Movie Recommendation System Using Collaborative Filtering

Ching-Seh (Mike) Wu
Dept. of Computer Science
San Jose State University
San Jose, CA, USA
ching-seh.wu@sjsu.edu

Deepti Garg
Dept. of Computer Science
San Jose State University
San Jose, CA, USA
deepti.garg@sjsu.edu

Unnathi Bhandary
Dept. of Computer Science
San Jose State University
San Jose, CA, USA
unnathi.bhandary@sjsu.edu

Abstract-As the business needs are accelerating, there is an increased dependence on extracting meaningful information from humongous amount of raw data to drive business solutions. The same is true for digital recommendation systems which are becoming a norm for consumer industries such as books, music, clothing, movies, news articles, places, utilities, etc. These systems collect information from the users to improve the future suggestions. This paper aims to describe the implementation of a movie recommender system via two collaborative filtering algorithms using Apache Mahout. Furthermore, this paper will also focus on analyzing the data to gain insights into the movie dataset using Matplotlib libraries in Python.

Keywords: Collaborative filtering, recommender system, mahout, user-based recommender, item-based recommender.

I. INTRODUCTION

A recommendation system or recommendation engine is a model used for information filtering where it tries to predict the preferences of a user and provide suggests based on these preferences. These systems have become increasingly popular nowadays and are widely used today in areas such as movies, music, books, videos, clothing, restaurants, food, places and other utilities. These systems collect information about a user's preferences and behavior, and then use this information to improve their suggestions in the future.

A large number of companies are making use of recommendation systems to increase user interaction and enrich a user's shopping experience. Recommendation systems have several benefits, the most important being customer satisfaction and revenue. Many a times, customers tend to look at the recommendations provided based on their previous transaction because they think that they will find better options. If these recommendations are fine tuned to the user's needs, the customer will be satisfied with their purchase. Thus, the customer would use this application once again. With customers using these applications frequently, a huge amount of revenue is generated, which is why many e-commerce compames are turning to Improve their recommendation engines.

Although recommendation systems are common, developing systems that provide good and appropriate suggestions is a challenge. Each user has different preferences and likes. Additionally, a user's preference depends on many aspects such as their mood, the occasion, the reason for their purchase etc. If a website or app is not able to predict and provide suitable recommendations as per the liking of the user, then the user is likely to stop using that website or app. Thus, there is always a need for companies to improve their recommendation systems.

One goal of this paper IS to design a mOVIe recommendation system that considers the past movie ratings given by various users to provide suggestions to the user. We implemented this system usmg collaborative filtering algorithms and Apache Mahout framework The second goal is to compare the performance and efficiency of user-based recommender system and item-based recommender system.

This paper is organized as follows: First, a brief overview of a few relevant, recent research done in the space of recommender system will be discussed. Second, we will present the understanding on the technique of collaborative filtering. Third, the data preparation and data analysis approach using Mahout will be discussed. Finally, a qualitative evaluation on the techniques used will be presented.

II. OVERVIEW

A. Recommendation Systems

In this section we briefly introduce the different types of recommendation systems. There are three main types of recommendation systems, namely collaborative filtering, content-based filtering and hybrid system.

Collaborative Filtering: Collaborative filtering systems analyze the user's behavior and preferences and predict what they would like based on similarity with other users. There are two kinds of collaborative filtering systems; user-based recommender and item-based recommender.

Content Based Filtering: Content-based systems considers the description and features of an item along with the user's preferences to provide suggestions.

Hybrid System: Hybrid recommendation systems are a combination of both collaborative and content-based filtering methods. In these type of systems, collaborative and

978-1-5386-6565-7118/$31.00

content-based predictions are performed separately and then the results of both techniques are combined to provide reconnnendations.

B. Mahout

Apache Mahout is a scientific and highly analytical library based on the Apache Software Foundation. It aims to produce free, distributed and scalable implementations of advanced machine learning algorithms used in the fields of clustering, classification, collaborative filtering, and frequent pattern matching. Many of the implementations use the Apache Hadoop platform. It includes implementations of powerful algoritlnn like Loglikelihood Similarity, Pearson Coefficient, Cosine Similarity- to name a few. Mahout is still not yet fully developed as it continues to grow. Nevertheless, it still offers a full stack option of incorporating machine learning on the big data which is managed by the underlying Hadoop platform.

