's satisfaction towards that movie which can be useful to envision the variation in users view and then recommend the movie to the user.

RELATED WORK

Many types of research are being carried out to recommend the movies to a user based on the user ratings for the benefits of users and online entertainment providers. One part of the research is to predict the movie and the other part is to recommend the movies to the users based on users desire and also based on movie ratings. According to "The Netflix Algorithm" [1], the algorithm used by the Netflix is Cinematch. In the Cinematch algorithm, movies are matched with the users rather than matching the users to the movies.

In another research paper named "Movie Rating Prediction Using Singular Value Decomposition" [2], application of SVD for collaborative filtering was explored. Incremental SVD method was employed on Netflix dataset for predicting the movie ratings based on previous user's preferences.

An approach of faster and approximative version of ALS [3] was proposed by Pilaszy et al. The speedup is obtained by replacing exact least squares by coordinate descent method. The author observed a marginal loss of accuracy when compared with the significant decrease in the training time.

DATA SOURCES

For this project, Netflix dataset [4] is used. It has been downloaded from Kaggle. The dataset consists of training dataset and qualifying dataset which is the test dataset. The training data set consists of 100,480,507 ratings that 480,189 users gave to 17,770 movies. Each row in the training dataset is a quadruplet of the form <user, movie, date of grade, grade>. The user and movie fields are integer IDs, while grades are from 1 to 5 (integral) stars.

The qualifying data set contains over 2,817,131 triplets of the form <user, movie, date of grade>. All the ratings are predicted for this dataset.

Data Pre-Processing

As the dataset consists of millions of product details data pre-processing stage became challenging. After going through the collected datasets and understanding each metrics there are few discrepancies. The major challenge handled in this preprocessing stage was loading the data into the cluster as the data was of huge size the cluster was going out of space frequently.

Therefore, we have divided the original file into 8 different files and started loading them into the cluster. All the preprocessing is done using Amazon AWS with the help of m4xlarge Spark cluster. We need to start 10 m4xlarge clusters to the minimum as the data was so huge and so were the computations.

Data pre-processing is performed in Amazon AWS – EMR services platform as the dataset is large. A cluster with ten m4. xlarge nodes is created and the datasets are loaded using PYSPARK. The computations and normalization of the data are done using SPARK-SQL and PYSPARK. 6 Simple steps were followed in the pre-processing stage and are mentioned below. But before bumping into that we have faced some memory issues while performing computations over the data, so we have manipulated the data in the following manner.

- Remove movie with too fewer reviews (they are relatively not popular).
- Remove customer who gives too fewer reviews (they are relatively less active).

Having done all the above modifications, the cluster will have a significant improvement in efficiency, since all the unpopular movies and non-active customers will be removed. This should help improve the statistical significance too.

Step 1: Load all the 8 CSV files into the RDD using pySpark.

Step 2: Then the RDD is divided again into 2 parts as we were unable to perform all the on the single RDD so the RDD was divided and the RDD is normalized and then split with respect to the delimiter.

Step 3: After the splitting, the data is passed to a function where it further normalizes the data and creates a data frame from the resultant array or list.

Step 4: After the data frame is created the data frame is appended into to the list and the list is returned to the main function.

Step 5: The above steps are repeated for all the remaining files and we obtain 8 data frames i.e. a data frame from each of the file.

Step 6: After getting the data frames we merge all the data frames into a single data frame and output the file as CSV file.

METHODS AND ALGORITHM

For the Project, we have used 3 different Algorithms to perform the prediction of the ratings and 2 of the Algorithms are used to provide recommendations for the users along with the rating prediction. Along with these 2 algorithms we have designed a new algorithm which is used for the prediction of ratings of the movies in the test dataset. This algorithm gives us a range score or range of values for prediction of ratings for each user. The prediction thus obtained is compared with the actual ratings given in the meta data or the original data to determine the correctness factor of the algorithm.

The Algorithms used were:

- SVD

- ALS
- MEAN-STANDARD DEVIATION APPROACH

All the algorithms are implemented in Python using PYCHARM IDE. Various packages like numpy, pandas, scipy etc are used for the implementation. Filtering and recommending based on information given by other users is known as **collaborative filtering**. This assumes that if there are people existing with similarities i,e same taste for the movies then they are likely to give same ratings to the movies. We have specifically chosen collaborative filtering because it provides filtering information or patterns using techniques involving collaboration among multiple agents and is quite useful for our data.

SVD

Singular Value Decomposition (SVD) to recommend other movies based on user ratings. The main aim of SVD is that it breaks a matrix of any shape into a product of 3 matrices such that it satisfies the following formula. This algorithm is mainly used for low-rank approximations.

$$A = USV^T$$

Here U, S and V are matrices having dimensions m×m, m×n, and n×n respectively. Matrix S is a diagonal matric=x having only r nonzero values, which makes the dimensions as m×r, r×r, r×n. U and V are orthogonal matrices and S is a diagonal matrix. The matrices U and V are also normalized and reduced to produce matrices Uk and Vk where these matrices are produced by removing (r-k) columns from matrix U and by removing (r-k) rows from matrix V. When these matrices are multiplied we get a matrix AK. This reconstructed matrix is the closest approximation to the original matrix A where k being the rank in matrix Ak.

