COMPARISON OF COLLABORATIVE FILTERING ALGORITHMS WITH VARIOUS SIMILARITY MEASURES FOR MOVIE RECOMMENDATION

Taner Arsan, Efecan Köksal, Zeki Bozkuş

Department of Computer Engineering, Kadir Has University, Istanbul, Turkey

ABSTRACT

Collaborative Filtering is generally used as a recommender system. There is enormous growth in the amount of data in web. These recommender systems help users to select products on the web, which is the most suitable for them. Collaborative filtering-systems collect user's previous information about an item such as movies, music, ideas, and so on. For recommending the best item, there are many algorithms, which are based on different approaches. The most known algorithms are User-based and Item-based algorithms. Experiments show that Item-based algorithms give better results than User-based algorithms. The aim of this paper isto compare User-based and Item-based Collaborative Filtering Algorithms with many different similarity indexes with their accuracy and performance. We provide an approach to determine the best algorithm, which give the most accurate recommendation by using statistical accuracy metrics. The results are compared the User-based and Item-based algorithms with movie recommendation data set.

KEYWORDS

Collaborative Filtering, Recommendation Systems, User-based Algorithms, Item-based Algorithms

1. INTRODUCTION

Collaborative Filtering (CF) is became most popular method for decreasing information conflicts. Works Collaborative filtering is working like creating a database of preferences for users and items. The system has significant success on the Internet and most big companies use CF. The idea under this paper is about selecting right information to the right user in the given database. Automated collaborative filtering systems aim that finding users who that the same tastes or information according to the specific purpose. To build the database, users share information or preferences with the system so the system can decide better choices for the other users. To achieve that users should give their feedback truly [1, 2].

In this paper, database of Collaborative Filtering System includes the data of users and the movies as shown in Table 1.

Table 1. Collaborative filtering System is about prediction of missing rate in User-Item matrix. Prediction for theNathan's rate for Titanic.

	Star Wars	Hoop Dreams	Contact	Titanic
Joe	5	2	5	4
John	2	5		3
Al	2	2	4	2
Nathan	5	1	5	?

Collaborative filtering algorithms are divided into two different recommender systems that are User-based recommender system and Item-based recommender system as shown in Figure 1 and Figure 2 respectively.

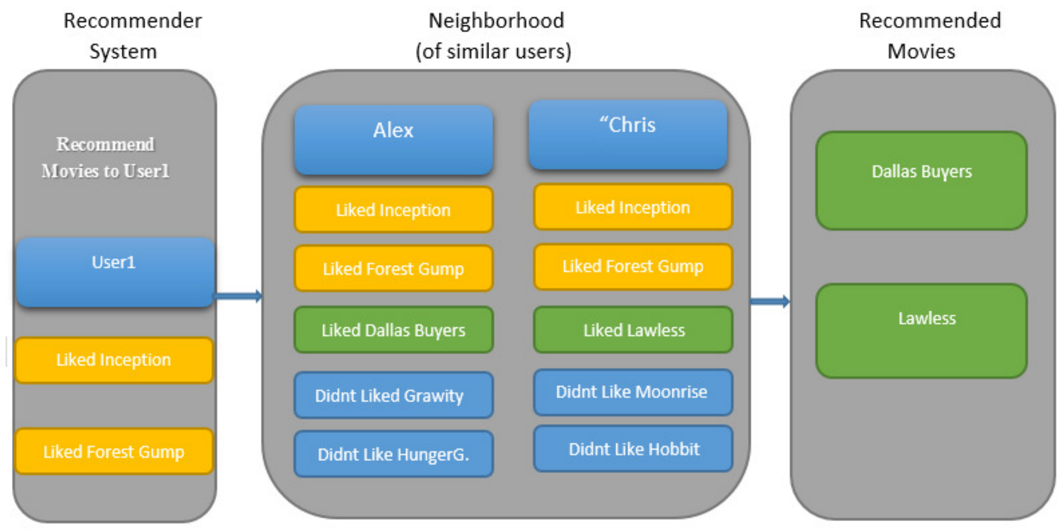


Figure 1. Collaborative filtering Systems applied in User-based Recommender System.

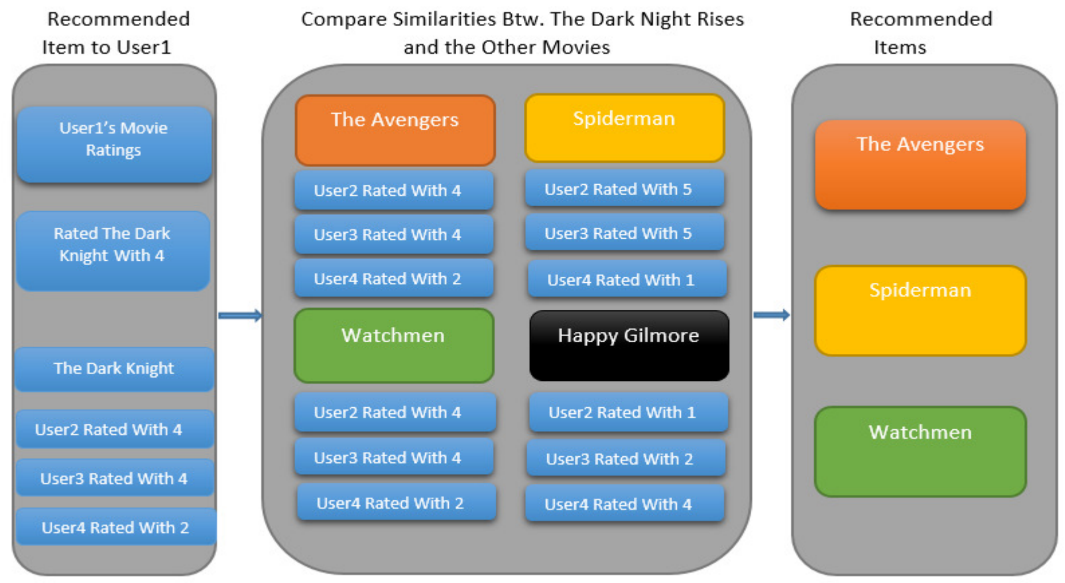


Figure 2. Collaborative filtering Systems applied in Item-based Recommender System.

2. Framework

2.1. Introduction to Apache Mahout

In this paper, Apache Mahout is used as an implementation framework, whichallows developers to generate strong and scalable recommender.These are publicly available sources machine learning library. Apache Lucane Project pioneer to start Mahout as a by-product in 2008.Mahout principally provides in content search and technologies of receiving information. When the collection of data is too large, Mahout intends became the first choice as a library for collaborative filtering. Mahout is coded as Java programming Language. Mahout doesn't supply a user interfaces or installer. After coding part it is the job to the developer to complete interfaces for the algorithm. The Mahout library includes a lot of recommender systems. This studyalso discusses how Mahout has adapted the User-based Recommender Systems and Item-based Recommender Systems [3, 4].

2.1.1. Further Subsections

To build up the inputs for the purpose of the paper, first datasets need to be converted to csv extension file. This file consists of some data, which are User ID, Item ID and the given preferences (rates).Ids in Mahout are always number (integer) and the preference has the property that is the larger number is positive strong preferences. According to the Movie Lens data sets, these preferences are between 1 and 5 as an integer. After converting data file to the csv file, first column shows user id, second column shows item id and the last column shows the rates [4].

2.1.2 Recommender Input File, Intro Csv

Csv file is shows the numbers separating with commas. To be more clear the table shows which column shows which identity [3,4]. Table 2 shows what information includes csv file.

