A Profile Based Movie Recommendation System

Đinh Xuân Hùng, Lâm Lê Đình Khang, Nguyễn Thị Khánh Hà

University of Information Technology (UIT) – VietNam National University HCM City

18520791@gm.uit.edu.vn, 18520885@gm.uit.edu.vn, 18520692@gm.uit.edu.vn

ABSTRACT

In the context of the current covid 19 epidemic, the entertainment needs of each person increase sharply every day and watching movies is one of the top choices that people love.Through exploiting the thoughts and aspirations of customers will help businesses improve the quality of products and services. With the desire to contribute to helping entrepreneurs win the trust and interest of customers, we proceed to build a recommendation system in order to suggest movies to customers. Our project centers around building a movie recommendation system based on Collaborative filtering and Content-based Filtering for handling Big Data. In particular we look at implementing five algorithms of collaborative filtering which are K-Nearest Neighbours(KNN), Alternating Least Squares(ALS) and Singular Value decomposition (SVD), and Content-based, which are Cosine similarity and Jaccard similarity, and finally compare their performance in terms of standard measures.

Key word: Movies Recommendation System, Movies Recommendation System with Spark, Machine Learning, Big Data.

1. INTRODUCTION

Recommender systems are one of the most important applications of machine learning and are part of everyday life.. They are well-studied and proven to provide tremendous value to internet businesses and their consumers. Recommender systems can be loosely broken down into three categories: content-based systems, collaborative filtering systems, and hybrid systems, out of which we are going for item-based collaborative filtering systems. Item-based shared filtering recommends a product based on the browsing habits of other users who looked at the same item as me (users who looked at my item also looked at these other items). Further, Collaborative filtering systems can be further broken down into Memory-Based Collaborative Filtering and Model-based collaborative filtering. Model based algorithms uses the training data to create a model that can then be used to generate predictions while memory based collaborative filter just uses training data to generate a predictions.

We intend to implement Collaborative Filtering and Content-based Filtering.The algorithm implemented are K Nearest Neighbours (KNN), Alternating Least Square(ALS) and Singular Value decomposition (SVD), Cosine similarity, Jaccard similarity where KNN is memory based while ALS and Truncated Singular Value decomposition are model based.

Our report consists of 6 parts. Part 1 introduces the topic and motivation to study the above problem. Part 2 examines related works on the recommendation system problem. This is followed by Section 3 detailing the dataset used in movie recommendations. Section 4 presents the machine learning methods that we will apply on the above dataset. Section 5 presents our results when applying machine learning models on the above dataset. Finally, Section 6 will conclude the problem and suggest the next development direction.

2. RELATED RESEARCH WORKS

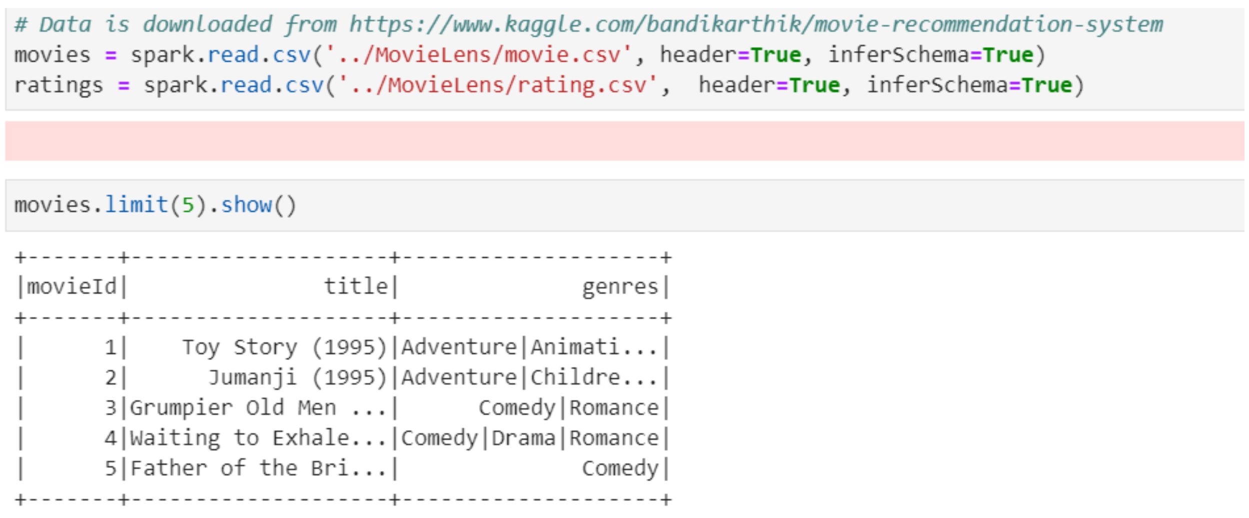
Many papers are available in literature on Big Data Analysis for Recommendation System using different platforms. Linyuan Lü,[1], reviewing recent developments in recommender systems and discuss the major issues. They have compared and evaluated available algorithms and examined their roles in the future development. In addition to algorithm, physical aspects are described to illustrate macroscopic behavior of recommender systems. Potential impacts and future directions are also discussed. They are emphasized that recommendation has great scientific depth and combines diverse research fields which makes it interesting for physicists as well as interdisciplinary researchers Sandra Garcia Esparza [2], proposed a collaborative filtering approach for producing recommendation for different products using reviews and opinions available on twitter or other social communication sites. Hongyan Liu[3], proposed recommendation method that analyzes the difference between the ratings and opinions of the user to identify the user's preferences. This method considers explicit ratings and implicit opinions, an action that can address the problem of data sparseness. Based on these methods, they also conduct an empirical study of online restaurant customer reviews to create a restaurant recommendation system and demonstrate the effectiveness of the proposed methods. Li Chen [4], proposed a novel clustering method based on Latent Class Regression model (LCRM), which is essentially able to consider both the overall ratings and featurelevel opinion values (as extracted from textual reviews) to identify reviewers' preference homogeneity. In the experiment, they tested the proposed recommender algorithm with two realworld datasets. More notably, they compared it with multiple related approaches, including the non-review based method and non-LCRM based variations Klavdiya [5], study the implementation of a predictive model on cloud using Hadoop and MapReduce programming concept.This paper represents a cloud recommendation system using mahout machine learning algorithm. Jai Prakash Verma [6], recommended summarization of reviews and feedback given by students or different stakeholders for an educational institution.

