Journal of Information & Optimization Sciences
ISSN 0252-2667 (Print), ISSN 2169-0103 (Online)
Vol. 45 (2024), No. 4, pp. 1029–1039
DOI : 10.47974/JIOS-1625

Bi-clustering based recommendation system

Mahesh Mali *
Dhirendra Mishra †
Department of Computer Engineering
SVKMs NMIMS Mukesh Patel School of
Technology Management and Engineering
Mumbai 400056
Maharashtra
India

M. Vijayalaxmi §
Department of Computer Engineering
V.E.S. College of Engineering
Mumbai University
Mumbai 400071
Maharashtra
India

Abstract

Users in the new age have countless choices of movies to watch, which creates the need for recommendation algorithms that will suggest a list of the best movies. The Recommendation algorithms will help users select the best movie to watch based on their personal choices and movie features. The first challenge among researchers is to find the most suitable dataset for the evaluation of recommendation algorithm performance. The work in the paper introduced a new MFRISE dataset to fulfil this challenge partially. The dataset is constructed using social data like Wikipedia, YouTube and other web sources with the help of web scraping data.

The proposed algorithm uses Collaborative as well as Content-Based Filtering to produce more accurate results. This architecture is based on the biclustering technique to improve the accuracy of the predicted ratings for better recommendations. The biclustering method involves simultaneous clustering of the user, as well as movie dimensions. The biclustering method will uncover the hidden insights from the intersection of two clusters. The method has produced more relevant recommendations by reducing error values in the prediction of user ratings. The RMSE for the new algorithm is observed as 0.83, which is

* *E-mail:* **mahesh.mali@nmims.edu (Corresponding Author)**
† *E-mail:* **dhirendra.mishra@nmims.edu**
§ *E-mail:* **m.vijayalakshmi@ves.ac.in**

better than that of many popular algorithms. The paper presents several experiments for analysing the proposed bi-clustering method using meta-features of the new proposed dataset. The future development of the biclustering method is suggested at the end of our paper.

Subject Classification: *Primary 68T99.*

Keywords: *Recommendation system, Clustering-based RecSys, Bi-clustering based recommender system.*

1. Introduction

A good movie recommendation is a movie that the user likes to watch. During the last decade, due to excessive information availability from the web and social media, the need for recommendation systems (RecSys) has continued to increase in the domain of movies. So, the most prominent market players, Amazon, Netflix, and Hotstar, are offering the best suggestions to users. A variety of algorithms are proposed to meet the needs of users by suggesting the best movie preferences in a wide range of applications. Finding the best movie to watch will explore recommender system evaluation metrics like accuracy, Precision, ranking, and execution time to increase the usefulness and quality of recommendations [1].

The current RecSys algorithms suffer from cold start issues for suggesting movies for new users without any rating history [2]. The data sparsity issue of RecSys happens because most viewers give a lesser number of ratings [3]. Scientists propose many new algorithms, but there are still many open research challenges to building a movie recommender system that is accurate and time-efficient [4].

1) Content-Based (CB) RecSys [4]: This RecSys model will make use of characteristics like the type of genre, duration, and country to make a movie profile. Movie similarity is calculated based on similarity measures.
2) Collaborative Filtering (CF) RecSys: This recommender system is based on user and movie interactions like ratings, feedback, etc., and User-Movies rating Similarity is used to generate the rating for the user [1].

This paper has implemented a new hybrid RecSys approach by combining the advantage of bi-clustering methods. The movies and users, both dimensions, are clustered to find groups of clusters with similar

interest patterns. It will help to minimise issues like cold start and data sparsity[4].

2. Augmented Dataset (MFRISE)

The RecSys performance is dependent on the dataset used to train an algorithm. The novel RecSys implementation is done using a new dataset extracted from MovieLens, IMDB, HetRec 2011, Wikipedia, and YouTube [5]. The dataset will now have multiple data points for analysing and recommending relevant movies [6].