C. Related Work

Over the years, many reconnnendation systems have been developed using either collaborative, content based or hybrid filtering methods. These systems have been implemented using various big data and machine learning algorithms.

In [1] the authors propose a collaborative reconnnendation system which is designed to work on the Hadoop platform, using the MapReduce framework. The authors have used the set-similarity join method to build this system, employing both user-based and item-based collaborative filtering techniques.

In [2] the authors proposed a movie reconnnendation system using collaborative filtering that focuses on the ratings given by the users to provide reconnnendations. The proposed system is built using K-means algorithm to sort the movies according to the ratings.

In [4] the authors propose a fully content-based movie reconnnendation system to reconnnend movies. The proposed system makes use of a neural network with the content information of the movies to obtain features and learn the similarities between movies. The movies are reconnnended based on the similarity between them.

In [7] the authors implement a reconnnendation system that combines both user based and item-based collaborative filtering approach. The system is built using nearest neighbors machine learning technique and develop a new algorithm that unifies used based and item-based reconnnendations.

Based on the research we conducted, collaborative filtering was found to be one ofthe popularly used approaches to build reconnnendation systems. Many of the systems used machine learning algorithms such as clustering using K-means, neural networks and so on to reconnnend items.

III. COLLABORATIVE FILTERING MODEL

We propose a collaborative movie reconnnendation system that focuses on the ratings given by the users to provide reconnnendations. This system is designed using Mahout which is built on top of the Hadoop platform. We have used the similarity/correlation between items to build this system, employing both user-based and item-based collaborative filtering techniques. In this section, we describe the intricacies of the two approaches of collaborative filtering.

A. Use-basedfiltering

User-based preferences are very connnon in the field of designing personalized systems. This approach assumes that the user's likings are not random if analyzed historically.

The process starts with users giving ratings (1-5) to some catalog items. These ratings can be implicit or explicit. Explicit ratings are when the user explicitly rates the item on some scale or indicates a thumbs-up/thumbs-down to the item. Often explicit ratings are hard to gather as not every user is much interested in providing feedbacks. In these scenarios, we gather implicit ratings based on their behavior. For instance, if a user buys a product more than once, it indicates a positive preference. In context to movie systems, we can imply that if a user watches the entire movie, he/she has some likeability to it. Note that there are no clear rules in determining implicit ratings.

Next, for each user, we first find some defined number of *nearest neighbors.* We calculate correlation between users' ratings using Pearson Correlation algoritlnn. The assumption that iftwo users' ratings are highly correlated, then these two users must enjoy similar items and products is used to reconnnend items to users.

B. Item-basedfiltering

Unlike the user-based filtering method, item based focuses on the similarity between the items users like instead of the users themselves. The most similar items are computed ahead of time. Then for reconnnendation, the items that are most similar to the target item are reconnnended to the user.

Figure. 1 further explains the two types of collaborative reconnnender systems. This type of filtering is often called *Pure CF.*

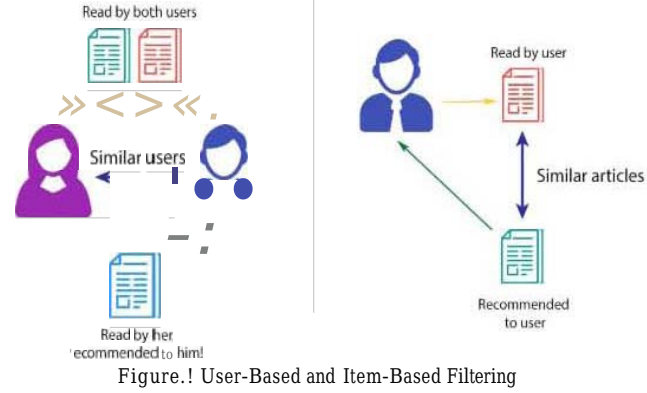


Figure.! User-Based and Item-Based Filtering

IV. IMPLEMENTATION

In this section we describe how we implemented our system.

