Natl. Acad. Sci. Lett.
https://doi.org/10.1007/s40009-021-01051-0

SHORT COMMUNICATION

Movie Recommender System Using K-Nearest Neighbors Variants

Sonu Airen[1] · Jitendra Agrawal[1]

Received: 26 July 2020 / Revised: 26 February 2021 / Accepted: 19 March 2021

Abstract Information overload is a major problem for many internet users which occurs due to overwhelming amounts of data made available to a user. In order to deal with this problem filtering tool, like Recommender System is required for providing relevant information for the users which personalizes the search according to user preferences. The Collaborative Filtering Recommender System finds the nearest neighbour set of active user by using similarity measures on the rating matrix. This paper proposes different variations of K-nearest neighbors (KNN) algorithm with different similarity measures namely cosine, msd, pearson and pearson baseline for Movie Recommender System. These different variations of KNN algorithms have been implemented for real data from MovieLens dataset and compared on accuracy metrics like fraction of concordant Pairs, mean absolute error, mean squared error, root mean squared error, precision@k and recall@k for Movie Recommender System. For real life application, Movie Recommender System filtering tool may be used as plugin by customizing the web browser.

Keywords Accuracy metrics · Collaborative filtering · K nearest neighbour · Machine learning · Recommender system

✉ Sonu Airen
sonugoyal24@rediffmail.com

Jitendra Agrawal
jitendra@rgtu.net

[1] School of Information Technology, Rajiv Gandhi Proudyogiki Vishwavidyalaya, Airport Road, Gandhi Nagar, Bhopal, Madhya Pradesh, India

Introduction

In today's world, digital online platform is necessity of human life. This online platform is needed for online shopping, digital payments and communications. In most of the digital applications Recommender System is required for improving the delivery of online contents. Recommender System is information filtering application or tool which recommends items according to user personal choices or preferences [9, 13].

The task of a Recommender System is to generate recommendation for given user/item. In general, Recommender System techniques are classified as follows: Content-based Recommender System [9, 13] suggests item to users by matching the target item description with the set of items that user has rated in the past. Collaborative Filtering Recommender System [9, 13] first finds an active user's neighbourhood (also known as "K Nearest Neighbors" or KNN, for short) by finding similar users to active user. Here similar user means that the user have the same preferences as the active user and they both co-rated the same items in the past. The neighbourhood of users has positive correlation with active user. After finding rating from neighbourhood users, these ratings are combined to generate recommendation for active user. Hybrid recommendation combines two or more methods for achieving higher performance. These seek to get advantage of one technique for removing disadvantage of another.

Collaborative Filtering with KNN [2, 3] is a memory-based Recommender System algorithm. Collaborative Filtering algorithm's main task is to calculate similarity among users or items. The common similarity measures such as cosine, msd, pearson and pearson baseline are used for similarity calculation among users. To judge the performance of different KNN algorithms, they are compared on four error measures and two accuracy measures on benchmark dataset. The performance of KNN algorithm has also been experimentally investigated and presented in this paper.

The organization of rest of the paper is as follows: In "Related Work" section, existing work related to KNN algorithms-based Recommender System are discussed. The proposed approach for KNN algorithms-based Recommender System, similarity module are described in details in "Proposed Architecture for Movie Recommender System" section. Dataset, evaluation metrics and performance comparison of the proposed system with respect to different KNN algorithm and similarity measures on benchmark dataset are reported in "Experiments" section. Finally, "Conclusion and Future Work" section is presented with the concluding remarks.

Related Work

Few existing works related to Movie Recommender System based on KNN, similarity measures and performance metric are discussed in this section. Similarity computation between users is an important step in KNN-based Collaborative Filtering algorithms. There are several ways to calculate the similarity between user/items such as Pearson correlation coefficient [2], cosine distance [15], and adjusted cosine similarity and mean squared difference [18].

A new similarity measure improved PCC weighted with RPB (IPWR) is proposed in [4]. IPWR uses rating preference behavior (RPB) of users and RPB is based on user average rating value and standard deviation. [14] formed neighborhood using K nearest-neighbor-based Collaborative Filtering algorithm. Design of rating system and acquiring ratings and privacy issues of Collaborative Filtering are discussed in [16]. Many authors [1] propose collaborative filtering with clustering. In ClustKnn [3], clustering model is built beforehand, and at time of recommendations KNN approach is used for rating prediction. In [2] authors discuss how recommendation for movies is generated by grouping KNN with k-means clustering. Authors performed experiments and found that the value of RMSE decreases with the decreasing number of clusters. Different approaches like deep learning, evolutionary computing and matrix factorization is also clubbed with Collaborative Filtering Recommender System. Deep learning approach is also used for movie recommendations. [12] discusses Auto encoders-based Movie Recommender System. In [7] movie swarms is created to solve cold start problem of new users and items in Movies Recommender System. Movie genres-based movie mining is used to find popular and interesting movies to solve new users problem.