The dimensionality reduction approach in SVD is very useful for the collaborative filtering processes produces a set of uncorrelated eigenvectors. Each customer and product is represented by its corresponding eigenvector. The process of dimensionality reduction helps customers who rated similar products (but not the same products) to be mapped into the space spanned by the same eigenvectors.

Prediction using SVD

Once the m×n ratings matrix is decomposed and reduced into three SVD component matrices with k features Uk, Sk, and Vk, prediction can be generated from it by computing the cosine similarities between m pseudo-customers and n pseudo-products. The prediction scores are calculated by adding the row average to the similarity. After the decomposition is done the prediction process involves only a dot product.

Algorithm Workflow

After loading the datasets by using the required packages the pivot matrix is constructed by making movieid as the column and the user id as the index. The resultant pivot table is converted into a matrix and the mean of the ratings for each column is calculated. The generated matrix is modified by reshaping and then the matrix is given as an input for the svds method which has been implemented from scipy.sparse.linalg package. Which gives us 3 reduced matrices as mentioned in the algorithm. Now, we have all the prerequisites to make predictions for each user. This can be done by performing simple matrix multiplication with respect to the value of k.

We must optimize the number of latent features this can be done by decreasing the RMSE values. This can be done in a continuous manner by merely increasing the value of k. We have taken the k value as 50 because we have seen that when k values lie between 20 and 100 then the unseen data is generalized better when compared to the k values less than 20.

Finally, the movies are recommended by using the prediction matrix for every user. This is done by merely returning the movies with the highest predicted rating that the specified user has not already rated. As we do not have any content features like genre or title we just merged the information and print the list of movies.

Limitations

- We have some limitations as the major challenge is dimensionality reduction.
- It works most effectively with small data.

ALS

Alternating Least Square Algorithm. Let us consider we have u users and i items in the dataset. Then,

$$\boldsymbol{Q_{ui}} = \begin{cases} \boldsymbol{r\ if\ user\ u\ rate\ item\ i} \\ \boldsymbol{0\ if\ user\ u\ did\ not\ rate\ item\ i} \end{cases}$$

Here, r is referred or related as rating values. Suppose, we have m users and n items then we have to declare a matrix of factors which represent the movies. The matrix of factor indicates factor vector for each movie and that would be how we represent the movie and we don't have any data about the type of the movie present. We also have to declare a factor vector for each user in a similar way as we have done for the movies. Factor matrix for movies is given by $\mathbf{Y} \in \boldsymbol{R^{fxn}}$ and factor matrix (each movie is a column vector) for users $\mathbf{X} \in \boldsymbol{R^{mxf}}$.

From the factorization, we can clearly see that there are two unknown variables present. Therefore, we will be adopting an alternating least square method with regularization. By doing so we first estimate **Y** using **X** and estimate **X** using **Y**. After enough or a sufficient number of iterations, we are aiming to have a final point where both of the estimates converge and either the matrices both X and Y are no longer change or the change that occurred is quite small.

Here, In the dataset, we depend mainly on the movies that have a rating from the users. Let's, declare the weight matrix in such a way that

$$\boldsymbol{w_{ui}} = \begin{cases} \boldsymbol{0\ if\ \ q_{ui}\ = 0} \\ \boldsymbol{1 \qquad else} \end{cases}$$

The cost function minimized is,

$$J(x_u) = (q_u - x_u Y) W_u (q_u - x_u Y)^T + \lambda x_u x_u^T$$
$$J(y_i) = (q_i - X y_i) W_i (q_i - X y_i)^T + \lambda y_i y_i^T$$

To prevent overfitting of the data we perform regularization. Ideally, the regularization parameters are tuned using cross-validation in the dataset. Factor vectors are defined as,

$$x_u = (Y W_u Y^T + \lambda I)^{-1} Y W_u q_u$$
$$y_i = (X^T W i X + \lambda I)^{-1} X^T W_i q_i$$

Where $\boldsymbol{W_u} \in \boldsymbol{R^{nxn}}$ and $\boldsymbol{W_u} \in \boldsymbol{R^{mxm}}$ are the diagonal matrices.

Algorithm Workflow

After loading the datasets by using the required packages the pivot matrix is constructed by making movieid as the column and the user id as the index. The resultant pivot table is converted into a matrix and the values of the matrix are stored as a list. Later the estimations X and Y are calculated by using linalg package and these values are determined by iteration for a specific number of iterations.

