e-ISSN : 0976-5166
p-ISSN : 2231-3850

WEIGHTED HYBRID MODEL FOR IMPROVING PREDICTIVE PERFORMANCE OF RECOMMENDATION SYSTEMS USING ENSEMBLE LEARNING

Raghavendra C K

Research Scholar, Computer Science & Engineering, SJB Institute of Technology, Bangalore, Affiliated to Visvesvaraya Technological University,

Belagavi, Karnataka 590018, India

Assistant Professor, Computer Science & Engineering, BNM Institute of Technology, Bangalore, Karnataka 560070, India

raghav.ck.clk@gmail.com

Srikantaiah K C

Professor, Computer Science & Engineering, SJB Institute of Technology, Bangalore, Affiliated to Visvesvaraya Technological University,

Belagavi, Karnataka 590018, India

srikantaiahkc@gmail.com

Abstract

Recommending items to the users based on their likes and dislikes has become a need in e-marketing sectors. Content Based Filtering (CBF) and Collaborative Filtering (CF) are the two common methods for generating recommendations both uses different set of input sources to make recommendations. CBF method requires more information on items rather than user preferences. Similarly, to generate accurate predictions CF needs large dataset with more number of active users. To overcome the limitations of traditional methods and to improve the accuracy of predictions a weighted sum hybridization method is proposed by applying ensemble learning approach to find best combination of models which improves predictive performance. The recommendations using baseline model, CBF models and CF models are individually obtained and best two models are used for weighted hybridization method where each of the these approaches are experimented with different weights to obtain the best hybridized recommendations. Experimental results shows hybrid approach gives better predictions with Root Mean Squared Error (RMSE) value of 0.88 and Mean Absolute Error (MAE) value of 0.67 which are less compared to all other approaches.

Keywords: **Baseline model; Collaborative filtering; Content based filtering; Ensemble learning; Hybrid filtering; Movie recommendation system.**

1. Introduction

In recent years, there has been an exponential rise in the volume of information available online due to electronic resources and online services. This overload of information created a potential filtering problem in delivering required information for users. This emphasize the necessity of having automated systems which can extract relevant information which is unseen by user’s. Such Systems are commonly referred as Recommender Systems (RS). This system performs well in filtering the products, music, movies and so on.

To boost the willingness of customers in order to purchase items on an e-commerce website, a system is very much required to provide information of unseen products. One of the approaches for managing the overload problem on the customer's side is to use a recommender system. A recommender system can increase the business of e-commerce sites by increasing sales in three ways: turning browsers into purchasers, cross-selling, and loyalty. People feel that a recommender system can benefit e-commerce businesses in assisting clients in selecting the most appropriate products or services for their needs. The global e-commerce giants, such as Amazon.com and

Ebay.com, have embraced the recommender system as one of their website's standout features. The wide varieties of choices offered in most of the e-commerce websites and the heterogeneous and dynamic nature of these websites often leave the users overwhelmed. The recommender system acts as a guide in this scenario helping the users by recommending appropriate products.

Content based and collaborative filtering are two traditional methods of RS, content based method makes predictions based on past behavior of user i.e. items similar to items purchased in past whereas, collaborative filtering will neither require users history nor content it mainly works on finding similar items or similar users *i.e.,* similarity matching. Similarity measures are the core component of RS and the performances of these metrics directly impact the accuracy and recommendations getting generated.

Both approaches have drawbacks; CBF can recommend a new item, but it requires more user preference data to incorporate the optimal match. Similarly, collaborative filtering requires a large dataset containing active users who have previously rated an item in order to provide effective predictions. To address this issue, we suggest a hybrid technique in which best two approaches are combined by assigning optimal weights to obtain the most efficient recommendation.

The remaining part of the paper is organized as: Section II highlights related work with their outcomes. Section III Methodologies, Section IV states the problem with the proposed system architecture. Section V discuses about experimental set up and results and at last a conclusion and further enhancement is given in section VI.

2. Related Work

From the last two decades, several researches have been working in the area of recommendation system and their applications and proposed several ways of recommending items using CBF, CF (User-item, User-user), context based and hybrid methods to solve this problem.

Authors [Venugopal *et al.,* (2018)] given comprehensive study on various strategies applied for developing RS. They also outlined advantages and disadvantages of various RS currently in use and also provided further directions of research in this area. [R. C K *et al.,* (2021)] conducted experiments on analyzing performance of various similarity metrics used in building RS. A hybrid RS [Mohamed Elyes Ben Haj Kbaier *et al.,* (2017), R. C K *et al.,* (2022)] are proposed for tourism and books is implemented which uses two hybridization techniques i.e. switching and weighted to combine three recommender filtering methods. In weighted technique a linear programming model is applied for setting weights and values automatically. The approach used has the best accuracy for anticipated rates, according to many evaluation metrics. The findings are much improved when the two hybridization strategies are combined.