A. Dataset

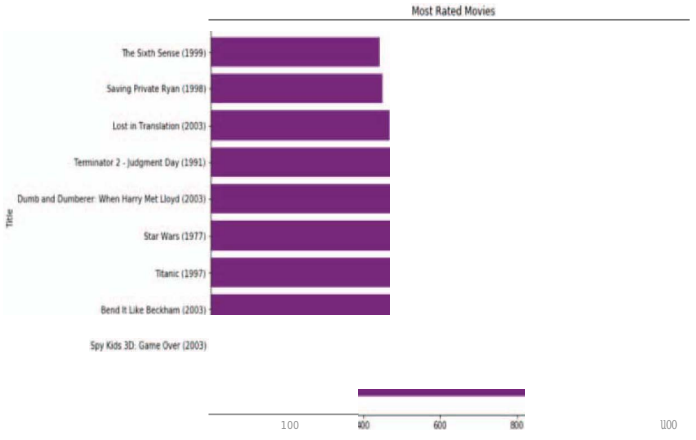
The movies dataset used in our experiments is obtained from the Yahoo Research Webscope database. The database provides two files namely Yahoo! Movies User Ratings and Yahoo! Descriptive Content Information, v.l.O. The Yahoo! Movies Users Ratings file contains 211231 records and consist of User ID, Movie ID and Ratings. The Yahoo! Movies Descriptive Content Information file contains 54058 records and consists Movie ID, Title, Genre, Directors, Actors and so on.

B. Data Cleaning

The Movies Descriptive Content Information file contained about 40 columns. Most of these columns were not required for our experiments and hence were removed. The dataset also contained a lot of blank values and duplicate values which needed to be resolved. In addition, there were some entries for movies in the Movies Users Ratings files that didn't correspond to any movie in the Movies Descriptive Content Information file. These entries were removed for easier processing.

C. Data Analysis

We analyzed the dataset to gain insight into the movie dataset that could help in developing our system using the Matplotlib libraries in Python. We identified patterns such as most rated movies, most rated genres, the number of movies in each genre and the number of movies rated in each rating category as shown below.



