Empirical Analysis of Variations of Matrix Factorization in Recommender Systems

Srilatha Tokala[1], Murali Krishna Enduri[2], T. Jaya Lakshmi[3], Koduru Hajarathaiah[4], Hemlata Sharma[5]
Department of CSE, SRM University-AP, India[1,2]
Department of Computing, Sheffield Hallam University, United Kingdom[3,5]
School of Computer Science and Engineering, VIT-AP University, India[4]

***Abstract*—Recommender systems recommend products to users. Almost all businesses utilize recommender systems to suggest their products to customers based on the customer's previous actions. The primary inputs for recommendation algorithms are user preferences, product descriptions, and user ratings on products. Content-based recommendations and collaborative filtering are examples of traditional recommendation systems. One of the mathematical models frequently used in collaborative filtering is matrix factorization (MF). This work focuses on discussing five variants of MF namely Matrix Factorization, Probabilistic MF, Non-negative MF, Singular Value Decomposition (SVD), and SVD++. We empirically evaluate these MF variants on six benchmark datasets from the domains of movies, tourism, jokes, and e-commerce. MF is the least performing and SVD is the best-performing method among other MF variants in terms of Root Mean Square Error (RMSE).**

Keywords*—Recommendations; matrix factorization; content-based; collaborative filtering; RMSE

I. Introduction

A large number of websites offer products to their users. Users purchase products based on a variety of necessities and tastes. By giving customers the best products, one can help to accelerate the purchasing process and raise customer contentment. As the state of technology advances at a rapid pace, it becomes increasingly difficult to anticipate user preferences and meet their requirements. Recommendations are quite useful in many aspects of our daily lives. We employ some external features to learn and make choices about a user's preferences for a specific product [1]. The recommender system is developed to address this issue. These systems learn from user actions and preferences to predict what content may most likely catch the user's interest [2]. Many commercial sites like YouTube, Amazon, and Netflix are highly benefited by using highly sophisticated recommender systems. Potential applications include suggesting books on Amazon, movies on Netflix, products on Flipkart, and so on. These sites continuously monitor the user's watch/view/purchase history and attempt to make educated guesses about what other products the user might find interesting. Many times, systems ask users to provide explicit ratings on used products. This rating information is a significant input to the recommender systems [3].

In 1979, a computer-based librarian introduced the first iteration of the recommender system to offer customers advice on what books to read. Then it advanced in the 1990's with a lot of research achievements in various fields. A research lab Group Lens at the University of Minnesota in the United States launched another recommender system implementation in the 1990's to assist people [4]. After that, they started calling it a Group Lens Recommender System. The use of recommender systems in advancing research across disciplines and sectors has grown in recent years. Recommender systems are essential components of many online platforms, providing users with tailored content and product suggestions. These systems significantly boost user engagement and satisfaction by forecasting user preferences through historical data and behavior analysis [5]. However, despite their extensive use, traditional recommender systems encounter issues with accurately modeling preferences, ensuring fairness, mitigating bias, and maintaining transparency. Researchers have been exploring advanced methodologies to overcome these obstacles and enhance the effectiveness and dependability of recommender systems.

Causal inference methods are essential for uncovering the fundamental causes of user preferences and behaviors, thereby improving the precision and dependability of recommendations in recommender systems [6]. By leveraging these techniques, effective recommendation algorithms can greatly enhance user satisfaction through personalized content that matches individual interests and preferences. Nonetheless, selection bias remains a significant challenge, as it can result in biased and inaccurate recommendations by disproportionately representing certain user groups or preferences based on skewed data [7]. Achieving fairness in these systems is vital to ensure that all users receive equitable recommendations, regardless of their demographics or past behaviors. The integration of large language models into recommender systems can further refine the understanding of user context and intent, resulting in more sophisticated and effective recommendations [8]. Combining these advanced methodologies helps to mitigate issues of bias and fairness, ultimately improving the overall performance and trustworthiness of recommender systems [9].

There are many applications of recommender systems. Various recommender system techniques are proposed for a variety of applications related to Government, Business, Online Shopping, Library, Learning, Tourism, Group activities, and Healthcare.

1) Government: The government may greatly improve its communication with its constituents and its ability to serve the public by adopting the internet recommender system. The citizen services discover and recommend to users more significant and interesting services. One-time items will receive ratings from the business perspective services [10]. By considering the citizen's profile, more relevant and engaging services to the citizen are recommended. In business perspective, one-time items will receive ratings from the business perspective.

2) Business: Various recommender systems are developed for business promotions. Some of the systems pay attention to the recommendations initiated by individual customers that are Business-to-Customer (B2C) systems. Recommendations produced for business users on products and service is called Business-to-Business (B2B) systems.

3) Online shopping: One of the most significant tools in the realm of online purchasing is the recommender system [11]. Ratings for the purchased products by a user is the primary information that depicts the interest of the user [12]. Almost all commercial applications like Amazon, Netflix, and Flipkart provide the option for giving ratings for the products.

4) Library: To propose resources for research in the university's online libraries, Porcel *et al.* conducted research and created a recommender system [13], [14]. Applications for online libraries can employ systems of recommendations to help users search and select knowledge and data resources.

5) Learning: Learning recommender systems guide learners to select the subjects, courses, and learning information to perform learning activities [15]. Digital libraries contain a huge amount of e-documents that a user can choose from [16].

6) Tourism: Recommender Systems are also used to give recommendations for tourism places to tourists. It is mainly suggested on transportation, restaurants, and lodging for users to feel comfortable reaching their destinations. Users are directed to a wide range of online resources. These services contain different perspectives according to videos, music, and learning materials that are uploaded by users [17].

7) Group activities: As interactions through online have increased, the use of group activities has become more popular. Giving recommendations to a group of users having different opinions is a crucial task. The idea of group activities is to learn interactions between the users from the known group ratings [18]. In various cases, the decision has to be made by the users in both online or in without internet access. In such instances, the entire organization makes the decision to balance users' expectations in online as well as offline formats. The online group is to be formed by the system, but the offline group will be already formed [19].

8) Healthcare: Efficient and effective communication is very important in healthcare. A growing number of patients require the care of healthcare professionals from many specialties, especially those who have chronic illnesses or diseases [20].

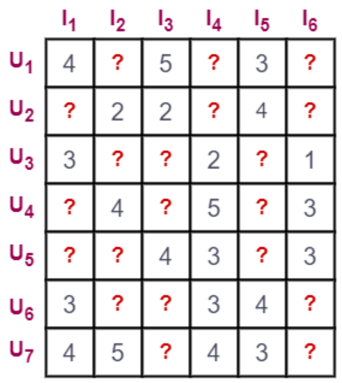
There are many other applications in the fields of medicine, banking, telecom, media, social networks, e-commerce, internet of things (IoT) [21], [22] other than the ones already mentioned.

In Section II a brief analysis of the problem statement is discussed. In Section III we discussed about the taxonomy of recommender systems followed by in Section IV, a discussion on matrix factorization techniques is mentioned. In Section V the empirical evaluation of the datasets and the rating distribution plots are discussed. Section VI discusses about the results and discussion and in Section VII gives a brief comparison of different matrix factorization methods. Finally, Section VIII ends up with final conclusion and future plan. The supplementary information for the results is placed in Section VIII-B.

II. Problem Statement

Consider a set of m users $U = \{U_1, U_2, \ldots, U_m\}$ and a group of n items $I = \{I_1, I_2, \ldots, I_n\}$ and a rating matrix R of size $m \times n$, R_{ij} denotes the rating provided by the user U_i to item I_j. Making user recommendations for unrated items presents a challenge.

The problem of recommender systems is depicted in Fig. 1.



	I_1	I_2	I_3	I_4	I_5	I_6
U_1	4	?	5	?	3	?
U_2	?	2	2	?	4	?
U_3	3	?	?	2	?	1
U_4	?	4	?	5	?	3
U_5	?	?	4	3	?	3
U_6	3	?	?	3	4	?
U_7	4	5	?	4	3	?

Fig. 1. Example for recommender systems.

This correlates to the issue of matrix completion, which is to fill the empty cells of R with rating information from the filled matrix entries. This problem is challenging because the real-world rating matrices are huge in size and sparse in nature. For instance, Amazon product recommendation is represented as a matrix containing around 197 million users and 12 million products. As a single user may not rate many products, the product vector of the user has more empty cells than ratings. Hence, it is extremely challenging to predict recommendations for the next user action based on these fewer interactions and it is called a data sparsity problem.

Handling sparse rating data in recommendation systems is challenging due to extreme sparsity, where missing data makes it hard to find patterns and leads to less accurate predictions. Overfitting occurs as models may capture noise from sparse data, reducing their ability to generalize. The cold start problem also arises when new users or items lack enough interaction history, limiting accurate recommendations. Solutions include regularization, hybrid models, and using implicit feedback or additional information to address these issues.

There are numerous methods of solving matrix completion issues. One of the traditional methods for matrix completion is matrix factorization (MF). This study focuses on solutions based on MF. Table I provides the notations utilised in this work.

Next section describes the existing methods of recommendation.