3. DATASETS

This dataset is taken from Kaggle [10].The datasets describe ratings and free-text tagging activities from MovieLens, a movie recommendation service. GroupLens Research has collected and made available rating data sets from the MovieLens web site (https://movielens.org). The data sets were collected over various periods of time, depending on the size of the set.[11]

It contains 20000263 ratings and 465564 tag applications across 27278 movies. These data were created by 138493 users between January 09, 1995 and March 31, 2015. This dataset was generated on October 17, 2016.

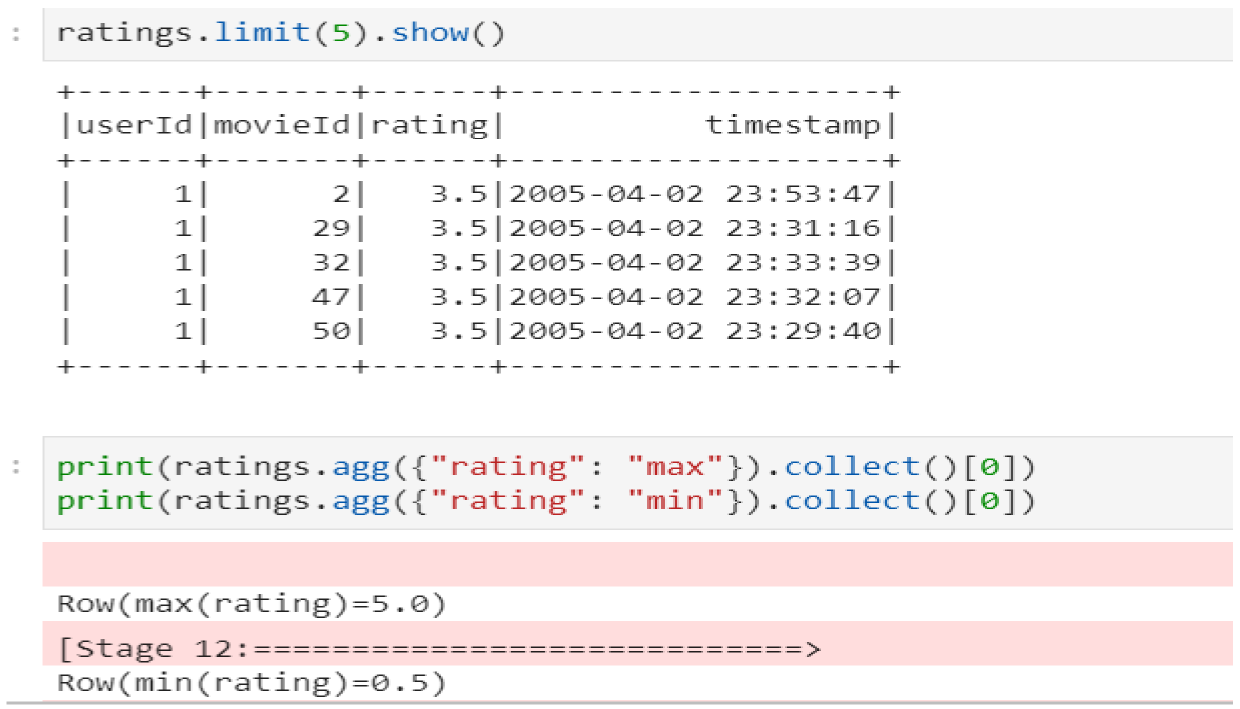
Users were selected at random for inclusion. All selected users had rated at least 20 movies.



```
# Data is downloaded from https://www.kaggle.com/bandikarthik/movie-recommendation-system
movies = spark.read.csv('../MovieLens/movie.csv', header=True, inferSchema=True)
ratings = spark.read.csv('../MovieLens/rating.csv',  header=True, inferSchema=True)
```

```
movies.limit(5).show()
```

```
+-------+--------------------+--------------------+
|movieId|               title|              genres|
+-------+--------------------+--------------------+
|      1|    Toy Story (1995)|Adventure|Animati...|
|      2|      Jumanji (1995)|Adventure|Childre...|
|      3|Grumpier Old Men ...|      Comedy|Romance|
|      4|Waiting to Exhale...|Comedy|Drama|Romance|
|      5|Father of the Bri...|              Comedy|
+-------+--------------------+--------------------+
```



```
ratings.limit(5).show()
```

```
+------+-------+------+-------------------+
|userId|movieId|rating|          timestamp|
+------+-------+------+-------------------+
|     1|      2|   3.5|2005-04-02 23:53:47|
|     1|     29|   3.5|2005-04-02 23:31:16|
|     1|     32|   3.5|2005-04-02 23:33:39|
|     1|     47|   3.5|2005-04-02 23:32:07|
|     1|     50|   3.5|2005-04-02 23:29:40|
+------+-------+------+-------------------+
```

```
print(ratings.agg({"rating": "max"}).collect()[0])
print(ratings.agg({"rating": "min"}).collect()[0])
```

```
Row(max(rating)=5.0)
[Stage 12:=============================>
Row(min(rating)=0.5)
```

4. METHODS OF IMPLEMENTATION

Collaborative filtering approaches build a model from a user's past behavior (items previously purchased or selected and/or numerical ratings given to those items) as well as similar decisions made by other users; then use that model to predict items (or ratings for items) that the user may be interested in. Content-based filtering approaches utilize a series of discrete characteristics of an item in order to recommend additional items with similar properties. These approaches are often combined in Hybrid Recommender Systems. In this report, our team takes a collaborative filtering approach to a movie recommendation system.

4.1 KNN Algorithm

When it comes to recommending items/movies, we are interested in recommending only top K items/movies to user [7]. KNN relies on labeled input data to learn a function that generates appropriate output for unlabeled data. Our first step is to clean the given dataset and reduce the number of user and movies while keeping dimension of rating constant and reshape the dataset into a format which can be given as parameters. Reducing the number of users and movie is necessary as it reduces sparsity and more importantly, KNN doesn't work properly in case of a large number of dimension. When a movie name is given as input, the KNN model selects movies from a dataset which have a fuzz ratio of more than 50 for the given input. It then calculates Euclidean distance for each selected movie for the input movie and selects K nearest movies with the smallest Euclidean distance as per calculation. Here, a simpler approach to select k is to set $K = \sqrt{n}$ where n is the number of data points in the training dataset(unique movies). From above K-nearest movies, we select top N movies which can be recommended to user.