- *Movie-related metadata:* The movie data is mainly extracted from the GroupLens dataset. The dataset has movie data coming from IMDb. The movieID is scrapped from the IMDb movie list. MovieID from IMDB was then used to get movie-related data such as directors, writers, cast, run time, votes, ratings, etc. [5]. The selected attributes are MovieId, Title, Genres, Critics Rating, and Critics Number of Reviews.
- *Wikipedia data:* The popularity of movies was collected. The popularity of movies was collected through the Wikipedia web page of the movie, and the links were extracted by Google search API [5]. The selected attributes are MovieId, Title, Wikipedia View Count, Release date, and Collection_amount.
- *Web data[8]:* Movie-related websites like Rotten Tomatoes IMDb are scraped for additional information like cast and crew, collection, studio, etc. These data are also used to confirm release dates, actors and other details. Some data were extracted from the HetRec 2011 dataset, which is available on the GroupLense website. The attributes are MovieId, Title & matched from above movie data, Actors & Number of actors, Directors & Number of actors, Tags & Number of times movie tagged in social comments.

2.1 *Data Augmentation to extracted data to generate the MFRISE dataset*

The raw movie data is retrieved from multiple platforms and is processed for uniformity and to remove the missing values [6]. A Python script was prepared to make the data uniform by removing missing data points. Some features are added to the dataset by extracting data using Python scripts. The extracted features are listed in Table 1.

Table 1
Extracted Features

Feature Name	Details
Movie Sum	The number of users who rated a particular movie
Movie Rating Mean	Average of users rating particular movie
Movie Release Month	Release month of the movie(Extracted)
Movie Release Year	Release year of the movie(Extracted)
Number of Users Rated	Number of movies rated by a particular user
Average User Ratings	Average movie rating by a particular user
Directors Movie Sum	Total number of movies made by the directors
Directors Rating Mean	Average of the ratings of movies by director
Actors Movie Sum	Average of total movie number by actor
Actors Rating Mean	Average of the ratings of movies by the actor

The correlation matrix is derived from movie ratings of actors, directors and other relevant data. The popularity of the director's actors is computed by taking the mean of the movie ratings in which they were involved. The critics ratings and tag relevance are also considered.

3. Bi-Clustering based Recommendation System

The proposed algorithm is a hybrid recommendation system using CB and CF-based methods using biclustering to predict user ratings [7]. The algorithm generates recommendations based on past similarities between user's ratings of movies [6]. For better recommendations, we can use user ratings data as explicit data and some other data like watch time [1]. The proposed biclustering-based recommender algorithm works in the following steps,

Step 1: Data Prepossessing

Step 2: Movie and User Clustering

Step 3: Find Intersection of Clusters as Co-Cluster

Step 4: Bi-clustering based predicted rating value (CB based)

Step 5: SVD algorithm based on predicted ratings (CF-based)

Step 6: Combine ratings and Present recommendations

Our research work introduced the bi-clustering method with hybrid recommenders in the recommendation system.

3.1 *Data Prepossessing*

The data is taken from the MRISE dataset, which has a set of users, movies, ratings, actors, directors and many such entities. The data is processed for error values and for checking the sparsity matrix of ratings. The cold users are identified to reduce the problem of cold recommendations.

3.2 *Movie and User Clustering*

Neighbourhood algorithms like KNN are good for clustering movie data sets. The numeric data can be handled very efficiently by k-means[7]. The k-proto algorithm is used to handle a combination of categorical and numerical data. The k-Means algorithm makes smaller clusters with greater efficiency for larger datasets[7]. The average rating of the user and movie cluster is computed. It acts like all users in the same cluster have a similar type of rating [6].

3.3 *Find Intersection of Clusters as Co-Cluster*

The movie and user cluster average rating values computed from the above step are used to calculate the average movie-user cluster (R) value, as shown in Table 2. The intersection of two ratings is computed by the weighted average method [8]. The user weight is the ratio of total users in the cluster to total users in the dataset. Similarly, movie weight is the ratio of total movies in the cluster to total movies in the corpus.