For calculating pairwise similarities between items, similarity matrix is built in neighborhood-based Collaborative Filtering Recommender System. To solve the scalability problem of similarity matrix, neighborhood model is factor in [10]. Matrix factorization Collaborative Filtering method [6, 11] considers ordinal rating on product and item ratings as predicted using probability distribution. In [5], authors find optimal number of nearest neighbours for movie recommender service using item based KNN Collaborative Filtering. Several metrics to measure user prediction performance like accuracy metrics, error metrics and Statistical metrics have been discussed in [17].

Proposed Architecture for Movie Recommender System

Proposed architecture for Movie Recommender System using different KNN algorithms and different similarity measures is shown in Fig. 1. Here historical user data has been used for recommendation purpose. Similarity between different users is calculated using user-item rating matrix. Four types of similarities cosine, msd, pearson and pearson baseline are calculated for given user-item rating matrix. After calculating similarities, variation of KNN-based Collaborative Filtering recommendation algorithms is used with five fold cross validation and movie recommendation is generated . For generated results metrics like MSE, RMSE, MAE and FCP on different values of number of nearest-neighbors are compared. To the best of our knowledge no research has been conducted on Movie Recommender System using K Nearest Neighbors variants like KNN-Basic, KNN-WithMeans, KNN-WithZScore, KNN-Baseline on four different similarity measurement for neighborhood calculation i.e. cosine, msd, pearson and pearson baseline similarities.

K-Nearest Neighbors(KNN) Algorithm

In this section, a discussion on KNN algorithm and its different variations with different similarity measures has been presented. Accuracy metrics to evaluate the performance of these algorithms has been presented next. We assume that U = $\{U_1, U_2, \ldots, U_N\}$ and I = $\{I_1, I_2, \ldots, I_M\}$ are set of users and items, respectively. The user-item rating matrix is denoted as $R = r_{ij_{NM}}$ for $i = \{1, 2, \ldots, N\}$ and $j = \{1, 2, \ldots, M\}$. Equation 1 is an example of user-item rating matrix for six users U = {A,B,C,D,E,F} and six items I = {U,V,W,X,Y,Z}. "–" represents the missing rating.

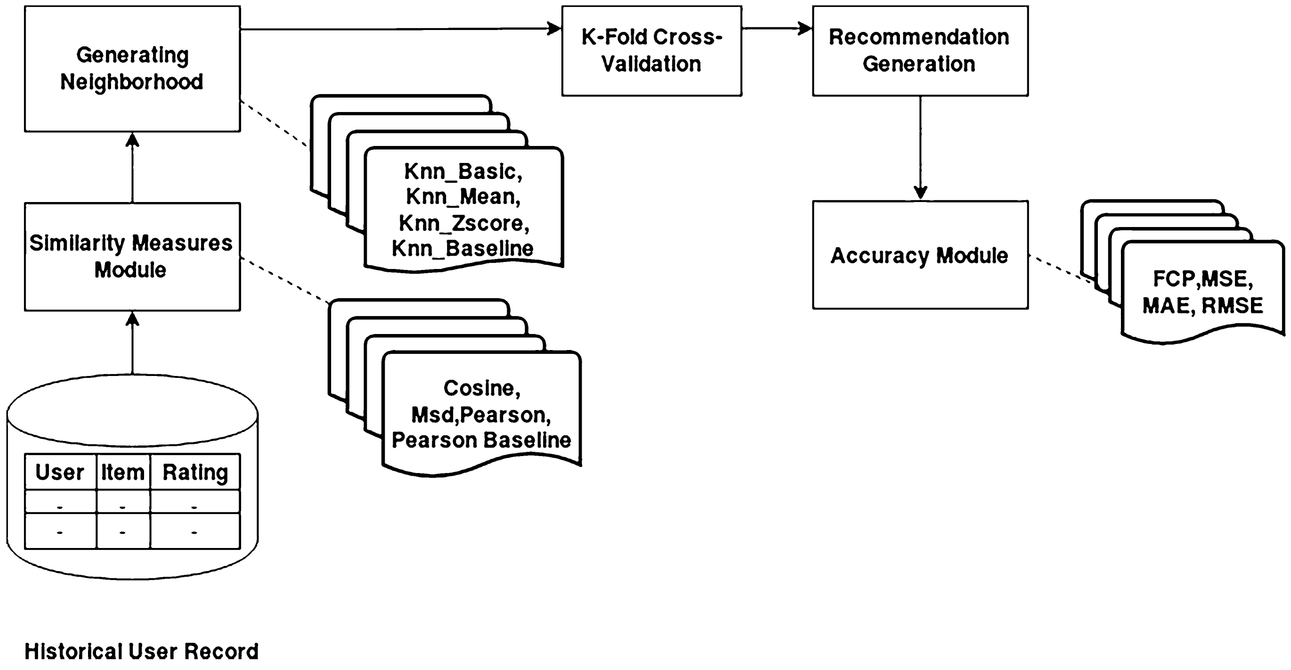
Generating Neighborhood
K-Fold Cross-Validation
Recommendation Generation
Knn_Basic, Knn_Mean, Knn_Zscore, Knn_Baseline
Similarity Measures Module
Accuracy Module
FCP,MSE, MAE, RMSE
Cosine, Msd,Pearson, Pearson Baseline
User | Item | Rating
Historical User Record