The work carried out [Sanya Sharma *et al.,* (2017)] here uses cosine similarity as the similarity measure and uses composite search algorithm to combine the candidate algorithms. The results are then compared with a website called limeroad.com. Users will receive better recommendations as a result of this. The user's history is crucial in determining the user's interests. The user's search history is not taken into account by Composite Search. Researchers employed weighted parallel hybrid techniques, which are a blend of content-based filtering and collaborative filtering, to construct recommender systems for an e-commerce platform in Indonesia. Weighted parallel hybridization, when employed alone, can overcome the limitations of the two techniques. The system is unable to make recommendations to new users or to recommend the new products.

The use of tag and rating [Chunxia Zhang *et al.,* (2018)] information to calculate user similarity is presented as an improvement to the collaborative filtering algorithm. Furthermore, it incorporates a hot-item penalty into the process of determining a user's similarity in order to mitigate the impact of the hot item on their co-rated products. It also considers the time component when calculating user interest, as it varies over time. [Qin Zhao *et al.,* (2018)] a novel function is utilized to more precisely compute the similarity between two items, overcoming the shortcoming of the content-based method. The semantic similarity of two objects is seen to be a form of virtual link between them. As a result, even if there is no physical link path between them, the link can be computed.

Matrix factorization [Chia-Yu Lin *et al*., (2018)] recommendation strategies are explored from two perspectives: first, from the perspective of implicit feedback, and then, to address the implicit feedback problem, the original cost function is changed. [M. T. Himel *et al*., (2017)] developed Recommendation systems for movies by co relating user's information and weights using a formula. It forms movie clusters using the k-means method and recommends the cluster with the highest mean movie rating to the user. [M. K. Kharita *et al.,* (2018)]. In this paper, a movie recommender system is described, which essentially uses an item-based collaborative filtering technique to generate dynamic item recommendations that learn from positive feedback.

The performance of Collaborative Filtering and Hybrid-based techniques in producing movie suggestions is analyzed and compared in this research [N. Ifada *et al.,* (2020)]. In terms of Precision and NDCG metrics, the Collaborative Filtering-based strategy always outperforms the Hybrid-based approach at any top-N position.

These findings suggest that the Hybrid approach to movie recommendation does not always outperform the Collaborative Filtering strategy. [Q. Du *et al.,* (2021)] to begin, they employed KNN to gather similar data from the target user and item in order to make a preliminary forecast. Simultaneously, it mined users' fundamental information features as well as possible user and item attributes from the original data. Then they tried to use the XGBoost, LightGBM, and CatBoost algorithms to develop regression models, all of which were based on the gradient boosting decision tree concept. Finally, the trained model was used to forecast the target user's movie rating, and a suggestion list was constructed based on the expected findings.

[C. Cai and L. Wang (2020)] Proposed a clustering and reclassification method for movie recommendation. They applied improved K-means algorithm to cluster according to the scores of similar users. [M. Gupta *et al.,* (2020)] This method employs cosine similarity, k-nearest neighbour, and a collaborative filtering mechanism to avoid the drawbacks of content-based filtering. Despite the goal for Euclidean distance, cosine similarity is used since cosine angle accuracy and movie equidistance are virtually equivalent.

A feature combination hybridization method [Wairegi *et al*., (2020)], which will entail using feature ratings obtained by the use of the collaborative filtering approach to enhance the content-based recommendation of articles. [Singh, P.S *et al.,* (2021)] proposed a RS for travelers called Advanced Tour Sequence Recommender (ATSR) which outperforms the other traditional methods.

3. Problem Definition and Architecture

Traditional RS's lacked the ability to provide relevant recommendations based on the user's preferences, and they simply recommended the most popular things. The proposed hybrid technique solves the difficulties of a typical recommendation system and can be utilized to generate an optimum list of recommendations. The hybrid approach is defined and its architecture is explained in this section.

3.1. *Problem Definition*

Given a movieLens dataset with details of users, movies and ratings of n users for m movies. Our objective is to design and develop a weighted hybrid model by combining content based and collaborative filtering for recommending movies and also to conduct performance analysis of various RS models.

3.2. *Architecture*

The main components include: feature extraction and data preprocessing, Content based and collaborative weighted hybrid model and evaluation. Figure 1 shows the architecture of the hybrid recommender model.

3.2.1 Feature Extraction and Data Preprocessing

In this step latest Movie Lens small dataset is obtained from kaggle website. We transformed data into appropriate format by removing unwanted features and redundant information present in the dataset. Based on userId we split 70% of dataset into training data and 30% into testing data by using scikit-learn library.

3.2.2 Baseline Model

It is a basic recommendation model based on popularity of movies. Here, we calculate missing rating value by finding the average rating of the movie. This model is non personalized and will not considers user preferences but works on idea that most users rate movies close to the ratings of other users i.e., close to the average rating value. The missing rating *R (p,j)* of user *p* for movie *j* is predicted as:

$$\boldsymbol{R(p,j) = \frac{1}{q}\sum_{i=1}^{m} R(i,j)} \qquad (1)$$

Where, *m*- number of users, *q*-number of users rated for the movie, *R(i,j)*- rating of i^{th} user for j^{th} movie.

