Recommendation System Based on Double Ensemble Models using KNN-MF

Krishan Kant Yadav[1], Dr. Hemant Kumar Soni[2], Dr. Nikhlesh Pathik[3]
Computer Applications Department, Prestige Institute of Management & Research, Gwalior, Madhya Pradesh, India[1]
Computer Science & Engineering Department, Amity University, Gwalior, Madhya Pradesh, India[2, 3]

***Abstract*—In today's digital environment, recommendation systems are essential as they provide personalised content to users, increasing user engagement and enhancing user satisfaction. This paper proposes a double ensemble recommendation model that combines two collaborative filtering algorithms, K Nearest Neighbour (KNN) and Matrix Factorization (MF). KNN is a neighbourhood-based algorithm that uses the similarity between users or items to make recommendations. At the same time, MF is a model-based algorithm that decomposes the user-item rating matrix into lower-dimensional matrices representing the latent user and item factors. The proposed double ensemble model uses KNN and MF to predict missing ratings matrix and combines their predictions using stacking. To evaluate the performance of the proposed ensemble model, we conducted experiments on three datasets i.e. Movielense, BookCrossing dataset and Hindi Movie dataset and compared the results with those of single algorithm approaches. The experimental results demonstrate that the double ensemble model outperforms the single algorithm approaches regarding accuracy metrics such as Mean Square Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE). The results indicate that stacked KNN and MF lead to a more robust and more accurate recommendation system.**

***Keywords*—Recommendation system; k nearest neighbour; matrix factorization; predictions using stacking; ensemble model**

I. Introduction

A recommendation system is essential in everyone's life since it allows people to choose from all available possibilities. A recommendation system is an integral component of machine learning algorithms that provides meaningful recommendations to users. There are two popular types of recommendation systems:

- Content-based.
- Collaborative Filtering.

Content-based recommendations are insufficient since they evaluate the user's history when making suggestions. A collaborative filtering strategy is being utilised to address such issues, in which it seeks comparable groups of users to make judgments. The content based recommendation system is a supervised machine learning technique, which is used to induce a classifier to discriminate netween interesting and uninteresting items for the customer.

Filtering is estimating a consumer's interest by recognising predilections and information from a huge number of users. This is done by using strategies that need cooperation between several data sources, agents, etc., for data filtering to fetch information or patterns. According to the central tenet of collaborative Filtering, two clients, X and Y, may have similar choices in one product if they have a similar option in other products.

Examples of collaborative filtering algorithms:

- YouTube content suggestion to users — It proposes videos to you based on other users who have followed or viewed similar videos as you.
- Coursera course recommendation — It proposes courses based on the completion of existing courses by others.

The collaborative filtering focus on past experience, past knowledge and user behaviour. The collaborative filtering recommends items based on similarity measures between users and items. The basic purpose of such approach is to identify users with similar interests or common preferences about any product or item. The collaborative filtering approach's main advantage is its extensive coverage. The objective of any recommendation system is to make user-relevant product recommendations. Understanding item content is not always necessary; for instance, a movie's genre does not always contain the complete story. No cold-start issue allows us to predict item ratings without prior information about that item and without waiting for a user to purchase. It displays how user preferences have changed over time. Latent factor models, which capture minor intrinsic qualities if most customers buy two unrelated, are particularly susceptible to Filtering.

The main weakness of this strategy is that it is unfriendly for suggesting new things because there is no item interaction with it. Memory-based approaches are infamous for having poor performance on highly sparse datasets.

The cold-start problem is caused by collaborative filtering systems' reliance on the utilisation of publicly available data from similar users. If you were creating a brand-new recommendation system, you wouldn't have any helpful information. Scalability problems of the algorithm arise as the number of users increases. We would have to build a sparse matrix with tons of elements if we had millions of clients and many thousands of films.

Lack of reliable; Input data may not always be accurate because human rating performance is not always dependable.

Ratings are not as crucial as user participation. Item-based suggestions provide a superior answer in this case.

In this paper, we have presented a double ensemble model for recommendation using KNN and MF. Overall, the proposed double ensemble recommendation model provides a promising approach for improving the accuracy of recommendation systems by combining different algorithms. This model can be applied to various domains, including e-commerce, social networks, and multimedia.

The organisation of this paper is as follows :

Section II throws light on work already being done in the field. Section III represents the proposed work. The experiment setup and result is discussed in Section IV. Section V concludes with future directions.

II. Related Work

Lots of work has been done in the last decade in the field of recommendation systems. We have considered the recent work of the previous 5-6 years.