Fig. 1 Proposed architecture for movie recommender system

$$\begin{array}{c|cccccc} & U & V & W & X & Y & Z \\ \hline A & 3 & 7 & 4 & 9 & 9 & 7 \\ B & 7 & - & 5 & 3 & 8 & 8 \\ C & 7 & 5 & 5 & - & 8 & 4 \\ D & 5 & 6 & 8 & 5 & 9 & 8 \\ E & 5 & 8 & 8 & 8 & 10 & 9 \\ F & 7 & 7 & - & 4 & 7 & 8 \end{array} \qquad (1)$$

KNN [19] algorithm is distance-based lazy learning type of supervised ML algorithm which can be used for classification of categorical as well as numeric data. To predict the rating in our proposed KNN-based Collaborative Filtering Movie Recommender System, similarity is weighted differently in a different variant of KNN-based Collaborative Filtering algorithm, which has been presented next (Table 1).

KNN-Basic

In a simple KNN-based collaborative filtering algorithm the rating prediction for item i from user A is given by:

$$\hat{r}_{Ai} = \frac{\sum_{B \in N_i^K(A)} \text{LIKE(A,B)} \cdot r_{Bi}}{\sum_{B \in N_i^K(A)} \text{LIKE(A,B)}} \qquad (2)$$

where $\hat{r}_{Ai}$ is the estimated rating of user A for item i. r_{Ai} is the true rating of user A for item i. $N_i^K(A)$ is the K nearest neighbors of user A that have rated item i and LIKE(A,B) is similarity or likeness between user A and user B.

Table 1 K-nearest neighbors (KNN)'s pseudo-code

```
Input: $K$ := The desired number of nearest neighbors
$D$ := Set of training data in the d-dimensional
$t$ := Test tuple in the d-dimensional
Output: $K$ Nearest neighbors of the given $t$
Begin:
Step 1 : For test tuple $t$ compute the distance between $t$ and all other training tuple in data set $D$.
Step 2 : Sort the distance $d$ in ascending order.
Step 3 : Pick the first $K$ tuples from sorted distance list.
Step 4 : Assign class label to test tuple $t$ according to majority votes from $K$ nearest neighbors.
end
```

KNN-WithMeans

To adjust the different rating behaviour, mean rating of user is subtracted from the user rating and used as weight for similarity caluculation in KNN-WithMeans algorithm. The rating prediction for item i from user A is given by :

$$\hat{r}_{Ai} = \mu_A + \frac{\sum\limits_{B \in N_i^K(A)} \mathrm{LIKE}(A,B) \cdot (r_{Bi} - \mu_B)}{\sum\limits_{B \in N_i^K(A)} \mathrm{LIKE}(A,B)} \tag{3}$$

here μ_A is the mean of all ratings given by user A.

KNN-WithZScore

To normalize the ratings, firstly mean is subtracted and divided by standard deviation from the rating and this z-score normalized rating is used as a weight for similarity. The rating prediction for item i from user A is given by :

$$\hat{r}_{Ai} = \mu_A + \sigma_A \frac{\sum\limits_{B \in N_i^K(A)} \mathrm{LIKE(A,B)} \cdot (r_{Bi} - \mu_B)/\sigma_B}{\sum\limits_{B \in N_i^K(A)} \mathrm{LIKE(A,B)}} \tag{4}$$

here σ_A is the standard deviation of all ratings given by user A.

KNN-Baseline

In KNN baseline, The rating prediction for item i from user A is given by :

$$\hat{r}_{Ai} = b_{Ai} + \frac{\sum\limits_{B \in N_i^K(A)} \mathrm{LIKE(A,B)} \cdot (r_{Bi} - b_{Bi})}{\sum\limits_{B \in N_i^K(A)} \mathrm{LIKE(A,B)}} \tag{5}$$

here b_{Ai} is baseline ratings for item i from user A.