TABLE I. NOTATIONS

Notation	Usage
R	Rating Matrix
m	Number of Users
n	Number of Items
i	user
j	item/product
X	Latent features for the Users
Y	Latent features for the Items
k	Number of features extracted
U	Set of Users
I	Set of Items
δ	Regularization Constant
γ	Learning Rate
$\widetilde{R}$	Prediction rating
β	Distribution Parameter Set
α	Distribution Hyper parameter
P,Q	Orthogonal Matrices
s	Singular Matrix

III. LITERATURE

Information filtering in recommender systems involves choosing and displaying relevant information or items for users based on their preferences, behaviors, and interactions [23]. This entails processing large amounts of data to find and present content, products, or services that are most likely to appeal to the user [24]. By providing tailored suggestions that suit each user's particular preferences and needs, the goal is to enhance the user experience. There are many classifications of recommender systems in practice. Fig. 2 shows a popular classification.

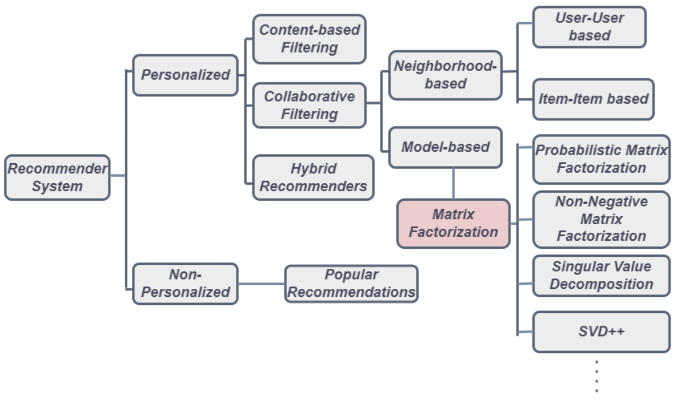


Fig. 2. Classification of recommender systems.

Non-personalized and personalised recommender systems are the two main classifications of recommender systems. Non-personalized recommenders show users only the most popular items, regardless of their purchases/interests. Based on their purchases and reviews, personalised recommenders analyse the tasks of users and make pertinent product recommendations. For instance, suggesting the most popular web series such as $MoneyHeist$ being recommended to all Netflix subscribers irrespective of their genre choice can be regarded as a non-personalized recommendation. On the other hand, suggesting movies/TV shows belonging to a genre that the user watches/rates more often is a personalized recommendation.

Additional categories for personalised recommender systems include content-based filtering, collaborative filtering, and hybrid models. Content-Based (CB) Filtering utilises the item attributes in recommendation. The algorithm creates a list of products with characteristics comparable to those of products the customer has already bought or reviewed. A few items from the list with top similarity will be recommended to the user. For this purpose, metrics such as cosine, euclidean, pearson, or spearman are used [25]–[28].

An example showing content-based filtering is depicted in Fig. 3.

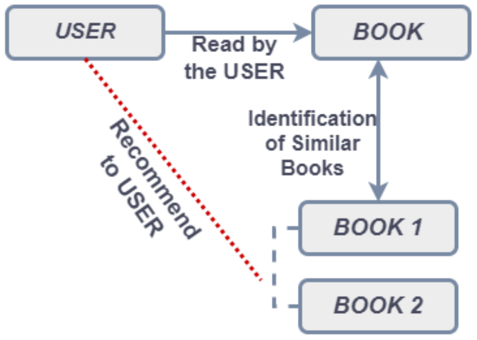


Fig. 3. Content-based filtering.

The recommender system finds additional books (let's say BOOK1 and BOOK2) that are comparable to the one the USER has already read and recommends those to the USER.

The content-based recommendation requires domain knowledge to identify attributes that may be non-available due to privacy concerns. This is a major limitation of this class of recommendation systems. However, CB addresses the cold start problem effectively. Collaborative Filtering (CF) suggests items/products based on the user's previous choices [29]–[34]. For a specific user i, CF finds additional users who share i's preferences and makes suggestions based on their choices. Their interactions with various products that user i purchased/rated can be used to determine if they have comparable tastes. Collaborative filtering key benefit is that it doesn't require domain knowledge [35]. The process of CF for book recommendation is given in Fig. 4.

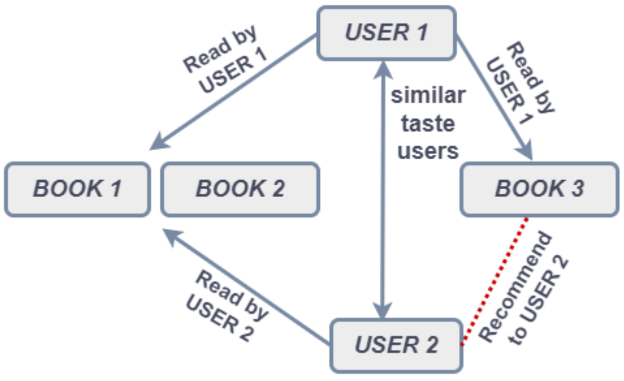


Fig. 4. Collaborative filtering.

In this example, there are two different users (say USER1 and USER2) who are having similar tastes. If there are two different books (say BOOK1 and BOOK2) that are read by both users, the recommender system identifies the books that are read by only one user (say USER1) and recommends the remaining books (say BOOK3) that are not read to the other user (say USER2).

The two approaches of calculating user similarity are Neighborhood-based and Model-based methods.

Neighborhood-based Collaborative Filtering (NCF) methods are also called Memory-based models or Heuristic-based models. The NCF technique forecasts the similarity between users and items by analyzing user-item interactions through heuristics. It employs two approaches: user-user collaborative filtering and item-item collaborative filtering [36].

- User-User based Collaborative Filtering: When producing predictions, the user-based collaborative filtering finds the other users who are engaging in similar behaviors. For user's having similar interactions, the items are recommended. It predicts the interest of an item that depends on the rating information from similar users [37]. The steps to compute user-user similarities are given below:
    - Build a user vector A_i for each individual user i. A_i will be of size n, n being the number of items. $A[j]$ is 1 if user buys item j, otherwise it is zero.
    - Compute the similarity matrix M, of size $m \times m$ where m is the number of users such that $M[i_1, i_2]$ = similarity(i_1, i_2).
    - For every user i, identify a set of users $S \in U$, where S contains the users with top similarity score with i.
    - Suggest the items that are bought by the users in S, and that are not bought by i.

The example of user-user based collaborative filtering is depicted in Fig. 5.

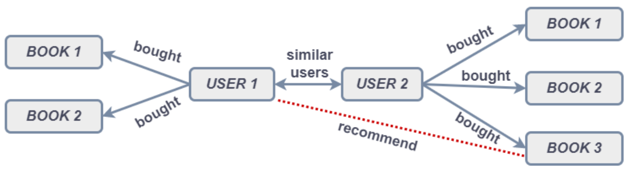


Fig. 5. User-user based collaborative filtering.

To recommend books to USER1, the recommender computes the similar users of USER1, which is USER2 in the first step. BOOK3 bought by USER2 is not bought by USER1, and is recommended to USER1.

Item-item based Collaborative Filtering (ICF): By detecting the associated subjects that users have previously rated, the item-item based collaborative filtering determines how similar the items are and provides predictions. It computes the similarity of how the target item is selected from the k-most similar items [38]. Additionally, the corresponding parallels are found. When comparable things are discovered, the prediction is made using the target user's rating as well as the average of the related items. The following are the steps to compute item-item similarities:

- Find the previously liked items of the target user from the historical data.
- Identify the most similar items for the previously liked items.
- Select the maximum likelihood items from similar item sets.
- Introduce the products to the target user.

The example of item-item based collaborative filtering is depicted in Fig. 6.

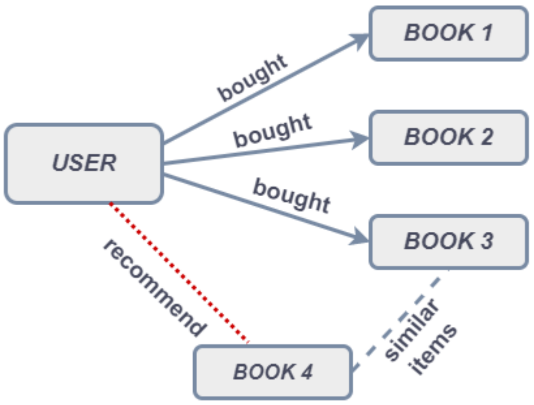


Fig. 6. Item-item based collaborative filtering.

To suggest books to a USER, the recommender system computes the maximum likelihood of similar books (say BOOK4) bought by the USER from the historical data and recommends them to the USER.

Model-based Collaborative Filtering trains a model using historical information on user-item ratings. Once the model is trained, ratings can be predicted using the model [39]. One of the well-liked model-based techniques is $Matrix\ Factorization\ (MF)$. The goal of this study is to compare and assess the performance of several MF approaches in different areas. The next section discusses MF and its variations in detail.

IV. Matrix Factorization Techniques

In this section, various matrix factorization methods are discussed. In every matrix factorization method, ratings are predicted and the recommendations are given to the users. So, the evaluation metric for our analysis is limited to Root Mean Square Error (RMSE).