Wu et al. present a novel deep learning-based collaborative filtering (CF) approach that utilises a marginalised denoising auto-encoder (MDAE) to capture high-order interactions among users and items. They have experimented on three real-world datasets to demonstrate that the proposed method outperforms several state-of-the-art CF methods[1].

A novel CF approach based on Recurrent Neural Networks (RNNs) is presented by Hidasi et al., which can model sequential user behaviour. The proposed method outperforms traditional CF methods on two benchmark datasets and also shows promising results on a real-world dataset[2].

Tan and Wang proposed a new factorisation machine-based CF approach that utilises funnel-structured embeddings to capture both the sequential and holistic user behaviour information for the session-based recommendation. Experimental results on three real-world datasets show that the proposed method outperforms several state-of-the-art approaches[3].

Kaur and Singh presented a comprehensive survey of neighbourhood-based CF methods, which are the most commonly used CF methods in practice. The paper provides a detailed overview of the existing methods, their variants, and their strengths and weaknesses[4].

A novel CF approach is presented by Zhou et al. for implicit feedback datasets that utilise adaptive regularisation to handle the sparsity and noise of the data. Experimental results on three real-world datasets demonstrate the superiority of their method.[5].

Yuan et al. propose a novel CF approach incorporating temporal dynamics into the recommendation process. The proposed method uses a recurrent neural network (RNN) to model the temporal patterns of user-item interactions and achieves better performance on two datasets[6].

A deep Bayesian CF approach is presented by Wang et al. that can handle cross-domain recommendations. The proposed method uses a hierarchical Bayesian model to capture the relationships between different domains and performs better than several other CF methods on different datasets [7].

Wang et al. propose a deep learning-based clustering method to improve the performance of CF. The proposed method utilises a neural network to learn user and item representations and then performs clustering on these representations to group similar users and items. Experimental results on two datasets show that the proposed method leads to better results than others [8].

Ma et al. presented a joint embedding method for CF that uses orthogonal regularisation to improve the quality of user and item representations. The proposed method achieves state-of-the-art performance on two benchmark datasets[9].

Jia et al. proposed a hybrid CF model that combines reinforcement learning with traditional CF methods to improve personalised recommendations. Experimental results on a real-world dataset show that the proposed method outperforms several state-of-the-art CF methods[10].

In this paper, Chen et al. propose a user modelling method for recurrent neural network (RNN) based CF, which uses a hierarchical attention network to capture the user preferences and generates personalised user representations for better recommendation accuracy[11].

Hu et al. presented an Attentional factorisation machine-based CF approach for a session-based recommendation. The proposed method uses attention mechanisms to capture the importance of different items in a session and performs better on two considered datasets[12].

Li et al. presented a multi-relational graph convolutional network-based CF approach for a cross-domain recommendation. The proposed method uses graph convolutional networks to learn user and item representations across multiple domains and achieves better performance than several state-of-the-art CF methods on a real-world dataset[13].

Li et al. presented a neural collaborative filtering (NCF) approach using long-term and short-term user representations to capture user preferences. The proposed method achieves better accuracy when tested with two datasets[14].

Wu et al. provide a comprehensive survey of the literature on bias and fairness in recommendation systems in this paper. It discusses the various types of bias in such systems, the methods used to detect and mitigate bias, and the ethical and legal considerations associated with fair recommendation [15].

Zhang et al. proposed a graph neural network-based recommendation algorithm that uses graph convolutional networks to model user-item interactions. The authors evaluated their approach on several benchmark datasets and showed that it outperforms several state-of-the-art recommendation algorithms[16].

Ricci et al. wrote a recommender system handbook covering various aspects of recommendation, including algorithms, evaluation, and applications[17].

We can conclude that ensemble models can improve the performance of recommendation systems. KNN and MF approach combining can perform better.

III. Proposed Method

This study developed a CF technique based on KNN and SVD. Generally, SVD provides a more accurate prediction compared with KNN. KNN method is based on k nearest neighbour users or items followed by top K users, and items are chosen. The accuracy depends on the source of the data and the target objective. Usually, KNN is better in data sets with low missing data proportions. SVD is better in massive-size data sets.

This research attempted to go beyond traditional ensemble learning by investigating multi-level ensemble learning concerning recommender systems. We concentrated on stacking generalisation when developing the Recommender System. We tried to examine the impact of the shift from single-level to multi-level ensemble learning on overall accuracy. We used the three datasets and three ensemble approaches based on Collaborative Filtering to evaluate accuracy. The results reveal that 2-level stacking outperforms single-level stacking and any individual recommender system.