The higher the number of iterations the more the X and Y values converge and less the RMSE value. These X and Y values are calculated just as mentioned in the above formula by performing the dot products and multiplying it with constant lambda.

We create a recommendation function for printing the recommendations to the users along with the movies they don’t like, and they might like and also the list of the movies they have liked.

MEAN-STANDARD DEVIATION APPROACH

This Algorithm was designed to predict the values using the statistical approach. The basic analogy for this has been taken from using mean and standard deviation in predicting the values for the time series values. In this approach, we first analyze the data and compute the quantiles to know how much the data lies in each quantile.

Here we create a matrix from the data by converting the dataset for the data frame to a pivot table and hence we have all the data combined with the required values needed for computation. We calculate mean for each column.

$$\boldsymbol{\mu} = \sum_{i=1}^{N} x_i / N$$

Later we calculate the standard deviation for each of the row to know how much deviation is from the mean. So, the standard deviation is calculated like

$$\boldsymbol{\sigma} = \sqrt{\sum_{i=1}^{N}(x_i - \mu) / N}$$

Algorithm Workflow

Import the required packages and the division package so that it supports the float values else the python takes all the values as 0.

Then the data is loaded as a data frame and store all the movie id and the user id in the lists and make them unique. After loading all the data, the original data is converted as a pivot table making movie id as columns and user id as rows and filling all the NA values as "0". Later the pivot table is converted to a matrix and its transpose is also stored as another matrix.

Now on applying the Algorithm the mean of each column is calculated by using the transpose matrix as all the columns will be rows which means the rows represent the movie id and the mean of each movie is computed and is pushed into a list.

Secondly, the standard deviation for each user is calculated to see how much deviation there original rating from the mean rating for that movie is. Then all the deviations for a user are taken for all the movies that he has rated then the mean deviation is computed for each user. As we have the mean deviation and mean rating for each movie. This algorithm is tested on the qualifying data provided by NETFLIX. This data mainly contains all the movies whose ratings for a user are not given in the original data. Example consider a movie id 20 and user id 3 this user rating is not present in the original dataset, but the movie has a mean rating and also the user has mean deviation. So, the prediction is done by subtracting the mean deviation from the mean value of the movie and similarly by adding the mean deviation to the mean of the movie.

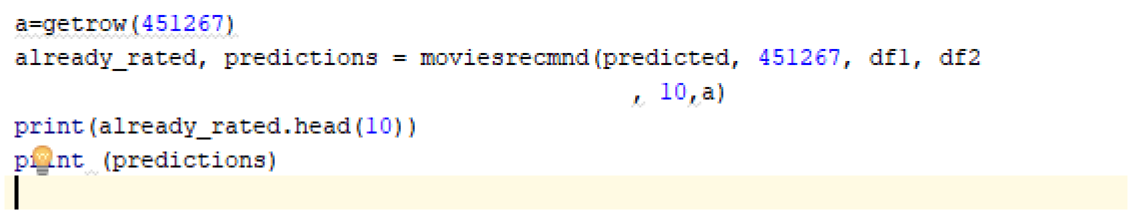
Finally, this gives us the entire range of the ratings for the movie. If the exact value lies in between the predicted range, we are providing a success value "1" else we are giving a failure value as "0". This is mainly done to calculate the error rate and the accuracy of the algorithm.

Limitations

•The main limitations of the Algorithm are we need to have a huge training data as the previous algorithm mainly depends upon the iterations it mainly depends upon the data. The larger the data the accurate is the prediction and less is the error rate.
•Although we can recommend the movies by using this algorithm it might not be accurate enough as there are no categories present in the dataset the movies having the same rating or more than the rating can be recommended to the users.