Similarities Measures

The similarities measures calculate similarity between users or items.

Cosine Similarity

Cosine similarity is vector-based similarity with range 0(minimum similarity) to 1(maximum similarity). The cosine similarity between user A and B is defined as:

$$\text{cos-sim}(A,B) = \frac{\sum\limits_{i \in I_{AB}} r_{Ai} \cdot r_{Bi}}{\sqrt{\sum\limits_{i \in I_{AB}} r_{Ai}^2} \cdot \sqrt{\sum\limits_{i \in I_{AB}} r_{Bi}^2}} \tag{6}$$

Here I_{AB} is set of items co-rated by both A and B.

Mean Squared Difference Similarity (MSD)

The MSD distance [13, 18] computes the rating distance by taking the mean of square difference of rating of co-rated items by both A and B.

$$\text{msd-dist}(A,B) = \frac{1}{|I_{AB}|} \cdot \sum_{i \in I_{AB}} (r_{Ai} - r_{Bi})^2 \tag{7}$$

The MSD similarity is then defined as:

$$\text{msd-sim}(A,B) = \frac{1}{\text{msd-dist}(A,B) + 1} \tag{8}$$

Pearson Similarity

The value of pearson similarity [2] is calculated using equation 9. It is also known as Pearson Correlation Coefficient (PCC). Its range is -1 (for $-$ve correlation) to $+1$ (for $+$ correlation). The Pearson similarity is defined as:

$$\text{pearson-sim}(A,B) = \frac{\sum\limits_{i \in I_{AB}} (r_{Ai} - \mu_A) \cdot (r_{Bi} - \mu_B)}{\sqrt{\sum\limits_{i \in I_{AB}} (r_{Ai} - \mu_A)^2} \cdot \sqrt{\sum\limits_{i \in I_{AB}} (r_{Bi} - \mu_B)^2}} \tag{9}$$

Pearson Baseline Similarity

Pearson baseline similarity is also known as shrunk coefficient similarity. It uses the baselines rating b_{Ai} for item i from user A for centering of ratings. b_{Ai} is calculated as $b_{Ai} = \mu + \beta_A + \beta_i$ here μ is global average rating, β_A is user bias and β_i is item bias. The Shrunk correlation coefficient $\hat{\rho}_{AB}$ is defined as:

$$\hat{\rho}_{AB} = \frac{\sum\limits_{i \in I_{AB}} (r_{Ai} - b_{Ai}) \cdot (r_{Bi} - b_{Bi})}{\sqrt{\sum\limits_{i \in I_{AB}} (r_{Ai} - b_{Ai})^2} \cdot \sqrt{\sum\limits_{i \in I_{AB}} (r_{Bi} - b_{Bi})^2}} \tag{10}$$

now Pearson baseline similarity is

$$\text{pearson-baseline-sim}(A,B) = \frac{|I_{AB}| - 1}{|I_{AB}| - 1 + \text{shr}} \cdot \hat{\rho}_{AB} \tag{11}$$

here shr is shrinkage parameter. The above-mentioned similarity metrics for the rating matrix given in Eq. 1 are as follows:

(a) Cosine Similarity Measure

$$\begin{array}{c|cccccc} & A & B & C & D & E & F \\ \hline A & 1 & 0.7992 & 0.7792 & 0.9346 & 0.9738 & 0.8846 \\ B & & 1 & 0.8747 & 0.9058 & 0.8661 & 0.8270 \\ C & & & 1 & 0.9095 & 0.8654 & 0.8532 \\ D & & & & 1 & 0.9893 & 0.8656 \\ E & & & & & 1 & 0.8816 \\ F & & & & & & 1 \end{array}$$

(b) MSD Similarity Measure

$$\begin{array}{c} \\ A \\ B \\ C \\ D \\ E \\ F \end{array} \begin{array}{c} \begin{array}{cccccc} A & B & C & D & E & F \end{array} \\ \begin{pmatrix} 1 & 0.0833 & 0.1388 & 0.1363 & 0.1818 & 0.0980 \\ & 1 & 0.2 & 0.2173 & 0.1041 & 0.6666 \\ & & 1 & 0.1388 & 0.0892 & 0.16 \\ & & & 1 & 0.2857 & 0.3333 \\ & & & & 1 & 0.1388 \\ & & & & & 1 \end{pmatrix} \end{array}$$

(c) Pearson Similarity Measure

$$\begin{array}{c} \\ A \\ B \\ C \\ D \\ E \\ F \end{array} \begin{array}{c} \begin{array}{cccccc} A & C & D & E & F & \end{array} \\ \begin{pmatrix} 1 & -0.1374 & -0.3573 & 0.2081 & 0.7619 & 0.2773 \\ & 1 & 0.4538 & 0.5159 & 0.1124 & 0.2183 \\ & & 1 & 0.4513 & -0.0428 & 0.2973 \\ & & & 1 & 0.7633 & -0.0577 \\ & & & & 1 & 0.0396 \\ & & & & & 1 \end{pmatrix} \end{array}$$