Table 2

Extracted Features

Total movies (mv) = 1000 Total Users (us) = 100		**MC 1** (R_m= 3)	**MC 1**	..	**MC n** (R_m= m)
		M=100 movies | $\mathbf{m} = \frac{100}{1000} = 0.1$	**250**	..	**x**
UC 1 (R_u= 2)	U=25 users $u = \frac{25}{100} = 0.25$	$R = \frac{3*0.1 + 2*0.25}{(0.1+0.25)} = 2.29$			Bi-Cluster n
UC 2	**57**				
..					
UC n (R_u= u)	$Y = \frac{u}{us}$				$R_p = \frac{m*x + n*y}{(x+y)}$

UC = User Cluster
MC = Movie Cluster
R_m = Movie Cluster Rating
R_u = User Cluster Rating

3.4 *Co-Clustering based predicted rating value*

The movie clusters and user cluster intersection form the smaller co-clusters, which is a group of users with similar interests. The rating computed in the above calculations of Table 2 is the predicted bicluster rating for each co-cluster. This rating will help to predict a list of movies best for co-cluster users.

3.5 *SVD algorithm based on predicted ratings*

The Singular Value Decomposition means the SVD algorithm [9] is used to reduce the size of the matrix using linear algebra methods [12]. The rating matrix has a row that presents users (U) and columns that represent movies (V). The SVD algorithm is used for the implementation of a hybrid recommender system as it is faster and more accurate for movie datasets, as per our last paper [1]. The matrix equation used to calculate the latent vector S is given as ($R = U\ S\ V^{T}$).

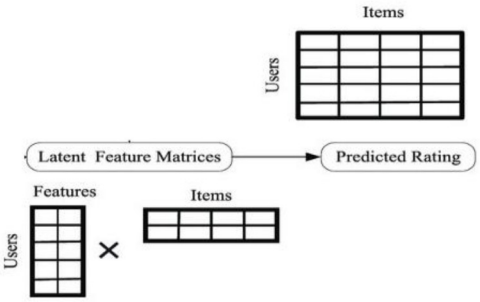

Figure 1

SVD Algorithm to Predict Rating

The SVD algorithm shown in Figure 1 is used to predict the rating based on the matrix factorisation method [8]. The rating is CF-based recommendation system ratings.

3.6 *Combine CB and CF-based Ratings*

The predicted ratings from bi-cluster-based (Section 3.4) and SVD algorithms (Section 3.5) are added to get the combined ratings of the proposed hybrid algorithm [10,11]. The proposed algorithm makes use of CB and CF methods, so it is also called the hybrid recommendation method.

3.6 *Present Recommendations*

For any user, the recommendations are generated by organising the combined rating in decreasing order, and *Top-k* movies are presented to the user as a list of recommendation results.

4. Experimental setting and Evaluation metrics

We build experiments based on MFRISE datasets, HetRec Dataset and MovieLens datasets. Data sparsity in Table 3 indicates the percentage number of ratings available for all user-movie combinations. The data Sparsity issue indicates very few viewers give a rating after watching.

Table 3
Data sets used for experimentation

Dataset	movies	users	ratings	Data sparsity (%)
ml-latest	9742	610	100836	1.7
ml-25M	62423	6545	25000095	6.12
MFRISE* (New)	10197	2113	855598	3.98

The K-Means clustering is used to find groups of similar movies and groups of similar users. The clustering is done for users and then also for movies to get an advantage of bi-clustering by generating more accurate results. The evaluation metrics for validating recommendation results are used to analyse the performance of recommenders.

4.1 *MAE (Mean Absolute Error) Value*

MAE calculates the average absolute difference between the predicted rating values using the method and the actual observed rating values in the dataset[6].

$$MSE = \frac{1}{N} \sum_{\hat{r}_{ui} \in N} (r_{ui} - \hat{r}_{ui})^2 \tag{1}$$

Where, r_{ui} = Predicted Rating by algorithm | $\hat{r}_{ui}$ = Actual Rating

4.2 *RMSE*

RMSE calculates the square root of the average squared differences between the predicted rating values using the algorithm and the actual observed rating values in the dataset [3,4].