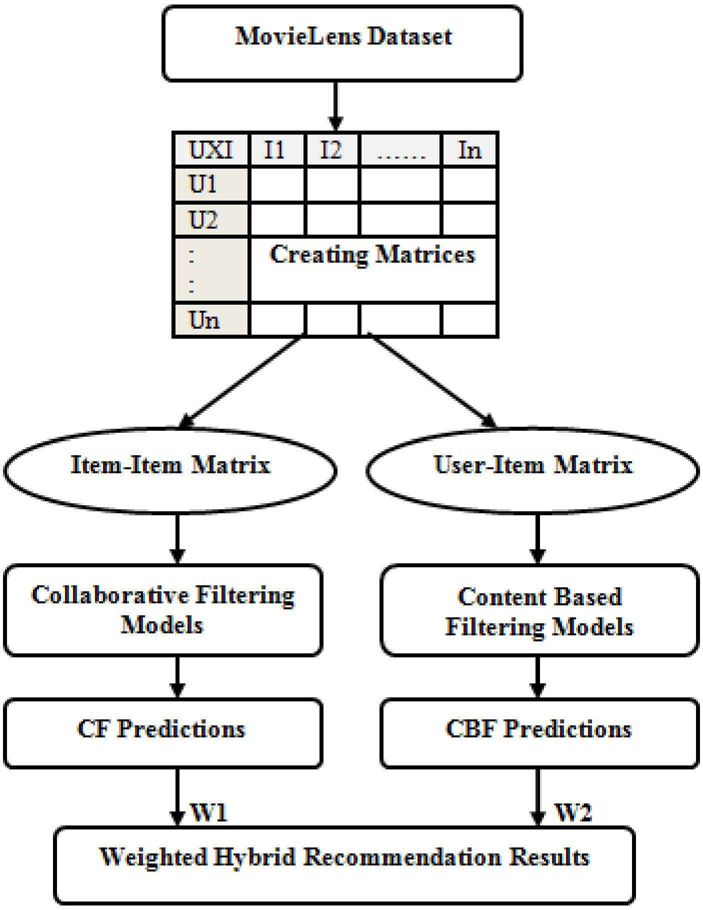

Fig.1. Architecture of the hybrid recommender model

Movies Users	movie 1	movie 2	movie 3	movie4	movie 5	....	movie n
user1	4	4.5	3.5	4	3.5		4
user2	3	4	3.5	4	3		3
user3		4	3	4.5			2.5
user4	4.5	5	3.5	5	2.5		3.5
user5	3.5	4.5	3.5	3	2.5		3.5
: :	: :	: :	: :	: :	: :	: :	: :
user m	5	4	3	4	3.5		5

Table 1. User to Movie rating matrix R

Consider an example of user to movie matrix shown in Table 1, which contains ratings of m users on n movies. Consider row number 3 *i.e.,* user 3 ratings here, empty cells indicates the movies which are not seen or rated by user 3. In the rating matrix R (3,1) and R(3,5) are missing which indicates user 3 have not rated movies 1 and 3. Here, we have to find missing values i.e., predicted ratings of user 3 for movies 1 and 3 *i.e.,* R (3,1) and R(3,5) using Equation (1), by finding average ratings of other users who rated those movies. In the matrix shown in Table 1, assuming q users have rated those movies then R(3,1) and R(3,5) is calculated as average rating of q users for movies 1 and 3. After predicted the ratings, movies which are not seen by the users are recommended based on popularity because highly rated movies will have higher rating average. Here movie 1 will have higher rating average compared to movie 5 so probability is more that movie 1 will be part of the recommendation list for user 3 compared to movie 5. The detailed algorithm for baseline model is given Algorithm 1.

```
Algorithm1: Baseline(R, p, j, m)
//Purpose:
//Input: R: Rating matrix, p:p^th user, j:j^th movie, such that R(p,j) is the missing
rating value, m-number of users
//output: R(p,j)- Predicted rating p^th user and j^th movie
R(p,j)=0
q=0
for(i=1;i<=m;i++)
        if( R(i,j) is NOT NULL)
                R(p,j) = R(p,j)+R(i,j)
                q++
        End If
End For
R(p,j)=R(p,j)/q
Return R(p,j)
```

3.2.3 Content based filtering (CBF)

These recommender systems suggest items based on the content i.e., things that are similar to items purchased by the user in the past. In this technique, similarity scores between two items are calculated based on which similar items are suggested to the active user. Any attribute or feature of the item can be considered while calculating similarity. For movies, we can use various features like genres, release year, tags and reviews to generate content profile. In our work, we used genres feature of movies as content profile to build users and movies vectors for recommending movies.