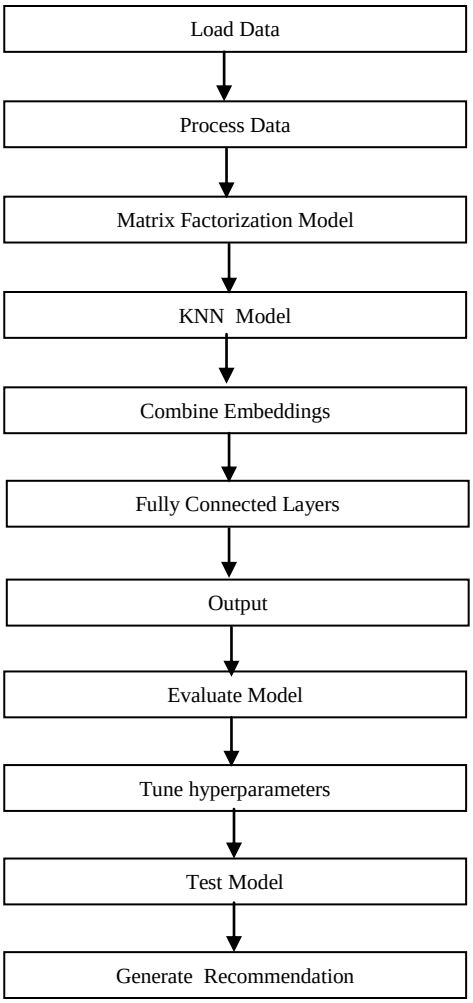

Fig. 1. Flow chart of the proposed work.

This proposed work uses KNN and Matrix Factorization (MF) as reference models. We have first implemented KNN and computed the results for various configurations like change in K etc. After taking multiple configurations, we have performed self ensemble model of KNN. We call it self-ensemble because we have considered the single base model and ensemble its variations for better performance. After this, we considered the Matrix factorisation model and computed the results on its variations like SVD. Similar to the previous one here, we also perform self-ensemble for MF. Now finally, we have proposed the double ensemble hybrid model. Hybrid as it uses both KNN and MF. We are here stacking both these models, so we named it Double ensemble. The proposed model is applied to three popular datasets. Experiment results show the proposed model's superiority.

Bagging, Boosting, Stacking, and other models are included in ensemble learning. Stacking will be the primary topic of this article. However, the recommendations mentioned above suggest a particular aspect and do not recognise the recommendation of specific movies submitted by individual users, causing the recommended material to stray from users' needs and impacting the user experience. Compared to the traditional CF-based algorithms on KNN and MF, the proposed method can realise the recommendation of specific movie input by particular users, make more personalised recommendations, and deal with the issue of cold start and sparse matrix processing to some extent.

We must train a broad set of learners to create an effective ensemble model. We used Collaborative filtering models such as KNN and MF in our scenario. Each student uniquely manipulates the dataset and makes errors based on their personal biases. We trained these models on a subset of the dataset and assessed their accuracy and diversity on the remainder. Initially, we divided the total dataset 80:20 into training and test datasets. The individual learners were then trained on the training dataset, and predictions were generated on the test dataset. We calculated accuracy using the MSE, RMSE and MAE value and diversity using the Pearson correlation coefficient. We have implemented 3 Algorithms and evaluated all three algorithms on 3 Datasets, explained in detail in the next section.

This work integrates the approaches mentioned earlier and presents three ensemble recommendation algorithms:

1) Ensemble K-Nearest Neighbors (E-KNN).
2) Ensemble Matrix Factorization(E-MF).
3) Stacked Nearest Neighbors- Matrix Factorization (S-KNNMF).

Algorithm 1: E-KNN (Ensemble K Nearest Neighbour Algorithm)

We have implemented the vanilla KNN algorithm and then taken its three variants by varying the parameters like K and distance. After that, through stacking, we made self ensemble model of KNN. The flowchart of the proposed work is given in Fig. 1. The steps are as follows:

Step 1: Input Datasets

The input of this algorithm is recommendation datasets. We have considered three datasets. The three datasets are taken to check the coverage and better performance check.

Step 2: Determine Similarity Using the Distance Function.

We have to take distance as a Euclidean distance, and another is manhattan distance. There are different distance functions, but Euclidean is the most often used. It is most commonly utilised when the data is continuous. For continuous variables, the Manhattan distance is also quite popular.

Euclidean distance =

$$d(x,y) = \sqrt{\sum_{i=1}^{m}(x_i - y_i)^2} \dots\dots\dots \text{i}$$

Manhattan distance =

$$d(x,y) = \sum_{i=1}^{m}|x_i - y_i| \dots\dots\dots\dots\dots \text{ii}$$

Step 3: Determining the Best K value.