The basic procedure of MF is shown in Fig. 7.

Further, five variations of MF techniques namely matrix factorization (MF), probabilistic matrix factorization (PMF), non-negative matrix factorization (NMF), singular value decomposition (SVD), and SVD++ are elaborated. Each of the variant addresses the challenges of collaborative filtering like cold start problem, and data sparsity in different ways.

MF addresses data sparsity by decomposing the user-item interaction matrix into lower-dimensional user and item matrices, capturing latent factors that help to estimate missing values

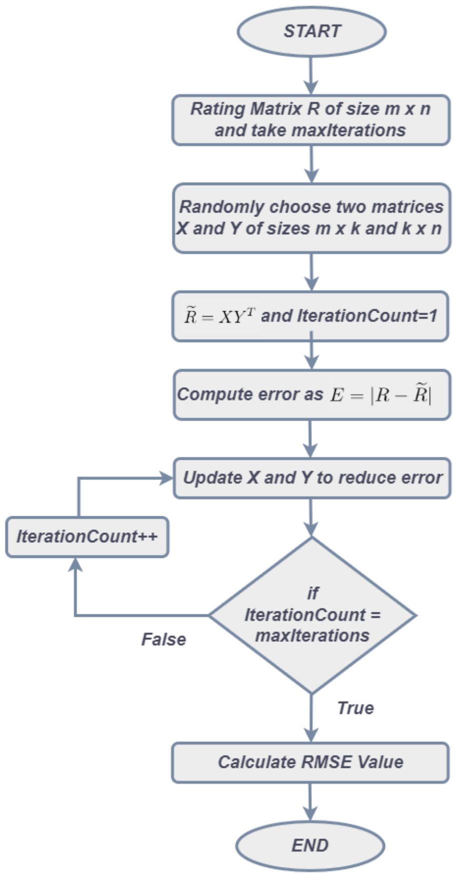

START
Rating Matrix R of size m x n and take maxIterations
Randomly choose two matrices X and Y of sizes m x k and k x n
$\widetilde{R} = XY^T$ and IterationCount=1
Compute error as $E = |R - \widetilde{R}|$
Update X and Y to reduce error
IterationCount++
if IterationCount = maxIterations
False
True
Calculate RMSE Value
END

Fig. 7. Flow chart for the procedure of Matrix Factorization (MF).

and fill gaps created by unobserved ratings. However, MF faces challenges with the cold start problem, as it depends on historical interactions to learn these latent factors, making it difficult to generate accurate recommendations for new users or items without prior rating data. PMF extends MF with a probabilistic framework that regularizes factorization, helping to manage sparse data. While PMF also faces cold start challenges due to reliance on historical data, Bayesian approaches can mitigate this by incorporating priors on latent factors. NMF decomposes the matrix into non-negative latent factors, capturing additive relationships and handling sparse data more effectively. However, it still relies on sufficient observed data and struggles with cold start, though variations incorporating content-based data can help address this limitation. SVD factorizes the matrix into orthogonal components, capturing key features with reduced dimensions and approximating missing values through low-rank approximations. However, it requires observed data for decomposition, making it less effective for cold start scenarios, as it lacks a direct mechanism for handling users or items without prior interactions. SVD++ extends standard SVD by incorporating both explicit ratings and implicit feedback, such as clicks and views, which helps mitigate data sparsity by providing additional data points. While it improves cold start handling for users through implicit feedback, it still requires some interaction data and remains limited for completely new users or items.

A rating matrix R is of size $m \times n$ is input to any MF method, where m, n are the number of users and items. R is factored into two latent feature matrices X and Y [40], [41].

Three steps are common in these MF methods:

1) Initialization of latent feature matrices X and Y: It is a common practice to initialize the matrices X and Y randomly. Different MF techniques use different data distributions to generate these random numbers.
2) Computation of predicted rating matrix: The predicted rating matrix $\widetilde{R}$ is computed by extracting k users and items latent features. The value of k can be fixed empirically. These latent feature matrices are multiplied to get the overall predicted matrix. The sample process is shown below.

$$\underset{m \times n}{\begin{bmatrix} r_{11} & .. & r_{1n} \\ . & .. & . \\ . & .. & . \\ . & .. & . \\ r_{m1} & .. & r_{mn} \end{bmatrix}} = \underset{m \times k}{\begin{bmatrix} x_{11} & .. & x_{1k} \\ . & .. & . \\ . & .. & . \\ . & .. & . \\ x_{m1} & .. & x_{mk} \end{bmatrix}} \underset{k \times n}{\begin{bmatrix} y_{11} & .. & y_{1n} \\ . & .. & . \\ . & .. & . \\ . & .. & . \\ y_{k1} & .. & y_{kn} \end{bmatrix}}$$

Where, k is the number of features extracted, m is the users, n is the items, X is a matrix representing latent features of the users, Y denotes the latent features of the items. The relation between R and $\widetilde{R}$ is shown in Eq. 1.

$$R \approx XY^T \tag{1}$$
$$\widetilde{R} = XY^T$$

The error in the prediction is determined using the difference between corresponding cells in R and $\widetilde{R}$ after computing $\widetilde{R}$. A few common error metrics include Mean Absolute Error (MAE), Regularised Square Error (RSE), and Root Mean Square Error (RMSE), as illustrated in Eq. 2, Eq. 3, and Eq. 4, respectively.

Eq. 2 calculates Root Mean Square Error (RMSE) by subtracting the original rating from the predicted rating values.

$$RMSE = \sqrt{\frac{1}{N} \sum (r_{ij} - \widetilde{r}_{ij})^2} \tag{2}$$

where N is the number of predictions, $\widetilde{r}_{ij}$ is the predicted rating, and r_{ij} is the original rating.

According to Eq. 3, the Regularised Square Error (RSE) is produced by subtracting the original rating from the predicted rating values and adding regularisation factors.

$$RSE = (r_{ij} - \widetilde{r}_{ij})^2 + \delta \tag{3}$$

Mean Absolute Error (MAE) is calculated as shown in Eq. 4, by subtracting the original rating's absolute value from the anticipated rating values.

$$MAE = \frac{1}{|N|} (|r_{ij} - \widetilde{r}_{ij}|) \tag{4}$$

Minimising the discrepancy between the actual and predicted rating matrices is the key job here.

In general, an objective function shown in Eq. 5 is used for that task.

$$min\ \frac{1}{2}||R - XY||^2 \tag{5}$$

The updating of the latent user and item matrices, which are covered below, is necessary for the goal function.

1) Update the latent feature matrices X and Y: Different update rules are used by MF methods to reduce the error computed in step 2.
2) Step 3 is repeated until either error doesn't remain the same in two successive steps or the error is less than a chosen threshold. But in most of the programming solutions, a fixed number of iterations is taken as a terminating point.

To summarise, different MF techniques vary in steps 1 (Initialization), and 3 (update rule). The following sections describe the variations in detail.

A. Matrix Factorization

1) Initialization of latent feature matrices: The initialization of X and Y are purely random values with 0 to 1 distribution in basic MF [42], [43].

2) Update rule to reduce the error between actual and predicted rating matrices:: By multiplying the rating vectors for the person and the object, as stated in Eq. 6, one can find the original rating.

$$r_{ij} \approx x_i y_j^T \tag{6}$$

Utilising the observed ratings while reducing the squared error is one method for computing the empty ratings in the matrix. The square error minimization is shown in Eq. 7.

$$min \sum_{i,j} (r_{ij} - x_i y_j^T)^2 \tag{7}$$

The result will overfit the training data and to overcome the overfitting in squared error a regularization term is incorporated is shown in Eq. 8. Regularization is controlled by using a regularization constant δ known as Regularized Square Error (RSE).

$$min \sum_{i,j} (r_{ij} - x_i y_j^T)^2 + \delta(||x_i||^2 + ||y_j||^2) \tag{8}$$

where $||.||$ is the frobenius norm. Alternating least squares or stochastic gradient descent can be used to estimate this value. According to Eq. 9, every rating within the training data has been predicted via stochastic gradient descent, and the prediction error is calculated.

$$e_{ij} = r_{ij} - x_i y_j^T \tag{9}$$

Then update the vectors y_j and x_i with a constant γ called the learning rate, and δ as the regularization constant. Updating the values of y_j and x_i is shown in Eq. 10.

$$\begin{aligned} y_j &\longleftarrow y_j + \gamma(e_{ij}x_i - \delta y_j) \\ x_i &\longleftarrow x_i + \gamma(e_{ij}y_j - \delta x_i) \end{aligned} \tag{10}$$

B. Probabilistic Matrix Factorization

1) Initialization of latent feature matrices: The initialization of X and Y are random values of 0 to 1 with normal distribution.