MovieId	Title	Genres
1	Toy Story (1995)	['Adventure', 'Animation', 'Children', 'Comedy', 'Fantasy']
2	Jumanji (1995)	['Adventure', 'Children', 'Fantasy']
3	Grumpier Old Men (1995)	['Comedy', 'Romance']
4	Waiting to Exhale (1995)	['Comedy', 'Drama', 'Romance']

Table 2. Movies Table

Generating Movie Vector (MV)

Consider the Table 2, which describes the movies information stored in moives.csv file of MovieLens dataset. It has three attributes MovieId is unique identifier for movies, Title indicates title of the movie and Genres indicates the type of genres to which movie belongs to. Here genres attribute is used for developing *MV*. Consider a list of *m* users i.e., $U= \{u_1,u2,u3\ldots\ldots u_m\}$, list of n movies., $M=\{m1,m2\ldots m_n\}$ and Rating matrix *R(mXn),* Movie vector (MV) is a *nXg* Boolean matrix and is defined as,

$$\boldsymbol{MV(i,j)} = \begin{cases} \boldsymbol{1\ if\ jth\ genre\ is\ in\ ith\ movie} \\ \boldsymbol{0\ otherwise} \end{cases} \tag{1}$$

MovieId	Title	Genres	Movie Vector
1	Toy Story (1995)	['Adventure', 'Animation', 'Children', 'Comedy', 'Fantasy']	[1.1.1.1.1.0.0.0.0.0.0.0.]
2	Jumanji (1995)	['Adventure', 'Children', 'Fantasy']	[1.0.1.0.0…]
3	Grumpier Old Men (1995)	['Comedy', 'Romance']	[0.0.0.1.0.0.1]
4	Waiting to Exhale (1995)	['Comedy', 'Drama', 'Romance']	[0.0.0.1.0.0.1….]

Table 3. Movies Table with Movie Vector

In the Table 3, Movie Vector attribute which is added into the movies table indicates the list of zeros and ones of length equal to number of unique genres present in the entire dataset. Consider MV of movie 1 [1,1,1,1,1,0,0,0,0…] where one indicates that movie belongs to a particular genre and otherwise it is zero. Here, first five sequence of one’s indicates that movie 1 will fall under adventure, animation, children, comedy, and fantasy genres whereas next zeros indicates movie 1 will not fall under drama, romance and other genres.

Generating user vector (UV)

UV is a list of average ratings of a user for each genres combined from the list of movies which are rated by user. Consider Rating matrix in Table 4 indicating ratings of users for movies watched by them, from this table we derive Rating vector *RV(i,j)* by multiplying with row rating *R(i,j)* with movie vector of j^{th} movie *MV(j)* i.e., *RV(i,j) = R(i,j) X MV(j)*. The resulting *RV* is added into the rating table as shown in Table 5

Movies Users	**movie 1**	**movie 2**	**movie 3**	**movie 4**
user1	3	4		
user2			5	4
user3		2		

Table 4. Rating Matrix

UserId	**MovieId**	**Rating**	**Rating Vector**
1	1	3	[3.3.3.3.3.0.0.0…..]
1	2	4	[4.0.4.0.0….]
2	3	5	[0.0.0.5.0.0.5…]
2	4	4	[0.0.0.4.0.0.4….]
3	2	2	[2.0.2.0.0….]

Table 5. Rating Matrix with Rating Vector

After obtaining the *RV* of each user for each movie we group the Table 5 based on user to get the total number of movies watched by user with, count of each genre i.e., Count Vector *CV* and total sum of the rating vector. Once we get the total sum of *RV* we obtain *UV* by dividing the total rating of particular genre in *RV* with its value in *CV* as given in Algorithm 2. Table 6 indicates *UV* of each user.

Finally, Predicted rating of *P(i,j)* for each user *i* and movie *j* is obtained by using Eq. (3).

$$\boldsymbol{P(i,j) = \frac{1}{q}\sum_{k=1}^{g} UV(i) X MV(j,k)} \tag{2}$$

Where, *q* is number of movies watched by user and *g* is number of genres.

```
Algorithm2: UV(RV, MV)
//Purpose:
//Input: RV: Rating vector, MV: Movie Vector
//output: UV- User Vector for each user u in U
// UV and CV are one dimensional array of size g and their elements are
//initialized to zero
For each genre k in G
          UV(k)=0, CV(k)=0
End for
For each user u_i belongs to U
For each movie m_j belongs to M
        If R(i,j) is Not NULL
                For each genre k in G
                        UV(i) = UV(i) + R(i,j) * MV(j,k)
                        CV(i) = CV(i) + MV(j,k)
                End for
        End If
        UV(i) = UV(i) / CV(i)
        End For
End For
Return UV
```

Consider an example, from Table 3. User 3 have not watched movie 1 so the predicted rating P(3,1) is calculated by multiplying corresponding elements of UV(3) [2,0,2,0,0,0……] with MV(1) [1.1.1.1.1.0.0.0.0.0.0.0.] which is equal to [2*1+0*1+2*1+0*1…..=4] 4/2 where 2 is number movies watched by user 3. Similarly predicted ratings of other movies for user 3 are calculated and listed in descending order of predicted ratings.