The lower value of K is sensitive to outliers; at the same time, a higher K-value is more immune to outliers since it considers more votes when projecting. Cross-validation is an effective method for determining the best K value. It calculates the validation error rate by excluding a subset of the training data from the model construction process. Cross-validation (say, 10-fold validation) entails randomly splitting the training set into ten groups, or folds, of about similar size. 90% of the data is utilised for training the model, with the remaining 10% used to validate it. Based on the 10% validation data, the misclassification rate is calculated. This method is repeated ten times. Every ten times, a different collection of observations is handled as a validation set. It yields ten validation error estimates, which are then averaged out.

Step 4: Ensemble of the KNN Model

For the ensemble, we have used the stacking method. The primary distinction between voting and stacking is how the final aggregate is accomplished. In voting, user-specified weights aggregate the classifiers, whereas stacking uses a blender/meta classifier to do this aggregation. As we have used the self-ensemble model, we have used stacking for the ensemble.

Step 5: Assessing the Model Performance.

We have evaluated model performance on the error terms MSE, RMSE, and MAE.

Step 6: Final Output is Prediction

Finally, we would want to make predictions using the proposed model. We have considered the three variants of KNN and ensemble them using the stacking method. We have applied three different datasets to evaluate the performance of this model.

Fig. 2 represents the block diagram showing the ensemble of KNN using the stacking approach.

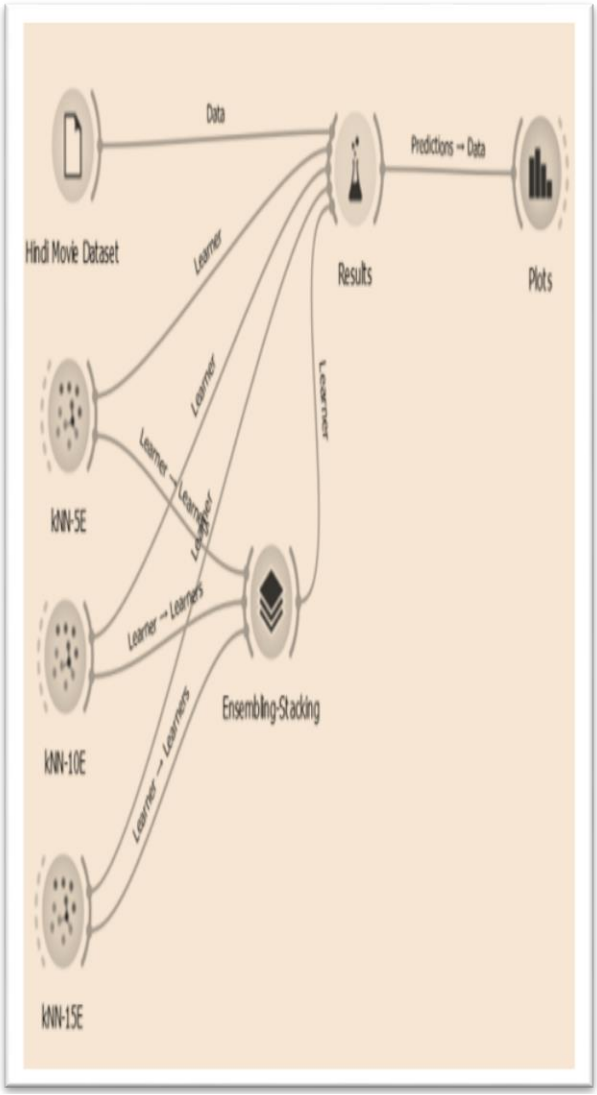

Fig. 2. E-KNN model on hindi movie dataset.

Algorithm 2: E-MF(Ensemble Matrix Factorization Algorithm)

We have implemented the Various Matrix algorithms. We have considered the three variants of matrix factorisation NMF, SVD and SVD++. After that, we made a self-ensemble model of MF through stacking.

E-MF algorithm for a stacking ensemble model that combines NMF, SVD, and SVD++ models for recommendation:

Step 1 Split the user-item rating matrix into training and validation sets.

Step 2 Train an NMF model on the training set using gradient descent to minimise the reconstruction

error between the actual ratings and the low-rank approximation of the matrix.

Step 3 Train an SVD model on the training set using gradient descent to minimise the MSE between the actual ratings and the predictions of the model.

Step 4 Train an SVD++ model on the training set using gradient descent to minimise the MSE between the actual ratings and the predictions of the model.

