Reducing Data Sparsity in Recommender Systems

Nadia F. Al-Bakri[1 *] and Soukaena Hassan Hashim[2]
[1]Department of Computer Science, Al Nahrain University, Baghdad-Iraq.
[2]Department of Computer Science, University of Technology, Baghdad-Iraq.
[*]Corresponding author: Nadiaf_1966@yahoo.com.

Abstract

Recommender systems are used to find user's interested things among a huge amount of digital information. Collaborative filtering is used to generate recommendations. However, the data sparsity problem leads to generate unreasonable recommendations for those users who provide no ratings. From this point, this paper presents a modest approach to enhance prediction in movielens dataset with high sparsity by applying collaborative filtering methods. The proposal consists of three consequence phases: preprocessing phase, similarity phase, prediction phase. The experimental results obtained conducting similarity measures against movielens user rating datasets show that the result of prediction is enhanced about 10% to15% with the non-sparse rating matrix.
[DOI: 10.22401/JNUS.21.2.20]

Keywords: Recommender system, Collaborative filtering, Pearson similarity measure, Prediction.

1. Introduction

Recommender system is an intelligent suggestion maker that suggests interested items to users such as books, cd, and products on Amazon.com and movies on Movielens websites. The users are enabled to filter large information [1]. Recommendation systems apply:

1. Data mining techniques to deal with the problem of information overload due to the dynamic generated information.
2. Prediction algorithms to predict users interest among huge amount of available items.
3. Different Recommendation algorithms to recommend products (items) to active users.

Recommender systems benefits both service providers and users by reducing the costs of selecting items in an online shopping and improving decision making process [2][3]. This paper, is concentrated and contributed mostly on memory-based collaborative filtering, and conducting methods to measure the performance against movielens user rating datasets. The paper is organized as follows: the related works are presented in section 2, on section 3 a passing background in recommendation process and collaborative filtering algorithms and its variants memory-based approach is described, in section 4 the proposed model is presented, the experimental results in section 5 and to end with the conclusion.

2. Related Work

In what follows, some of the previous research literatures related to the techniques used in user-based and item-based collaborative filtering are presented.[4] Proposed a dynamic model that exploits the concept of time decay obtained from e-commerce databases; by giving its mathematical formula using the time decay factor to redefine the item-to-item similarity function. The model was evaluated by actual ecommerce data. The results show that there is no increase in the computational complexity of item-to-item similarity matrix. [5] Proposed a personalized recommendation technique using the data contained in both ratings and profile contents of users. A user preference is described by designing a set of sequential characteristics of users' interest based on time series analysis (TSA) techniques. A linear model of the features is constructed to detect changes in user preferences. Then a recommendation is generated by giving a weight to the dynamic features on rating and profile contents by using the item based recommendation algorithm through its similarity. [6] Presented a heuristic similarity model to increase the recommendation performance when only few ratings are available in estimating similarities for each user. This improved similarity measure is composed of three factors of similarity: proximity, significance and singularity. The model not only considers rating between users,

but also the global preference of user behavior in which different users have different rating preferences. The model uses the mean and variance of the rating to describe the rating preference of user. [7] Presented a model of item-item similarity to achieve an increase in online scalability. Different techniques such as item-item Pearson vs. cosine similarities between item vectors are conducted and different techniques for prediction such as weighted sum prediction method vs. regression model techniques were conducted and the results were compared with the basic user-based approach. The results show that new item-based algorithm provided better performance and quality of prediction than user-based algorithm. [8] proposed a system that combines two types of recommendation systems: content-based to nominate products to new customers similar to the products that have been previously purchased by customers. Cosine similarity function is used between the products described by texts. The other type is item-based collaborative filtering method which is used for registered customers. The proposed system contributed in overcoming problems with high efficiency and accuracy, such as the cold start for new item, scalability due to increasing items, synonymy and sparsity.[9] Applied three different recommendation algorithms, they are: Pearson's Coefficient, Bayesian Network and KNN (K nearest neighbored). They computed the similarities using these three techniques for gaining recommendation. The authors concluded that using Pearson's Correlation Coefficient achieves good quality in which it allows scaling large dataset.

3. Recommendation System

The Formal Definition of Recommender System is shown below:

There are two sets of entities, which they are users and items. They are represented as: [1]

U: This denotes the number of training users.

I: This denotes the number of items in the dataset.

F: A rating function that measures the utility of a specific item $i \in I$ to the user $u \in U$, where is

F: $U \times I \rightarrow R$, such that R is rating scalar.

For each user $u \in U$, an item $i \in I$ is chosen such that it makes the most of the user's rating. Rating is done on integer scale of 1 to 5 or sometimes on real scale of 0.5 to 2.5 according to the recommendation model type. Each user can be well-defined in his profile features like user id, age, gender, occupation. Each movie in movies datasets can be well-defined having characteristics such as its movie id, movie release date, movie director and movie actors.