UserId	User Vector	Count Vector
1	[3.5, 3, 3.5, 4, 3.5, 0……]	[2, 1, 2, 1, 1, 0, 0…]
2	[0, 0, 0, 4.5, 0, 4……]	[1, 0, 1, 1, 0, 0, 1….]
3	[2, 0, 2, 0, 0, 0…]	[1, 0, 1, 0………]

Table 6. User vector and count vector of each user

3.2.4 Collaborative filtering (CF)

In this method, Neighborhood based approach is applied for generating recommendations by finding similar users or similar items. Based on Neighborhood approach CF are of two types user-user based or item-item based recommendation system. For both approaches user to item matrix R as shown in Table 1 is given as input. Figure 2 gives Architecture of collaborative filtering.

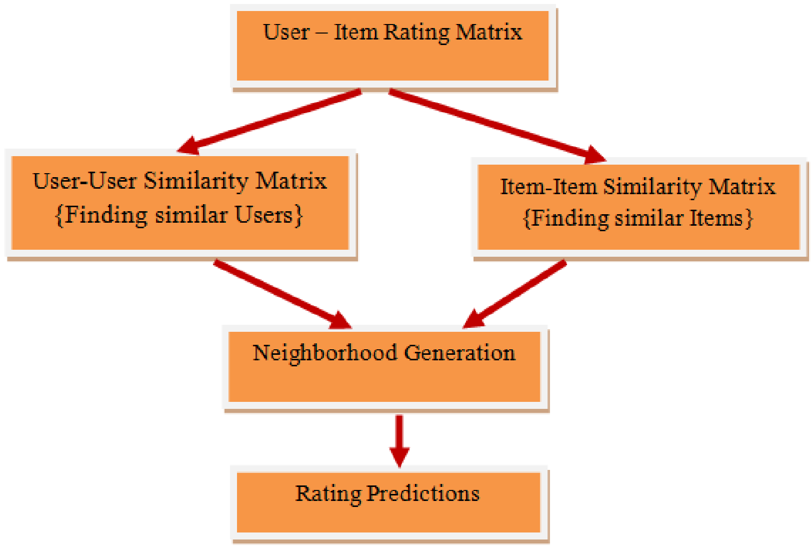

Fig. 2. Architecture of CF

In user based CF approach, the neighborhood of user is defined by similar users who given similar ratings for the movies rated by the user and user to user similarity is found by using cosine similarity metric . The predicted rating *P(i,j)* of user i for movie j is calculated using Eq. (4).

$$P(i,j) = \frac{\sum_{v \in K(i)} r_{v,j} * sim(v,i)}{\sum_{v \in K(i)} sim(v,i)} \quad (4)$$

Where, *K(i)* is list of *k* nearest users for user *i*, *v* is user who belongs to list of similar user for user *i*, $r_{v,j}$ is rating of user v for movie j and *sim(v,i)* represents similarity between user *v* and *i*.

In item based CF approach, the neighborhood of item is defined by movies which are having similar ratings for the movies rated by the user and item to item similarity is found by using cosine similarity metric. The predicted rating *P(i,j)* of user *i* for movie *j* is calculated using Eq. (5).

$$P(i,j) = \frac{\sum_{v \in K(j)} r_{i,v} * sim(v,j)}{\sum_{v \in K(j)} sim(v,j)} \quad (5)$$

Where, *K(j)* is list of *k* nearest movies for movie *j*, *v* is movie which belongs to list of similar movies for movie *j*, $r_{i,v}$ is rating of user i for movie v and *sim(v,j)* represents similarity between movie *v* and *j*.

In both approaches, predictions of a target user i.e., *P(i,j)* is obtained by finding ratings of similar users or similar movies based on the Equation 5 and 6 for all the movies which are unseen by the user . After finding predictions, movies are sorted based on the ratings and top n movies in the list with higher rating values are recommended for the user.

3.2.4 Hybrid technique

In recent times, hybrid models are of greater use because of more accurate recommendations compared to traditional models. Hybrid recommender systems are combination of two or more models where predictions of two, three, or more filtering algorithms are combined to obtain final predictions. These recommender systems integrate the good features of each technique to provide recommendations more effectively. Hybrid filtering is divided into feature augmentation, weighted, cascade, mixed, switching, and meta-level based.

In this work, weighted hybridization is applied in this weights are allotted to each technique based on their performance. This technique finds combined predictions by summing the final predictions retrieved from all models used in hybridization based their weights. The detailed algorithm for hybrid model is given Algorithm 3.

Algorithm3: Hybrid(PR_1,PR_2..PR_n, W[ ])
//Purpose:
//Input: *PR1*: Combined predictions of Model1, *PR2:* Combined predictions of Model2, n: number of models considered for hybridization, *W[ ]:* array of weights of length n.
//output: *CP(p,j)*- combined prediction for p^{th} user and j^{th} movie
for each user p in U:
for each model k in K:
    CP(p,j) = CP(p,j) + R_k(p,j)*W_k
End for
End for
Return *CP*

The algorithm takes inputs of predicted ratings of individual models and an array of weights for each model. Predicted ratings of all the models are combined by applying suitable weights to find missing ratings of movies which are not seen by user. After finding combined predictions, Hybrid combination which gives better results is used to generate recommendations for target user.