2) Update rule to reduce the error between actual and predicted rating matrices: By fixing the parameters, the log-posterior value on the predicted rating matrix $\widetilde{R}$ is observed from the original rating matrix R [44]. To maximize the log-posterior for the user's and item's latent features, some additional regularization hyperparameters are added and fixed to minimize the sum of squares as shown in Eq. 11.

$$\begin{aligned} E = &-\frac{1}{2}\left[\sum_{i=1}^{m}\sum_{j=1}^{n}(r_{ij} - x_i^T y_j)^2_{(i,j)\in\Omega_{R_{ij}}}\right] \\ &- \frac{1}{2}\left[\delta_X \prod_{i=1}^{m} ||x_i||^2_{Fro} + \delta_Y \prod_{j=1}^{n} ||y_j||^2_{Fro}\right] \end{aligned} \tag{11}$$

where

$$\delta_X = \frac{\sigma_X^2}{\sigma^2}, \delta_Y = \frac{\sigma_Y^2}{\sigma^2}$$

The procedure for calculating the log-posterior distribution is as follows: An approach that offers a statistical framework using the Bayes theorem for the model rating matrix R called Probabilistic Matrix Factorization (PMF) which is proposed by Salkhutdinov and Mnih [45]. PMF is a probabilistic linear model with gaussian distribution which is used for initial latent feature matrices X and Y. By fixing the parameters, the log-posterior value on the predicted rating matrix $\widetilde{R}$ is observed from the original rating matrix R [44].

$$p(\beta|Z,\alpha) = \frac{p(Z|\beta,\alpha)p(\beta|\alpha)}{p(Z|\alpha)} \propto p(Z|\beta,\alpha)p(\beta|\alpha) \tag{12}$$

In this case, Z represents dataset, β represents the distribution parameter set, and α represents the distribution hyper parameter. The posterior distribution, also known as a-posteriori, is denoted by $p(\beta|Z,\alpha)$. $p(Z|\beta,\alpha)$ is the likelihood and $p(\beta|\alpha)$ is the prior. More information about the data distribution can be obtained through the training process, and the model parameter β can be adjusted to fit the data. Let R_{ij} represent the rating of the user i on the item j and let $X \in R^{m\times k}$ and $Y \in R^{k\times n}$ are the users and items latent feature vectors respectively. Here we assume that the entries of R are normally distributed around the inner product of (X_i, Y_j)

with a common variance. We will now use our rating matrix for the predictions.

$$\beta = \{X, Y\}, \quad Z = R, \quad \alpha = \sigma^2$$

where σ^2 is the variance of the Gaussian distribution. We get this by substituting these values in Eq. 12.

$$p(X, Y|R, \sigma^2) = p(R|X, Y, \sigma^2)p(X, Y|\sigma_X^2, \sigma_Y^2) \tag{13}$$

In Eq. 13, X and Y values are independent of each other, and hence the equation can be rewritten as shown in Eq. 14.

$$p(X, Y|R, \sigma^2) = p(R|X, Y, \sigma^2)p(X, \sigma_X^2)p(Y, \sigma_Y^2) \tag{14}$$

Let I_{ij} be defined as the likelihood of R entries such that if the value is 1, the entry is observed, and if the value is 0 the entry is not observed. Adopt a gaussian-distributed probabilistic linear model and specify the conditional probability across the observed ratings as per Eq. 15.

$$p(R|X, Y, \sigma^2) = \prod_{i=1}^{m}\prod_{j=1}^{n}[N(r_{ij}|x_i^T y_j, \sigma^2)]^{I_{ij}} \tag{15}$$

The new assumption about the likelihood is that R's entries are independent, every entry has a normal distribution, and entries all have the same variance σ^2.

The prior distributions of X, Y are shown in Eq. 16 and Eq. 17.

$$p(X|\sigma_X^2) = \prod_{i=1}^{m} N(x_i|0, \sigma_X^2) \tag{16}$$

$$p(Y|\sigma_Y^2) = \prod_{j=1}^{n} N(y_j|0, \sigma_Y^2) \tag{17}$$

In these priors we assume that X and Y rows are correlated, every entry has a normal distribution, and entries all have the same variance σ^2.

Replacing Eq. 15, Eq. 16 and, Eq. 17 in Eq. 14 we get,

$$\begin{gathered} p(R|X, Y, \sigma^2) = \prod_{i=1}^{m}\prod_{j=1}^{n}[N(r_{ij}|x_i^T y_j, \sigma^2)]^{I_{ij}} \\ \prod_{i=1}^{m} N(x_i|0, \sigma_X^2) \prod_{j=1}^{n} N(y_j|0, \sigma_Y^2) \end{gathered} \tag{18}$$

For training our model, we apply logarithms on both sides of Eq. 18 and then apply derivatives on both sides of the equation. Then the expression for log-posterior is like

$$\begin{gathered} lnp(X, Y|R, \sigma^2) = -\frac{1}{2\sigma^2}\prod_{i=1}^{m}\prod_{j=1}^{n} I_{ij}(r_{ij} - x_i^T y_j)^2 \\ -\frac{1}{2\sigma_X^2}\prod_{i=1}^{m}||x_i||_{Fro}^2 - \frac{1}{2\sigma_Y^2}\prod_{j=1}^{n}||y_j||_{Fro}^2 \end{gathered}$$

Here, the Fro suffix is called the Frobenius norm is given by

$$||x||_{Fro}^2 = x^T x$$

C. Non-Negative Matrix Factorization

There are many applications in which the data is analyzed to be non-negative, and many of the tools follow this property [46]. The idea of NMF, which forces the data to be non-negative, gave rise to the need for low-rank approximation for development. NMF is used as a tool for the analysis of high-dimensional with non-negative entries in the data. NMF should consist of only non-negative constraints as a part of the representation. There are different variants of NMF algorithms proposed [47] and we are using basic NMF. NMF was introduced with the name positive matrix factorization by Paatero and Tapper [48]. Researchers paid attention to NMF after the work given by Lee and Sung. They discussed more on usage and importance of NMF [49].

1) Initialization of latent feature matrices: The initialization of latent feature matrices X and Y are taken as non-negative random values.

2) Update rule to reduce the error between actual and predicted rating matrices: Finding an estimate of a non-negative matrix R, which is represented as the product of latent feature matrices X and Y, is the primary goal of NMF.

$$R \approx XY$$

where R is a $m \times n$ rating matrix. The approximation of R with product of matrices X $(m \times k)$ and Y $(k \times n)$ by considering $k \leq (m, n)$. The latent feature matrices X and Y, can be derived by using multiplicative update method that consist of some update rules. The rules are explained in [49]. The non-negativity property is maintained by both matrices X and Y.

$$\begin{aligned} X &= X \cdot \times ((R \cdot /(X \times Y + (R == 0))) \times Y^T) \\ Y &= Y \cdot \times (X^T \times (R \cdot /(X \times Y + (R == 0)))) \end{aligned}$$

Similarly, the dot division of X and Y is $X \cdot /Y$, where $X \cdot \times Y$ is element-wise division calculated as the dot product of X and Y. The product of two matrices X and Y is $X \times Y$. The transposed version of the matrix X is X^T. In the denominator, the expression $R == 0$ is used to avoid division by zero.

The properties of NMF are only non-negative values are allowed into the resultant matrix, since non-negative values are only allowed, the matrix is allowed to only add but not subtract, and the result of the factorization is not unique.

D. Singular Value Decomposition

The singular value decomposition (SVD) method was first applied for recommender systems [50]. In MF using SVD, the rating matrix R decomposes into three latent feature matrices P, s, and Q, as shown in Eq. 19 where the rating matrix R is of size $m \times n$ and the latent factor matrices P is of size $m \times m$, s is of size $m \times n$, and Q is of size $n \times n$. Here, P and Q are orthogonal matrices, and s is a singular matrix. The latent feature matrices X and Y are computed as follows: X=$P.s$ and Y= Q [51].

$$R \approx PsQ^T \tag{19}$$

Better performance is achieved by using the SVD approach on its applications that reduces the dimensionality of user and item matrix [52].

E. SVD++

The main purpose of SVD++ is to identify the missing ratings in the matrix by adding implicit feedback to the user's latent feature matrix [53]. This technique is observed to be more accurate in many cases, because of including implicit feedback to user latent feature matrix [54].

The implicit feedback matrix UV is calculated as follows: The matrix U is calculated as $U = [u_{ij}]\ \forall(i,j)$ is 1 if R_{ij} is present in the original rating matrix 0 otherwise. For every non-zero entry in the matrix i^{th} row is written as $\frac{1}{\sqrt{|I_i|}}$ and is an $m \times n$ matrix. V matrix is calculated as $V = [v_{ij}]\ \forall(i,j)$ which is same as an item feature matrix of order $n \times k$. Calculate the dot product of matrices UV of order $m \times n$ and $n \times k$, resulting in a matrix of the order of $m \times k$. The implicit feedback matrix UV is to be added to X (say $X = X + UV$) before performing the dot product of X and Y^T. V matrix is assigned to Y (say $Y = V$).

The variations of different MF methods used in this work are tabulated in Table II.

V. Empirical Evaluation

The era of each dataset, where it was downloaded from, and the information that is contained in it are all fully described in this section. Exploratory data analysis reveals a full description of the dataset ratings. additionally explains the setup for conducting the analysis and arriving at the RMSE value.

A. Datasets

The primary goal of utilizing the datasets is to run the simulations. The datasets are downloaded from Kaggle, Konect, and Github. Datasets namely Movie Lens-100K, Movie Lens-1M, Film Trust, Trip Advisor, Jester, and Market datasets are taken. Each dataset contains information about the users, items, ratings, and timestamps.