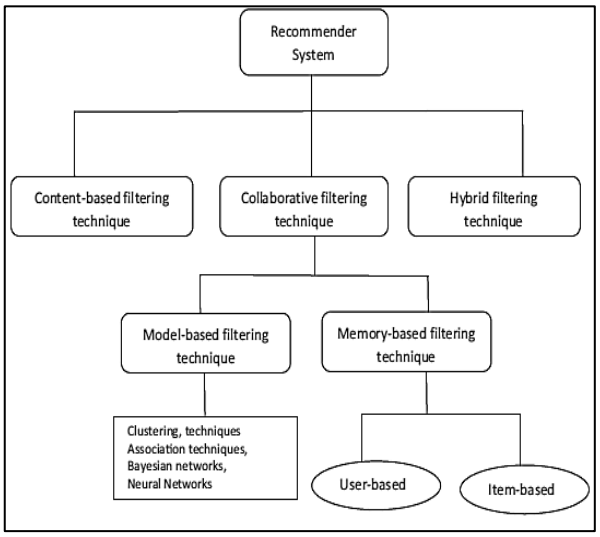


Fig.(1) : Recommendation techniques [3].

Recommendation system techniques are categorized, as shown in Fig.(1) into:

- Content-based filtering techniques,
- Collaborative Filtering techniques, and
- Hybrid filtering techniques.

With content based filtering approach, recommendation is based on matching items preferred by the active user in the past with dedicated items to the user. In collaborative filtering (CF), items are recommended based on other users with similar interests and favorites to these items. Hybrid filtering technique is used to overcome the limitations of both techniques [1]. In this paper, collaborative filtering technique is utilized.

Collaborative filtering process, as shown in Fig.(2) represents the user-item as a rating matrix. It calculates similarities between user preferences then a group called neighborhood is built and a user gets recommendations to items rated by users in his neighborhood. Collaborative filtering produces either prediction which is a numerical, that stating the predicted rate of particular item i for a

particular user u, or a Recommendation which lists top N items to the user [3].

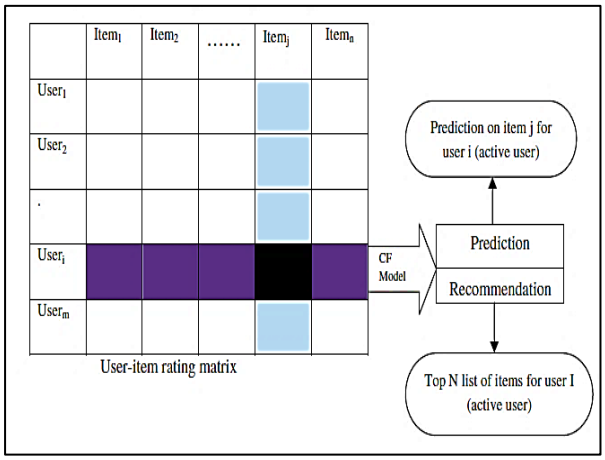


Fig.(2): collaborative filtering process.

The techniques of collaborative filtering (CF) are classified into two main categories:

- Memory-based.
- Model-based.

In this paper, a memory based collaborative filtering is adopted.

3.1 Memory based collaborative technique

This technique identifies related neighbors for active user, then a prediction rating to unknown item i for active user can be yielded. They have succeeded an extensive achievement in existent life applications [11]. They can be achieved in two ways:

A. User-based collaborative technique

This technique is based on a comparison between user's ratings on same items by calculating a similarity *using Pearson correlation* between users, and then a computation is done to predict rating for an item by the active user [3]. The correlation of user u to user v is calculated using equation 1 which is called the *Pearson correlation* for users: [11]

$$Sim(u,v) = \frac{\sum_{i\in I}(r_{u,i}-\overline{r_u})(r_{v,i}-\overline{r_v})}{\sqrt{\sum_{i\in I}(r_{u,i}-\overline{r_u})^2}\sqrt{\sum_{i\in I}(r_{v,i}-\overline{r_v})^2}} \quad \text{.....Eq. (1)}$$

Where: $i \in I$, I is the overall co-rated items for both user u and user v given their rates. $r_{u,i}$ is the rating for user u to item i and $\overline{r_u}$, $\overline{r_v}$ are the users u ,v mean rating.

B. Item-based collaborative technique

This technique is based on a Computation of similarity between items determining the similarity degree between the target item and the past purchased items by the active user, then a selection is made to choose the most similar items [3]. To compute item similarity, The *Pearson Correlation* coefficient equation between items i and j is: [11]

$$Sim(i,j) = \frac{\sum_{u\in U}(r_{u,i}-\overline{r_i})(r_{u,j}-\overline{r_j})}{\sqrt{\sum_{u\in U}(r_{u,i}-\overline{r_i})^2}\sqrt{\sum_{u\in U}(r_{u,j}-\overline{r_j})^2}} \quad \text{....Eq. (2)}$$

Where: $u \in U$, U is the co-rated users given rating for both items *i* and *j*.