Table 2.Information includedby csv file

User ID	Movie ID	Rates
1	102	3
2	35	2
2	75	5
91	102	3
101	54	3
101	102	4

The following codes are how u.data can be converted to csv file in Java

```
publicclass MovieDataConvert {
        publicstaticvoid main(String[] args) throws IOException {
                BufferedReader br = new BufferedReader(new
FileReader("data/u.data"));
                BufferedWriter bw = new BufferedWriter(new
FileWriter("data/movies.csv"));
                String line;
                while((line = br.readLine()) != null) {
                        System.out.println(line);
                        String[] values = line.split("\\t", -1);
                        bw.write(values[0] + "," + values[1] + "," + values[2] +
"\n");
                }
                br.close();
                bw.close();
        }
}
```

FileReader: Creates a new FileReader, given the name of the file to read from

FileWriter: Constructs a FileWriter object given a file name.
BufferedReader: Creates a buffering character-input stream that uses a default-sized input buffer.
BufferedWriter: Writes text to a character-output stream, buffering characters so as to provide for the efficient writing of single characters, arrays, and strings.
readLine: Reads a line of text. A line is considered to be terminated by any one of a line feed ('\n'), a carriage return ('\r'), or a carriage return followed immediately by a linefeed.

2.1.3 Creating a Recommender

A little piece of code given as an example of how to create recommendation to users [3, 4].

```
class UserBasedPearsonCorrelationSimilarity {
Public static void main(String[]args) throws Exception{
DataModel model = new FileDataModel(new File("data/movies.csv"));
 ←Load Data files
UserSimilarity similarity = new PearsonCorrelationSimilarity(model);
        UserNeighborhood neighborhood = new
NearestNUserNeighborhood(50,
                                    similarity, model);
        Recommender recommender = new
GenericUserBasedRecommender(model,
                                    neighborhood, similarity);
←Create Recommender Engine
HashMap<String,
String>getMovieNameById=MovieItemConvert.getGetMovieNameById();
                 List<RecommendedItem>recommendations =
recommender.recommend(2,5);
← For User  2,Recommend  5 items
for (RecommendedItem recommendation : recommendations) {
System.out.println(recommendation);
        }
  }
}
```

With *DataModel* the program can reach all preferences which are user and item data and rates.

With *UserSimilarity* the program can find how similar the users.

With *UserNeighborhood* the program can find the most similar user for the selected user.

With *Recommender* the program can recommend items to the users.

2.1.4Analyzing the Output

When developer runs the code, output of this code should be like

```
RecommendedItem[Item:106, value:4.96451]

RecommendedItem[Item:205, value:4.36231]

RecommendedItem[Item:100, value:4.26752]

RecommendedItem[Item:12, value:4.13121]

RecommendedItem[Item:502, value:4.01531]
```

The components of User-based recommendation in Mahout is given in Figure 3. This figure also shows interaction of the components.

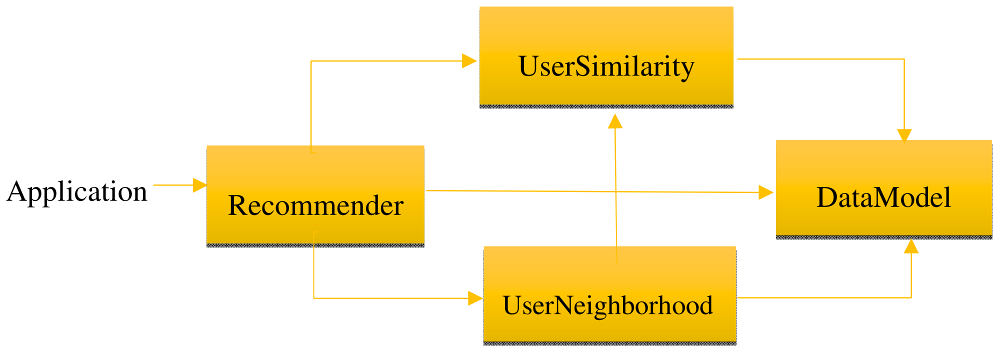


Figure 3. Interaction of components in Mahout User-based recommendation.

3. SIMILARITY MEASURES

Recommender systems contain many similarity metrics that come from machine learning. They are important for recommender systems. Each similarity metrics are related with vector space methods; but there are various ways for defining the similarity. They can be categorized in a way that distance and degree measurement. There are different similarity calculation techniques for computing similarity between users. Since each similarity have different formulas, they give different measures from each other. Some similarity computation techniques are explained in the following sub headings [5].

In the Collaborative Filtering Systems, there is a mutual point that is establishment of similarity between users and items. The Mahout library has concerted a lot of similarity algorithms and gives permission to the developers for integrating them into collaborative Filtering Recommender Systems for the purpose of clarifying similar neighborhoods to the users or computing similarities between items. Mahout has concerted similarity algorithms, which are,

1. Euclidean Distance Similarity
2. Log Likelihood Ratio Similarity algorithms
3. Pearson Correlation Coefficient Similarity
4. Tanimoto Coefficient Similarity
5. Uncentered Cosine Similarity
6. Spearman Correlation Coefficient Similarity

3.1 Euclidean Distance Similarity

In the code implementing EuclideanDistanceSimilarity (model) to UserSimilarity will work for this method. The method based on distance between users.

This method is working as users is a point in many items. The table has the rates of the each user to the each item. This metric converts Euclidean distance d between 2 such users. Distance value is smaller when these users are more similar. This method gives the value of $1/(1+d)$.It never gives negative value as a similarity and when the value increases it means that they are more similar [3].

The equation is given in (1) as

$$r_2\,(\mathrm{x},\mathrm{y})=\sqrt{(x_1-y_1)^2+(x_2-y_2)^2}=\sqrt{\sum_{i=1}^{2}(x_i-y_i)^2} \qquad (1)$$

Table 3. Similarities between user 1 and the other users.

Euclidean	Item 1	Item 2	Item 3	Distance	Similarity to User 1
User 1	5.0	3.0	2.0	0.000	1.000
User 2	3.0	2.0	5.0	3.937	0.203
User 3	2.0	-	-	2.500	0.286
User 4	5.0	-	3.0	0.500	0.667
User 5	4.0	3.0	2.0	1.118	0.472

In the code implementing EuclideanDistanceSimilarity (model) to ItemSimilarity will work for this method.

As shown in Table 3, this method compares rates of the items for one item not for one user to items. Item similarity gives better results because user based similarity affected by mood of user or tastes of user can change over time. Item similarities are more fixed and better for precomputation. It speeds up computation as runtime.

3.2 Log Likelihood Similarity

In the "Accurate Methods for the Statistics of Surprise and Coincidence" paper Ted Dunning created Log Likelihood Ratio Similarity. Log Likelihood similarity is similar to Tanimoto similarity, but it is more complex to understand. It can explain with Math and it doesnot take individual preference. The value gives how unlikely the user to have so much conflicts and also it is based on total number of items out and total number of each user has preferences. It means to dissimilar user will have some common items, but two similar user will conflict. For example, if two users have 4 preferences in common, but have both only taken 10 preferences into the data model, they will be considered more similar than two users who have 4 preferences in common but have both taken over 50 preferences into the data model[3]. Table 4 shows the similarity between users according to the Log Likelihood Similarity Measurement.

Table 4. Example for Log Likelihood Similarity Measurement.

	Item1	Item2	Item3	Item4	Item5	Item6	Item7	Similarity to the user1
User 1	X	X	X					0.90
User 2	X	X	X	X				0.84
User 3								0.55
User 4	X			X	X		X	0.16
User 5	X	X	X	X	X	X		0.55

3.3 Pearson Correlation Similarity

It is used for converting similarity between two users or items by measuring obliquity of two series of preferences to act together in a comparative and linear manner. It considers preferences of conflicting users and items. It tries to find each users' or items' derivations from their average rates while recognizing linear adjustment between two items or users.

$$\text{P, C(w,u)}=\frac{\sum_i (r_{w,i}-\bar{r}_w)(r_{u,i}-\bar{r}_u)}{\sqrt{\sum_i (r_{w,i}-\bar{r}_w)^2 \sum_i (r_{u,i}-\bar{r}_u)^2}} \quad (2)$$

w and u shows the two users or items for which the coefficient is calculated, i is an item, $r_{w,i}$and $r_{u,i}$ are individual ratings from w and u for i, and average ratings of $\bar{r}_w$and $\bar{r}_u$are ,for user (or item) w and u [3, 4]. Table 5 shows Pearson Correlation Similarity of user1 and the others based on three items common.

Table 5.Pearson Correlation Similarity.