4. Experimental results and discussions

4.1. *Experimental set up*

The suggested recommendation algorithms are implemented on a Windows 10 Home Premium computer with an Intel Core i5 processor running at 3.38 GHz and 8.0GB of RAM, and a jupyter notebook. Experiments are conducted using the MovieLens benchmark dataset, which contains 1 lakh ratings for 1642 films from 943 individuals.

4.2. *Performance evaluation metrics*

To evaluate the predictions of recommender systems RMSE and MAE values are calculated.
Root Mean Square Error (RMSE): Estimates the rate of error while predicting a not rated item for an active user. The equation is as follows

$$RMSE = \sqrt{\frac{\sum i, j \in X \left| r_{ij} - r'_{ij} \right|^2}{|X|}} \tag{6}$$

r_{ij} is the actual rating, r'_{ij} is predicted rating, $|X|$ is test data.
Mean Absolute Error (MAE): It's the average of absolute errors. Absolute error is calculated as differences between predicted value and actual value.

$$MAE = \frac{1}{n}\sum_{j=1}^{n} |p_j - a_j| \tag{7}$$

Where, n is number of samples, p_j is predicted value and a_j is actual value.

4.3. *Performance Evaluation of Various Models*

4.3.1 Baseline model

This Model predicts the missing rating based on popularity of a movie. It fills the missing rating based on average rating of the movie. RMSE and MAE values for baseline model are shown in Table 7.

Recommender System	RMSE	MAE
Baseline Model	0.9694	0.75049

Table 7. Results of baseline model

4.3.2 Content based models

Content based model is experimented by following two approaches:

(1). Generating User profile and Movie profile based on genres feature of movies. Using Genres feature user vector and movie vector is generated and Similarity is calculated based on these vector comparison.

(2). Experimented with various machine learning algorithms present in scikit- learn library. Experimental results shows that support vector regression gives better results compared to other models as shown in Table 8.

Recommender System	RMSE	MAE
CBF based on movie genre	0.925339	0.713335
Linear regression	0.965603	0.738663
Lasso	0.943501	0.734317
K Nearest Neighbor Regressor (k=5)	0.971668	0.747365
K Nearest Neighbor Regressor (k=10)	0.939256	0.722360
Random Forest Regressor (RFR)	0.973810	0.746850
Support Vector Regressor (SVR)	0.927255	0.697850

Table 8. Results of CBF models

Content based filtering based on movie genre and SVR have lower RMSE values and will give better results when compared to other models. Therefore we can consider either of the models for building hybrid model.

4.3.3 Collaborative Filtering Models

We experimented both user based and item based CF models by finding user to user similarity and movie to movie similarity using cosine similarity and KNN model is applied for generating nearest neighbors for both users and movies. We experimented by changing the number of neighbors to find the optimal value for K to get better results as shown in Figure 3 and 4. Results shows that for users based approach at K = 20 and for item based approach at K =10 we got lesser values for RMSE and MAE.

Experimental results shows that item-item based CF outperforms user-user based CF in all variation of K values. Similarity between items is much stronger compared to users because number of items are comparatively less than users with less overlapping between items features.

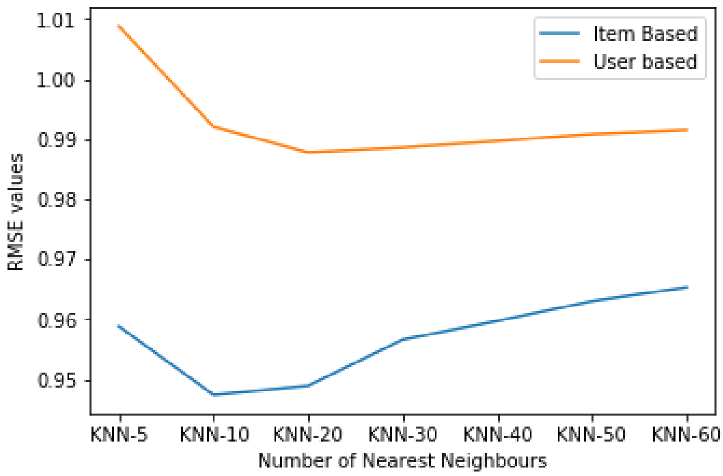

Fig. 3. Effect of K value on RMSE values for user and item based CF

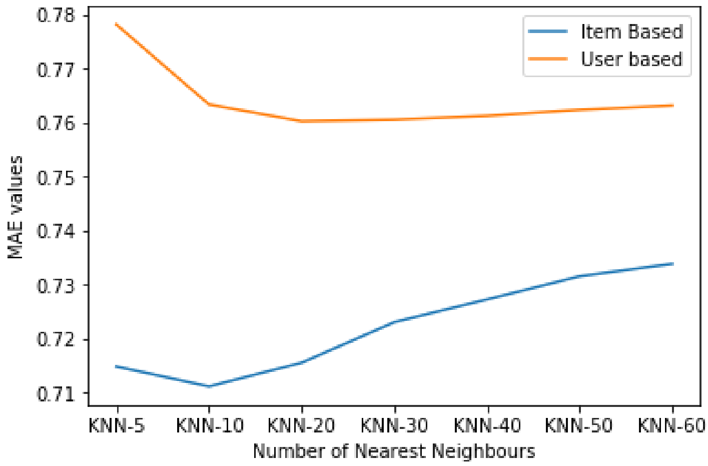

Fig. 4. Effect of K value on MAE values for user and item based CF

Recommender System	RMSE	MAE
Item Based CF with K = 10	0.9474	0.7112
User Based CF with K = 20	0.987820	0.760323