Step 5 Use the NMF, SVD, and SVD++ models to generate predictions for the validation set.

Step 6 Combine the predictions of all models using stacking ensemble method.

Step 7 Calculate the MSE between the actual ratings and the combined predictions for the validation set.

Step 8 Choose the best ensemble model based on the validation MSE.

Step 9 Use the chosen ensemble model to generate predictions for new users or items.

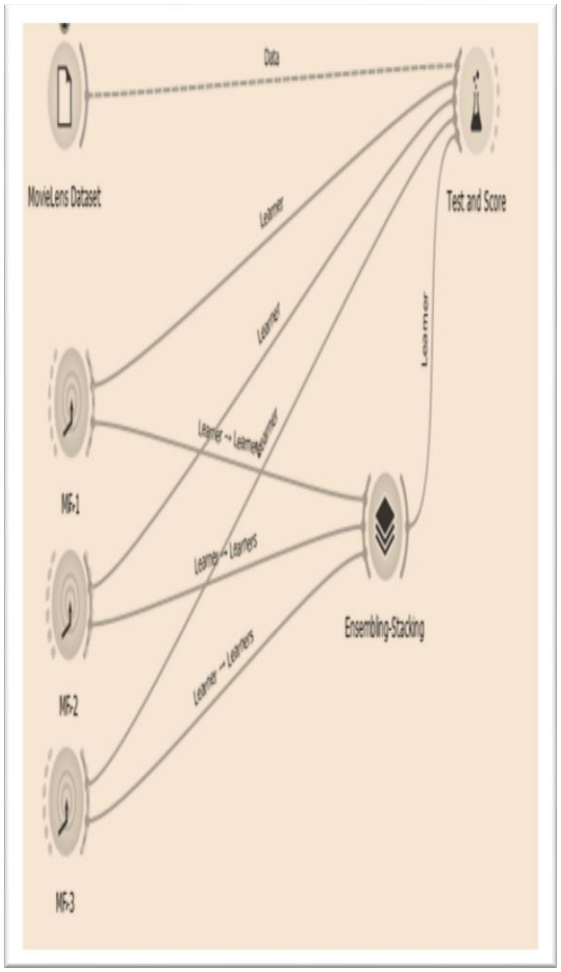

Fig. 3. Block diagram of E-MF model.

Algorithm 3: Proposed Double Ensemble Algorithm Based on KNN and MF

We have taken KNN and MF as reference models in this proposed work. Firstly we have implemented KNN and computed the results for various configurations like change in K etc. After taking multiple configurations, we have performed self ensemble model of KNN. We call it self-ensemble because we have considered the single base model and ensemble its variations for better performance.

After this, we considered the Matrix factorisation model and computed the results on its variations like SVD and SVD++. Similar to the previous one here, we also perform self-ensemble for MF. Now finally, we have proposed the double ensemble hybrid model. Hybrid as it uses both KNN and MF. We are here stacking both these models, so we named it Double ensemble.

The KNN-MF-based double ensemble recommendation system is a double ensemble MF algorithm variation that combines matrix factorisation with KNN for better recommendation accuracy. Here are the steps of the algorithm:

Input:

User-item interaction matrix (R), where each element r_ij represents the rating or preference of user i for item j.

Number of factors (k) to use for matrix factorisation.

Number of matrix factorisation algorithms to use (m).

Number of ensemble methods to use (n).

Number of neighbors (K) to use for KNN.

Output:

A set of recommendations for each user.

Steps:

1. Split the data into training and test sets.
2. For each matrix factorization algorithm (i = 1 to m):
   a. Apply matrix factorisation to the training data using algorithm i to generate a set of recommendations (R_i).
   b. Compute the prediction error of R_i on the test set using a suitable evaluation metric (e.g., RMSE, MAE).
3. For each ensemble method (j = 1 to n):
   a. Combine the recommendations from all matrix factorisation algorithms using ensemble method j to generate a set of ensemble recommendations (R_j).
   b. Compute the prediction error of R_j on the test set using the same evaluation metric as step 2.
4. Choose the combination of the matrix factorisation algorithm and ensemble method that produces the lowest prediction error on the test set.
5. For each user:
   a. Compute the K nearest neighbours based on the user's rating history using a suitable similarity measure (e.g., cosine similarity, Pearson correlation).
   b. For each item not yet rated by the user, predict a rating using a weighted average of the ratings of the K nearest neighbours.

c. Combine the KNN predictions with the predictions from the chosen combination of matrix factorisation algorithm and ensemble method using stacking.
d. Use the final set of combined predictions as the recommendations for each user.