(d) Pearson Baseline Similarity measure

$$\begin{array}{c} \\ A \\ B \\ C \\ D \\ E \\ F \end{array} \begin{array}{c} \begin{array}{cccccc} A & B & C & D & E & F \end{array} \\ \begin{pmatrix} 1 & 0.6588 & 0.7491 & 0.8173 & 0.9220 & 0.7523 \\ & 1 & 0.8811 & 0.8999 & 0.8414 & 0.9859 \\ & & 1 & 0.8385 & 0.8276 & 0.8455 \\ & & & 1 & 0.9694 & 0.9345 \\ & & & & 1 & 0.8972 \\ & & & & & 1 \end{pmatrix} \end{array}$$

Experiments

Data Set

During experimentation, MovieLens dataset [8] was used. The MovieLens dataset used for this work is called ML-100K, there are 100,000 ratings, ranging from 1 to 5 stars, with 943 persons and 1682 movies. ML-100K is least sparser dataset(Sparsity level 93.696%) in MovieLens data set.

Evaluation Metrics

The experimental work is carried out in python. In our experiment, k fold cross-validation is used with k is equal to 5. For different accuracy metrics, the mean values of five folds are used.

For rating prediction of Movie Recommender System, we have used KNN-based Collaborative Filtering Recommender System algorithm. We have used four variants of KNN which are KNN-Basic, KNN-WithMeans, KNN-WithZScore, KNN-Baseline. Each of these four algorithms have been tested on four different similarity measures for neighborhood calculation i.e. cos-sim, MSD-sim, Pearson-sim and Pearson baselines similarity. The performances of different similarity measures on different KNN methods are measured in terms of MSE, RMSE, MAE and FCP. Precision@k and recall@k for TOP-N recommendation are also calculated. The performance evaluation metrics are defined as follows:

FCP (Fraction of Concordant Pairs)

FCP [11] is ranking quality measurement metric. FCP measures ration of ordered items pairs (Concordant Pairs) to total number of pairs.

$$\mathrm{FCP} = \frac{n_c}{n_c + n_d} \tag{12}$$

$$\begin{gathered} n_c = \sum_u n_c^u \qquad n_d = \sum_u n_d^u \\ n_c^u = |(\mathrm{i},\,\mathrm{j})|\hat{r}_{u_i} > \hat{r}_{u_j} \qquad r_{u_i} > r_{u_j}| \end{gathered} \tag{13}$$

here n_c^u is concordant Pairs for user u and n_d^u is discordant pairs for user u.

MAE (Mean Absolute Error)

$$\mathrm{MAE} = \frac{1}{|\hat{R}|} \sum_{\hat{r}_{ui} \in \hat{R}} |r_{ui} - \hat{r}_{ui}| \tag{14}$$

MSE (Mean Squared Error)

$$\mathrm{MSE} = \frac{1}{|\hat{R}|} \sum_{\hat{r}_{ui} \in \hat{R}} (r_{ui} - \hat{r}_{ui})^2. \tag{15}$$

RMSE (Root Mean Squared Error)

$$\mathrm{RMSE} = \sqrt{\frac{1}{|\hat{R}|} \sum_{\hat{r}_{ui} \in \hat{R}} (r_{ui} - \hat{r}_{ui})^2}. \tag{16}$$

Precision@k and Recall@k

For recommendation, generated ratings are in the numerical scale (commonly 1–5) and Precision and Recall are binary metrics which work on binary output generated from model. To convert non-binary metrics (ratings) to binary metrics (Precision and Recall), relevant and non-relevant items are introduced in Precision@k and Recall@k. Here k is user defined parameter which denotes relevancy threshold for item's rating . Item is consider as relevant if its true/actual rating is greater than k otherwise item is irrelevant. Relevancy of item is known beforehand the generation of recommendation.

$$\mathrm{Precision@k} = \frac{|\{\text{Recom-and-Relev-Items}\}|}{|\{\text{Recom-Items}\}|} \tag{17}$$

$$\text{Recall@k} = \frac{|\{\text{Recom-and-Relev-Items}\}|}{|\{\text{Relev-Items}\}|} \tag{18}$$

Here Relev-Items is items that are relevant

Recom-Items items that are recommendated

Recom-and-Relev-Items are items that are recommendated and relevant and |||| denotes the cardinality of set.