	Item1	Item2	Item3	Correlation with user1
User 1	5.0	3.0	2.0	1.000
User 2	2.0	2.0	5.0	-0.764
User 3	2.0	-	-	-
User 4	5.0	-	3.0	1.000
User 5	4.0	3.0	2.0	0.945

3.4 Tanimoto Coefficient Similarity

As shown in Figure 4 and Table 6, this is a similarity that ignores the preference values so that it does focus on the value that the user given for the item. It only checks that the user expressed a preference or not. It is also known as Jaccard coefficient. Its formula is the number of items that both users showed their interest, divided by the number of items that either usershows some interest. When they do not have any similar preference, the result will be zero. The similarity value cannot be greater than one [4]. The equation for Tanimoto Coefficient Similarity is given in (3):

$$T(A,B)=\frac{\text{A.B}}{|\text{A}|^2+|B|^2-A.B}=\frac{\sum_{i=1}^{n} A_i x B_i}{\sum_{i=1}^{n} A_i^2+\sum_{i=1}^{n} B_i^2-\sum_{i=1}^{n} A_i x B_i} \quad (3)$$

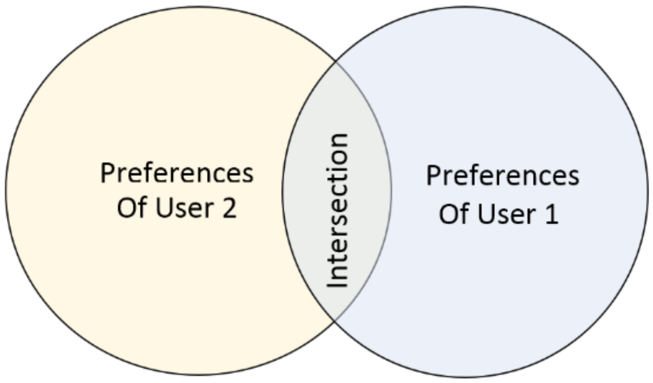


Figure 4. The Tanimoto coefficient is the ratio of the intersection that means both users express their feelings about the same items, to the union of the users preferred items.

Table 6. By using Tanimoto Coefficient similarity, the similarity values are calculated between user one and the other users.

	Item 1	**Item 2**	**Item 3**	**Item 4**	**Item 5**	**Item 6**	**Item 7**	**Similarity to User 1**
User 1	√	√	√					**1.0**
User 2	√	√	√	√				**0.75**
User 3	√			√	√		√	**1.17**
User 4	√		√	√		√		**0.4**
User 5	√	√	√	√	√	√		**0.5**

3.5 Uncentered Cosine Similarity

It is a similarity that measures cosine of the angle created from the two vectors in the coordinate system. The result changes from -1 and 1. This similarity does not center the data, it moves the user's preference values, it makes their means is 0. Also it does not adjust the preference values, therefore it is called uncentered cosine similarity. The equation for Uncentered Cosine Similarity is given in equation (4):

$$Cos(\theta) = \frac{\sum_{i=1}^{n} A_i x B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \tag{4}$$

3.6 Spearman Correlation Similarity

As shown in Table 7, Spearman Correlation Similarity is similar to Pearson Correlation Similarity but instead of preference values, it uses ranks. For each user, the preference item's preference values are ordered from the least-preferred to the most-preferred. Then this value modify with starting from 1. Now if Pearson correlation is computed with these values, it will give the Spearman Correlation Similarity. This similarity is better for smaller data sets because computing and storing the ranks take long time [5, 6, 7]. The equation for Spearman Correlation Similarity is given in equation (5):

$$\omega_{(a,u)} = \frac{\sum_{i=1}^{n}(rank_{(a,i)} - \overline{rank_a}) * (rank_{(u,i)} - \overline{rank_u})}{\sigma_a * \sigma_u} \tag{5}$$

Table 7. After changing the values of the preferences into the ranks, the results are found by using (2.5).

	Item 101	Item 102	Item 103	Correlation to User 1
User 1	3.0	2.0	1.0	1.0
User 2	1.0	2.0	3.0	-1.0
User 3	1.0	-	-	-
User 4	2.0	-	1.0	1.0
User 5	3.0	2.0	1.0	1.0

4. IMPLEMENTATION

Item based collaborative Filtering Algorithm is chosen for this part of the paper. To recommend something to the user Adjusted Cosine Similarity Method is chosen.

4.1 Adjusted Cosine Similarity

The difference between the User Based Collaborative Filtering and the Item Based Collaborative Filtering is that User based takes the rows and Item based takes the columns for similarity measurement. Basic Cosine similarity computation has important disadvantage which is rating scale between different users are ignored. By subtracting the selected user mean from every co-rated place, Adjusted Cosine Similarity takes advantage. The equation for Adjusted Cosine Similarity is given in equation (6),

$$\text{Sim(i,j)}=\frac{\sum_{u\in U}(R_{u,i}-\overline{R_u})(R_{u,j}-\bar{R}_u)}{\sqrt{\sum_{u\in U}(R_{u,i}-\overline{R_u})^2}\sqrt{\sum_{u\in U}(R_{u,j}-\bar{R}_u)^2}} \qquad (6)$$

This equation shows similarity between i and j items, and $\bar{R}_u$is average of the ratings of the uth user.

4.2 Prediction Computation

The significant part of the collaborative filtering system is output of the recommendation. The items found with adjusted cosine similarity will be used for target users rates and finally recommend something to the user.

4.2.1 Weighted Sum

For item *i* to *useru*, finding sum of the rates to user u on the items similar to *i*this method is used. Every rates are weighted by adjusted cosine similarity $Sim_{i,j}$ between items *i* and *j*. Weighted Sum equation is

$$P_{u,i}=\frac{\sum_{all\ similar\ items,N}(S_{i.N}*R_{u,N})}{\sum_{all\ similar\ items,N}(|S_{i,N}|)} \qquad (7)$$

This equation simply achieves to find how the active user rates the similar items.

4.3 Creating Database

First ‘Movierecommender’ database is created. Then tables has to be created.

Movies table added and this table includes data of id, movie titles, movie genres are available. Rates table are built. This table includes data of id, movie, user_ and rates. Also in movie and user_section primary keys are added so conflicts are blocked.

Finally, Users table added and this table includes information about user id, name, last name, age, sex and email. Also into the email part primary key added. So when users come to the system each user can enter the system with their own email addresses.

Flowchart of the Proposed Algorithm is given in Figure 5. On the other hand, flowchart of Mahout Library used in algorithm is given in Figure 6.