3.2 Prediction and Recommendation Computation

After a similarity computation, a group of nearest neighbors for an active user is chosen. Then a prediction is generated by aggregating weighted ratings. Prediction can be achieved in two ways: [11]

A. User based prediction

A prediction for active user (a) on a specific item i is computed by summing the mean of active user (a) with summation of the rating's weights on that item i, as shown in equation 3:[11]

$$predict(a,i) = \overline{r_a} + \frac{\sum_{u\in U} sim(a,u).(r_{u,i}-\overline{r_u})}{\sum_{u\in U}|sim(a,u)|} \quad \text{..... Eq. (3)}$$

B. item-based prediction

A simple weighted average to predict the rating, P (a,j), for user (a) on item j as shown in equation 4:[11]

$$predict(a,j) = \frac{\sum_{i\in I} sim(i,j).(r_{a,i})}{\sum_{i\in I}|sim(i,j)|} \quad \text{.....Eq. (4)}$$

4. The proposed recommendation system

The proposed recommendation system consists of three phases, they are: Preprocessing phase, Similarity phase, and Prediction phase. Fig.(3) shows the proposed system.

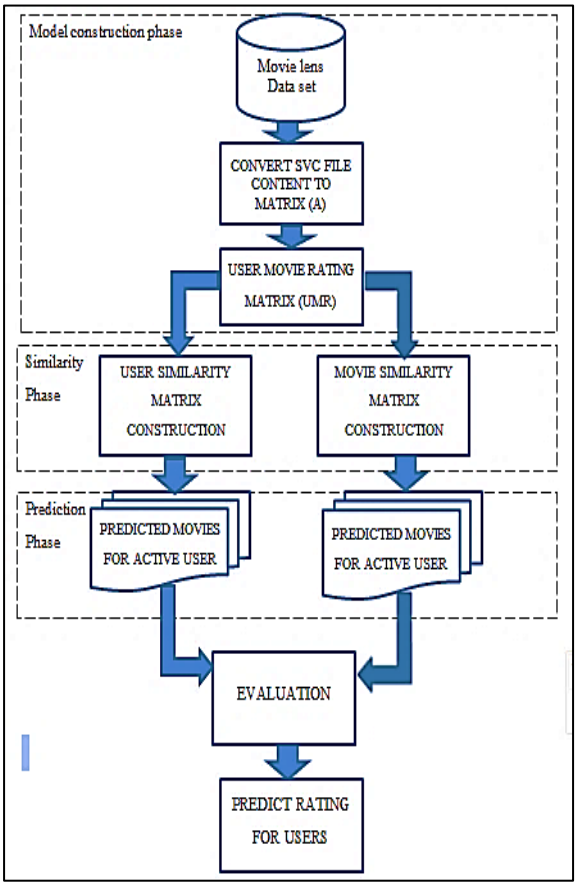


Fig.(3) the block diagram of the proposed system.

4.1 Dataset description

A MovieLens 100k dataset is used from the GroupLens Research Group web-based research recommender system.it consists of 100,000 ratings from 943 users on 1682 movies In this dataset at least 20 movies each user rated. The file ratings.dat is conducted for information extraction.

4.2 The preprocessing phase

This phase consists of two parts:

A. Model description

The proposed model consists of:

u ∈ U a user u belongs to all co-rated users. **m ∈ M** a movie m belongs to all co-rated movies that both users rated. **rate** $_{u,m}$ means rating of user u to movie m.

avg rate$_u$ means average rating for user u.

Sim (mi,mj) means the similarity between movies mi and mj.

user ID	item ID	Rating	Timestamp
1	17	3	875073198
1	47	4	875072125
1	64	5	875072404
1	90	4	878542300
1	92	3	876892425
1	113	5	878542738
1	222	4	878873388
1	227	4	876892946
1	228	5	878543541
1	253	5	874965970
2	257	4	888551062
2	279	4	888551745
2	299	4	888550774
2	301	4	888550631
2	303	4	888550774
2	307	3	888550066
2	308	3	888979945
2	313	5	888552084

Fig.(4): The movielens Svc file Snapshot.

B. Matrix Construction

In this section of the preprocessing phase, the data structure is created to capture the relations between different users and different movies as in the following steps:

1. Load movielens100k dataset from grouplens website. The loaded file is converted from u.data files to Cvs format to be further accessed and processed. Users and items are numbered consecutively from 1. The data is ordered at random fashion. These files show the numeric values separated by tabs such that the first column is user ID, the second column is movie ID, the third column is rating given by user to the movie (Integer numbers between 1 to 5) and the last column is the Timestamp field that represent time in seconds as shown in Fig.(4).
2. Populate Svc file to a matrix called (A) in the proposed system from the pure data contained in the Svc file. Then construct the user-movie-rating (UMR) matrix. The rows in (UMR) matrix represent users ID and the columns represent movie ID extracted from the second column of Svc file.

The programming steps of matrix construction is depicted in Fig.(5) and shown in algorithm (1).

1	2	3	4	5	6	7	8	9
5	3	4	3	3	5	4	1	5
4	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	2	4	4
0	0	0	5	0	0	5	5	5
0	0	0	0	0	0	3	0	0
0	0	0	0	0	0	4	0	0
4	0	0	4	0	0	0	0	4
0	0	0	0	0	0	0	4	5
0	0	0	5	0	0	0	0	0
3	3	0	5	1	0	2	4	3
0	0	0	0	0	0	5	0	4

Fig.(5): user-movie-rating matrix (UMR).

Algorithm (1): building user-movie-rating matrix (UMR) from excel file (movielens).