4.2 ALS Algorithm

In collaborative filtering, matrix factorization is one of the solutions for sparse data problems. Alternating Least Square (ALS) [8] is also a matrix factorization algorithm that minimizes two loss functions alternatively obtained from matrix factorization.

In matrix factorization, the main goal is to divide the Rating Matrix into two lower dimension matrix which are the User matrix and Item Matrix. This algorithm assumes that there are k attributes or features which define each user and similarly there are k attributes or features that define each movie. Multiplying each feature of the user by the corresponding feature of the movie and adding it together gives a reasonable estimate of the user's rating.

4.3 SVD Algorithm

SVD is one of the matrix factorization algorithm which generates the matrices with the specified number of columns. Hence, SVD decreases the number of output and better works on the sparse matrices for features output. [9]

4.4 Cosine similarity Algorithm

Cosine similarity is a method to measure the difference between two non zero vectors of an inner product space.

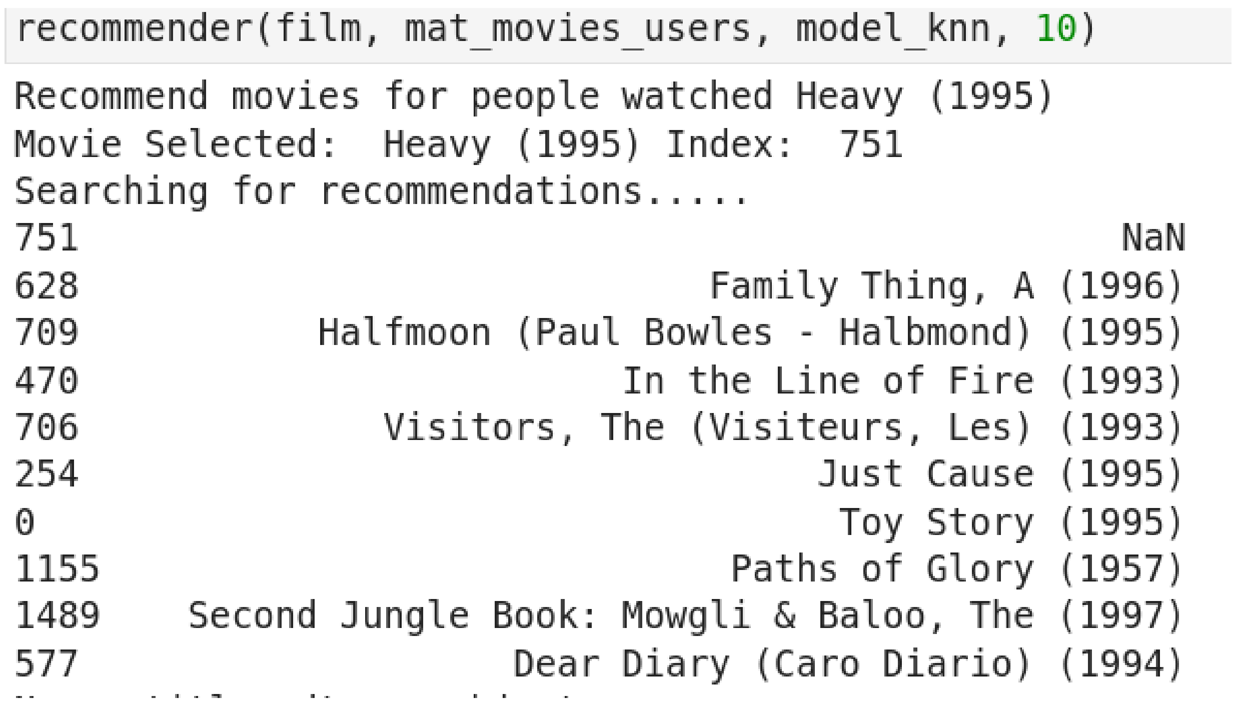
4.5 Jaccard similarity Algorithm

Jaccard Similarity, also called the Jaccard Index or Jaccard Similarity Coefficient, compares members for two sets to see which members are shared and which are distinct.

5. RESULT

5.1 KNN Algorithm

Given below is the output of recommended movies to the user when a movie is given as an input string. The recommended movies seem similar to the given input movie.



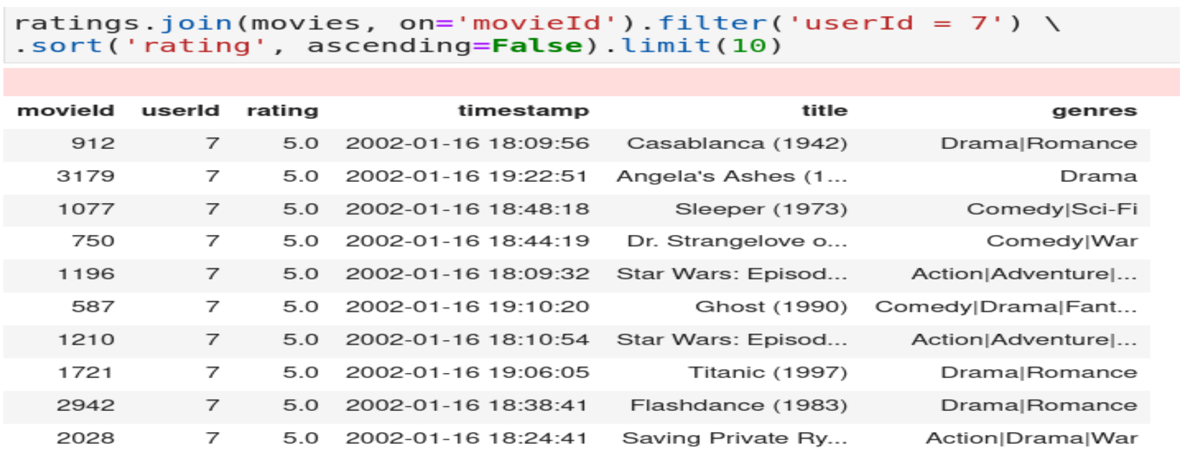
```
recommender(film, mat_movies_users, model_knn, 10)
```

```
Recommend movies for people watched Heavy (1995)
Movie Selected:  Heavy (1995) Index:  751
Searching for recommendations.....
751                                                   NaN
628                                  Family Thing, A (1996)
709             Halfmoon (Paul Bowles - Halbmond) (1995)
470                              In the Line of Fire (1993)
706                 Visitors, The (Visiteurs, Les) (1993)
254                                       Just Cause (1995)
0                                          Toy Story (1995)
1155                                  Paths of Glory (1957)
1489     Second Jungle Book: Mowgli & Baloo, The (1997)
577                         Dear Diary (Caro Diario) (1994)
```