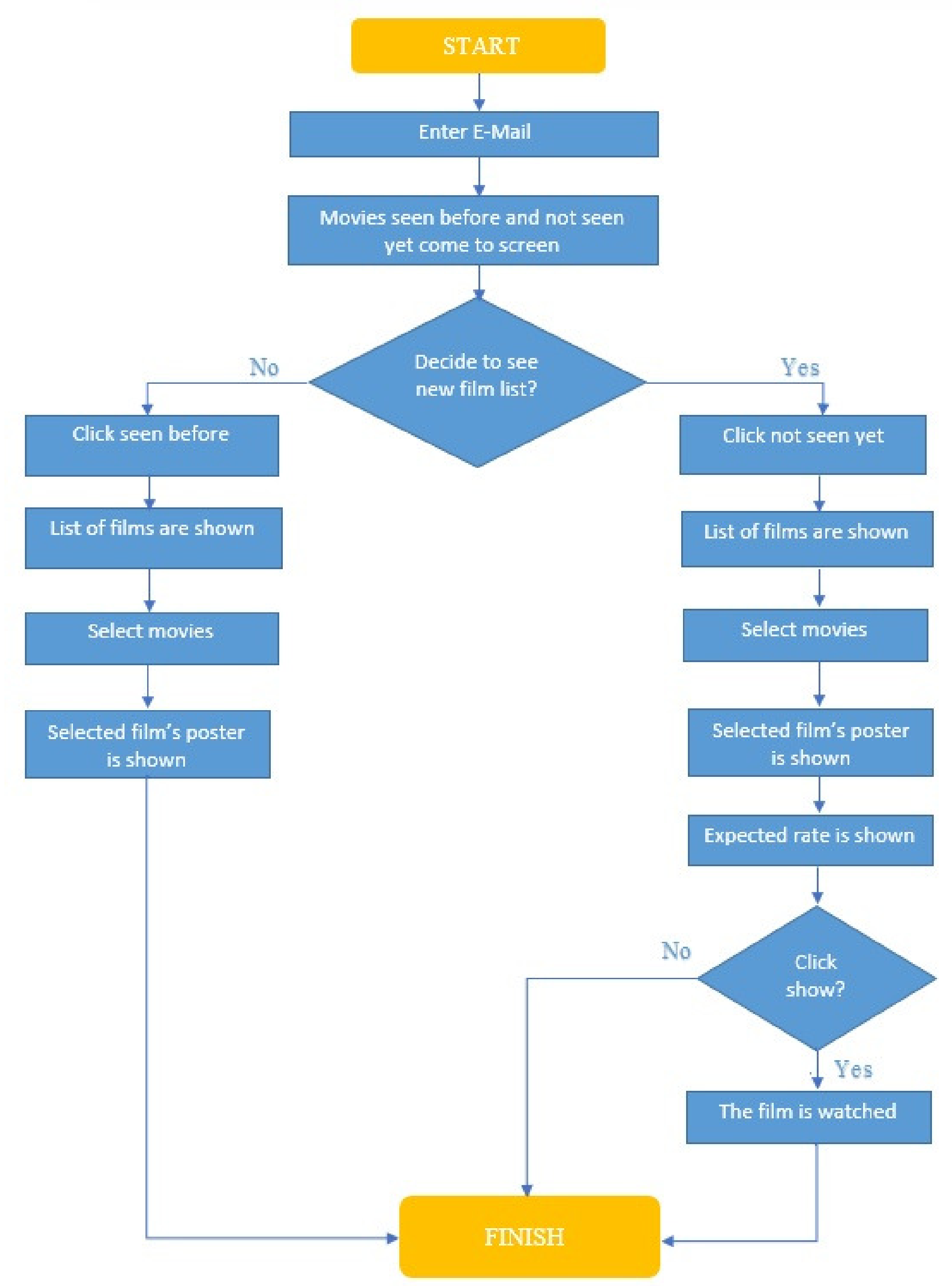


Figure 5. Flowchart of the Proposed Algorithm.

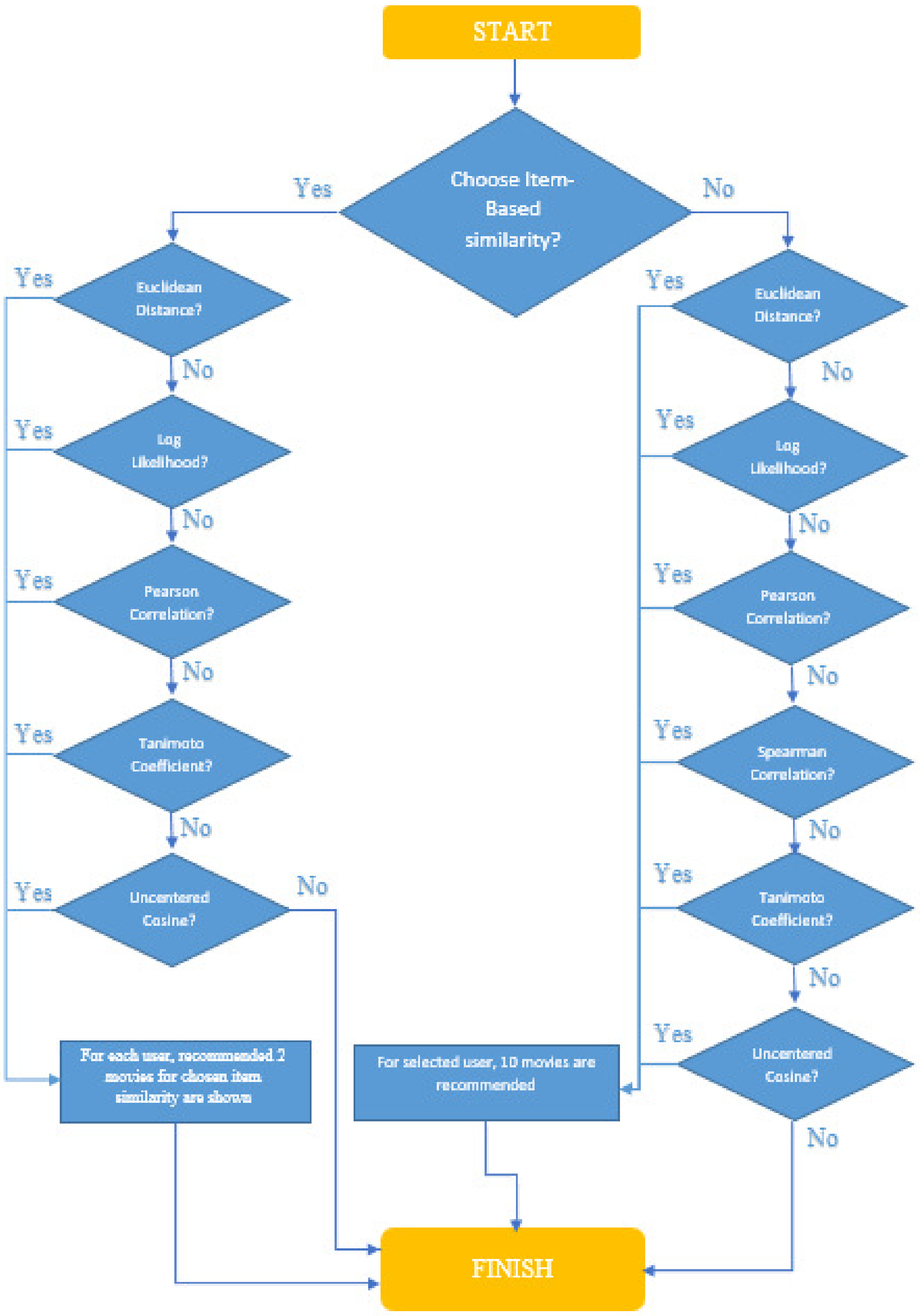

START
Choose Item-Based similarity?
Yes
No
Euclidean Distance?
Log Likelihood?
Pearson Correlation?
Tanimoto Coefficient?
Uncentered Cosine?
Spearman Correlation?
For each user, recommended 2 movies for chosen item similarity are shown
For selected user, 10 movies are recommended
FINISH

Figure 6. Flowchart of Mahout Library Used Algorithm.

5. Experimental Evaluation

5.1 Data Set

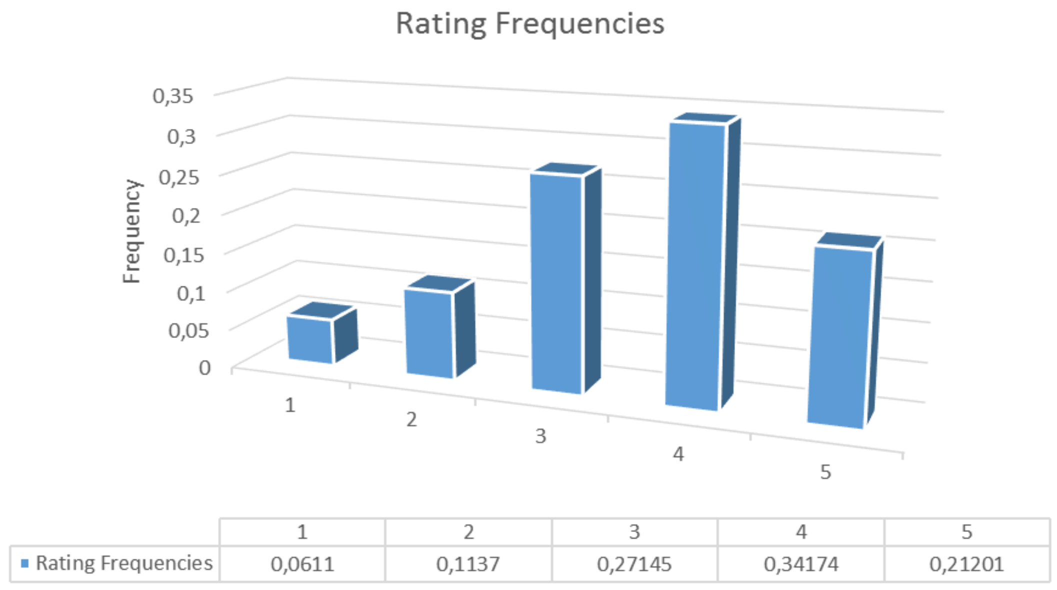
In this paper, algorithms applied to the MovieLens-100K data sets. It contains 100,000 ratings from 943 users on 1682 movies. All users in the data sets rated at least 20 movies. There are 2 types of data file. In u.data set, it has user id, item id, rating, timestamp sections. In the u.item data set, it contains information about movies such as movie id, movie title, release date, genres. Since movie ids are the same in the both data sets, we connected these data sets in our experiments.