Input: movielens file, A MATRIX
Output: user-movie-rating matrix (UMR).

```
Begin
Step1 Import (movielens) file and store content in a two dimension array (A).
  //such that no. of rows equals to movielens file rows and first column contain user id second column is
        movie id and last column is rating.
Step2 initializing UMR matrix with zeros.
Step3 Counter=1                    //initialize no. of users to 1.
Step4 For each row i of A do     // loop to store every user in movlens and its rating .
Step5 if   A(row i  ,1)=counter
            UMR(counter, A(row i, column 2))=A(row i, column 3);
         else                                                  // column 2 of A matrix is movie ID
         Counter=counter+1;                                    //take next user.
          UMR(counter, A(row i, column 2))=A(row i, column 3); //column 3 contains  rating
     end if                                                  //use column 2 as column index in(UMR)
  end for
End
```

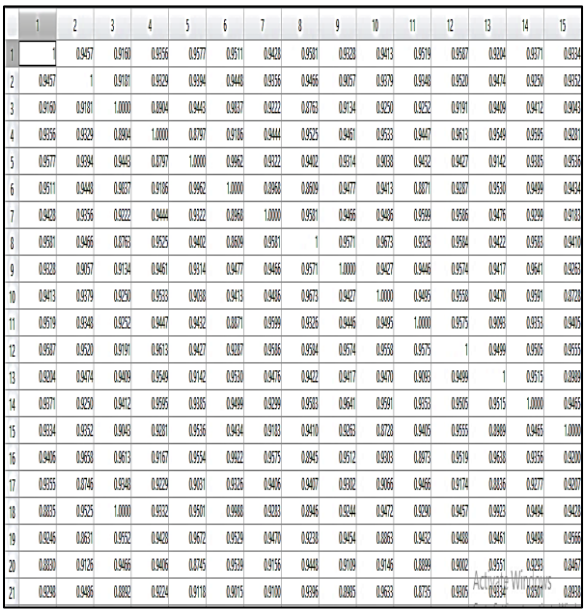
4.3 Similarity phase.

After building the user movie matrix (UMR) in the previous step, the next step is computing:

- The similarity between users then stores it in user –user similarity matrix (UUSM) as shown in Fig.(6). The framework described in Algorithm (2). The rows and columns represent users.
- The similarity between movies then stores it in movie–movie similarity matrix (MMSM) matrix. The framework described in Algorithm (3). The rows and columns represent movies as shown in Fig.(7).

	1	2	3	4	5	6	7	8	9	10
1	1	0.0919	-0.0052	0.0200	0.2922	0.3306	0.3235	0.2568	0.0565	0.2776
2	0.0919	1	0.1031	0.1046	0.0018	0.1803	0.0279	0.0603	0.0837	0.0739
3	-0.0052	0.1031	1	0.2269	-0.0199	0.0251	-0.0082	0.0504	-0.0124	0.0064
4	0.0200	0.1046	0.2269	1.0000	-0.0125	-0.0312	0.0110	0.1429	-0.0073	-0.0116
5	0.2922	0.0018	-0.0199	-0.0125	1.0000	0.1540	0.2710	0.1910	0.0593	0.1089
6	0.3306	0.1803	0.0251	-0.0312	0.1540	1.0000	0.3808	0.1072	0.0847	0.4610
7	0.3235	0.0279	-0.0082	0.0110	0.2710	0.3808	1.0000	0.2098	0.0881	0.3730
8	0.2568	0.0603	0.0504	0.1429	0.1910	0.1072	0.2098	1.0000	0.0156	0.1562
9	0.0565	0.0837	-0.0124	-0.0073	0.0593	0.0847	0.0881	0.0156	1	0.1428
10	0.2776	0.0739	0.0064	-0.0116	0.1089	0.4610	0.3730	0.1562	0.1428	1.0000
11	0.2237	0.0683	0.0354	0.0696	0.2584	0.1594	0.2782	0.0998	0.0039	0.1310
12	0.2417	0.0786	0.0115	0.0121	0.0495	0.1718	0.1848	0.1094	0.1705	0.1932
13	0.2717	0.1217	0.1192	0.0318	0.2229	0.3304	0.3842	0.2216	0.1016	0.3588
14	0.2481	0.1649	0.0160	0.0031	0.1913	0.2911	0.2028	0.1265	0.1152	0.2276
15	0.0961	0.3682	0.0699	0.1057	0.0127	0.1144	0.0084	0.0293	0.0544	0.0371
16	0.2921	0.1006	0.0186	0.0305	0.2009	0.2925	0.3449	0.2011	0.0613	0.3932
17	0.1402	0.1860	-0.0146	-0.0086	0.0298	0.1376	0.0376	0.0568	0.2178	0.0941
18	0.3728	0.1213	-0.0228	-0.0223	0.1860	0.4906	0.3983	0.0929	0.0692	0.4032
19	0.0284	0.0245	0.0893	0.0398	0.0322	0.0826	0.0681	0.0247	0.0634	0.0414