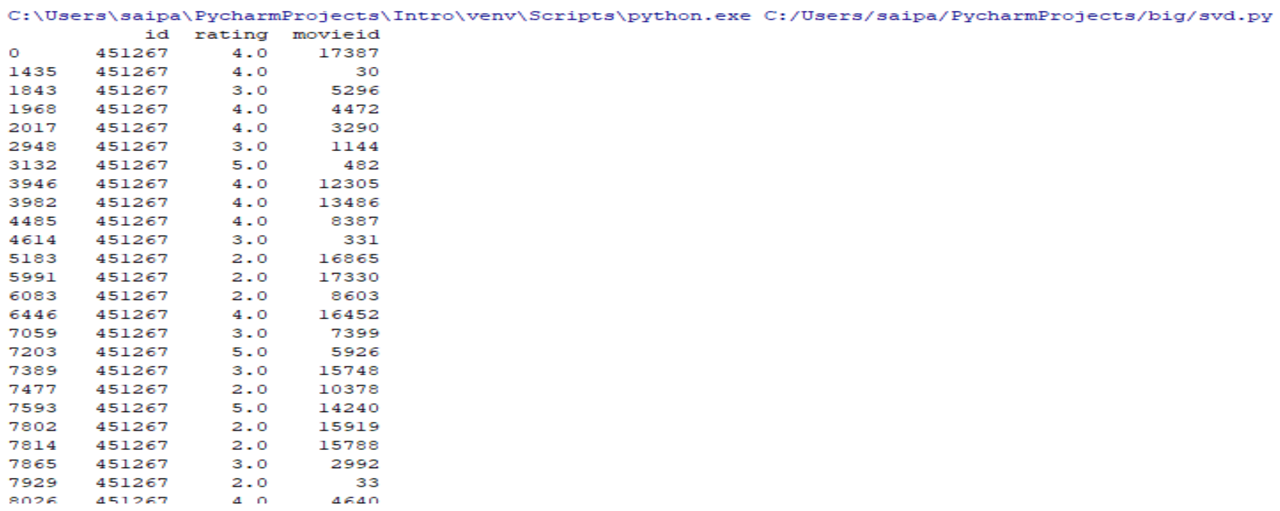
RESULTS

SVD



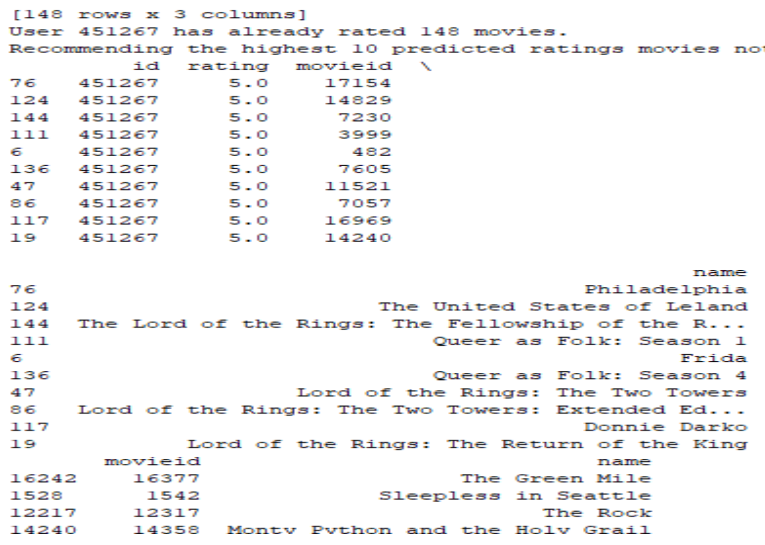
```
a=getrow(451267)
already_rated, predictions = moviesrecmnd(predicted, 451267, df1, df2
                                          , 10,a)
print(already_rated.head(10))
print (predictions)
```

The image is a snippet of the implementation of SVD Algorithm. The Snippet shows the prediction and recommendation being done for the user id *"451267"*



```
C:\Users\saipa\PycharmProjects\Intro\venv\Scripts\python.exe C:/Users/saipa/PycharmProjects/big/svd.py
           id  rating  movieid
0      451267     4.0    17387
1435   451267     4.0       30
1843   451267     3.0     5296
1968   451267     4.0     4472
2017   451267     4.0     3290
2948   451267     3.0     1144
3132   451267     5.0      482
3946   451267     4.0    12305
3982   451267     4.0    13486
4485   451267     4.0     8387
4614   451267     3.0      331
5183   451267     2.0    16865
5991   451267     2.0    17330
6083   451267     2.0     8603
6446   451267     4.0    16452
7059   451267     3.0     7399
7203   451267     5.0     5926
7389   451267     3.0    15748
7477   451267     2.0    10378
7593   451267     5.0    14240
7802   451267     2.0    15919
7814   451267     2.0    15788
7865   451267     3.0     2992
7929   451267     2.0       33
8026   451267     4.0     4640
```