Figurc.2 Most Rated Movies

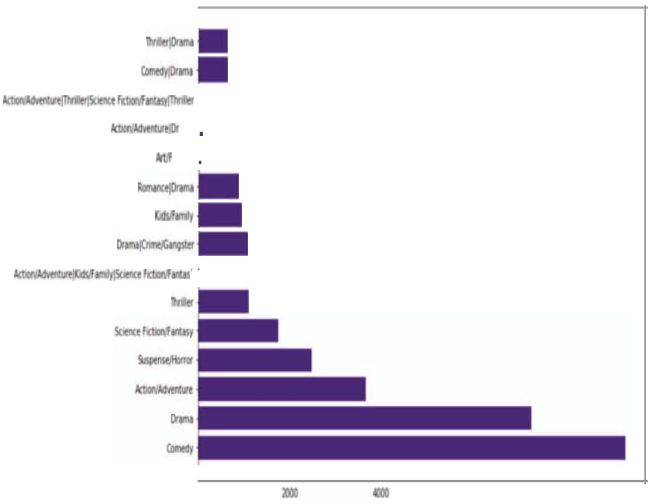


Figure.3 Most Rated Genres

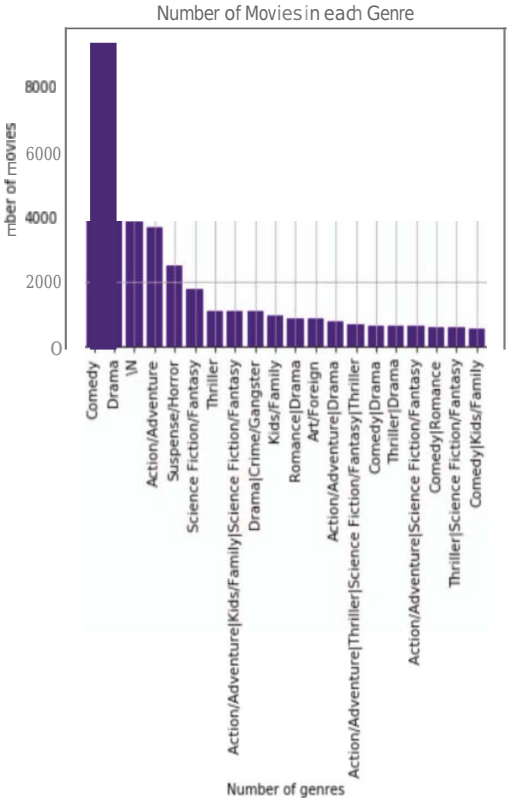


Figure.4 Number of Movies in Each Genre

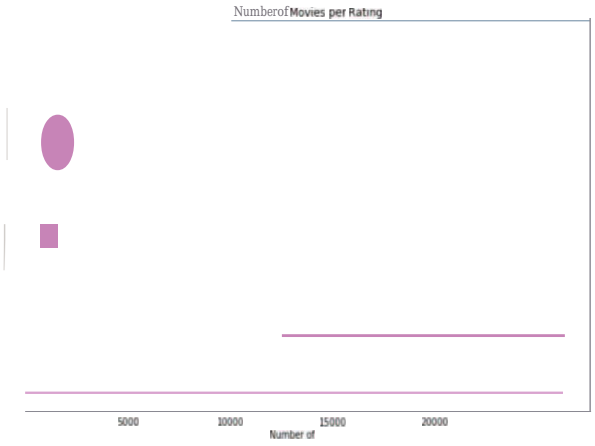


Figure.5 Number of Movies in Each Rating Category

D. *Model Building*

We used Mahout library to build the recommender system. For User based filtering, we used the *UserSimilarity* class in addition to the *PearsonCorrelationSimilarity* which uses the Pearson Correlation Coefficient to determine the similarity between users' ratings; hence the preference.

Figure.6 provides the mathematical formula for the Pearson Correlation. The higher the coefficient is, the more correlated the two users' choices are.

$$r = \frac{\sum (x-\bar{x})(y-\bar{y})}{\sqrt{\sum (x-\bar{x})^2 \sum (y-\bar{y})^2}}$$

Figure.6 Pearson Correlation Coefficient formula

The *UserNeighborhood* is computed by using another machine learning algorithm of distance-based clustering called, *NearestNUserNeighborhood* where N is defined in the program code. Nearest Neighbor algorithm searches the *N* nearest data-points around each data-point to get the most similar data-points and group them together.

The item-based filtering is implemented using the *ItemSimilarity* class of Mahout. The machine learning algorithm used to compute the item similarity is *LogLikelihoodSimilarity.* Since the items are static, their similarities based on the user ratings are not going to change over time, we can precompute them and store them offline.

The results from the Item Based recommender is loaded to the Hadoop Distributed File System (HDFS) to have a scalable and fault-resistant storage. The User Based recommender results have to be computed every time during a recommendation since unlike items, ratings provided by users.

V. MODEL EVALUATION

A. *Qualitative Evaluation*

The movie recommender system built in this paper facilitates the understanding of how a recommender system works. To evaluate the accuracy and relevancy of the results produced by our system, we analyze both the approaches differently.

Movie 1	Movie 2	Similarity
1800421139	1800379216	0.99959636
1800061638	1800111258	0.99959064
1800121659	1800379216	0.99955463
1807537463	1804738128	0.9995 903
8028 3191	1807858489	0.9995346
1800121659	1800111258	0.9995051
1800061638	1800121659	0.999477 5
1800421139	1800121659	0.99944425
1800111258	1800379216	0.99939984
1800421139	1800111258	0.99938726
1800061638	1800379216	0.9993557
1800421139	1800061638	0.999335
1800080788	1800080795	0.9992829
180743259	1807428853	0.9992593

Figure.7 Raw Output from Item Based Recommender

We compare the Item based similarity coefficient results as given in Figure.7 by mapping the MovieID of Movie 1 and Movie 2 to their titles. Using Python *pandas* libraries, we get the result as shown in Figure.8. As evident from the table, movies which are similar are given a higher similarity metric.