Fig.(6) : Sample of user-user similarity matrix.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	1	0.9457	0.9160	0.9556	0.9577	0.9511	0.9420	0.9501	0.9328	0.9413	0.9519	0.9587	0.9204	0.9577	0.9334
2	0.9457	1	0.9101	0.9329	0.9394	0.9448	0.9356	0.9466	0.9057	0.9379	0.9548	0.9520	0.9474	0.9259	0.9552
3	0.9160	0.9181	1.0000	0.9004	0.9443	0.9457	0.9222	0.8763	0.9134	0.9250	0.9252	0.9191	0.9409	0.9412	0.9043
4	0.9556	0.9329	0.8994	1.0000	0.8797	0.9106	0.9444	0.9525	0.9451	0.9533	0.9447	0.9613	0.9549	0.9595	0.9207
5	0.9577	0.9394	0.9443	0.8797	1.0000	0.9662	0.9322	0.9402	0.9314	0.9033	0.9432	0.9427	0.9142	0.9365	0.9516
6	0.9511	0.9448	0.9457	0.9106	0.9662	1.0000	0.8968	0.8939	0.9477	0.9413	0.8871	0.9207	0.9530	0.9499	0.9434
7	0.9420	0.9356	0.9222	0.9444	0.9322	0.8968	1.0000	0.9501	0.9466	0.9466	0.9599	0.9506	0.9476	0.9299	0.9183
8	0.9501	0.9466	0.8763	0.9525	0.9402	0.8939	0.9501	1	0.9571	0.9673	0.9526	0.9504	0.9422	0.9503	0.9410
9	0.9328	0.9057	0.9134	0.9451	0.9314	0.9477	0.9466	0.9571	1.0000	0.9427	0.9446	0.9574	0.9417	0.9641	0.9263
10	0.9413	0.9379	0.9250	0.9533	0.9033	0.9413	0.9466	0.9673	0.9427	1.0000	0.9495	0.9558	0.9470	0.9591	0.9228
11	0.9519	0.9548	0.9252	0.9447	0.9432	0.8871	0.9599	0.9526	0.9446	0.9495	1.0000	0.9575	0.9095	0.9353	0.9405
12	0.9587	0.9520	0.9191	0.9613	0.9427	0.9207	0.9506	0.9504	0.9574	0.9558	0.9575	1	0.9499	0.9505	0.9555
13	0.9204	0.9474	0.9409	0.9549	0.9142	0.9530	0.9476	0.9422	0.9417	0.9470	0.9095	0.9499	1	0.9515	0.9269
14	0.9577	0.9259	0.9412	0.9595	0.9365	0.9499	0.9299	0.9503	0.9641	0.9591	0.9353	0.9505	0.9515	1.0000	0.9465
15	0.9334	0.9552	0.9043	0.9207	0.9516	0.9434	0.9183	0.9410	0.9263	0.9228	0.9405	0.9555	0.9269	0.9465	1.0000
16	0.9406	0.9658	0.9613	0.9167	0.9554	0.9422	0.9575	0.8945	0.9512	0.9103	0.8973	0.9519	0.9630	0.9356	0.9200
17	0.9555	0.8746	0.9148	0.9229	0.9031	0.9126	0.9406	0.9407	0.9302	0.9066	0.9466	0.9174	0.8836	0.9277	0.9207
18	0.8835	0.9525	1.0000	0.9332	0.9501	0.9008	0.9203	0.8946	0.9244	0.9472	0.9260	0.9457	0.9023	0.9494	0.9423
19	0.9246	0.8631	0.9552	0.9428	0.9572	0.9529	0.9470	0.9230	0.9454	0.8865	0.9432	0.9408	0.9451	0.9490	0.9566
20	0.8830	0.9126	0.9466	0.9406	0.8745	0.9539	0.9155	0.9440	0.9109	0.9145	0.8899	0.9002	0.9551	0.9295	0.9457
21	0.9298	0.9486	0.8882	0.9224	0.9118	0.9015	0.9100	0.9396	0.8985	0.9633	0.8735	0.9305	0.9354	0.9297	0.8938

Fig.(7): Sample of movie-movie similarity matrix.

Algorithm(2): construct user based similarity matrix
Input: user-movie-rating matrix (UMR). Output: user –user similarity matrix (UUSM)

Begin
Step1 Construct user–user similarity matrix (UUSM) by calculating the correlation by iterating through all possible pairs of users.
//find the Pearson correlation of user 1 with all users in the matrix and user 2 with all users and so on until final user in the data set.
Step 2 The Pearson correlation formula used to find the similarity values between users is:
$$Sim(user\ u, user\ v) = \frac{\sum_{movie\ m \in M}(rate_{u,m} - avg\ rate_u)(rate_{v,m} - avg\ rate_v)}{\sqrt{\sum_{movie\ m \in M}(rate_{u,m} - avg\ rate_u)^2}\ \sqrt{\sum_{movie\ m \in M}(rate_{v,m} - avg\ rate_v)^2}}$$
End

Algorithm (3): construct movie based similarity matrix
Input: user-movie-rating matrix (UMR). Output: movie –movie similarity matrix (MMSM).

Begin
Step 1 Construct movie –movie similarity matrix (MMSM) by calculating the correlation through all possible pairs of movies.
// find the Pearson correlation of movie1 with all movies in the matrix and movie 2 with all movies and so on until the last movie in the data set.
 	