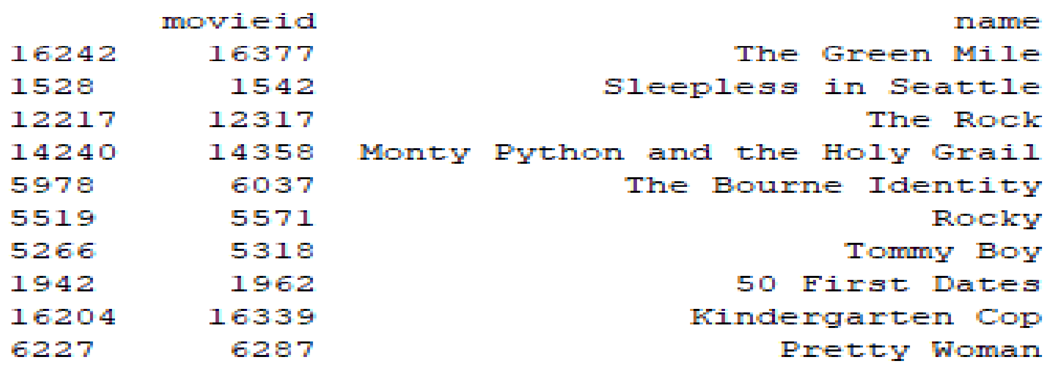
The below picture depicts the recommendation of ratings (that are predicted to be highest) and movies to the customers.



```
[148 rows x 3 columns]
User 451267 has already rated 148 movies.
Recommending the highest 10 predicted ratings movies not already rated.
         id  rating  movieid  \
76   451267     5.0    17154
124  451267     5.0    14829
144  451267     5.0     7230
111  451267     5.0     3999
6    451267     5.0      482
136  451267     5.0     7605
47   451267     5.0    11521
86   451267     5.0     7057
117  451267     5.0    16969
19   451267     5.0    14240

                                                   name
76                                         Philadelphia
124                             The United States of Leland
144  The Lord of the Rings: The Fellowship of the R...
111                                Queer as Folk: Season 1
6                                                 Frida
136                                Queer as Folk: Season 4
47                        Lord of the Rings: The Two Towers
86   Lord of the Rings: The Two Towers: Extended Ed...
117                                        Donnie Darko
19                  Lord of the Rings: The Return of the King
       movieid                           name
16242    16377                  The Green Mile
1528      1542             Sleepless in Seattle
12217    12317                        The Rock
14240    14358  Monty Python and the Holy Grail
```

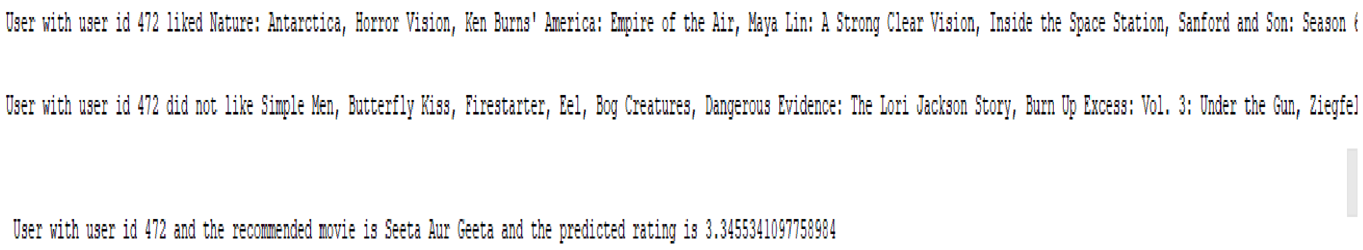
Picture depicting the recommendation of movies to the user having id “*451267”*



```
      movieid                              name
16242   16377                    The Green Mile
1528     1542               Sleepless in Seattle
12217   12317                          The Rock
14240   14358  Monty Python and the Holy Grail
5978     6037              The Bourne Identity
5519     5571                             Rocky
5266     5318                         Tommy Boy
1942     1962                    50 First Dates
16204   16339                  Kindergarten Cop
6227     6287                     Pretty Woman
```

ALS

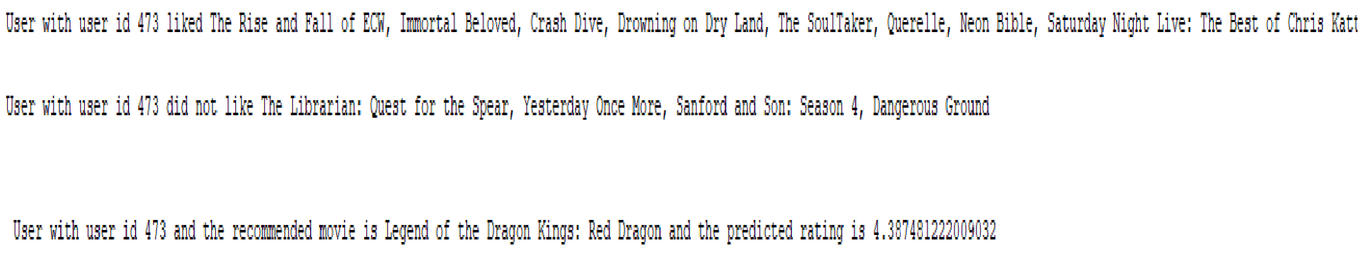
These pictures show the working of ALS algorithm along with the predicted rating and recommendation.



```
User with user id 472 liked Nature: Antarctica, Horror Vision, Ken Burns' America: Empire of the Air, Maya Lin: A Strong Clear Vision, Inside the Space Station, Sanford and Son: Season

User with user id 472 did not like Simple Men, Butterfly Kiss, Firestarter, Eel, Bog Creatures, Dangerous Evidence: The Lori Jackson Story, Burn Up Excess: Vol. 3: Under the Gun, Ziegfel


 User with user id 472 and the recommended movie is Seeta Aur Geeta and the predicted rating is 3.3455341097758984
```