5.1.1 Rating Distribution

As shown in Figure 7, the ratings in the Movie Lens data sets are integers. Ratings are between 1 and 5. Histogram is provided in the following section.

5.1.2 User and Movie Statistics

In this section, rating distributions are displayed. Mean of the ratings is calculated as 3,52986. Standard deviation of the ratings is 1,125674.



	1	2	3	4	5
▪ Rating Frequencies	0,0611	0,1137	0,27145	0,34174	0,21201

Figure 7. Rating frequency in the Movie Lens data sets.

5.2 Evaluation Metric

After many years of researches on Collaborative filtering algorithms, many researchers found different evaluation metrics in order to evaluate the quality of the prediction. Prediction accuracy metrics find values that show how much the prediction is close to the real preference. There are many prediction accuracy metrics are used by researchers for testing the prediction accuracy of their used algorithms, are Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) which are also implemented in Mahout. We selected Mean Absolute Error and Root Mean Squared Error as our choice of evaluation metrics for showing our experiment results.

5.2.1 Mean Absolute Error

This evaluation metric evaluates accuracy of an algorithm by comparing value of predictions against the actual user's ratings for the user-item pairs in the test dataset. For each rating-prediction pair, their absolute error is calculated. After summing up these pairs and dividing them by the total number of rating-prediction pairs, Mean Absolute Error can be found. It is the most commonly used and can be interpret easily. The equation of Mean Absolute Error is given in equation (8):

$$MAE = \frac{\sum_{i=1}^{n} |p_i - r_i|}{n} \tag{8}$$

5.2.2 Root Mean Square Error

This is a statistical accuracy metric that is slightly different from Mean Absolute Error. Once rating-prediction difference is calculated, its power of 2 is taken. After summing them up and dividing them by the total number of rating-prediction pairs and taking square root of it, Root Mean Square Error can be found. Equation of Root Mean Square Error is given in equation (9):

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (p_i - r_i)^2}{n}} \tag{9}$$

Where,

$p_i$is the prediction of user i
$r_i$is the real or true rating of user i
$n$is the number of ratings-prediction pairs

By using evaluation metrics, prediction accuracy and efficiency of the collaborative filtering methods can be calculated and compared. Therefore the results will show which algorithm should be used for given datasets.

5.3 Experimental Procedure

In this section experimental procedures are explained:

5.3.1 Experimental Steps

The data set has divided into a training and test portions. In the experiments, from 0.2 to 0.9 training test ratios are used in order to calculate and compare the prediction accuracy. For each similarity measures and collaborative filtering techniques, evaluation has been coded to find Mean Absolute Error and Root Mean Square Error.

5.3.2 Experimental Platform

All our experiments were implemented by using Java programming language. All the experiments are run on windows based PC with Intel core i7 processor having a speed of 2.40 GHz and 16GB of ram.

5.3.3 Experiment Results

Experimental results of User-based and Item-based collaborative filtering techniques for creating prediction are shown. There are some parameters that have to be determined. These parameters are, the neighborhood size, training/test ratio and effects of different similarity measures. All the classes that contains evaluation metric has been run separately. Then the results have been recorded in order to compare them. By using these information, histograms have been created.

5.3.4 Experiment Results with Different Neighborhood Size

The size of the Neighbor affects the prediction quality. By changing the number of neighbors, sensitivity of neighborhood is determined. As number of neighbors' increases, the quality of prediction is also increases.

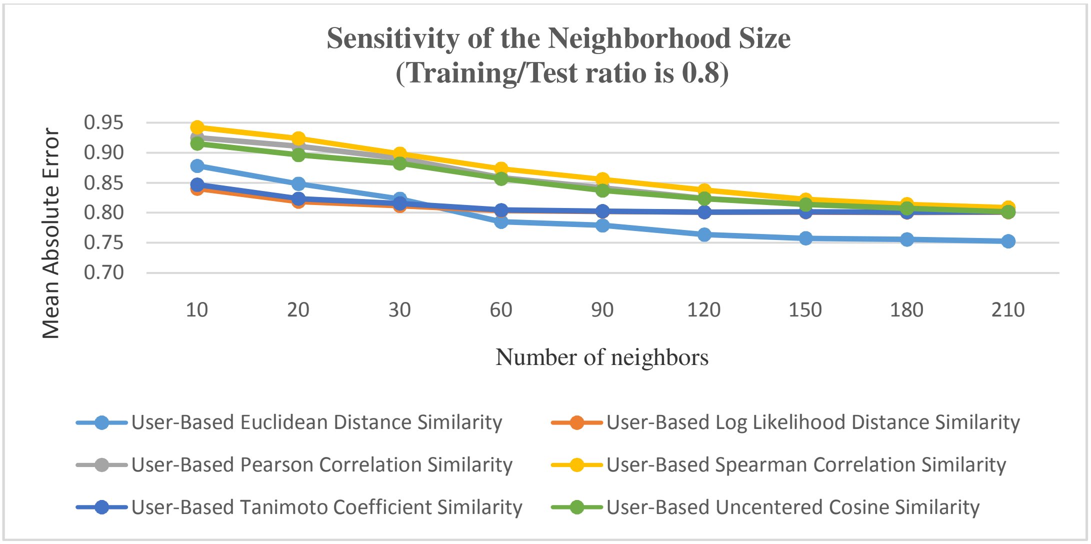


Figure 8. Mean Absolute Error for all user-based similarities as Training/Test Ratio changes.

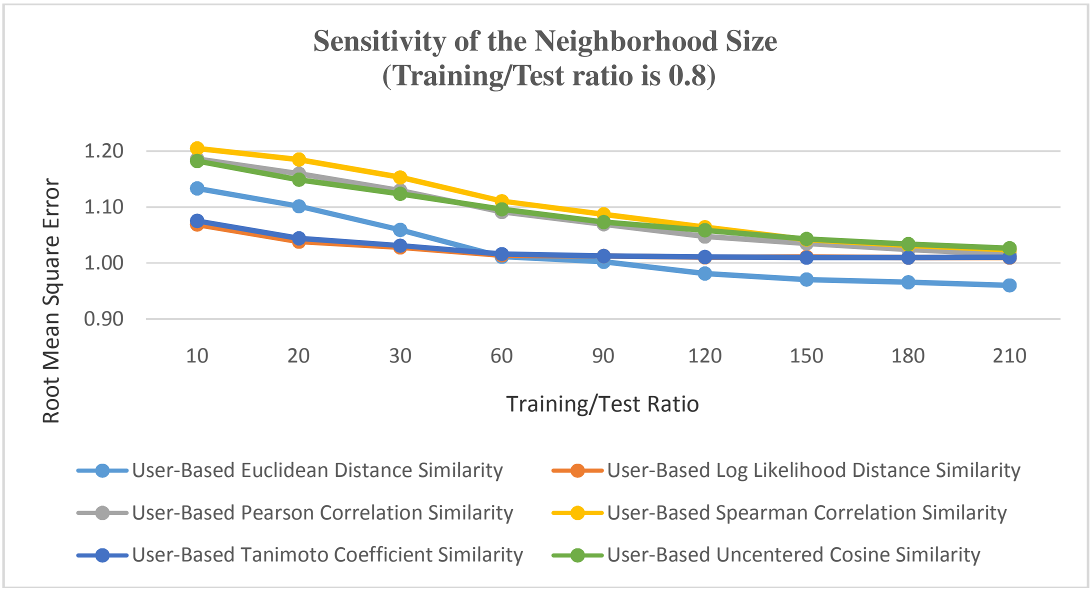


Figure 9. Root Mean Square Error for all user-based similarities as Training/Test Ratio changes.