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{\hat{r}_{ui}\in N}(r_{ui} - \hat{r}_{ui})^2} \tag{2}$$

4.3 *Processing Time*

RecSys algorithm speed is very important for producing faster recommendations. The processing time is the time required for predicting the rating and producing recommendations to the user. The time required for running the algorithm and generating recommended movies from data is processing time[7].

5. Performance Evaluation of BiClustering-based Recommender System

The performance of the proposed system is evaluated in this section of the work, and experimental results are obtained using the extracted MFRISE dataset. The MRISE dataset contains 2113 users, 10197 movies and 855598 user ratings. The movie and user data is clustered with the help of the K-means method. The elbow curves for movies and users are shown in Figure 2. The within-cluster sum of squares (WCSS) is used in clustering algorithms, particularly in K-Means clustering, to determine the optimal number of clusters. In this method, data is clustered in users and movies simultaneously, so the approach is called the bi-clustering approach. The new methods MAE, RMSE, execution, and processing time are compared in Table 4 to popular algorithms to understand performance. The algorithms were compared in some earlier literature [2], which is used here for further comparison with our work.

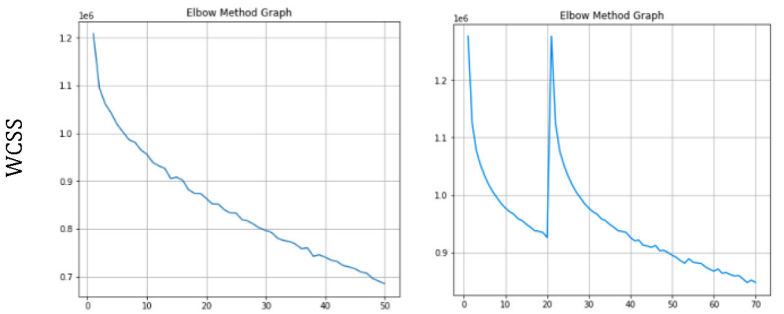

Figure 2

Elbow Curve for User and Movie Clustering (Optimum Cluster Size)

Table 4
Comparative Analysis of the Proposed System

Algorithm	MAE	RMSE	Processing Time
Base Line	1.5200	1.2224	0.22
KNN	0.9793	0.7736	2.70
CO-CLUSTERING	0.9654	0.9654	1. 02
SLOPE ONE	0.9434	0.7452	0.55
BiClustering * (Proposed)	0.8303	0.6801	0.21

The proposed algorithm was experimented with on the MFRISE dataset for comparative analysis. The RMSE of the new algorithm given in the work is 0.83, which is better than the Baseline, KNN, Co-clustering and Slope-one algorithms [7]. The results of our work are better in movie recommendations than other RecSys. The results are improved due to the combination of the SVD method and the bi-clustering method.

The comparison with respect to processing speed is also comparable or at par with existing methods irrespective of the Baseline algorithm. In our work, processing time is more than that of other RecSys, but this time is offline time. The RMSE and MAE results indicate that our proposed work is better in the case of user recommendation. Our work can also be used to address the challenge of generating effective group recommendations. We have a group recommendation issue as the next research gap, which can be addressed with the help of the bi-clustering method in the next paper.

6. Conclusion

The bi-clustering-based Recommender system is implemented using our MFRISE dataset. This dataset with extracted features is our research contribution to the recommender systems. After the experimentation, we conclude that the efficiency of the recommender algorithm is largely dependent on the dataset used for experimentation. The proposed algorithm is evaluated using standard evaluation metrics, including MAE, RMSE, and processing time (Test & Fit Time), as shown in Figure 3.