Here, movie, item-user-matrix-sparse, model, movie-ToIndex and N is passed as arguments to the recommendation function where movie is input movie string, item-user-matrixsparse is list which consists non-zeros values of movie-id and user-id, model is K-NearestNeighbors, movie-To-index is mapper that maps movie index and its title as we have two different files for movie title and movie index and finally N here is 7 which is top N movies recommended for user.

5.2 ALS Algorithm

In the ALS algorithm, the recommended movies are generated based on the previous

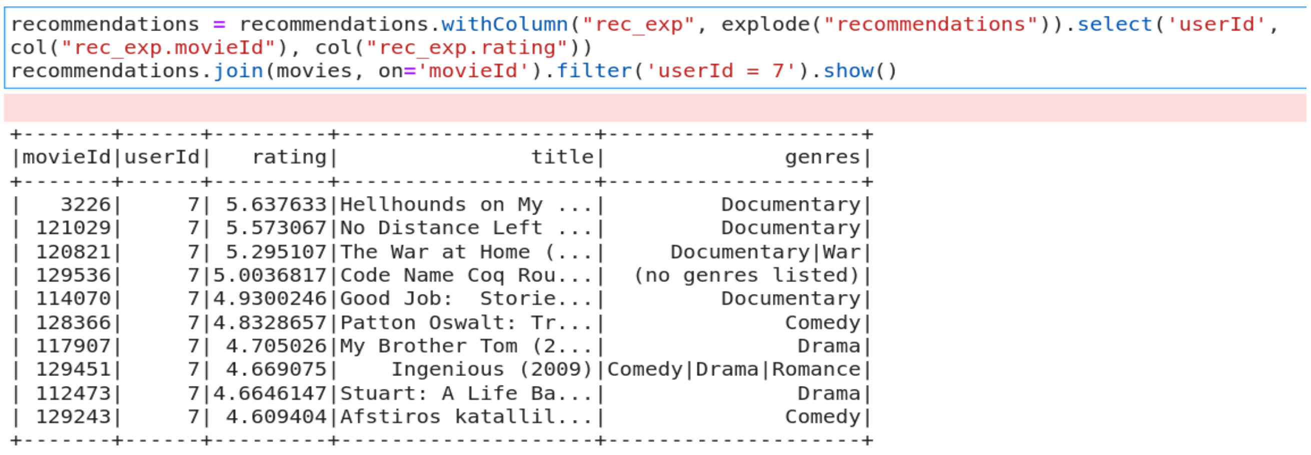


```
ratings.join(movies, on='movieId').filter('userId = 7') \
.sort('rating', ascending=False).limit(10)
```

movieId	userId	rating	timestamp	title	genres
912	7	5.0	2002-01-16 18:09:56	Casablanca (1942)	Drama|Romance
3179	7	5.0	2002-01-16 19:22:51	Angela's Ashes (1...	Drama
1077	7	5.0	2002-01-16 18:48:18	Sleeper (1973)	Comedy|Sci-Fi
750	7	5.0	2002-01-16 18:44:19	Dr. Strangelove o...	Comedy|War
1196	7	5.0	2002-01-16 18:09:32	Star Wars: Episod...	Action|Adventure|...
587	7	5.0	2002-01-16 19:10:20	Ghost (1990)	Comedy|Drama|Fant...
1210	7	5.0	2002-01-16 18:10:54	Star Wars: Episod...	Action|Adventure|...
1721	7	5.0	2002-01-16 19:06:05	Titanic (1997)	Drama|Romance
2942	7	5.0	2002-01-16 18:38:41	Flashdance (1983)	Drama|Romance
2028	7	5.0	2002-01-16 18:24:41	Saving Private Ry...	Action|Drama|War

preferences and ratings given by the user. The given figure depicts the list of the movie which are preferred by user-id 7

The below figure depicts the list of movie which are recommended to user-id 7 based on their preference and rating.