Performance Comparison

In this section, we compare different KNN algorithms with many similarity measures for Movie Recommender System on MovieLense ML100K data set. We conducted several experiments on this benchmark data set and compared these algorithms for two sets of parameters. First, accuracy metrics namely FCP, MAE, MSE and RMSE and second, Precision@k and Recall@k. We compare these measures wrt *K*. Here *K* is number of nearest neighbors. For our experiments, *K* varies from 10 to 100. Number of nearest neighbors is a parameter which can affect the performance of recommendation system. So we have compared the results with different values of nearest neighbors.

K v/s FCP, MAE, MSE and RMSE for Different KNN Algorithm

In this subsection, effect of *K* on different metrics like FCP, MAE, MSE and RMSE has been discussed. For better performance, higher value of FCP and lower value of MAE, MSE and RMSE should be preferred for Movie Recommender System.

Figure 2 gives the performance of four similarity measures with four KNN-based Collaborative Filtering Recommender System algorithms v/s number of nearest neighbors on ML-100K dataset. MSE in all four algorithms for cosine and pearson similarity has decreased first and became constant after *K* = 40. Further it is observed that it decreases and becomes constant after *K* = 20 in all four algorithm for pearson baseline and MSD similarities. MSE initially decreases and then increases in MSD similarity in KNN-Basic algorithm. It restarts decrease first and constant after *K* = 40 for remaining three algorithms. RMSE is same as MSE in all four algorithms and all four similarity measures except KNN-WithMeans with pearson baseline where RMSE first decrease, remain constant and increases. MAE in all four algorithms for cosine and pearson similarity decreases first and stays constant after *K* = 40. MAE for MSD similarity decreases initially and becomes constant after *K* = 40 in all four algorithms except KNN-Basic. In KNN-Basic, MAE first decreases and then slightly increases as K increases. MAE for pearson baseline slightly decrease and constant for all four algorithms. FCP increases and remains constant in all four algorithms using different similarity measures.

K v/s Precision@k and K v/s Recall@k

In this section we have discussed effect of *K* on precision@k and recall@k. We analyse the performance of different KNN algorithms i.e. KNN-Basic, KNN-WithMeans, KNN-WithZScore and KNN-Baseline with different value of *K* for cosine similarity only.

Figure 3a shows precision@k v/s *K*. For KNN-Basic , precision@k is almost constant. For KNN-WithMeans, precision@k fluctuates till K = 40 and thereafter increases only. In KNN-WithZScore, precision@k initially decreases and afterward almost constant.

In KNN-Baseline, precision@k initially decreases and then increases at *K* = 30 and remain constant after 40. Figure 3b shows precision@k v/s *K*. In KNN-Basic algorithm recall@k decreases with increases of *K*, in KNN-WithMeans and KNN-WithZScore it increases with increases in *K*. In KNN-Baseline it is almost constant.

Conclusion and Future Work

Recommender System is information filtering tool that is required in current digital era due to overloading of information. In the proposed system, Movie Recommender System is built using various KNN-based Collaborative Filtering algorithms. The famous ML-100K data set from MovieLens has been used for observing the performance of different KNN algorithms for different similarity measures. It has been observed that error like RMSE, MSE and MAE are stable after the neighborhood size of 40 neighbors. So we find that 40 is optimized value of *K* number of nearest neighbor for our dataset. The RMSE value for existing technique [2] is compared with the RMSE value of the proposed technique. The existing technique had the RMSE value of 1.233.The RMSE value using proposed technique of KNN-Basic, KNN-WithMeans, KNN-WithZScore, KNN-Baseline with different similarity measures namely cosine, msd, pearson and pearson baseline for Movie Recommender System are less than 1.233. Limitation of our approach is that it is performed well for ML100K dataset but has not been tested for other such dataset. Second limitation of our approach is that our approach works well for small dataset but for bigger dataset memory implication and run time implication may occur. The proposed system can be further improved using better distance measure like Mahalanobis distance which not only measures distance between two point, but also distance from all points using variance of data distribution. The performance of proposed system can be improved using other algorithms