Step2 The Pearson correlation formula used to find the similarity values between movies is:
$$Sim(movie\ i, movie\ j) = \frac{\sum_{user\ u \in U}(rate_{u,i} - avg\ rate_u)(rate_{u,j} - avg\ rate_u)}{\sqrt{\sum_{user\ u \in U}(rate_{u,i} - avg\ rate_u)^2}\ \sqrt{\sum_{user\ u \in U}(rate_{u,j} - avg\ rate_u)^2}}$$
//user U denotes the users that watched the movie i and movie j.
End

4.4 The prediction phase

A utilization of the entire user-movie-rating matrix (UMR) using Memory-based algorithm to generate a prediction by finding a set of users, known as neighbors, that have a past preferences agreed with the target user in seeing movies. After similarity computation, the prediction formula is applied to derive the prediction of active user to a movie and the top recommendations for an active user. The

details are presented in the following algorithms. Where algorithm (4) predict rating to a target user using weighted sum prediction with the adjustment and algorithm (5) use a simple sum prediction method.

Algorithm (4): find prediction of active user to a movie using weighted sum prediction method.
Input: user –user similarity matrix (UUSM), active user (AU). Output: predicted rating of user (AU) for movie m_i and top rated movie recommendation.

Begin
Step1 Retrieve the similarity values for the active user with co-rated users in the (UUSM) matrix.
Step2 Sort similarity values in descending order.
Step3 Select top values; these are the most similar users to (AU).
Step4 Calculate the predicted rating.
// using the selected similarity values as a weight value multiplied by user rate from UMR matrix, and then the result is normalized by dividing it by summation similarity values of similar users.
Step 5 The prediction formula for (AU) to movie m_i is:
$predict(user\ AU, movie\ m_i) = avg\ rate\ AU + \frac{\sum_{u \in user\ similarities}\ sim\ (AU,u).(\ rate_{u,mi} - avg\ \ rate_u)}{\sum_{u \in user\ similarities}\ sim\ (AU,u)}$
u denotes a letter over all users who rated movie m_i.
Step 6 Movies with the highest prediction rate are recommended to the active user (AU).
End

Algorithm (5): find prediction of active user to a movie using simple sum prediction method.
Input: movie –movie similarity matrix (MMSM). Output: predicted rating for active user and top rated movies recommendation

Begin
Step1 Retrieve the similarity values for target movie with all movies in the (MSIM) matrix.
Step2 Sort similarity values in descending order.
Step3 Select top values; these are the most similar movies to target movie.
Step4 Calculate the predicted rating
// using the selected similarity values as a weight value multiplied by user rate from (UMR) matrix, and then the result is normalized by dividing it by the summation similarity values of similar movies.
Step5 The prediction formula for active user(AU) to target movie m_i is:
$predict(user\ AU, movie\ m_i) = \frac{\sum_{n \in movie\ similarities} sim\ (m_i,n)(\ rate_{mi,n})}{\sum_{n \in movie\ similarities} sim\ (m_i,n)}$
// n denote a letter of all co-rated movies for the active user
Step6 Movies with the highest prediction rate are recommended to the active user (AU).
End

5.Experiments and Results

Offline analysis is used to understand the behavior and quality of recommender system model. User testing objective comparisons of memory based collaborative filtering algorithms is performed using evaluation measure [10]. A commonly-used CF quality metrics is Mean Absolute Error (MAE) which is sometimes called absolute deviation. This method is applied on the chosen test users. It computes the average of the absolute difference between the predicted and real ratings of users. Prediction value is the same scale of rating used and its quality is better when the MAE is low. The MAE equation is: [11].

$$MAE = \frac{\sum_{(i,j)} |\ p_{i,j} - r_{i,j}|}{n} \quad \text{.....Eq.(5)}$$

Where, n specifies the total number of tested users. $P_{i,j}$ is the predicted rating for user i on item j, and $r_{i,j}$ is the actual rating.

In This paper, experiments have been conducted using Movielens data sets, in which one of its problems is the sparsity problem. Movielens is the recommendation system research database reference. The experimental results are presented by applying the user-based and item-based memory collaborative filtering techniques. As shown in Fig.(8), the movielens dataset is a sparse matrix. The highest rating is the number of zeros in the matrix because users rate few movies. So predicting a rate to a movie is related mostly on a restricted neighbor. Fig.(9) shows the histogram of distribution to rating in movielens. The range scale is [1-5]. Fig.(10) and Fig.(11) show the similarity distributions after calculating user-based and item-based Pearson correlation on sparse and no sparse matrix.

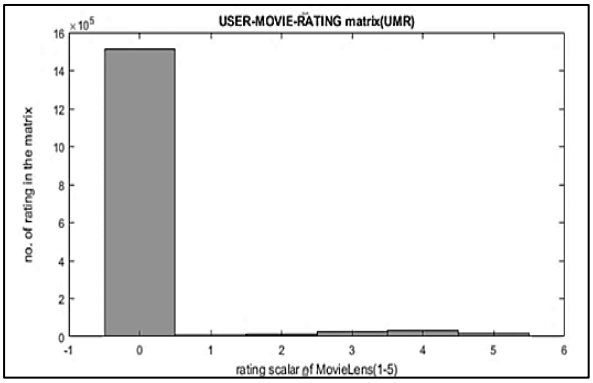


Fig.(8): The user-movie-rating matrix.