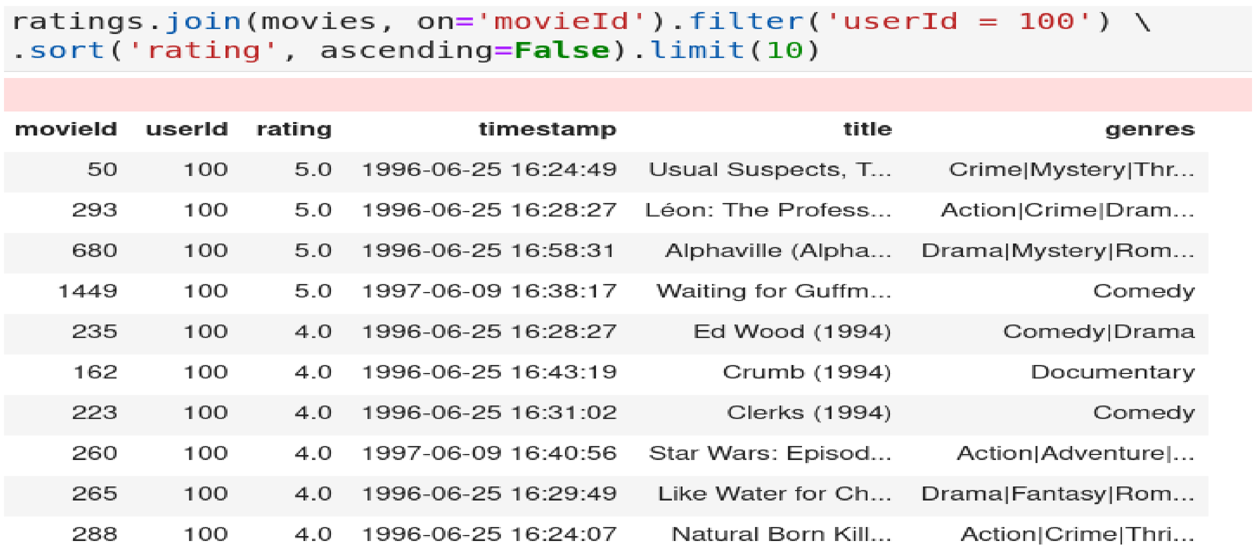


```
recommendations = recommendations.withColumn("rec_exp", explode("recommendations")).select('userId',
col("rec_exp.movieId"), col("rec_exp.rating"))
recommendations.join(movies, on='movieId').filter('userId = 7').show()
```

```
+-------+------+---------+--------------------+--------------------+
|movieId|userId|   rating|               title|              genres|
+-------+------+---------+--------------------+--------------------+
|   3226|     7| 5.637633|Hellhounds on My ...|         Documentary|
| 121029|     7| 5.573067|No Distance Left ...|         Documentary|
| 120821|     7| 5.295107|The War at Home (...|     Documentary|War|
| 129536|     7|5.0036817|Code Name Coq Rou...|  (no genres listed)|
| 114070|     7|4.9300246|Good Job:  Storie...|         Documentary|
| 128366|     7|4.8328657|Patton Oswalt: Tr...|              Comedy|
| 117907|     7| 4.705026|My Brother Tom (2...|               Drama|
| 129451|     7| 4.669075|    Ingenious (2009)|Comedy|Drama|Romance|
| 112473|     7|4.6646147|Stuart: A Life Ba...|               Drama|
| 129243|     7| 4.609404|Afstiros katallil...|              Comedy|
+-------+------+---------+--------------------+--------------------+
```

5.3 SVD Algorithm

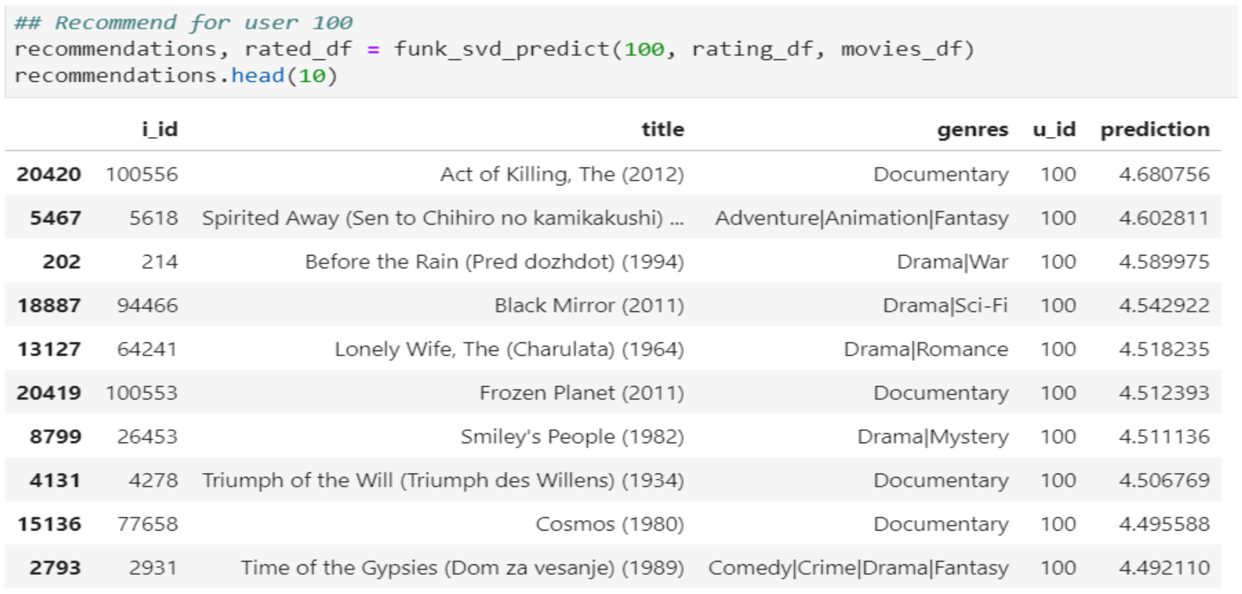
In the SVD algorithm, the recommended movies are generated based on the previous preferences and ratings given by the user. The given figure depicts the list of the movie, which are preferred by user-id 100



```
ratings.join(movies, on='movieId').filter('userId = 100') \
.sort('rating', ascending=False).limit(10)
```

movieId	userId	rating	timestamp	title	genres
50	100	5.0	1996-06-25 16:24:49	Usual Suspects, T...	Crime|Mystery|Thr...
293	100	5.0	1996-06-25 16:28:27	Léon: The Profess...	Action|Crime|Dram...
680	100	5.0	1996-06-25 16:58:31	Alphaville (Alpha...	Drama|Mystery|Rom...
1449	100	5.0	1997-06-09 16:38:17	Waiting for Guffm...	Comedy
235	100	4.0	1996-06-25 16:28:27	Ed Wood (1994)	Comedy|Drama
162	100	4.0	1996-06-25 16:43:19	Crumb (1994)	Documentary
223	100	4.0	1996-06-25 16:31:02	Clerks (1994)	Comedy
260	100	4.0	1997-06-09 16:40:56	Star Wars: Episod...	Action|Adventure|...
265	100	4.0	1996-06-25 16:29:49	Like Water for Ch...	Drama|Fantasy|Rom...
288	100	4.0	1996-06-25 16:24:07	Natural Born Kill...	Action|Crime|Thri...

The below figure depicts the list of movie, which are recommended for user-id 100 based on their preference and rating.