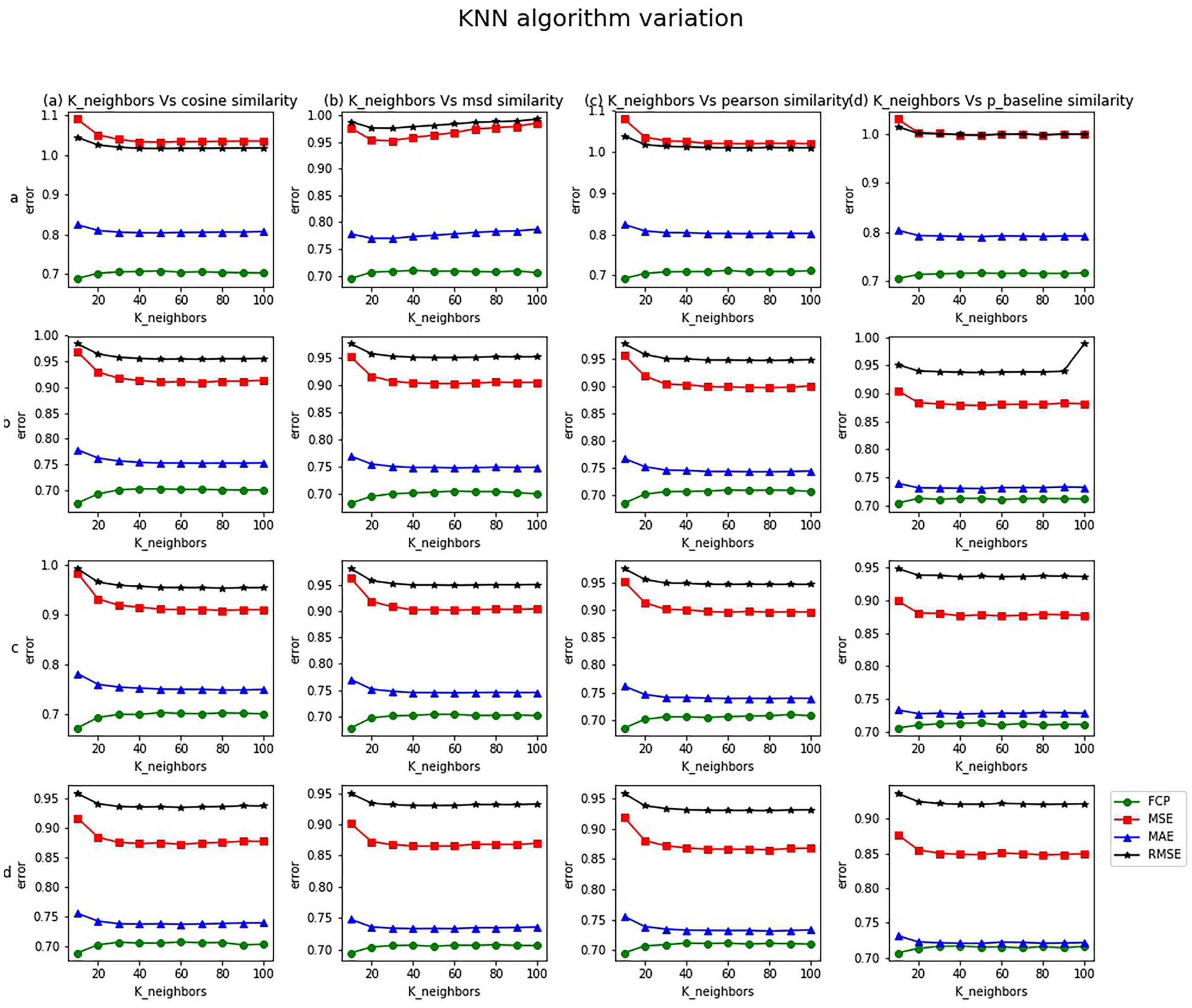

Fig. 2 Performance of different KNN methods(with four similarity measures) with different number of nearest neighbors(K) on ML-100K dataset. **a** KNN-Basic algorithm error v/s K. **b** KNN-WithMeans algorithm error v/s K. **c** KNN-WithZScore algorithm error v/s K. **d** KNN-Baseline algorithm error v/s K

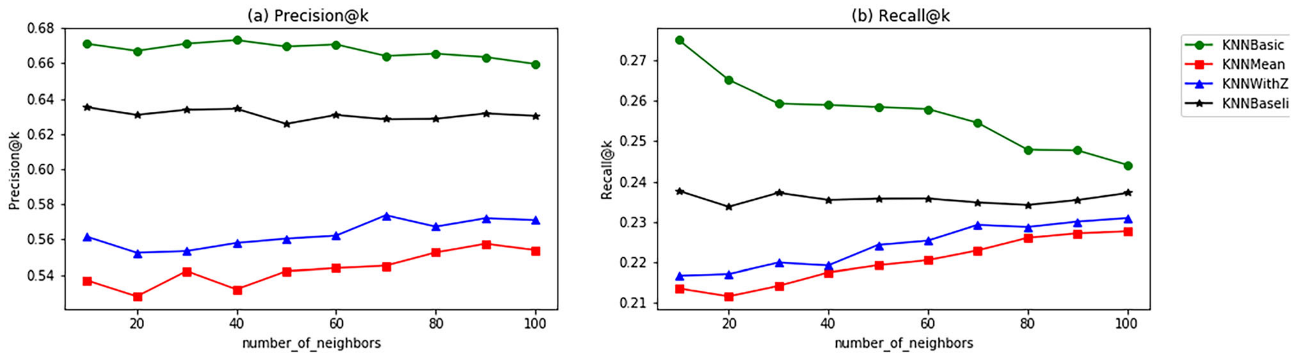

Fig. 3 Performance of different KNN methods (with cosine similarity measure) with different number of nearest neighbors (K) on ML-100K dataset. a precision@k v/s K. b recall@k v/s K

like SVM, GMM, ANN and CNN with Collaborative Filtering. Dataset other than ML-100K may be used for performance enhancement.