The period of Movie Lens-100K dataset is from September 19^{th}, 1997 to April 22^{nd},1998 [55]. The period of Movie Lens-1M dataset is on December 2015 [56]. The period of the Film Trust dataset is on 2011 [57]. The period of Trip Advisor dataset is from March 3^{rd}, 2001 to November 1^{st}, 2009 [58]. The period of the Jester dataset is from November 2006 to November 2012 [59]. The period of Market dataset is from January 1^{st}, to April 30^{th}, 2021 [60]. Table III tabulates the dataset statistics for numerous datasets.

B. Exploratory Data Analysis

In Fig. 8, the rating distribution plots for several datasets, including Movie Lens -100K, Movie Lens -1M, Film, Trip Advisor, Jester, and Market, are displayed. The rating distribution for films in the MovieLens-100K dataset ranges from 1 to 5. In the dataset, the users are the individuals, and the items are the films. A low rating of one is provided by users 6110 (6.11%) beyond 100000 ratings, while users 34174 (27.14%) offer a rating of four for films. The rating distribution in the movie lens-1M dataset ranges from 0.5 to 5. In the dataset, the users represent individuals, while the items refer to movies. It has been noted that users 306221 (29.20%) have given films a rating of four stars, while users 8559 (0.81%) have given films a poor rating of 0.5 out of 1048576.

According to Fig. 8, the rating distribution for the film trust dataset is between 0.5 and 4. The films in the dataset are the items, and the users are the people. It has been noted that out of 35494 ratings, users 1060 (2.98%) give films a low rating of 0.5 and users 9170 (25.83%) offer films a rating of five. The 1 to 5 rating distribution is part of the Trip Advisor dataset. The hotels in the dataset are the items, and the users are the individuals. Users 10082 (5.73%) out of 175765 ratings offer a low rating of one for hotels, while users 77668 (44.18%) provide a rating of five for hotels.

The rating distribution for the jester dataset is between -10 and $+10$, as shown in Fig. 8. In the dataset, the users are the individuals, and the objects are the jokes. Out of the 1048575 ratings, it is noted that over 4000 people have given ratings of 10 or above. The rating distribution in the market dataset ranges from 1 to 5. In the dataset, the users are the individuals and the items are the things. It has been noted that out of 1048575 users' ratings, users 609417 (58.11%) give items a rating of five, while users 63082 (6.01%) give products a poor rating of two.

VI. Results Analysis

To forecast the missing values in the rating matrix, we apply five different MF methods such as MF, PMF, NMF, SVD, and SVD++ with different latent features on six data sets which are given in Section IV and Table III. For all methods, we computed RMSE value for different latent features $k = 10, 50, 100, 200, 300, 400$ with 10 steps. In this section, we have shown patterns of RMSE values with different latent features for various MF methods on six datasets. We have shown other errors (RSE, RMSE, and MAE) for latent features $k = 1, 2, \cdots, 10$ with 100 steps in supplementary information as shown in Section VIII-B.

Fig. 9 describes RMSE value on six different datasets for the MF method with $k = 10, 50, 100, 200, 300, 400$ at different steps. In movie lens-1M, film trust, and jester datasets, if k-value is increasing there is a constant range of RMSE maintained. In the movie lens-100K dataset, it is observed that as k-value is increasing there is a constant range of RMSE for k values $10, 50, 100, 200$ and the RMSE decreases for k values

TABLE II. DIFFERENCES BETWEEN DIFFERENT VARIATIONS OF MATRIX FACTORIZATION (MF) METHODS

Method	Initialization of Latent Features	Updation of Latent Features
MF	Latent feature matrices X and Y are taken as random values between 0 and 1.	Update the parameters of X and Y by adding regularization constant and learning rate to minimize the error.
PMF	Latent feature matrices X and Y are taken as normal distribution random values between 0 and 1.	Update the parameters of X and Y by adding regularization hyperparameters to minimize the error.
NMF	Latent feature matrices X and Y are taken as non-negative random values.	Apply the rules of the multiplicative update method to update the parameters of X and Y.
SVD	Latent feature matrices X and Y are taken as floating point numeric dtype random values.	No update rule is used.
SVD++	Latent factor matrices X and Y are taken by adding an implicit feedback matrix to the user latent feature matrix X.	Adding an implicit feedback matrix to the user latent feature matrix X is itself an update that is performed.

TABLE III. INSIGHTS INTO SIX DIVERSE DATASETS: MOVIE LENS-100K, MOVIE LENS-1M, FILM TRUST, TRIP ADVISOR, JESTER, AND MARKET DATASETS

Dataset	Users	Items	Ratings	Rating Range	Average Rating	Sparsity
Movie Lens-100K	943	1682	100000	1-5	3.529	0.937
Movie Lens-1M	7848	65133	1048576	0.5-5	3.522	0.998
Film Trust	1508	2071	35494	0.5-4	3.002	0.988
Trip Advisor	145316	1759	175765	1-5	4.000	0.999
Jester	31958	140	1048575	-10 - +10	0.955	0.839
Market	941860	9849	1048575	1-5	4.062	0.999

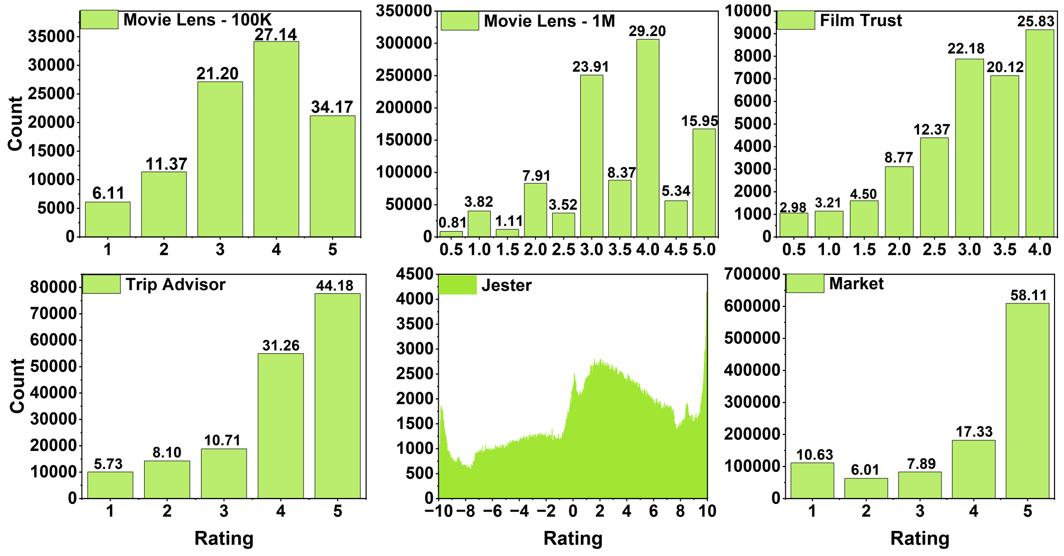


Fig. 8. Visual insights into Rating Distributions: Movie Lens-100K, Movie Lens-1M, Film Trust, Trip Advisor, Jester, and Market datasets.

300, 400. In the trip advisor dataset, the RMSE value decreases at finite steps for all k values and slightly increases. There is an increase in RMSE value if k is 10. In the market dataset, the RMSE value is decreased for all k values except for k value 200. Compared to all the datasets, the jester is giving less RMSE value. As there is a maximum of 140 items in the jester dataset, we can calculate RMSE value up to k less than or equal to $min(m, n)$.

Fig. 10 describes RMSE value on six different datasets for PMF at different steps. There are similar fluctuations in RMSE value as the k value is altered. In the movie lens-100K, for k is 10 latent features, it is observed that there is a decrease in RMSE value from 1.75 to 1.2. At 50 latent features, the RMSE value increases from 1.35 to 1.45. At 100 and 300 latent features, the RMSE increases from 1.65 to 1.75. At 200 latent features, it is observed that the RMSE decreases more from 2.0 to 1.5. For = 400 latent features, there is an increase in RMSE value from 1.55 to 1.9.