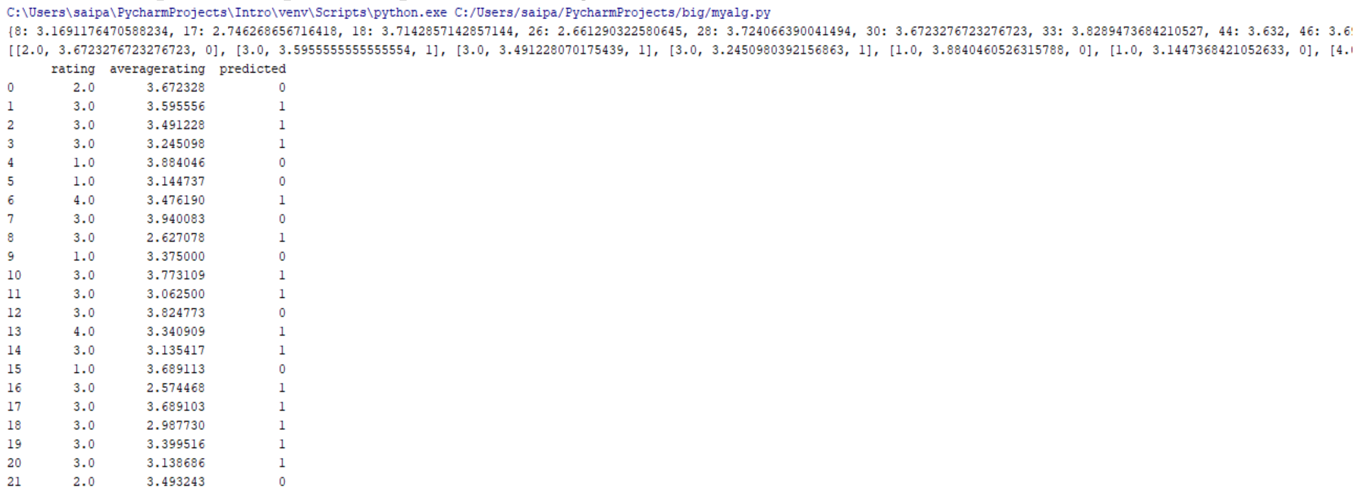
```
User with user id 473 liked The Rise and Fall of ECW, Immortal Beloved, Crash Dive, Drowning on Dry Land, The SoulTaker, Querelle, Neon Bible, Saturday Night Live: The Best of Chris Katt

User with user id 473 did not like The Librarian: Quest for the Spear, Yesterday Once More, Sanford and Son: Season 4, Dangerous Ground


 User with user id 473 and the recommended movie is Legend of the Dragon Kings: Red Dragon and the predicted rating is 4.387481222009032
```

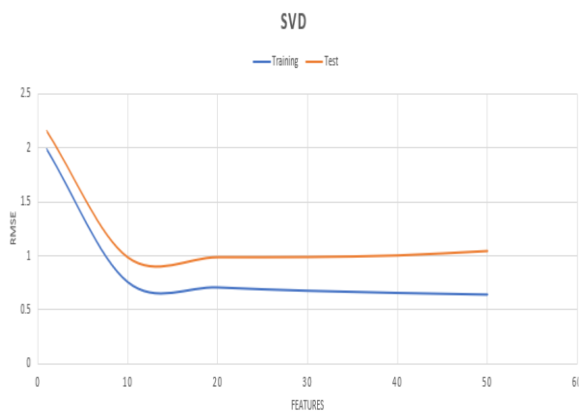
MEAN AND STANDARD DEVIATION APPROACH

The below picture depicts the predictive range of the values for the users.