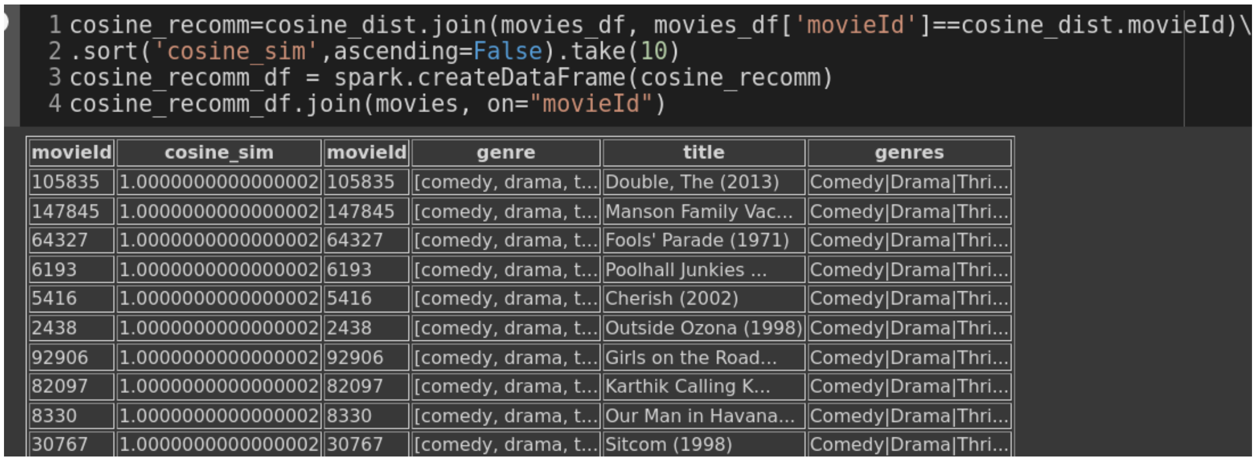


```
## Recommend for user 100
recommendations, rated_df = funk_svd_predict(100, rating_df, movies_df)
recommendations.head(10)
```

	i_id	title	genres	u_id	prediction
20420	100556	Act of Killing, The (2012)	Documentary	100	4.680756
5467	5618	Spirited Away (Sen to Chihiro no kamikakushi) ...	Adventure|Animation|Fantasy	100	4.602811
202	214	Before the Rain (Pred dozhdot) (1994)	Drama|War	100	4.589975
18887	94466	Black Mirror (2011)	Drama|Sci-Fi	100	4.542922
13127	64241	Lonely Wife, The (Charulata) (1964)	Drama|Romance	100	4.518235
20419	100553	Frozen Planet (2011)	Documentary	100	4.512393
8799	26453	Smiley's People (1982)	Drama|Mystery	100	4.511136
4131	4278	Triumph of the Will (Triumph des Willens) (1934)	Documentary	100	4.506769
15136	77658	Cosmos (1980)	Documentary	100	4.495588
2793	2931	Time of the Gypsies (Dom za vesanje) (1989)	Comedy|Crime|Drama|Fantasy	100	4.492110

5.4 Consine similarity Algorithm

The Consine similarity looks at the ratio of movie genres a user has seen to those that the user has never seen. The idea behind this method is to calculate a similarity coefficient between movies. From there, it will recommend users to watch the movies with the highest coefficients. Recommend movies for viewer who watched movie with movieId = 45

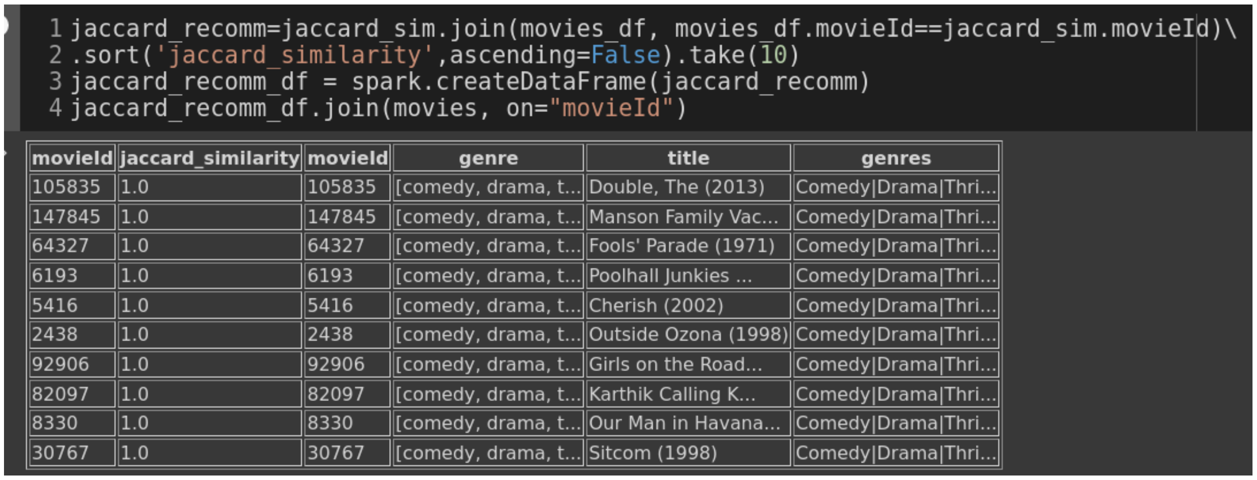


```
cosine_recomm=cosine_dist.join(movies_df, movies_df['movieId']==cosine_dist.movieId)\
.sort('cosine_sim',ascending=False).take(10)
cosine_recomm_df = spark.createDataFrame(cosine_recomm)
cosine_recomm_df.join(movies, on="movieId")
```

movieId	cosine_sim	movieId	genre	title	genres
105835	1.0000000000000002	105835	[comedy, drama, t...	Double, The (2013)	Comedy|Drama|Thri...
147845	1.0000000000000002	147845	[comedy, drama, t...	Manson Family Vac...	Comedy|Drama|Thri...
64327	1.0000000000000002	64327	[comedy, drama, t...	Fools' Parade (1971)	Comedy|Drama|Thri...
6193	1.0000000000000002	6193	[comedy, drama, t...	Poolhall Junkies ...	Comedy|Drama|Thri...
5416	1.0000000000000002	5416	[comedy, drama, t...	Cherish (2002)	Comedy|Drama|Thri...
2438	1.0000000000000002	2438	[comedy, drama, t...	Outside Ozona (1998)	Comedy|Drama|Thri...
92906	1.0000000000000002	92906	[comedy, drama, t...	Girls on the Road...	Comedy|Drama|Thri...
82097	1.0000000000000002	82097	[comedy, drama, t...	Karthik Calling K...	Comedy|Drama|Thri...
8330	1.0000000000000002	8330	[comedy, drama, t...	Our Man in Havana...	Comedy|Drama|Thri...
30767	1.0000000000000002	30767	[comedy, drama, t...	Sitcom (1998)	Comedy|Drama|Thri...