In the movie lens-1M, at 10 latent features, it is observed that there is a decrease in RMSE from 1.8 to 1.35. For k is 50 latent features, it is observed that the RMSE decreases from 1.55 to 1.35. For 100 latent features, the RMSE value reduces from 1.5 to 1.4. At 200 latent features, the RMSE value increases from 1.2 to 1.35. The RMSE value falls from 1.45 to 1.3 for 300 latent features. There is an increase in RMSE value from 1.35 to 1.55 at 400 latent features. In the film trust dataset, all are behaving in the same manner except for 50 latent features. There is a drastic change in RMSE value with different behavior from 1.20 to 0.99. For the remaining k values there are slight fluctuations in RMSE value. In the trip advisor dataset, different behavior is seen for $k = 10$ latent features. For all the remaining latent features, there is a decrease in RMSE value if the latent features are increased. In the jester dataset, there is an increase in RMSE value as

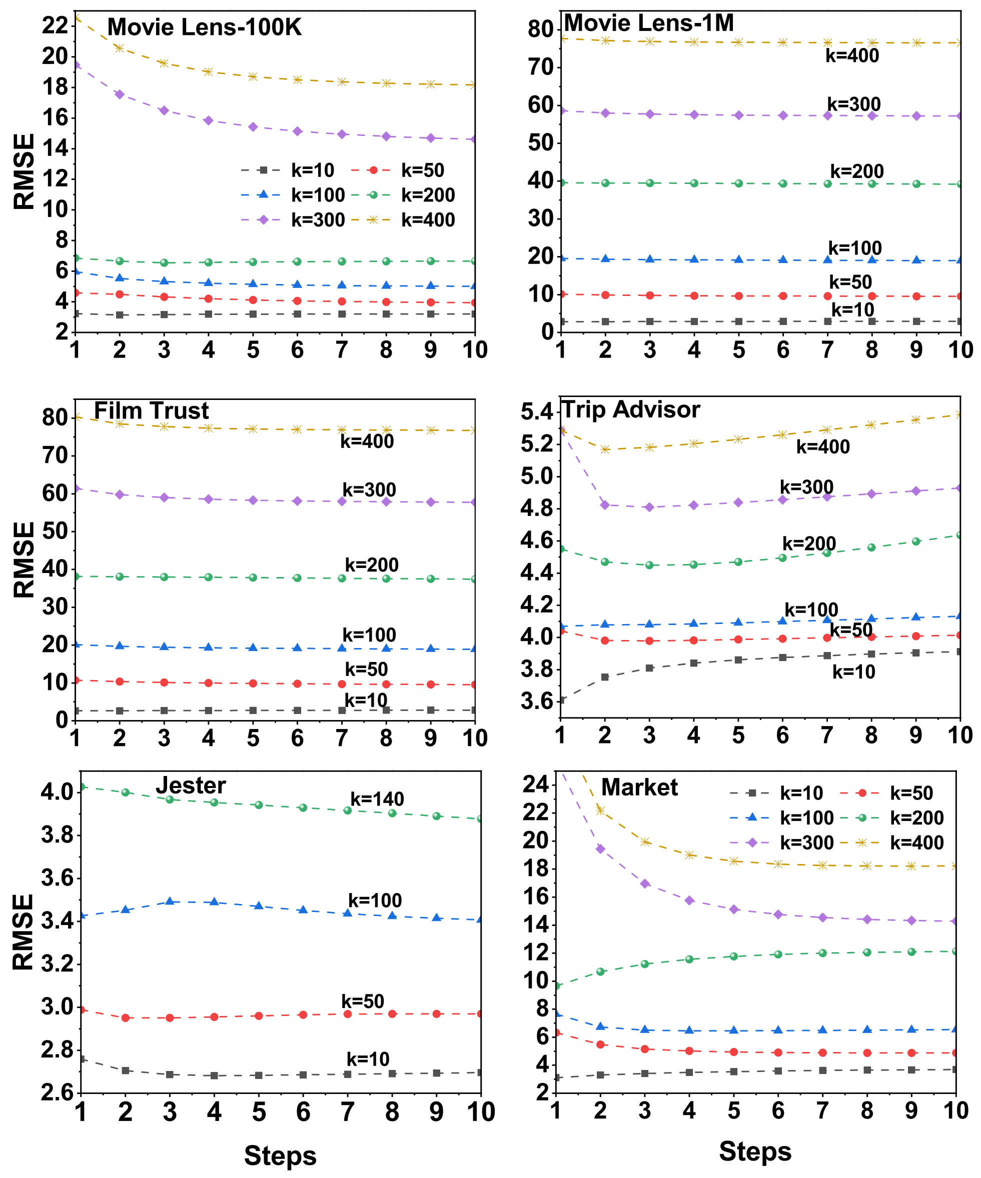


Fig. 9. Root Mean Square Error (RMSE) graphs for Matrix Factorization (MF) on six datasets, namely, Movie Lens-100K, Movie Lens-1M, Film Trust, Trip Advisor, Jester, and Market datasets for 10 steps and 400 latent features.

the latent features are increased. There is a decrease in RMSE value at 100. The constant RMSE is maintained at k value 140. In the market dataset, there is a decrease in RMSE value if the k-value increases.

For the NMF method, we have plotted RMSE values with different latent features on six datasets in Fig. 11. In all datasets, if the k-value is increasing RMSE decreases. Compared to all datasets, the NMF method gives less RMSE value for film trust and market datasets. More RMSE value for jester, movie lens-100K datasets. For any latent features, the RMSE value is decreasing with different steps.

Fig. 12 (a), (b), and (c) describes RMSE value of SVD method with various k values for pair of datasets movie lens-100K, jester, movie lens-1M, market, and trip advisor, film trust respectively. Across all datasets, an increase in the k-value is observed to correspond with a decrease in the RMSE value. Compared to all datasets, the SVD method gives less RMSE value for film trust and market datasets. More RMSE value for movie lens-100K datasets. For the jester dataset a drastic change in the RMSE value of the SVD method when increasing the k-value because the number of items is very less.

Fig. 13 describes RMSE value on six different datasets for SVD++ method at different steps. In all datasets, if the k-value is increasing then RMSE also increases. In the jester dataset, it is observed that the RMSE is maintained constant for different steps. Compared to all datasets, the SVD++ method gives less

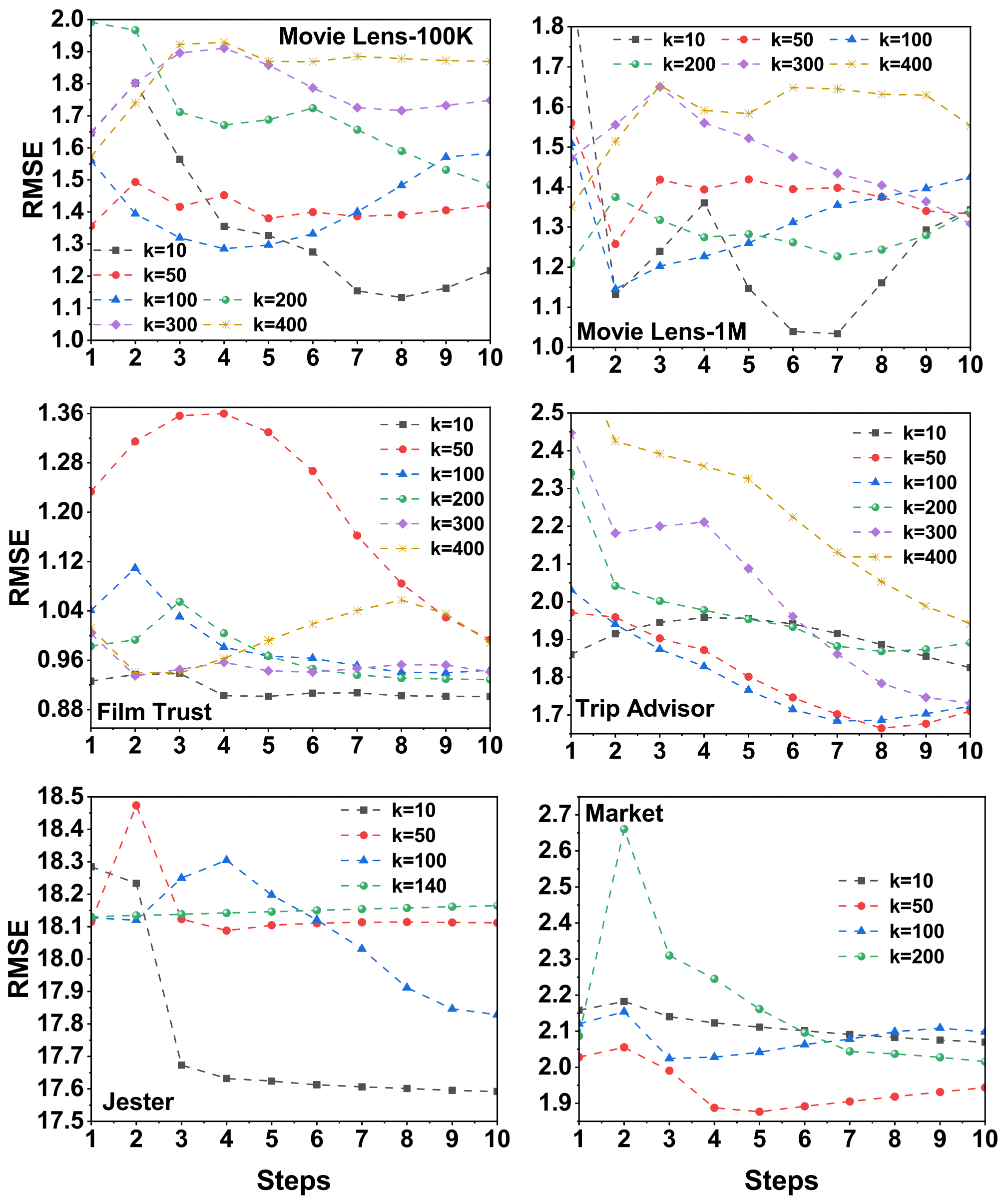


Fig. 10. Root Mean Square Error (RMSE) graphs for Probabilistic Matrix Factorization (PMF) on six datasets, namely, Movie Lens-100K, Movie Lens-1M, Film Trust, Trip Advisor, Jester, and Market datasets for 10 steps and 400 latent features.