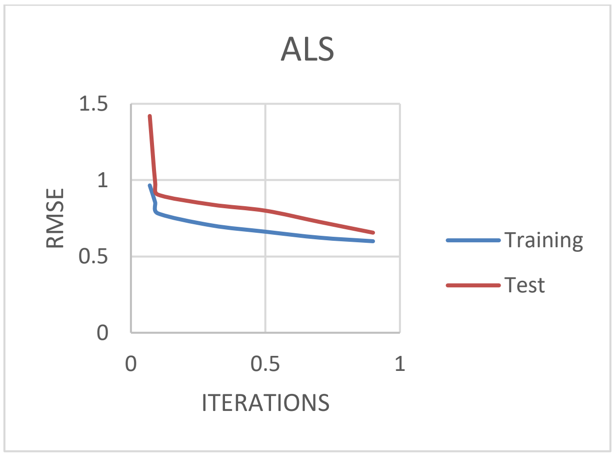


```
C:\Users\saipa\PycharmProjects\Intro\venv\Scripts\python.exe C:/Users/saipa/PycharmProjects/big/myalg.py
{8: 3.1691176470588234, 17: 2.746268656716418, 18: 3.7142857142857144, 26: 2.661290322580645, 28: 3.724066390041494, 30: 3.6723276723276723, 33: 3.8289473684210527, 44: 3.632, 46: 3.6
[[2.0, 3.6723276723276723, 0], [3.0, 3.5955555555555554, 1], [3.0, 3.491228070175439, 1], [3.0, 3.2450980392156863, 1], [1.0, 3.8840460526315788, 0], [1.0, 3.1447368421052633, 0], [4.
      rating  averagerating  predicted
0        2.0       3.672328          0
1        3.0       3.595556          1
2        3.0       3.491228          1
3        3.0       3.245098          1
4        1.0       3.884046          0
5        1.0       3.144737          0
6        4.0       3.476190          1
7        3.0       3.940083          0
8        3.0       2.627078          1
9        1.0       3.375000          0
10       3.0       3.773109          1
11       3.0       3.062500          1
12       3.0       3.824773          0
13       4.0       3.340909          1
14       3.0       3.135417          1
15       1.0       3.689113          0
16       3.0       2.574468          1
17       3.0       3.689103          1
18       3.0       2.987730          1
19       3.0       3.399516          1
20       3.0       3.138686          1
21       2.0       3.493243          0
```

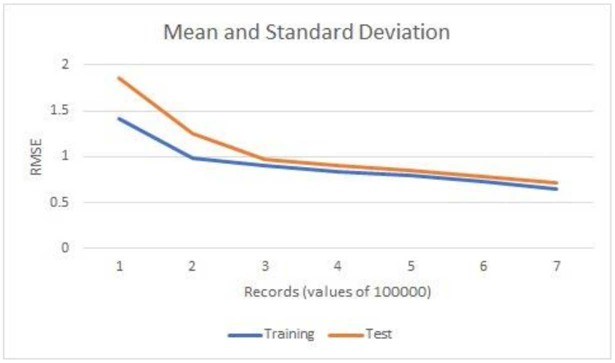
EVALUATION AND COMPARISON



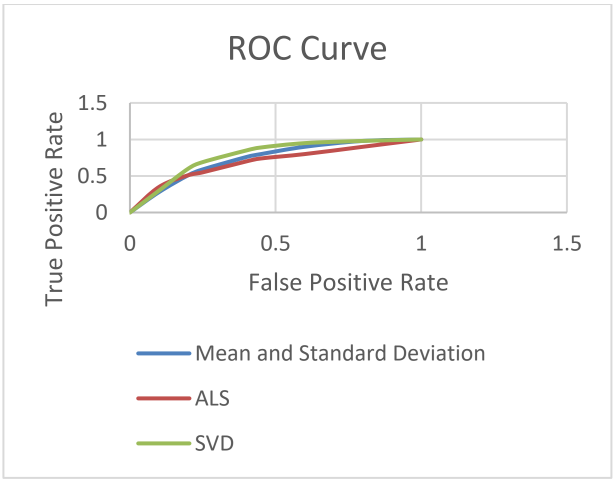
From the figure, it is quite evident that with an increase in the value of K the RMSE decreases and more accurate is the prediction for a user.



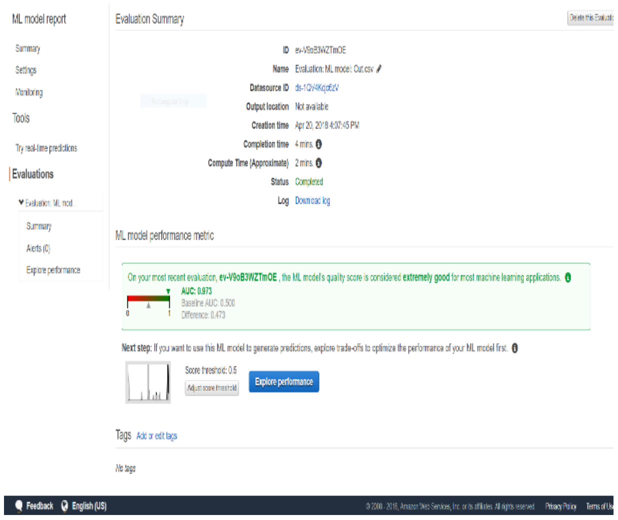
ALS picture depicting how iterations affect RMSE.



Mean and Standard Deviation picture depicting Records and RMSE variations.



This figure depicts ROC Curve for the 3 algorithms Mean-Standard Deviation, ALS and, SVD.



Evaluation Summary

ID	ev-V9oB3WZTmOE
Name	Evaluation: ML model: Out.csv
Datasource ID	ds-1QV4Kqjo6zV
Output location	Not available
Creation time	Apr 20, 2018 4:07:45 PM
Completion time	4 mins.
Compute Time (Approximate)	2 mins.
Status	Completed
Log	Download log

ML model performance metric

On your most recent evaluation, **ev-V9oB3WZTmOE** , the ML model's quality score is considered **extremely good** for most machine learning applications.