5.5 Jaccard similarity Algorithm

The jaccard similarity looks at the ratio of movie genres a user has seen to those that the user has never seen. The idea behind this method is to calculate a similarity coefficient between movies. From there, it will recommend users to watch the movies with the highest coefficients. Recommend movies for viewer who watched movie with movieId = 45

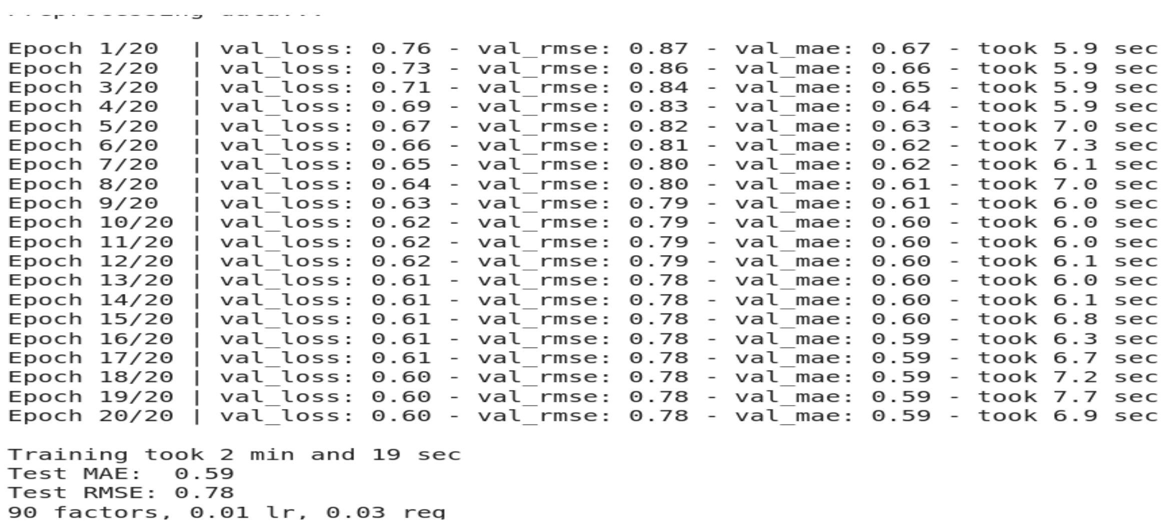


```
jaccard_recomm=jaccard_sim.join(movies_df, movies_df.movieId==jaccard_sim.movieId)\
.sort('jaccard_similarity',ascending=False).take(10)
jaccard_recomm_df = spark.createDataFrame(jaccard_recomm)
jaccard_recomm_df.join(movies, on="movieId")
```

movieId	jaccard_similarity	movieId	genre	title	genres
105835	1.0	105835	[comedy, drama, t...	Double, The (2013)	Comedy|Drama|Thri...
147845	1.0	147845	[comedy, drama, t...	Manson Family Vac...	Comedy|Drama|Thri...
64327	1.0	64327	[comedy, drama, t...	Fools' Parade (1971)	Comedy|Drama|Thri...
6193	1.0	6193	[comedy, drama, t...	Poolhall Junkies ...	Comedy|Drama|Thri...
5416	1.0	5416	[comedy, drama, t...	Cherish (2002)	Comedy|Drama|Thri...
2438	1.0	2438	[comedy, drama, t...	Outside Ozona (1998)	Comedy|Drama|Thri...
92906	1.0	92906	[comedy, drama, t...	Girls on the Road...	Comedy|Drama|Thri...
82097	1.0	82097	[comedy, drama, t...	Karthik Calling K...	Comedy|Drama|Thri...
8330	1.0	8330	[comedy, drama, t...	Our Man in Havana...	Comedy|Drama|Thri...
30767	1.0	30767	[comedy, drama, t...	Sitcom (1998)	Comedy|Drama|Thri...