RMSE value for the film trust dataset and more RMSE value for the jester dataset.

VII. Comparison of MF Methods

The comparison of different MF methods on different datasets namely movie lens-100K, film trust, movie lens-1M, trip advisor, jester, and market with $k = 5$ in Table IV, and $k = 10$ in Table V, and $k = 100$ in Table VI. Compared to all the MF methods NMF, and SVD methods gives less RMSE value with different k-values. It is observed that in MF, PMF, and SVD++ methods, by increasing the k-value RMSE also increases. Whereas in NMF, and SVD methods there is a decrease in RMSE value as k-value increases. The k-value in each method is the number of latent features that are divided into the items. As compared to all MF methods, if we consider more groups for items in the dataset, MF, PMF, and SVD++ give more errors for suggesting an item to the user. Whereas, if we increase the groups in NMF and SVD methods, the error value that is obtained is less while predicting a recommendation to the user.

For film trust and movie lens-1M datasets, the prediction performance of MF is drastically decreasing with an increase in the number of latent features. The dominant observation is that the rating distribution for the two datasets is right skewed and sparse. However, NMF and SVD are not affected much by this skewed rating as well as the number of latent features.

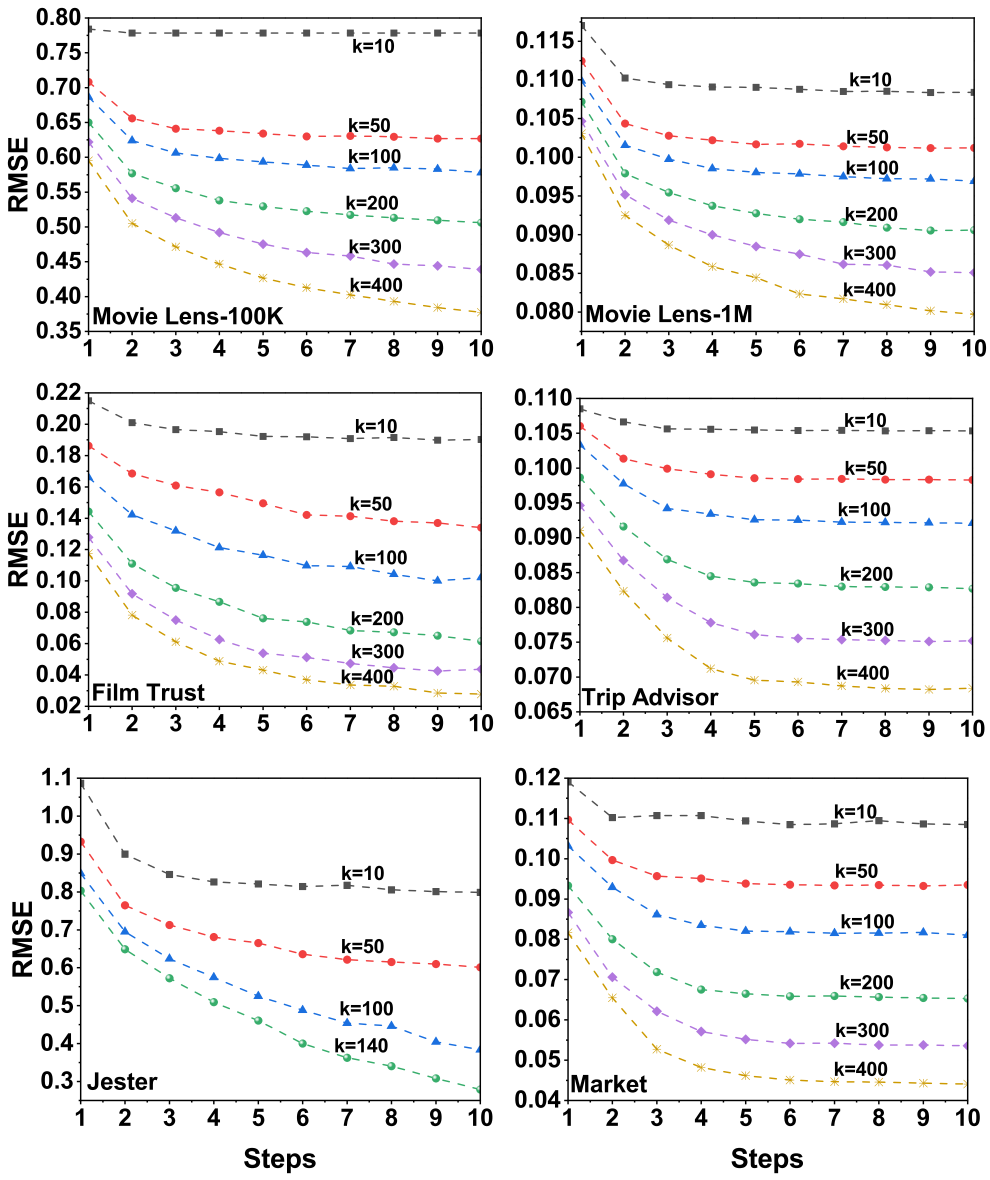


Fig. 11. Root Mean Square Error (RMSE) graphs for Non-Negative Matrix Factorization (NMF) on six datasets, namely, Movie Lens-100K, Movie Lens-1M, Film Trust, Trip Advisor, Jester, and Market datasets for 10 steps and 400 latent features.

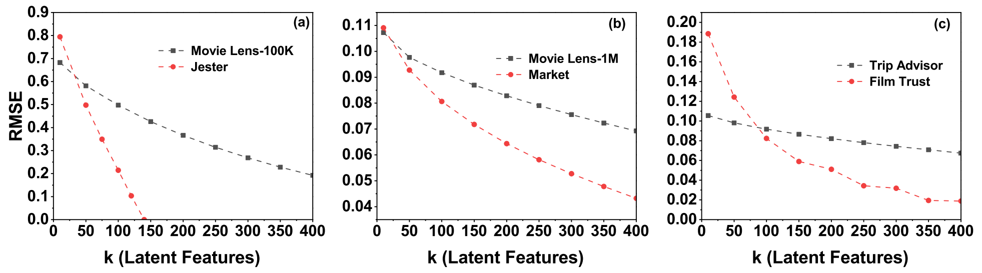


Fig. 12. Root Mean Square Error (RMSE) graphs for Singular Value Decomposition (SVD) on six datasets, namely, Movie Lens-100K, Movie Lens-1M, Film Trust, Trip Advisor, Jester, and Market datasets for 10 steps and 400 latent features.

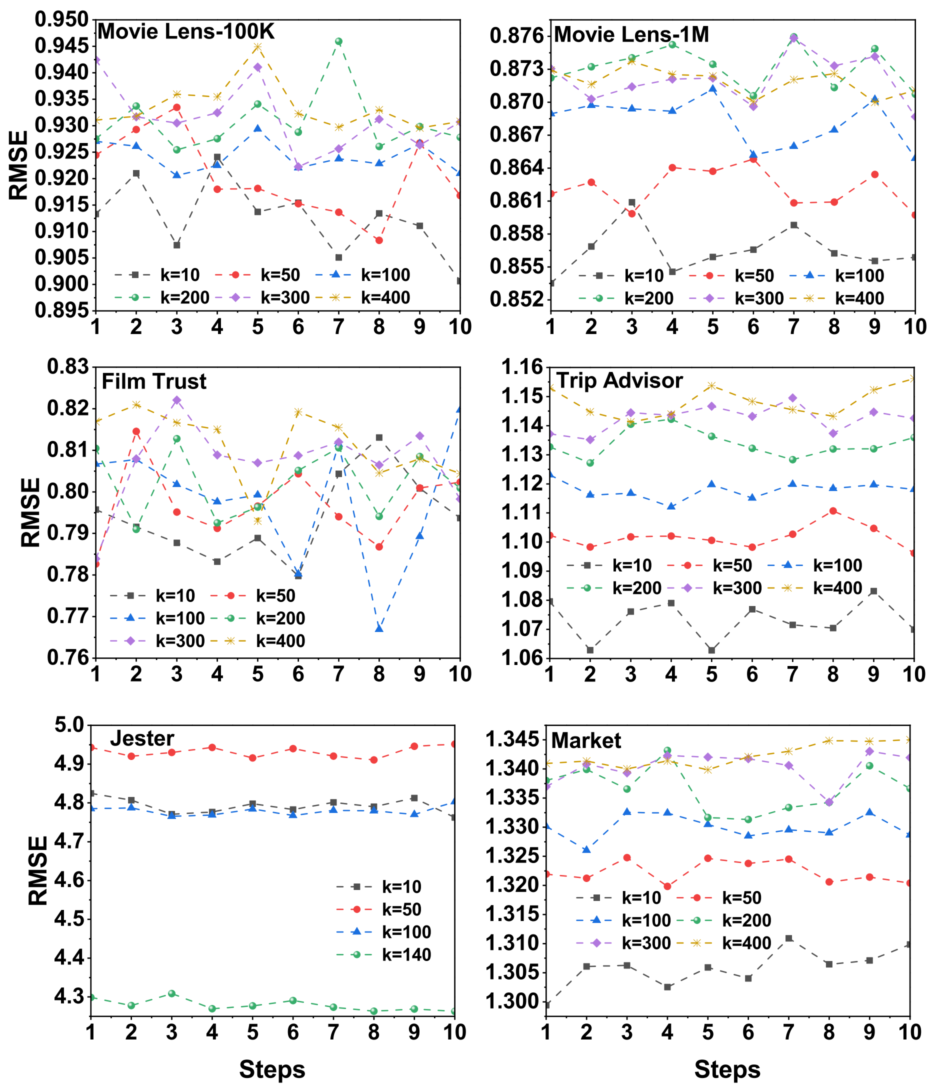


Fig. 13. Root Mean Square Error (RMSE) graphs for SVD++ on six datasets, namely, Movie Lens-100K, Movie Lens-1M, Film Trust, Trip Advisor, Jester, and Market datasets for 10 steps and 400 latent features.