AUC: 0.973
Baseline AUC: 0.500
Difference: 0.473

Next step: If you want to use this ML model to generate predictions, explore trade-offs to optimize the performance of your ML model first.

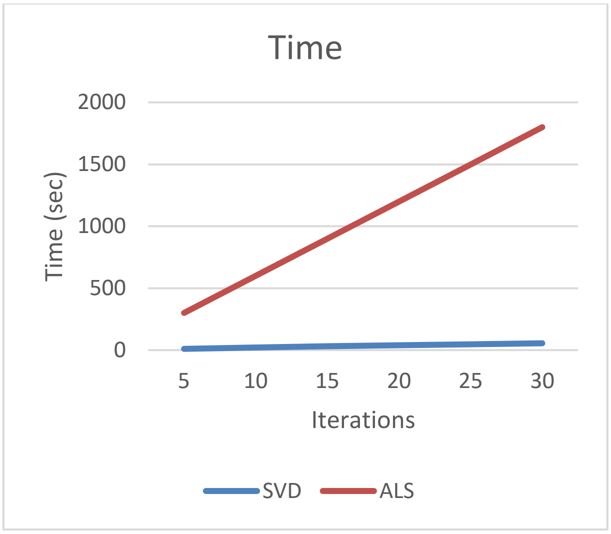
Score threshold: 0.5

Tags Add or edit tags

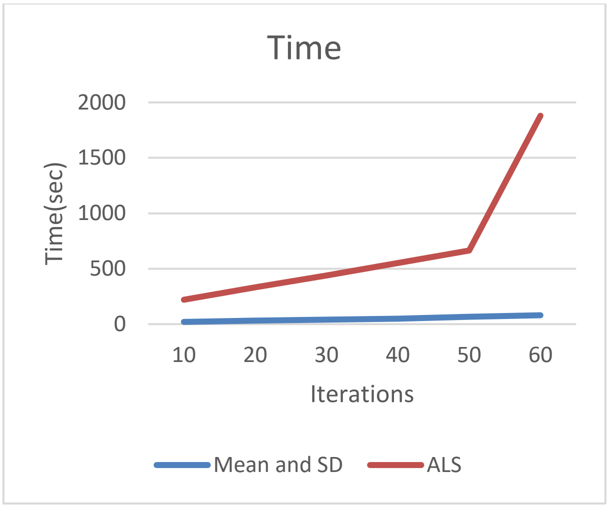
No tags

Feedback
English (US)
© 2008 - 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Us

The figure depicts the AUC for Mean Standard Deviation approach, which is 0.973. This is done using Amazon Machine Learning technique.



This figure compares the time taken for different iterations between SVD and ALS.



This figure compares the time taken for different iterations between Mean-Standard Deviation and ALS.

CONCLUSION

With the help of this project, we have predicted the movie ratings of the user on the Netflix data by using several different approaches like SVD, ALS, Mean and Standard Deviation approach (This was our own intuitive) and also provided recommendations to the user. We found out the tradeoffs between these Algorithms and compared our Algorithm to the other used algorithms with respect to time and error rates and many other factors. We have determined AUC, Accuracy, TPR, FPR for the algorithms and obtained all the above values for our Algorithm using Amazon machine Learning. This application not only helps the users but also NETFLIX to provide future ratings of the user for a movie and for the users to watch all similar types of movies they will like.

CONTRIBUTION

Task	People
1. Collecting and Pre-Processing data	Sai Pramod Yerubandi, Sri Navya Paruchuri, Sai Pavan Choudary
2. Implementing SVD	Sri Navya Paruchuri
3. Implementing ALS	Sai Pavan Choudary
4. Implementing Mean and Standard deviation Approach	Sai Pavan Choudary
5. Evaluating and comparison of Algorithms	Sai Pavan Choudary, Sai Pramod Yerubandi
6. Writing Report	Sai Pramod Yerubandi, Sri Navya Paruchuri, Sai Pavan Choudary
7. Slides, Demo and Presentation	Sri Navya Paruchuri, Sai Pramod Yerubandi

REFERENCES

[1]https://electronics.howstuffworks.com/netflix2.htm

[2]https://pdfs.semanticscholar.org/0e2e/439e6a5e2fd85205ccd4e8291c212f633f08.pdf

[3]http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.666.3609&rep=rep1&type=pdf

[4] http://www.netflix.com

[5]https://spark.apache.org/docs/latest/rdd-programming-guide.html

[6]https://web.stanford.edu/class/cme335/lecture6.pdf

[7]https://www.cs.cmu.edu/~venkatg/teaching/CStheory-infoage/book-chapter-4.pdf

[8]https://link.springer.com/chapter/10.1007/978-3-540-68880-8_32

[9]http://lucene.apache.org/hadoop/