Table 9. Results of CF models

4.3.4 Hybrid model

Weighted hybridization approach is used to build hybrid model. Combined predictions is obtained by considering predictions of each model and summing them by assigning appropriate weights for each model. As per our experimentation, CBF and item based CF models gives better results compared to other models so we considered those two models for hybridization. The proposed hybrid recommender system which is a combination of CBF and item based CF is experimented with weights (0.5, 0.5), (0.6, 0.4), (0.4, 0.6) and (0.7, 0.3). At weights of 0.6 for CBF and 0.4 for CF hybrid model shows the better results in comparison with other combination of weights as shown in Table 10. The proposed hybrid model gives better results compared to all other models at (0.6, 0.4) weights as shown in Figure 5.

Hybrid Model (CBF & Item Based CF)		RMSE	MAE
Weights of CBF	Weights of CF		
50%	50%	0.8837	0.6767
60%	40%	0.8803	0.6722
40%	60%	0.8932	0.6855
70%	30%	0.8831	0.6718

Table 10. RMSE and MAE values of hybrid model

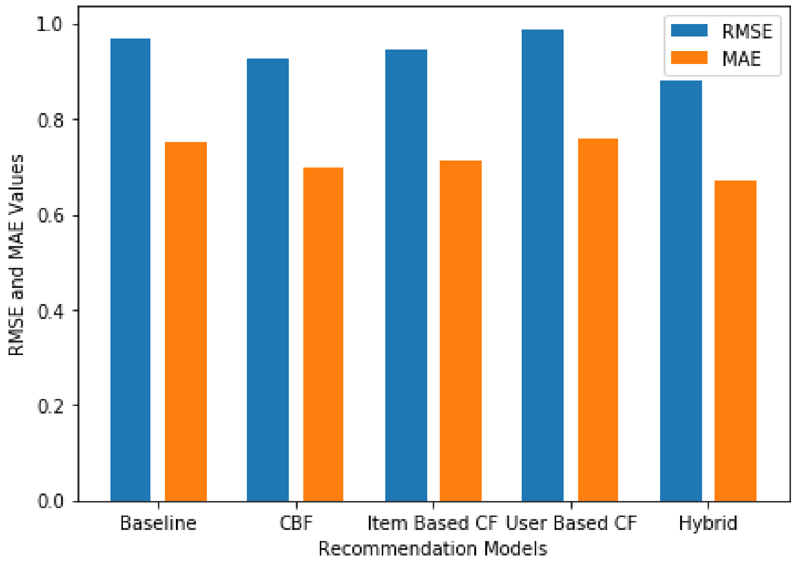

Fig. 5. Performance of various recommendation models

5. Conclusion

Recommendation System involves filtering suitable information for the user based on their interests and preferences. Content based and collaborative filtering are two traditional methods of RS both have certain limitations in tackling user's interests. In recent times, Hybrid recommendation systems are of greater use compared to traditional models because of its capability of combining best features from traditional models. In this work, we implemented various recommendation models including CBF, CF and Hybrid models.

Based on empirical analysis it is observed that, In CBF approaches based on genre and SVR gives better results compared to other models and In CF models item based approach outperforms user based approach. At k=10, item based approach gives better results whereas at k=20 user based approach gives better results. The proposed system has lesser RMSE and MAE values than the individual models.

Currently, we used Movie Lens ml-small dataset for implementation it can be replaced with larger dataset for better results. CBF model is implemented based on the vectors developed by genre feature of movies. As an enhancement of the work, single dimensional feature can be replaced with multi dimensional feature set by considering other features of movies like overview, tagline, and cast by combining the MovieLens dataset with TMDB dataset. Sentiment of users can be embedded while developing user and item profile by considering reviews of users and movies. CF model can be developed by applying model based approaches like Matrix factorization, SVD and Neural networks. Further, other hybrid approaches can be experimented with different combination of models by learning the importance of weights for each model.

References

[1] Raghavendra C K, Srikantaiah K.C, Venugopal K. R. (2018) 'Personalized Recommendation Systems (PRES): A Comprehensive Study and Research Issues', *International Journal of Modern Education and Computer Science (IJMECS)*, Vol.10, No.10, pp. 11-21. "doi: 10.5815/ijmecs.2018.10.02"

[2] R. C K and K. C. Srikantaiah. (2021) 'Similarity Based Collaborative Filtering Model for Movie Recommendation Systems', *in 5th International Conference on Intelligent Computing and Control Systems (ICICCS),* pp. 1143-1147. "doi: 10.1109/ICICCS51141.2021.9432354"

[3] M. E. B. H. Kbaier, H. Masri and S. Krichen.(2017) 'A Personalized Hybrid Tourism Recommender System', *2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA)*, pp. 244-250.