The lord of the Rings: The Fellowship of t e Ring 12(01)	The lo rd of th e Rings: The Two Towe rs 12(02)	0.999549
The Empire Strikes Back (1980)	Star Wars (1977)	0.9994775
I diana Jones and the la st Crusade (19B9)	I diana Jones and t e Temple of Doom (1984)	0.9992829
E.T. The Extr. -Terrestrl.1 (1982)	S .rW.n (1977)	0.9990453
The Godf. ther (1972)	The Godfat he r Part II (1974)	0.9989032
Pirates olthe Caribbean: The Curse of the Bl. ...	The Leag e of Extraordinary Gentlemen	0.99869
The M. ul. Reloaded (200 3)	Bruce 101 hty (200 3)	0.998663
Jeepers Creepers 2 12(03)	Freddy vs. Jasee (200 3)	0.99858
Harry Poner Andt e Cha berofSecrets 12(02)	The lord of th e Rings: The Fellowship olthe	0.998 73
Signs 12(02)	Shre (2001)	0.998355
2 Fast H rlous (200 3)	Bruce Al ighty 12(03)	0.998 32
H.rry Pott er Andt e Charnberef Secrets 12(02)	Shre (2001)	0.998276
The Texas Chains aw Massacre (200 3)	Scary Movie 3 12(03)	0.998155
The Leag e of Extraordinary Gentle en (200 3)	Bad Boys II(200 3)	0.998154
Ice Age (200 2)	S re 12(01)	0.99796
ill Bill Vol. 1 (200 3)	Underw orld (200 3)	0.997915
Austin Powers In Goldmemb er (200 2)	Signs 12(02)	0.997732
D. ddy D.y Care (200 3)	Bruce Al ighty (200 3)	0.997505
The Mu my (1999)	The Mu y Returns (200 1)	0.997426
2 Fast 2 F rious (200 3)	The Matrix Relo.ded (200 3)	0.997' 3
IIIBill Vol. 1 12(03)	The Te.as Ch.ins. Massacre (200 3)	0.997349
The F.st and the F rious 12(01)	XXX (200 2)	0.996123
How to De.1 12(03)	B.d Boys III2(03)	0.99 458
Down wit love (200 3)	owtolose a Guy; 10 D. ys (200 3)	0.992708
IThe Scorpion ing (200 2)	The Mummy Returns (200 1)	0.99107
Der alled (2002)	The Order{2(0 1)	0.9B559
Fr. n Sinatra - 3 P.ck(200 2)	Great Music. ls - Vol. 2 (1951	0.952113
Biogr.p y: Bette D.vis (1926)	The Bette Davis Collection (1993)	0.952113
Someon e like Yo (1991)	Someon e like You Where t e Heart 1512(02)	0.94 782
Be. uty a dthe Beast 12000 1	Be. uty and the Beast 119621	0.9 1396

Figure.8 Movies Similarities (Item Based)

For user-based recommender system, we evaluate the model using the *AverageAbsoluteDifferenceRecommenderEvaluator.* We divide the training data into test and train samples. Next, we evaluate the rating predictions on test data against the actual ratings as specified in the training data.

Figure.9 shows the raw output from the user based filtering technique. The system recommends 10 movies to user (User 5) and returns the nearest neighbors which have most similar taste preference as him. For each movie recommended, it also predicts the ratings by that user (User 5). We get an average absolute difference of 0 which proves that the predictions

made on the ratings of the recommended items are 100% accurate. Figure.lO shows the table of recommended movies with the movie title and predicted ratings.

```
----lG User Based Recommendations for User 5-----
RecommendedItem[item:18G4738128, value:5.GJ
RecommendedItem[item:18G8481189, value:5.GJ
RecommendedItem[item:18G7537463, value:5.GJ
RecommendedItem[item:18G8414381, value:5.GJ
RecommendedItem[item:18G84G3G3G, value:5.GJ
RecommendedItem[item:18G7432594, value:4.6666665J
RecommendedItem[item:18G84G4459, value:4.5J
RecommendedItem[item:18G84G4659, value:4.5J
RecommendedItem[item:18G84G6133, value:4.5J
RecommendedItem[item:18G84G4742, value:4.5J
-----4 Users similar to User 5-----
58
69
156
236
Average Absolute Difference= 0.00
```

Figure.9 Raw Output from User Based Recommender

The Lord of the Rings: The Two Towers (2002)	5
Freaky Friday (2003)	5
The Lord of the Rings: The Fellowship of the Ring (2001)	5
Bad Boys II (2003)	5
How to Lose a GUy in 10 Days (2003)	4.5

Figure.10 Recommended movies and the predicted rating score (User Based)

B. *Common Issues Evaluation*

With recommender systems, the following issues are always raised. We evaluate our system based on these issues and propose implementation design to resolve them.