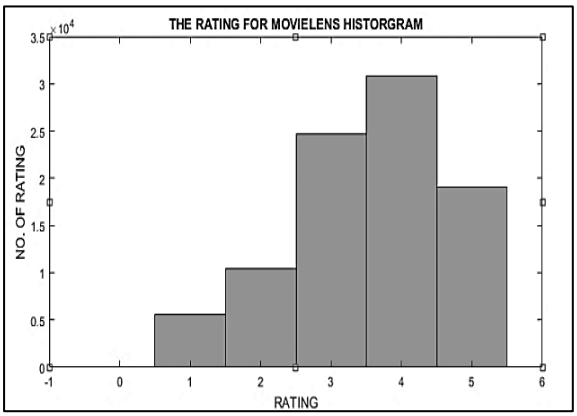
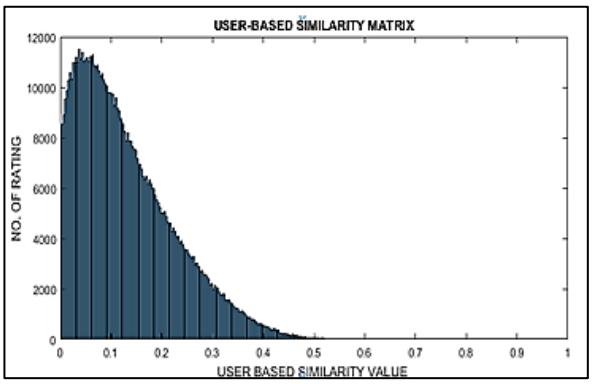
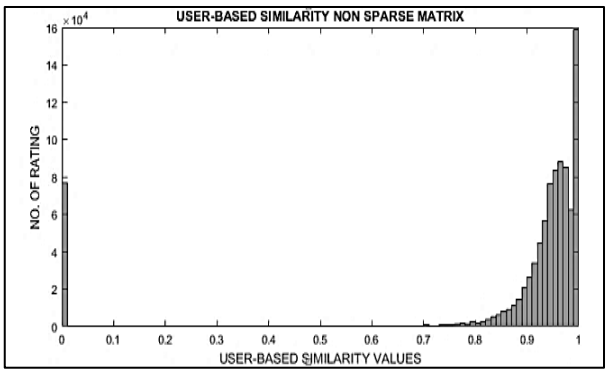


Fig.(9): Rating of movielens Histogram.

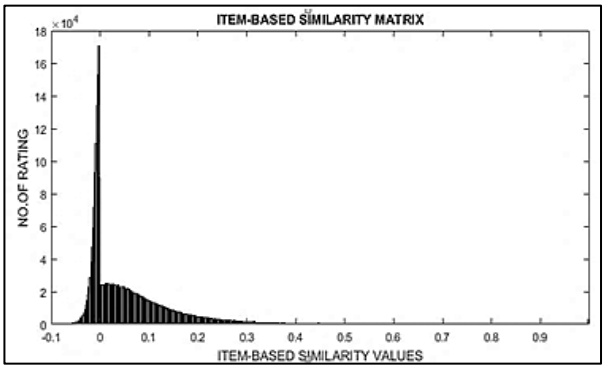


(a)

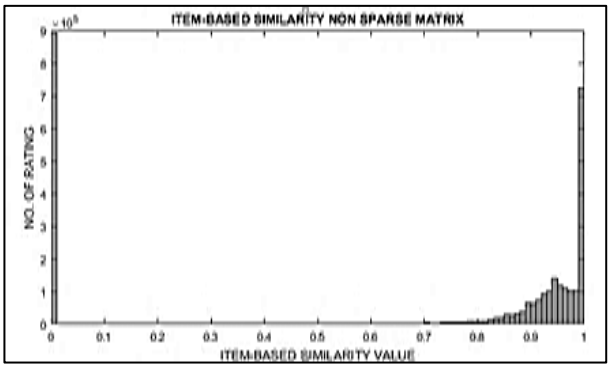


(b)

Fig.(10): Similarity computation (a) user-based histogram with no zero removal. (b) user-based histogram with zero removal.



(a)



(b)

Fig.(11): similarity computation (a) item based histogram (with zero) (b) item based histogram (with no zero).

5.1 Prediction Phase Results

An experiment is conducted on a sample from the user-movie-rating (UMR) matrix. As shown in Table (1) and Table (2), user ID 4 was chosen to predict his rating to movies he did not rate. After applying the user-based and item-based Pearson similarity, a user-based and item-based prediction formula is then applied to get predictions. The calculation is performed on the sparse matrix and on non-sparse matrix (removing zero rated movies).

Table (1)
Prediction using user based CF.

User id	AVG user rate	mov id	prediction with sparse matrix	Prediction with non-sparse matrix	size of co-rated users
User 4	4.357≅4	Mov id 6	3.345≅3	4.391≅4	23
User 4	4.357≅4	Mov id 9	3.710≅4	4.630≅5	268
User 4	4.357≅4	Mov id 11	3.234≅3	4.201≅4	217
User 4	4.357≅4	Mov id 17	3.165≅3	4.140≅4	85

Table (2)
Prediction using item based CF.

user id	AVG user rate	mov id	prediction with sparse matrix	prediction non sparse matrix	size of co-rated movies
User 4	4.357≅4	Mov id 6	4.380≅4	3.4423	14
User 4	4.357≅4	Mov id 9	4.360≅4	2.814≅3	14
User 4	4.357≅4	Mov id 11	4.371≅4	3.543≅4	14
User 4	4.357≅4	Mov id 17	4.356≅4	3.706≅4	14

5.2 Prediction accuracy measure

The accuracy of prediction in recommendation system is evaluated using the MAE to measure how close the predicted ratings are to the true user ratings.