Matrix factorisation, ensemble methods, and evaluation metrics are the same as described in the Double Ensemble Matrix Factorization algorithm. The KNN component adds a neighbourhood-based approach to the recommendation process, which can capture local and non-linear patterns in the data and improve the accuracy of recommendations, especially for cold-start users.

Fig. 3 shows the block diagram of our proposed S-KNNMF algorithm.

The proposed model performs better than the other two considered ensemble models.

IV. Experimental Setup

The experiment was conducted on a 64-bit Windows 8 machine with 8 GB RAM and an Intel Core i5-4200M processor with a 2.50 GHz clock speed. Algorithms were implemented in python using Anaconda. For visualisation, various tools are used that are part of the Anaconda. In a few cases, Google-Colab is also used for running our models.

A. Datasets

We have used three datasets to test our proposed algorithms' performance in this work. The reason for taking multiple datasets is that it will cover all possible dependencies of the dataset's nature, like biased, sparsity, etc. They are as follows:

1) Hindi Movie.
2) Book Cross.
3) Movielens.

B. Results

We have taken three datasets for comparative analysis. We have evaluated the performance of our proposed model based on the error scores, namely MSE, RMSE and MAE. We have compared all three models, i.e. ensemble KNN, ensemble matrix factorisation, and stacked KNN-MF model. Table I represents the evaluation result on Hindi Movie datasets.

The table shows that our proposed method gives better results than the other two ensemble models. Fig. 4 graphically represents the comparative analysis of all three considered models.

Table II represents the evaluation result of the Book-Crossing Dataset. It also shows our proposed algorithm's superiority over other considered algorithms.

TABLE I. Comparative Analysis of the Hindi Movie Dataset

Model	MSE	RMSE	MAE
E-KNN	0.900161946	0.948769	0.719056
E-MF	0.827225923	0.90952	0.700481
S-KNNMF	0.65895524	0.811761	0.624495

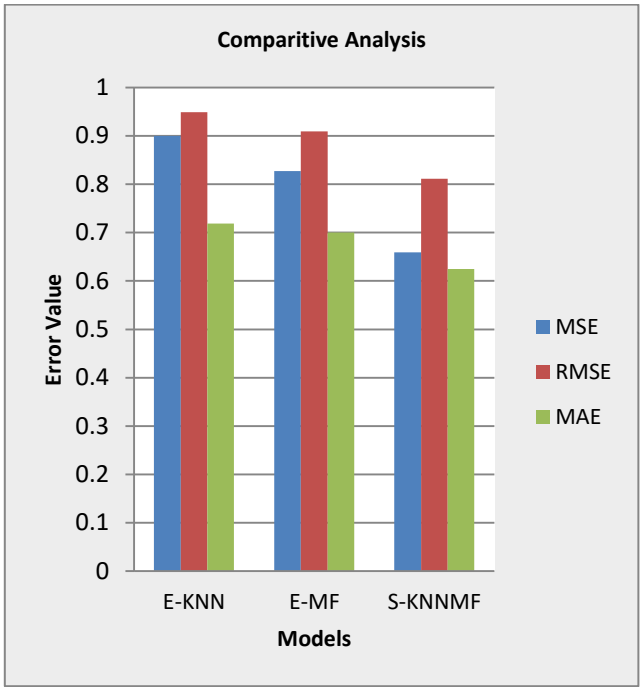

Fig. 4. Comparative graph for hindi movie dataset.

TABLE II. Comparative Analysis of the Book-Crossing Dataset

Model	MSE	RMSE	MAE
E-KNN	0.859679	0.927189	0.70268
E-MF	0.829634	0.910843	0.696308
S-KNNMF	0.789674	0.888636	0.681319

Table II shows our proposed method's superiority over the other two ensemble models. Fig. 4 graphically represents the comparative analysis of all three considered models for better representation.

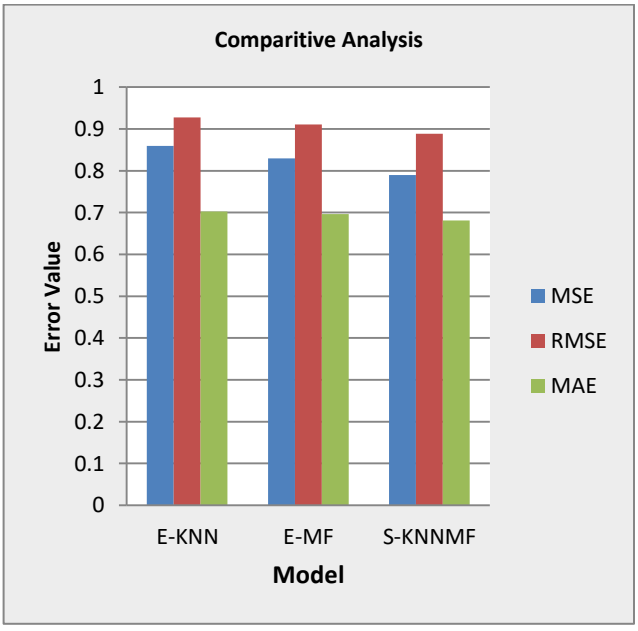

Fig. 5. Comparative graph for book-crossing dataset.