5.3.5 Experiment Results with Different Training/Test Ratio

By changing the Training/Test Ratio, sensitivity of the Training/Test ratio is determined. For this purpose, Training/Test ratio is changed from 0.1 to 0.9 by 0.1 for all similarity metrics. The results show that the quality of the prediction is increasing as Training/Test ratio increases. Moreover, User-Based Log likelihood Distance Similarity and Item-Based Tanimoto Coefficient Similarity have the lowest Mean Absolute Error and Root Mean Square Error which means they predict better. We picked 0.8 as an optimum value for the following experiments. The results are given In Figure 10, Figure 11, Figure 12 and Figure 13 respectively.

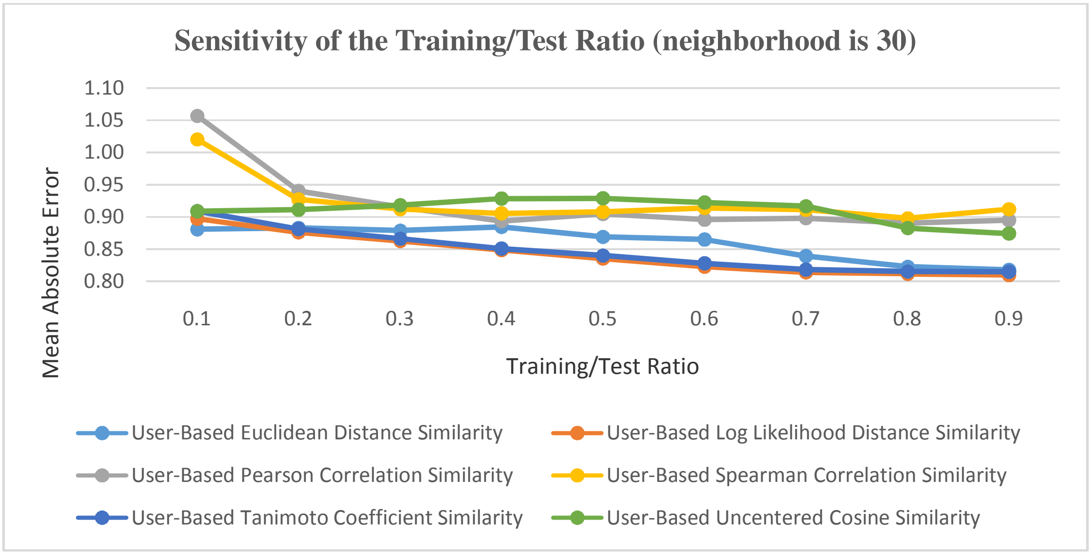


Figure 10. Mean Absolute Error for all user-based similarities as Training/Test Ratio changes.

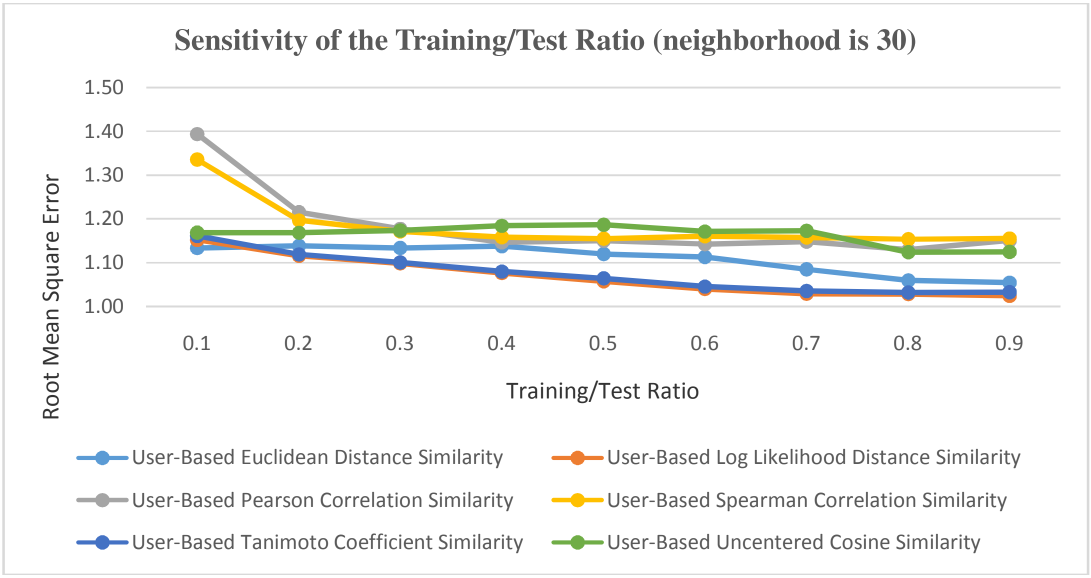


Figure 11. Root Mean Square Error for all user-based similarities as Training/Test Ratio changes.

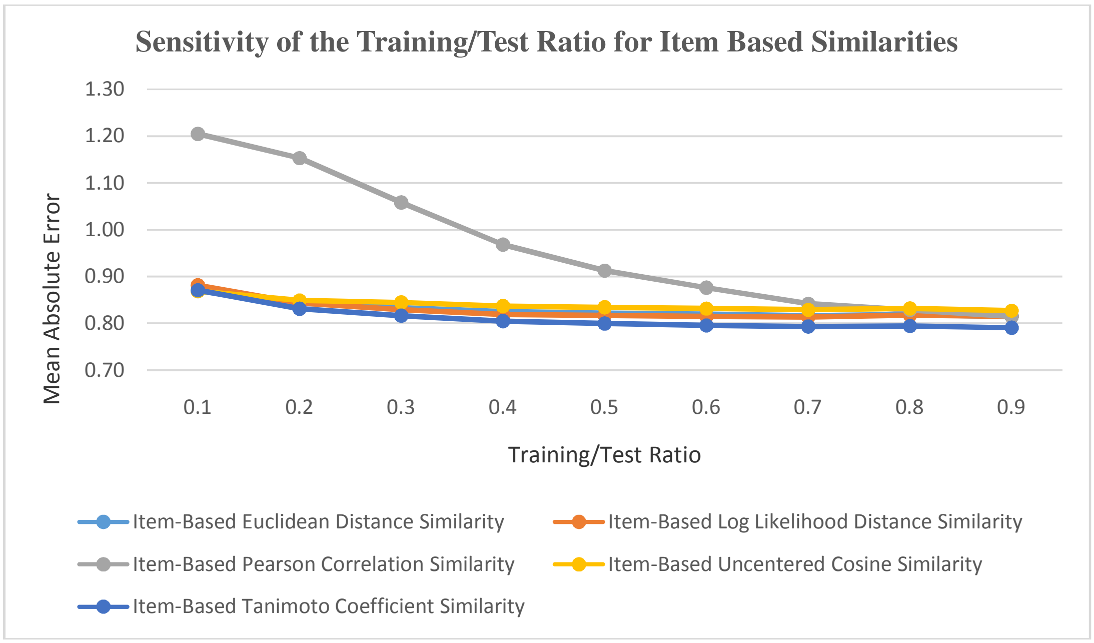


Figure 12. Mean Absolute Error for all item-based similarities as Training/Test Ratio changes.