[4] M. Aprilianti, R. Mahendra and I. Budi. (2016) 'Implementation of weighted parallel hybrid recommender systems for e-commerce in Indonesia', in *IEEE International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, pp. 321-326. "doi: 10.1109/ICACSIS.2016.7872772"

[5] C. Zhang, M. Yang, J. Lv and W. Yang. (2018) 'An improved hybrid collaborative filtering algorithm based on tags and time factor', in *Big Data Mining and Analytics*, vol. 1, no. 2, pp. 128-136. "doi: 10.26599/BDMA.2018.9020012"

[6] Q. Zhao, C. Wang, P. Wang, M. Zhou and C. Jiang. (2018) 'A Novel Method on Information Recommendation via Hybrid Similarity', *in IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 48, no. 3, pp. 448-459. "doi: 10.1109/TSMC.2016.2633573"

[7] C. Lin, L. Wang and K. Tsai. (2018) 'Hybrid Real-Time Matrix Factorization for Implicit Feedback Recommendation Systems', in *IEEE Access*, vol. 6, pp. 21369-21380, 2018. "doi: 10.1109/ACCESS.2018.2819428"

[8] M. T. Himel, M. N. Uddin, M. A. Hossain and Y. M. Jang. (2017) 'Weight based movie recommendation system using K-means algorithm', in *International Conference on Information and Communication Technology Convergence (ICTC)*, pp. 1302-1306. "doi: 10.1109/ICTC.2017.8190928"

[9] M. K. Kharita, A. Kumar and P. Singh. (2018) 'Item-Based Collaborative Filtering in Movie Recommendation in Real time', in *First International Conference on Secure Cyber Computing and Communication (ICSCCC),* pp. 340-342. "doi: 10.1109/ICSCCC.2018.8703362"
[10] N. Ifada, T. F. Rahman and M. K. Sophan. (2020) 'Comparing Collaborative Filtering and Hybrid based Approaches for Movie Recommendation', *6th Information Technology International Seminar (ITIS)*, pp. 219-223. "doi: 10.1109/ITIS50118.2020.9321014"
[11] Q. Du, N. Li, S. Yang, D. Sun and W. Liu. (2021) 'Integrating KNN and Gradient Boosting Decision Tree for Recommendation', *IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, pp. 2042-2049. "doi: 10.1109/IAEAC50856.2021.9390647"
[12] C. Cai and L. Wang. (2020) 'Application of improved k-means k-nearest neighbour algorithm in the movie recommendation system,' in *13th International Symposium on Computational Intelligence and Design (ISCID)*, pp. 314-317. "doi: 10.1109/ISCID51228.2020.00076"
[13] M. Gupta, A. Thakkar, Aashish, V. Gupta and D. P. S. Rathore. (2020) 'Movie Recommender System Using Collaborative filtering', in *International Conference on Electronics and Sustainable Communication Systems (ICESC)*, pp. 415-420. "doi: 10.1109/ICESC48915.2020.9155879"
[14] S. Wairegi W, Mwangi and R. Rimiru. (2020) 'A Framework for Items Recommendation System Using Hybrid Approach' *IST-Africa Conference (IST-Africa)*, pp. 1-15.
[15] Singh, P.S., Srikantaiah, K.C. (2021) 'ATSR: distance and time based advanced tour sequence recommendation system," *Int. j. inf. Technology,* 13, pp. 1531-1538. "doi:10.1007/s41870-021-00681-3"
[16] R. C K and K. C. Srikantaiah. (2022) 'Switching Hybrid Model For Personalized Recommendations By Combining Users Demographic Information', *Journal of Theoretical and Applied Information Technology,* Vol 100 No. 3, pp. 825-835.

Authors Profile

Raghavendra C K is currently working as an Assistant Professor in the Department of Computer Science and Engineering at B N M Institute of Technology, Bangalore and pursuing the Ph.D degree in the Department of Computer Science and Engineering at S J B Institute of Technology, Bangalore under Visvesvaraya Technological University, Belagavi, India. He obtained his B.E degree in 2009 and M.Tech degree in 2011from Visvesvaraya Technological University, Belagavi, India. His current research interests are in the area of Data Mining, Web Mining, Big Data Analytics, Recommendation systems Social Networks Analysis and Image Processing.

Dr. Srikantaiah K C is currently a Professor in the Department of Computer Science and Engineering at S J B Institute of Technology, Bangalore, India. He obtained his B.E from Bangalore Institute of Technology, M.E from University Visvesvaraya College of Engineering, Bangalore in 2002 and Ph. D degree in Computer Science and Engineering from Bangalore University, Bangalore. He has authored a book on Web Mining Algorithms. His research interest is in the area of Data Mining, Web Mining, Big Data Analytics, Cloud Analytics and Semantic Web.