The bi-clustering-based recommender system shows RMSE and MAE error values of 0.83 and 0.68 on the MFRISE dataset, which are found to be lower than other recommenders. The system is also tested with other datasets, which concludes that our system works at par with many popular recommender systems. The error values of the proposed system are better

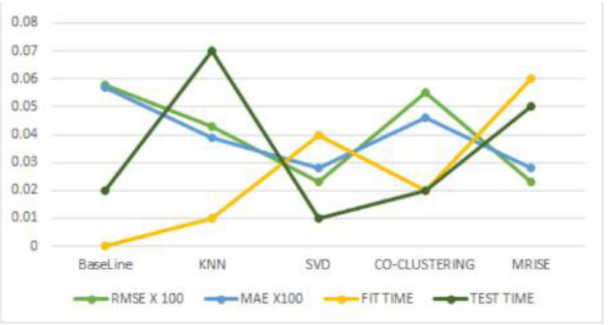

Figure 3

Performance Comparison of MFRISE

than those of other existing systems. The reason for the better error value is that the bi-clustering method has predicted more relevant results. The processing time is far better than the existing methods due to offline bi-clustering for rating predictions. Finally, It is observed that the proposed recommender algorithm with bi-clustering has improved the accuracy and time efficiency when tested on the extracted MFRISE dataset.

References

[1] Mali, Mahesh, Dhirendra S. Mishra, and M. Vijayalaxmi. "Multifaceted recommender systems methods: A review." *Journal of Statistics and Management Systems* 23.2 : 349-361 (2020).

[2] Chhikara, Monika & Malik, Sanjay Kumar (2023). A recommendation system for an online social semantic network using knowledge-based, content-based and collaborative filtering, *Journal of Information and Optimization Sciences*, 44:4, 795–806, DOI: 10.47974/JIOS-1356Mehta Y., Singhania A., Tyagi A., Shrivastava P., Mali M. (2020)

[3] Abdulhussein, Niran A. & Obaid, Ahmed J. A landscape view of news recommendation systems based on MIND dataset, *Journal of Discrete Mathematical Sciences and Cryptography*, 26:6, 1705–1716 (2023).

[4] Mehta Y., Singhania A., Tyagi A., Shrivastava P., Mali M. A Comparative Study of Recommender Systems. In: Kumar A., Paprzycki M., Gunjan V. (eds) ICDSMLA 2019. Lecture Notes in Electrical Engineering, vol 601. Springer, Singapore (2020).

[5] F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. A.C.M. Trans. Interact. Intell. Syst. 5, 4, Article 19 (January 2016), 19 pages. https://doi.org/10.1145/2827872

[6] F.O. Isinkaye, Y.O. Folajimi, B.A. Ojokoh, Recommendation systems: Principles, methods and evaluation, *Egyptian Informatics Journal,* Volume 16, Issue 3, Pages 261-273, ISSN 1110-8665, (2015).

[7] Mali, Mahesh, Dhirendra Mishra, and M. Vijayalaxmi. "Benchmarking For Recommender System (MFRISE)." 3C TIC. Cuadernos de desarrollo aplicados a las TIC, 11(2), 146-156. ISSN: 2254-6529

[8] Ricci, Francesco, Lior Rokach, and Bracha Shapira. "Recommender systems: introduction and challenges." Recommender systems handbook. Springer, Boston, MA, 1-34 (2015).

[9] Fayyaz, Zeshan, et al. "Recommendation systems: Algorithms, challenges, metrics, and business opportunities." *Applied Sciences* 10.21 : 7748 (2020).

[10] Hug, Nicolas. "Surprise: A Python library for recommender systems." *Journal of Open Source Software* 5.52 : 2174 (2020).

[11] Ahuja, Rishabh, Arun Solanki, and Anand Nayyar. "Movie recommender system using k-means clustering and k-nearest neighbor." 2019 9th International Conference on Cloud Computing, IEEE (2019).

[12] Mehta, Rachana, and Keyur Rana. "A review on matrix factorisation techniques in recommender systems." 2017 2nd International Conference on Communication Systems, Computing and I.T. Applications (CSCITA). IEEE (2017).