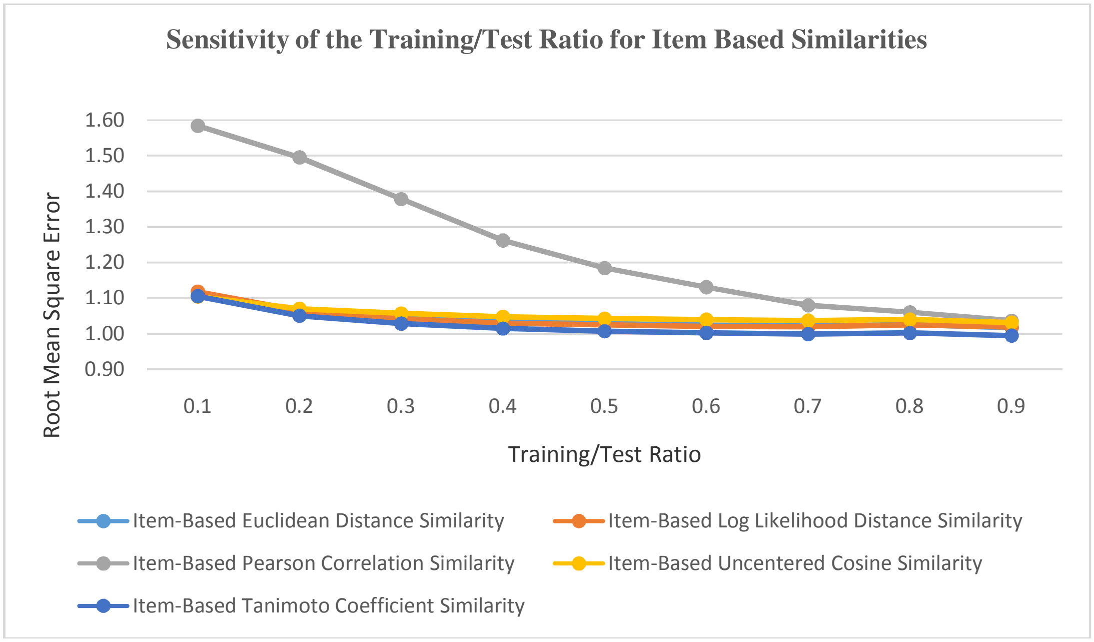


Figure 13. Root Mean Square Error for all item-based similarities as Training/Test Ratio changes.

5.3.6 Experiment Results with Different CF Algorithms

Since in the previous experiment, Log Likelihood Distance similarity and Tanimoto Coefficient similarity gave the lowest Mean Absolute Error and Root Mean Square Error, these similarities are picked in this experiment in order to compare Item-Based and User-Based algorithms. These similarities are tested with our data sets. As shown in Figure 14 and Figure 15, with different Training/Test ratio, Mean Absolute Error and Root Mean Square Error has calculated.

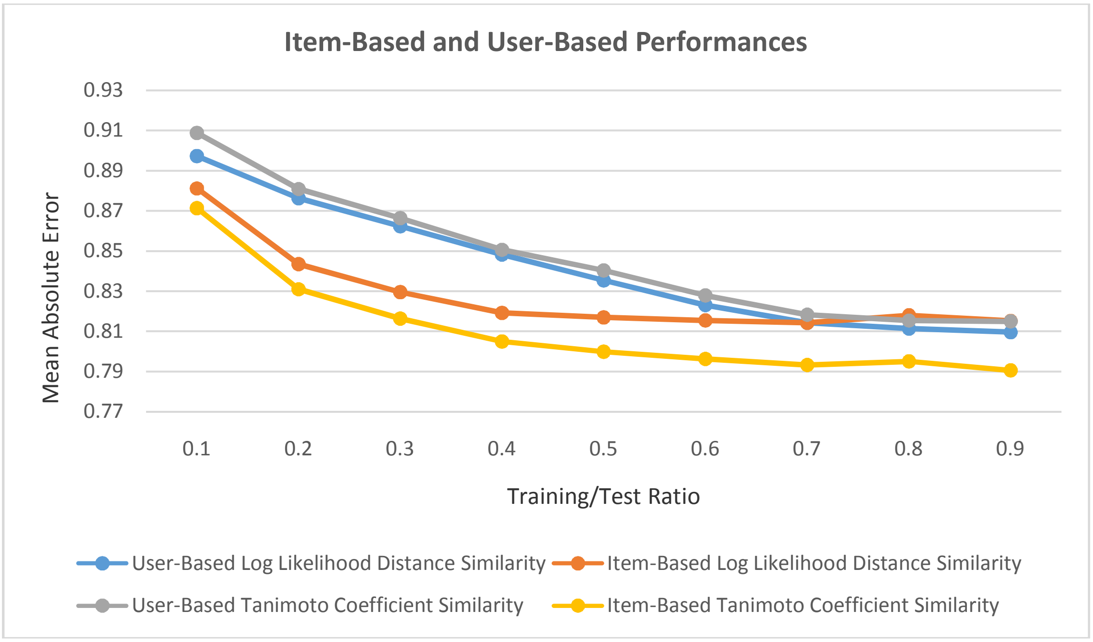


Figure 14. Impact of Training/Test Ratio on Item-Based and User Based algorithms by using Mean Absolute Error.

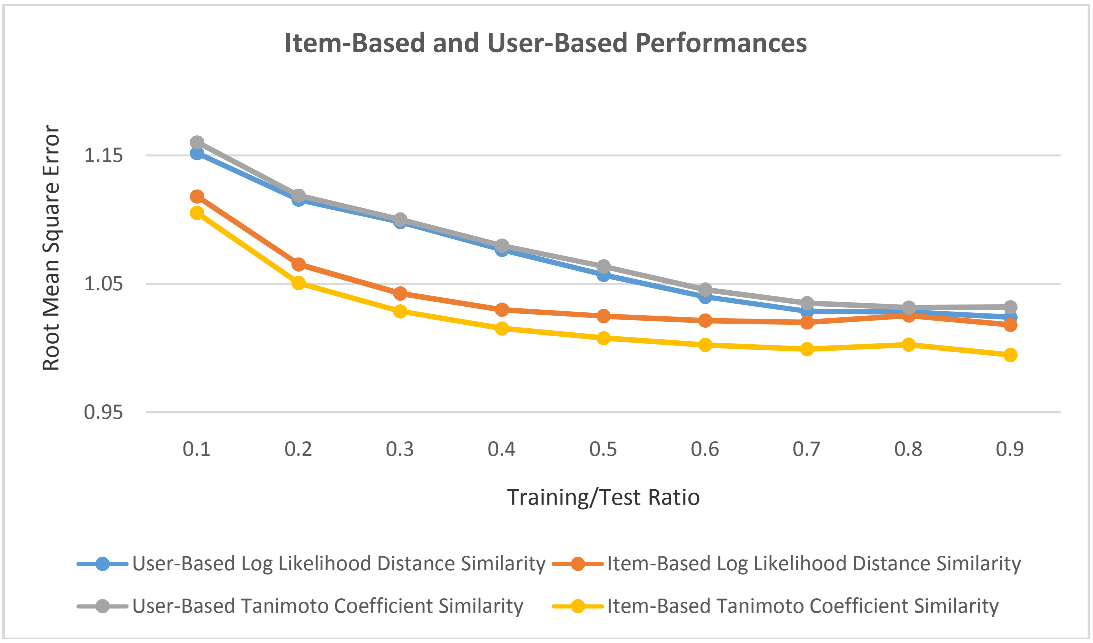


Figure 15 Impact of Training/Test Ratio on Item-Based and User Based algorithms by using Root Mean Square Error.

5.4 Performance Results

In this section performance results will be compared. Performance results are related with recommendation times. Since each similarity have different way to recommend item, their recommendation times are different. Recommendation times show that how fast the recommendation is created. By taking average recommendation time, it can be observed that which collaborative filtering technique gives the faster recommendations.

5.4.1 User-Based Similarities in Recommendation Times

In this section, the results for six User-Based similarities will be shown. Then by creating a histogram graph, their recommendation times will be compared. In this experiment, ten movies will be recommended to the selected user. For six User-based similarities, ten movies are recommended to the selected user and it is repeated ten times as shown in Table 8.

Table 8. For each repeat, recommendation times and average recommendation time for six User-Based similarities are shown as milliseconds.