A comparison of MAE values in Table (3) reveals that predictions using non sparse matrix are slightly better and less MAE values than predictions using sparse matrix. Fig.(12) shows the curves of MAE.

Table (3)
Prediction accuracy measured using MAE.

Tested user or Tested movie	The technique used	Number of tested Users or movies	MAE (sparse matrix)	MAE non sparse matrix
Movie ID 9	User-based	286 users	0.74	0.70
Movie ID 17	User-based	85 users	0.94	0.75
User ID 1	Item-based	262 items	0.84	0.67
User ID 4	Item-based	14 items	0.81	0.60
User ID 6	Item-based	201 items	0.84	0.67
User ID 9	Item-based	12 items	0.91	0.80

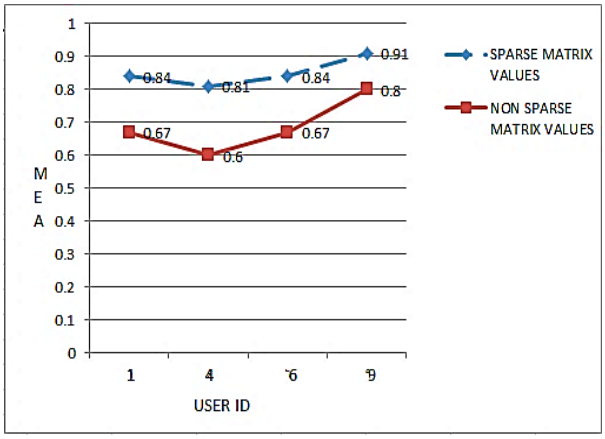


Fig.(12): The MAE, for sparse matrix and non-sparse matrix.

6- Conclusion

Collaborative filtering Recommendation methods rely mostly on implementing similarity measures because recommendation is suggested based on the relations between users and items. Sparsity is one of the problems when dealing with collaborative filtering techniques.

In this paper, an implementation of sparse reduction was done to improve prediction and to reduce the mean absolute error. The MEA results show that prediction of missing ratings are vulnerable to sparsity problem and its accuracy is considerably degrades with increased sparsity.

Using user-based and item-based prediction methods with non sparse matrix may improve the prediction accuracy and reduce the MAE values.

The complexity of computing the user-movie-rating matrix (UMR) increases as the number of users and items increases. Item-based method can provide better computational performance and better quality than user-based or nearest neighbored methods. User similarity approaches are more likely to produce diverse recommendation because it depends on user's similarity.

References

[1] Meenakshi S., Sandeep M., "A Survey of Recommender Systems: Approaches and Limitations", International Journal of Innovations in Engineering and Technology, ISSN: 2319-1058, Special Issue–ICAECE, 2013.

[2] Tejal A., R.S. Sonar, N. J. Uke, "A Survey on Recommendation System", International Journal of Innovative Research in Advanced Engineering (IJIRAE), ISSN: 2349-2163, 2(1), January 2015.

[3] Isinkaye F.O., Folajimi Y.O., Ojokoh B.A., "Recommendation systems: Principles, methods and Evaluation", Egyptian Informatics Journal, 16, 216-273, 2015.

[4] Chaolun X., Xiaohong J., Sen L., Zhaobo L., Zhang Y., "Dynamic item-based recommendation algorithm with time decay", IEEE, 242–247, 2010.

[5] Mayuri P. C. , Sonal P., and Ganesh D. "A Survey for Personalized Item based Recommendation System" International Journal of Emerging Trends & Technology in Computer Science (IJETTCS), 4(1), January-February 2015.

[6] Haifeng L. , Zheng H., Ahmad M., Hui T., Xuzhen Z., "A new user similarity model to improve the accuracy of collaborative filtering", State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Knowledge-Based Systems, 156–166, China, 2014.

[7] Badrul S., George K., Joseph K. and John R., "Item-Based Collaborative Filtering Recommendation Algorithms", in: WWW ACM, pp. 285–295, Hong Kong, 2001.

[8] Abbas A. R., Hamdoay S. A., "Design recommendation system in e-commerce site", Iraqi Journal of Science, 57(4), 2549-2556, 2016.

[9] Priyanka S., Manoj J., Abhishek S., Deepali V., "Web Usage Mining Using Pearson's Correlation Coefficient", International Journal of Engineering Research and Applications (IJERA), 3(2), 676-679, 2013.

[10] Michael D. E., John T. R. and Joseph A. K., "Collaborative Filtering Recommender Systems", Foundations and Trends R in Human–Computer Interaction, 4(2), 2010.

[11] Xiaoyuan S., Taghi M. K., "A Survey of Collaborative Filtering Techniques", Advances in Artificial Intelligence, 2009 (421425), 19 pages, 2009.