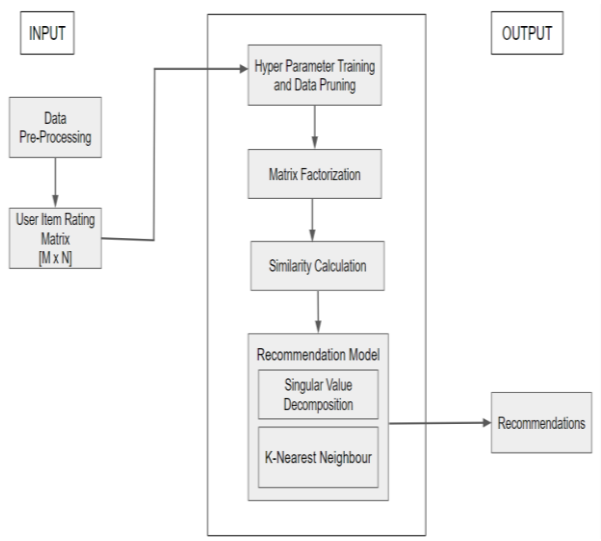

Fig. 6. Block diagram of an ensemble model.

Fig. 5 graphically represents the comparative analysis of all three approaches using Bookcrossing dataset. The Movielens dataset is very popular for performance analysis of recommendation systems. We have also used it for the evaluation of our proposed model. Fig. 6 is representing a block diagram of KNN and MF approaches. Table III represents the evaluation results of all considered algorithms on the Movielens dataset.

TABLE III. COMPARATIVE ANALYSIS FOR MOVIE-LENS DATASET

Model	MSE	RMSE	MAE
E-KNN	0.729304	0.853993	0.652247
E-MF	0.719795	0.848408	0.648123
S-KNNMF	0.658955	0.811761	0.624495

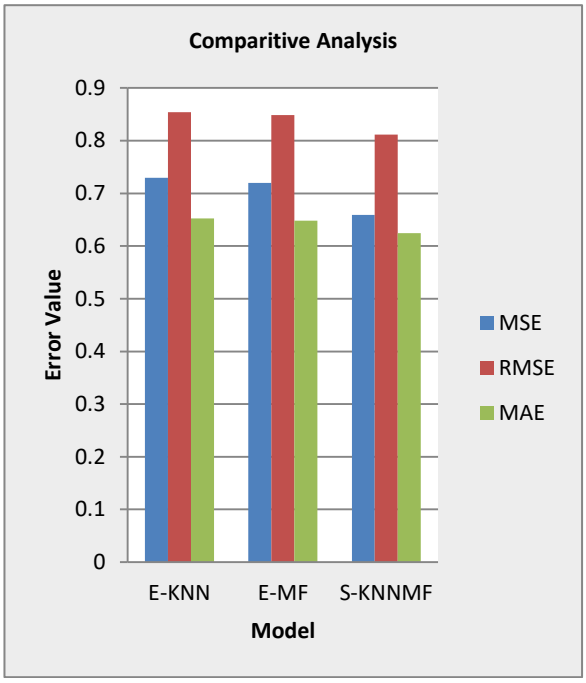

Fig. 7. Comparative graph for movielens dataset.

Table III supports our proposed method's superiority over the other two ensemble models. Fig. 7 is representing the ensemble model of KNN and MF approachs for better representation of recommendation.

It is clear from Tables I to III that the Ensemble model performs better than the normal models. Double ensemble further improves the performance of the model. Our proposed model gives a minimum error value when evaluated based on MSE, RMSE, and MAE.

V. CONCLUSION AND FUTURE DIRECTION

Double ensemble recommendation models that combine collaborative filtering methods like K-Nearest Neighbors (KNN) and matrix factorisation (MF) can offer improved accuracy and diversity in the recommended items.

KNN is a user-based collaborative filtering technique that recommends items based on user similarity. It suffers from the sparsity problem and is less effective for cold start problems. Conversely, MF is a matrix-based technique that learns latent factors for users and items and can address the sparsity problem. However, it may struggle with long-tail recommendations.