USER BASED SIMILARITIES					
EUCLIDEAN DISTANCE	**LOG LIKELIHOOD**	**PEARSON CORRELATION**	**TANIMOTO COEFFICIENT**	**UNCENTERED COSINE**	**SPEARMAN CORRELATION**
547	579	515	532	469	532
547	610	547	579	531	547
547	640	515	578	500	562
507	625	548	591	547	531
563	640	547	563	516	595
516	626	515	562	516	609
516	656	531	563	516	578
516	656	500	578	500	563
547	625	532	547	532	578
516	672	532	531	500	594
532,2	**632,9**	**528,2**	**562,4**	**512,7**	**568,9**

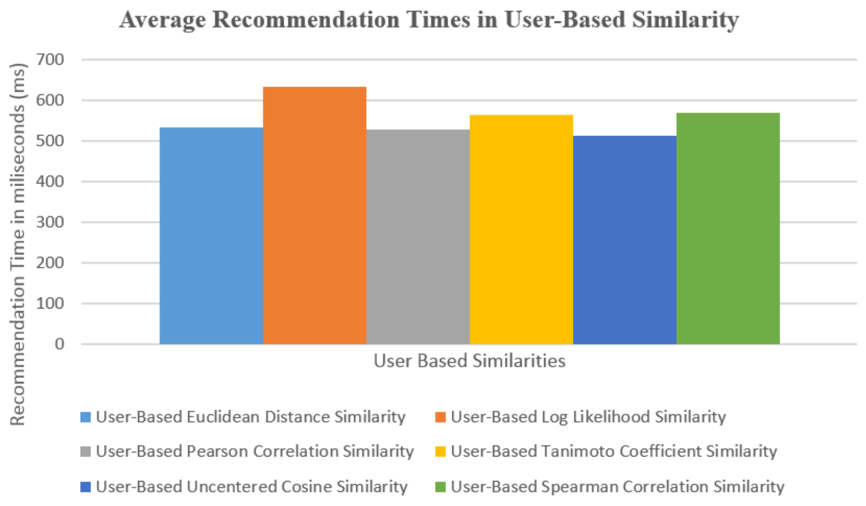


Figure 16. Impact of Item Based algorithms on recommendation times.

As shown in Table 8 and Figure 16, while Log Likelihood similarity is giving the slowest recommendations, Uncentered Cosine gives the fastest recommendations.

5.4.2 Item-Based Similarities in Recommendation Times

In this section, the results for five Item-Based similarities will be shown. Then by creating a histogram graph, their recommendation times will be compared. In this experiment, the most similar two items with the selected items will be displayed and it will be repeated ten times.

Table 9. For each repeat, recommendation times and average recommendation time for five Item-Based similarities are shown as milliseconds.

ITEM BASED SIMILARITIES				
EUCLIDEAN DISTANCE	**LOG LIKELIHOOD**	**PEARSON CORRELATION**	**TANIMOTO COEFFICIENT**	**UNCENTERED COSINE**
500	531	501	500	500
500	563	578	485	531
536	578	531	537	536
516	562	516	500	548
532	563	547	500	539
562	547	516	516	531
563	547	531	524	532
534	563	547	516	581
547	562	547	516	575
547	531	531	522	532
533,7	**554,7**	**534,5**	**511,6**	**540,5**

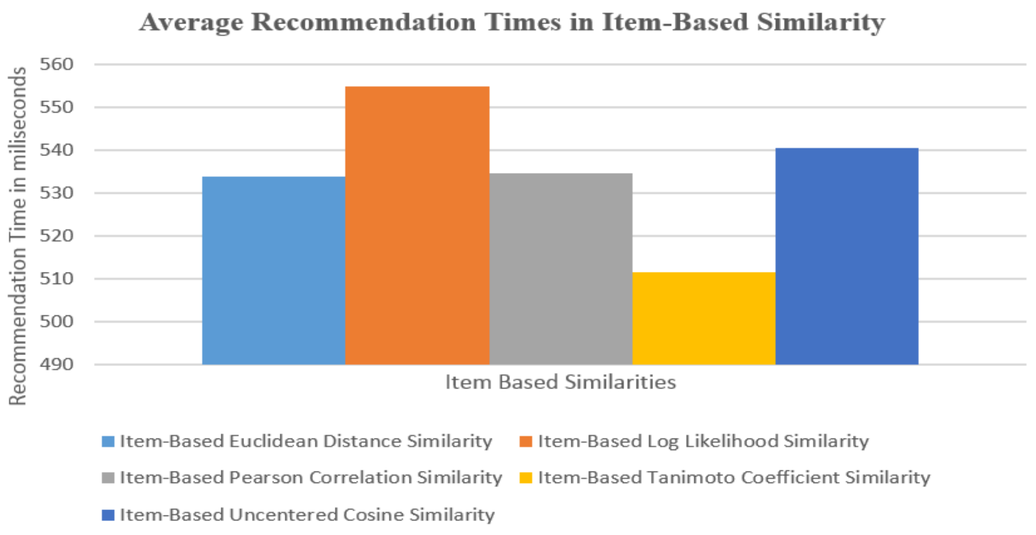


Figure 17. Impact of Item Based algorithms on recommendation times.

As shown in Table 9 and Figure 17, while Log Likelihood similarity is the slowest item-based recommendation algorithm and Tanimoto Coefficient Similarity is the fastest item-based recommendation algorithm.

6. CONCLUSION

Recommender systems offer users some items that they may desire to buy from a business. These system use user databases for taking supplement value for business. Recommender systems help user items that they would like to buy from business. Likewise these systems help the business by occurring more sales. Recommender systems are becoming an essential tool in e-commerce on the Web. New technologies are required that can develop the scalability of recommender systems which are being underlined by the huge volume of user data in current databases. Collaborative filtering is a new way to filtering data that can select from database. Collaborative filtering systems collect user's previous data about an item such as movies, book, music, ideas, feeling, and products. For recommending the best item, there are many algorithms that are based on different approaches. According to Collaborative Filtering Systems, there is a mutual point that is establishment of similarity between users and items. Collaborative-basedalgorithm extends to big data sets also support high quality recommendations.

In this paper, collaborative filtering algorithms are discussed, and showed the difference of these algorithms. We compare User-based and Item-based algorithms with different similarity index. By using these algorithms, we implemented them to the movie recommender system. These algorithms can be used in any other data sets in order to recommend items. There are much more work to be done in collaborative filtering algorithms. Our most important suggestions for improvements are below:

As we implemented the algorithms for making a movie recommender, these implemented algorithms can be used in many movie web pages for providing an option for their users. Furthermore, these algorithms can also be implemented in any other areas such as in a marketing department, looking the previous production tastes of the customers, and recommending them the best product. Also these algorithms can be used in web streaming areas such as music recommendation and also online bookstores and so on.

One challenge is that in the large amount of data sets, performance is not fast enough. Performance improvements must be done in the large data sets in order to recommend items as quick as possible.

REFERENCES

[1] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl, "Item-Based Collaborative Filtering Recommendation Algorithms", Tenth International World Wide Web Conference (WWW10), May 1-5, 2001, Hong Kong.

[2] Jonathan L. Herlocker, Joseph A. Konstan, Alborchers, and John Riedl, "An Algorithmic Framework for Performing Collaborative Filtering", Proceedings of the 22nd Annual International ACM SIGIR'99 Conference on Research and Development in Information Retrieval, Pages 230-237, New York, NY, USA.

[3] Keshav R, et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (3), 2014, 4782-4787.

[4] Sean Owen, Robin Anil, Ted Dunning, Ellen Friedman,"Mahout in Action", ISBN: 978-1935-18-2-689, Manning Publications Co., U.S America, 2012.

[5] Shuhang Guo (2014), Lapland University of Applied Sciences, "Analysis and Evaluation of Similarity Metrics in Collaborative Filtering Recommender System", Thesis of the Degree Programme in Business Information Technology, Tornio 2014.

[6] Hiroshi Shimodaira, Similarity and recommender systems School ofInformatics,The University of Edinburgh, 2014.

[7] Peter Casinelli, Advisor: Sergio Alvarez, "Evaluating and Implementing Recommender Systems As Web Services Using Apache Mahout", Boston College Honour Thesis, 2014.