Firstly, the *New User Problem* is concerned with the scenario when a new user is added to the recommender system. There are no ratings provided by him/her for any movie in the system yet. This is also called a *User Cold Start.* One simple possible resolution can be to recommend top rated movies or most recently added movies to this new user.

Secondly, there is a concern that no new item is recommended. It can be understood as an *Item Cold Start* problem. When a new movie is added to the system, it does not have any ratings associated with it. How can it be discovered and recommended? One resolution can be to recommend movies similar to the genre of top rated movies. If the new movie falls in that genre, it will get discovered. However, in this resolution, we will need to build a system based on the genre of the movies.

Keeping these known issues apart, either of the two approaches can be used based on the infrastructure. For item-based, the similarity computation is large of the order MxM (M: total movies) but since it is static, we can do it offline and re-compute it only after some threshold time. On the other hand, expensive to do at runtime as for each user, neighborhood is dynamic with new ratings from different users. Therefore, the former requires cache data storage, latter requires a dedicated processing server.

VI. CONCLUSION AND FUTURE WORK

In this paper we have implemented a movie recommendation system using collaborative filtering. This system is developed using Apache Mahout and takes the ratings given to movies into consideration to provide movie suggestions.

For future work, the recommender system could be developed using hybrid filtering approach instead of collaborative. Recent research indicates that hybrid systems are found to be more effective and provide more accurate recommendations. Hence, hybrid systems would be an improvement. Our system considers the user ratings to recommend movies. In the future, more features such as the genre of the movie, the directors, the actors and so on could be considered as well to provide suggestions. In addition, a new framework called Apache Prediction 10 could be looked into to develop the system instead of Mahout. The Apache Prediction 10 is a machine learning server that uses the technology stack of Apache Hadoop, Apache spark, Elastic Search and Apache Hbase to build Universal Recommender System.

REFERENCES

[1] A. V. Dev and A. Mohan, "*Recommendation system for big data applications based on set similarity of user preferences,*" *2016* International Conference on Next Generation Intelligent Systems (ICNGIS), Kottayam, 2016, pp. 1-6. doi: 1O.1109/ICNGIS.2016.7854058

[2] Kumar, Manoj & Yadav, D.K. & Singh, Ankur & Kr, Vijay, "*A Movie Recommender System: MOVREC*", 2015 International Journal of Computer Applications. 124. 7-11. 10.5120/ijca2015904111

[3] S. G Walunj, K Sadafale, "*An online recommendation system for e-commerce based on Apache Mahout framework*", 2017 ACM SIGMIS International Conference on Computers and People Research, pp. 153-158,2013.

[4] H. W. Chen, Y. L. Wu, M. K. Hor and C. Y. Tang, "*Fully content-based movie recommender system with feature extraction using neural network,*" 2017 International Conference on Machine Learning and Cybernetics (ICMLC), Ningbo, China, 2017, pp. 504-509. doi: 10.1109/ICMLC.2017.8108968

[5] Z. D Zhao, M. S Shang, "*User-Based collaborative filtering recommendation algorithms on Hadoop*", Proc. of Third International Workshop on Knowledge Discovery and Data Mining, pp. 478-481, 2016.

[6] B. Sarwar, G. Karypis, 1. Konstan, 1. Riedl, "*Item-based collaborative filtering recommendation algorithms*", Proceedings of the 10th international conference on World Wide Web, pp. 285-295, 2001

[7] Koen Verstrepen, Bart Goethals, "*Unifying nearest neighbors collaborative filtering*", Proceedings of the 8th ACM Conference on Recommender systems, October 06-10,2014, Foster City, Silicon Valley, California, USA doi: 10.1145/2645710.2645731

[8] Jain, A., & Vishwakarma, S. K., "*Collaborative Filtering for Movie Recommendation using RapidMiner*" International Journal of Computer Applications (0975 - 8887) Volume 169 - No.6, July 2017

[9] M. Gardener, "Statistics for Ecologists Using R and Excel (Edition 2) ". [Online]. Available: http://www.dataanalytics.org.uk [Accessed: 05-May-2018]

[10] Community Shared, "Apache Mahout". [Online]. Available: https://mahout.apache.org/ [Accessed: 02-April-2018]