References

1. Adomavicius G, Tuzhilin A (2005) Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. IEEE Trans Knowl Data Eng 17(6):734–749. https://doi.org/10.1109/TKDE.2005.99
2. Ahuja R, Solanki A, Nayyar A (2019) Movie recommender system using k-means clustering and k-nearest neighbor. In: 2019 9th international conference on cloud computing, data science & engineering (confluence), IEEE, pp 263–268. https://doi.org/10.1109/CONFLUENCE.2019.8776969
3. Al Mamunur Rashid SKL, Karypis G, Riedl J (2006) Clustknn: a highly scalable hybrid model- & memory-based cf algorithm. In: Proceeding of webKDD
4. Ayub M, Ghazanfar MA, Mehmood Z, Saba T, Alharbey R, Munshi AM, Alrige MA (2019) Modeling user rating preference behavior to improve the performance of the collaborative filtering based recommender systems. PLoS ONE 14(8):e0220129. https://doi.org/10.1371/journal.pone.0220129
5. Bahadorpour M, Neysiani BS, Shahraki MN (2017) Determining optimal number of neighbors in item-based knn collaborative filtering algorithm for learning preferences of new users. J Telecommun Electron Comput Eng (JTEC) 9(3):163–167
6. Bell RM, Koren Y (2007) Improved neighborhood-based collaborative filtering. In: KDD cup and workshop at the 13th ACM SIGKDD international conference on knowledge discovery and data mining, Citeseer, pp 7–14
7. Halder S, Sarkar AMJ, Lee YK (2012) Movie recommendation system based on movie swarm. In: 2012 Second international conference on cloud and green computing, pp 804–809. https://doi.org/10.1109/CGC.2012.121
8. Harper FM, Konstan JA (2015) The movielens datasets: history and context. ACM Trans Interact Intell Syst (TIIS) 5(4):1–19. https://doi.org/10.1145/2827872
9. Isinkaye F, Folajimi Y, Ojokoh B (2015) Recommendation systems: principles, methods and evaluation. Egypt Inform J 16(3):261–273. https://doi.org/10.1016/j.eij.2015.06.005
10. Koren Y (2010) Factor in the neighbors: scalable and accurate collaborative filtering. ACM Trans Knowl Discov Data (TKDD) 4(1):1–24. https://doi.org/10.1145/1644873.1644874
11. Koren Y, Sill J (2013) Collaborative filtering on ordinal user feedback. In: Twenty-third international joint conference on artificial intelligence
12. Lund J, Ng YK (2018) Movie recommendations using the deep learning approach. In: 2018 IEEE international conference on information reuse and integration (IRI), pp 47–54. https://doi.org/10.1109/IRI.2018.00015
13. Pazzani MJ (1999) A framework for collaborative, content-based and demographic filtering. Artif Intell Rev 13(5–6):393–408. https://doi.org/10.1023/A:1006544522159
14. Resnick P, Iacovou N, Suchak M, Bergstrom P, Riedl J (1994) Grouplens: an open architecture for collaborative filtering of netnews. In: Proceedings of the 1994 ACM conference on Computer supported cooperative work, pp 175–186. https://doi.org/10.1145/192844.192905
15. Sarwar BM, Karypis G, Konstan J, Riedl J (2002) Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. In: Proceedings of the fifth international conference on computer and information technology, vol 1, pp 291–324
16. Schafer JB, Frankowski D, Herlocker J, Sen S (2007) Collaborative filtering recommender systems. In: The adaptive web, Springer, pp 291–324
17. Shani G, Gunawardana A (2011) Evaluating recommendation systems. In: Recommender systems handbook, Springer, pp 257–297. https://doi.org/10.1007/978-0-387-85820-3_8
18. Shardanand U, Maes P (1995) Social information filtering: algorithms for automating word of mouth. In: Proceedings of the SIGCHI conference on Human factors in computing systems, pp 210–217. https://doi.org/10.1145/223904.223931
19. Tan PN, Steinbach M, Kumar V (2006) Classification: basic concepts, decision trees, and model evaluation. Introd Data Min 1:145–205