Combining KNN and MF can address the limitations of both techniques, leading to better performance. By leveraging the strengths of both models, the double ensemble approach can generate more diverse and accurate recommendations, especially for cold-start scenarios and long-tail items.

Overall the double ensemble approach of KNN and MF is a promising solution for improving the accuracy and diversity of recommendation systems. In the future, NN-based algorithms are also combined to get better performance. Other ensemble methods can also be applied.

CONFLICTS OF INTEREST

The authors declare no conflict of interest.

AUTHORS' CONTRIBUTIONS

Krishan Kant Yadav and Hemant Kumar Soni have conceptualised the problem statement. Nikhlesh Pathik contributed to the related work. Krishan Kant Yadav, Hemant Kumar Soni, and Niklesh Pathik developed the proposed algorithm. Hemant Kumar Soni and Krishan Kant Yadav did implementation work. Krishan Kant did formatting and editing work.

REFERENCES

[1] Wu, Y., DuBois, C., Zheng, A. X., & Ester, M. (2015). Deep Collaborative Filtering via Marginalised Denoising Auto-encoder. In Proceedings of the 24th ACM International Conference on Information and Knowledge Management (pp. 811-820).

[2] Hidasi, B., Karatzoglou, A., Baltrunas, L., & Tikk, D. (2016). Collaborative Filtering with recurrent neural networks. In Proceedings of the 4th International Conference on Learning Representations.

[3] Tan, Y., & Wang, Y. (2017). Factorization Machines with Funnel-structured Embeddings for Session-based Recommendation. In Proceedings of the 26th International Joint Conference on Artificial Intelligence (pp. 2992-2998).

[4] Kaur, S., & Singh, K. (2017). A Comprehensive Survey of Neighborhood-based Recommendation Methods. arXiv preprint arXiv:1707.01189.

[5] Zhou, Y., Wilkinson, D., Schreiber, R., & Pan, R. (2017). Collaborative Filtering for Implicit Feedback Datasets using Adaptive regularisation. In Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization (pp. 161-170).

[6] Yuan, F., Karatzoglou, A., & Jose, J. M. (2018). Collaborative Filtering with Temporal Dynamics. In Proceedings of the 12th ACM Conference on Recommender Systems (pp. 146-154).

[7] Wang, R., Fu, B., Fu, G., & Wang, M. (2018). Deep Bayesian Collaborative Filtering for Cross-domain Recommendation. In Proceedings of the 2018 ACM Conference on Recommender Systems (pp. 246-250).

[8] Wang, K., Gao, H., He, X., Deng, L., & Li, Y. (2019). Improved Collaborative Filtering via Deep Learning-based Clustering. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (pp. 6349-6355).

[9] Ma, H., Fan, Y., & Ji, X. (2019). Joint Embedding of User and Item with Orthogonal Regularization in Collaborative Filtering. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (pp. 6362-6367).

[10] Jia, C., Zhang, J., & Chen, K. (2020). A Hybrid Collaborative Filtering Model with Reinforcement Learning for Personalised Recommendations. In Proceedings of the 2020 ACM Conference on Multimedia (pp. 3123-3126).

[11] Chen, C., Xu, Z., & Wang, Y. (2020). User Modeling for Recurrent Neural Network based Collaborative Filtering. arXiv preprint arXiv:2002.04799.

[12] Hu, Y., Li, X., Lu, Z., & Zhou, X. (2020). Attentional Factorization Machines for Session-based Recommendation. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (pp. 7917-7926).

[13] Li, H., Li, X., Li, B., Li, C., & Li, P. (2021). Multi-Relational Graph Convolutional Network for Cross-Domain Recommendation. arXiv preprint arXiv:2101.08328.

[14] Li, Y., Tang, J., Li, Z., & Yang, B. (2021). Neural Collaborative Filtering with Long- and Short-Term User Representations. In Proceedings of the 2021 ACM Conference on Recommender Systems (pp. 178-187).

[15] Wu, L., Pan, S. J., Long, G., Jiang, J., & Zhang, C. (2020). A survey on bias and fairness in recommendation. arXiv preprint arXiv:2007.15551.

[16] Zhang, S., Yao, L., Sun, A., & Tay, Y. (2019). Deep learning based recommender system: A survey and new perspectives. ACM Computing Surveys (CSUR), 52(1), 1-38.

[17] Ricci, F., Rokach, L., Shapira, B., & Kantor, P. B. (2015). Recommender systems handbook (Vol. 1).