5.6 Performance Comparison

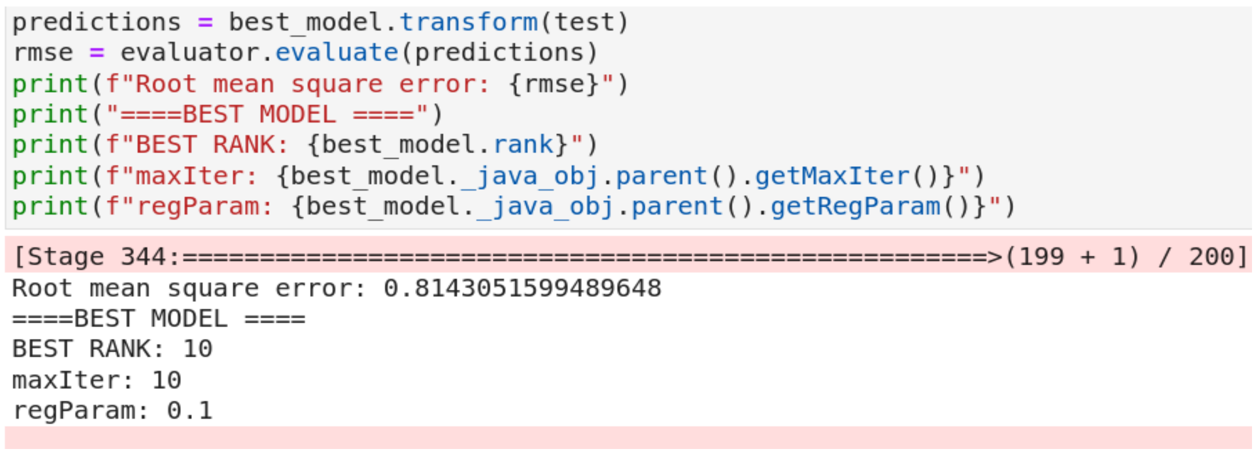
SVD:



```
Epoch 1/20  | val_loss: 0.76 - val_rmse: 0.87 - val_mae: 0.67 - took 5.9 sec
Epoch 2/20  | val_loss: 0.73 - val_rmse: 0.86 - val_mae: 0.66 - took 5.9 sec
Epoch 3/20  | val_loss: 0.71 - val_rmse: 0.84 - val_mae: 0.65 - took 5.9 sec
Epoch 4/20  | val_loss: 0.69 - val_rmse: 0.83 - val_mae: 0.64 - took 5.9 sec
Epoch 5/20  | val_loss: 0.67 - val_rmse: 0.82 - val_mae: 0.63 - took 7.0 sec
Epoch 6/20  | val_loss: 0.66 - val_rmse: 0.81 - val_mae: 0.62 - took 7.3 sec
Epoch 7/20  | val_loss: 0.65 - val_rmse: 0.80 - val_mae: 0.62 - took 6.1 sec
Epoch 8/20  | val_loss: 0.64 - val_rmse: 0.80 - val_mae: 0.61 - took 7.0 sec
Epoch 9/20  | val_loss: 0.63 - val_rmse: 0.79 - val_mae: 0.61 - took 6.0 sec
Epoch 10/20 | val_loss: 0.62 - val_rmse: 0.79 - val_mae: 0.60 - took 6.0 sec
Epoch 11/20 | val_loss: 0.62 - val_rmse: 0.79 - val_mae: 0.60 - took 6.0 sec
Epoch 12/20 | val_loss: 0.62 - val_rmse: 0.79 - val_mae: 0.60 - took 6.1 sec
Epoch 13/20 | val_loss: 0.61 - val_rmse: 0.78 - val_mae: 0.60 - took 6.0 sec
Epoch 14/20 | val_loss: 0.61 - val_rmse: 0.78 - val_mae: 0.60 - took 6.1 sec
Epoch 15/20 | val_loss: 0.61 - val_rmse: 0.78 - val_mae: 0.60 - took 6.8 sec
Epoch 16/20 | val_loss: 0.61 - val_rmse: 0.78 - val_mae: 0.59 - took 6.3 sec
Epoch 17/20 | val_loss: 0.61 - val_rmse: 0.78 - val_mae: 0.59 - took 6.7 sec
Epoch 18/20 | val_loss: 0.60 - val_rmse: 0.78 - val_mae: 0.59 - took 7.2 sec
Epoch 19/20 | val_loss: 0.60 - val_rmse: 0.78 - val_mae: 0.59 - took 7.7 sec
Epoch 20/20 | val_loss: 0.60 - val_rmse: 0.78 - val_mae: 0.59 - took 6.9 sec

Training took 2 min and 19 sec
Test MAE:  0.59
Test RMSE: 0.78
90 factors, 0.01 lr, 0.03 reg
```

ALS:



```python
predictions = best_model.transform(test)
rmse = evaluator.evaluate(predictions)
print(f"Root mean square error: {rmse}")
print("====BEST MODEL ====")
print(f"BEST RANK: {best_model.rank}")
print(f"maxIter: {best_model._java_obj.parent().getMaxIter()}")
print(f"regParam: {best_model._java_obj.parent().getRegParam()}")
```

```
[Stage 344:====================================================>(199 + 1) / 200]
Root mean square error: 0.8143051599489648
====BEST MODEL ====
BEST RANK: 10
maxIter: 10
regParam: 0.1
```

As you can see in above figure, RMSE of ALS is 0.81 while RMSE of SVD is 0.78 Hence, SVD works better than ALS.

6. CONCLUSIONS AND FUTURE WORK

In this report, we have introduced the movie recommendation system solution by applying machine learning algorithms such as KNN(k-nearest neighbors), ALS (Alternating Least Squares), SVD (Singular Value Decomposition), Consine similarity, Jaccard similarity.The most important problem that we aim to is to build an optimal movie recommendation system, as a result, we experiment the movie recommendation system with the data set obtained only at a good level on the entire datasets.

From the results in this model, in the future we want to perform the problem with a larger dataset and apply methods and techniques to increase system quality to bring the best results.

Link code:

REFERENCES

[1] Linyuan Lü, Matúš Medo, Chi Ho Yeung, Yi-Cheng Zhang, Zi-Ke Zhang,Tao Zhoua, Recommender systems, Elsevier Journal – Physics Reports 519 (2012) 1–49

[2] Sandra Garcia Esparza, Michael P. O'Mahony, Barry Smyth, Mining the real-time web: A novel approach to product recommendation, Elsevier Journal - Knowledge-Based Systems 29 (2012) 3–11.

[3] Hongyan Liu, Jun He, Tingting Wang, Wenting Song, Xiaoyang Du, "Combining user preferences and user opinions for accurate recommendation", Elsevier Journal- Electronic Commerce Research and Applications 12 (2013) 14–23.

[4] Li Chen, Feng Wang, "Preference-based clustering reviews for augmenting e-commerce recommendation", Elsevier Journal - Knowledge-Based Systems 50 (2013) 44–59.

[5] Klavdiya Hammond, Aparna S. Varde, "Cloud Based Predictive Analytics", 2013, IEEE 13'th International Conference on Data Mining Workshops.

[6] Jai Prakash, Bankim Patel, Atul Patel,"Web Mining: Opinion and Feedback Analysis for Educational Institutions", 2013, IJCA, Volume 84 – No 6, December 2013.

[7] Gupta, Meenu, et al. "Movie Recommender System Using Collaborative Filtering." 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC). IEEE, 2020.

[8] Li, Jung-Bin, et al. "Implementation of an Alternating Least Square Model Based Collaborative Filtering Movie Recommendation System on Hadoop and Spark Platforms." International Conference on Broadband and Wireless Computing, Communication and Applications. Springer, Cham, 2018.

[9] Kumar, D. (2021, January 12). Singular value DECOMPOSITION (SVD) in recommender system. Retrieved April 11, 2021, from https://analyticsindiamag.com/singular-value-decomposition-svdapplication-recommender-system/

[10] https://www.kaggle.com/grouplens/movielens-20m-dataset

[11] https://grouplens.org/datasets/movielens/