TABLE IV. RMSE Values for Different Variations of Matrix Factorization (MF) on Six Different Datasets Namely Movie Lens-100K, Movie Lens-1M, Film Trust, Trip Advisor, Jester, and Market at $k = 5$

Dataset (↓) / MF algorithm (→)	MF	PMF	NMF	SVD	SVD++
Movie Lens-100K	1.048	1.051	0.720	0.711	0.908
Movie Lens-1M	2.236	1.463	0.111	0.110	0.859
Film Trust	2.722	1.457	0.200	0.199	0.798
Trip Advisor	3.919	1.945	0.106	0.106	1.071
Jester	2.695	17.591	0.798	0.794	4.762
Market	3.597	2.041	0.112	0.112	1.294

TABLE V. RMSE Values for Different Variations of Matrix Factorization (MF) on Six Different Datasets Namely Movie Lens-100K, Movie Lens-1M, Film Trust, Trip Advisor, Jester, and Market at $k = 10$

Dataset (↓) / MF algorithm (→)	MF	PMF	NMF	SVD	SVD++
Movie Lens-100K	3.205	1.343	0.690	0.682	0.900
Movie Lens-1M	2.953	1.271	0.108	0.107	0.855
Film Trust	2.804	0.900	0.190	0.188	0.793
Trip Advisor	3.912	1.824	0.105	0.105	1.069
Jester	3.499	17.521	0.867	0.861	4.707
Market	3.692	2.069	0.108	0.109	1.309

VIII. Conclusion and Future Scope

A. Conclusion

Information theory is essential for improving the effectiveness of recommendation systems. By measuring the uncertainty and information content in user preferences and interactions, it offers a solid framework for creating more precise and efficient recommendation algorithms. This theoretical basis enhances the handling of sparse data, boosts prediction accuracy, and ensures more personalized user ex-

TABLE VI. RMSE VALUES FOR DIFFERENT VARIATIONS OF MATRIX FACTORIZATION (MF) ON SIX DIFFERENT DATASETS, NAMELY, MOVIE LENS-100K, MOVIE LENS-1M, FILM TRUST, TRIP ADVISOR, JESTER, AND MARKET AT $k = 100$

Dataset (↓) / MF algorithm (→)	MF	PMF	NMF	SVD	SVD++
Movie Lens-100K	5.014	1.583	0.577	0.497	0.920
Movie Lens-1M	18.994	1.425	0.096	0.091	0.864
Film Trust	18.886	0.944	0.102	0.082	0.879
Trip Advisor	4.131	1.722	0.092	0.091	1.118
Jester	3.407	17.828	0.383	0.219	4.803
Market	6.539	2.097	0.810	0.080	1.328

periences. As recommendation systems advance, the principles of information theory will continue to be vital in tackling challenges related to data complexity, user diversity, and changing preferences, resulting in more sophisticated and dependable recommendation solutions. In this study, various MF methods like MF, PMF, NMF, SVD, and SVD++ have been compared on different datasets namely movie lens-100K, film trust, movie lens-1M, trip advisor, jester, and market with different latent features on different steps. The performance of the MF methods is evaluated using RMSE. It is observed that MF is the least-performing MF method among all studied in this work. SVD is the outperforming method among all other MF algorithms. However, it has been observed that the number of latent features is affecting the prediction performance. The prediction power of MF, PMF, and SVD++ is reducing with an increase in the number of latent features. On the other hand, NMF and SVD are performing better with an increase in the number of latent features.

B. Future Scope

In the future, we would like to extend the concept of recommendation to different real-world contexts. For example, this study focuses solely on recommending a single item to an individual user. However, in practice, there are situations where recommendations are needed for a group of users. For example, a group of students might be advised on selecting an elective course based on their collective interests. This interest prompts the requirement for and creation of group recommender systems [61]. Cross-domain recommendation systems (CDR) can help mitigate this issue. CDRs use the ratings of the new item/user in one domain in another domain with transfer learning [62]–[64]. Utilizing these techniques will help recommendation systems work better by a variety of semantic information contained in knowledge graphs. A knowledge graph (KG) is a collection of relational facts, including information about the entities, entity categories, and collaborations among entities. KG embeds complex information about different relationships among real-world entities [65], [66].

REFERENCES

[1] S. Tokala, M. K. Enduri, T. J. Lakshmi, A. Abdul, and J. Chen, "Empowering quality of recommendations by integrating matrix factorization approaches with louvain community detection," *IEEE Access*, 2024.

[2] R. Chen, Q. Hua, B. Wang, M. Zheng, W. Guan, X. Ji, Q. Gao, and X. Kong, "A novel social recommendation method fusing user's social status and homophily based on matrix factorization techniques," *IEEE Access*, vol. 7, pp. 18 783–18 798, 2019.

[3] S. Tokala, J. Nagaram, M. K. Enduri, and T. J. Lakshmi, "Enhanced movie recommender system using deep learning techniques," in *2024 3rd International Conference on Computational Modelling, Simulation and Optimization (ICCMSO)*. IEEE, 2024, pp. 71–75.

[4] S. Tokala, M. K. Enduri, and T. J. Lakshmi, "Unleashing the power of svd and louvain community detection for enhanced recommendations," in *2023 IEEE 15th International Conference on Computational Intelligence and Communication Networks (CICN)*. IEEE, 2023, pp. 807–811.

[5] D. Roy and M. Dutta, "A systematic review and research perspective on recommender systems," *Journal of Big Data*, vol. 9, no. 1, p. 59, 2022.

[6] C. Gao, Y. Zheng, W. Wang, F. Feng, X. He, and Y. Li, "Causal inference in recommender systems: A survey and future directions," *ACM Transactions on Information Systems*, vol. 42, no. 4, pp. 1–32, 2024.

[7] J. Chen, H. Dong, X. Wang, F. Feng, M. Wang, and X. He, "Bias and debias in recommender system: A survey and future directions," *ACM Transactions on Information Systems*, vol. 41, no. 3, pp. 1–39, 2023.

[8] Y. Hou, J. Zhang, Z. Lin, H. Lu, R. Xie, J. McAuley, and W. X. Zhao, "Large language models are zero-shot rankers for recommender systems," in *European Conference on Information Retrieval*. Springer, 2024, pp. 364–381.

[9] Y. Wang, W. Ma, M. Zhang, Y. Liu, and S. Ma, "A survey on the fairness of recommender systems," *ACM Transactions on Information Systems*, vol. 41, no. 3, pp. 1–43, 2023.

[10] J. Lu, D. Wu, M. Mao, W. Wang, and G. Zhang, "Recommender system application developments: a survey," *Decision Support Systems*, vol. 74, pp. 12–32, 2015.

[11] J. K. Kim, Y. H. Cho, W. J. Kim, J. R. Kim, and J. H. Suh, "A personalized recommendation procedure for internet shopping support," *Electronic commerce research and applications*, vol. 1, no. 3-4, pp. 301–313, 2002.

[12] K.-j. Kim and H. Ahn, "A recommender system using ga k-means clustering in an online shopping market," *Expert systems with applications*, vol. 34, no. 2, pp. 1200–1209, 2008.

[13] C. Porcel and E. Herrera-Viedma, "Dealing with incomplete information in a fuzzy linguistic recommender system to disseminate information in university digital libraries," *Knowledge-Based Systems*, vol. 23, no. 1, pp. 32–39, 2010.

[14] C. Porcel, J. M. Moreno, and E. Herrera-Viedma, "A multi-disciplinar recommender system to advice research resources in university digital libraries," *Expert systems with applications*, vol. 36, no. 10, pp. 12 520–12 528, 2009.

[15] R. Sikka, A. Dhankhar, and C. Rana, "A survey paper on e-learning recommender system," *International Journal of Computer Applications*, vol. 47, no. 9, pp. 27–30, 2012.

[16] J. Lu, "A personalized e-learning material recommender system," in *International conference on information technology and applications*. Macquarie Scientific Publishing, 2004.

[17] J. Borràs, A. Moreno, and A. Valls, "Intelligent tourism recommender systems: A survey," *Expert systems with applications*, vol. 41, no. 16, pp. 7370–7389, 2014.

[18] Y.-L. Chen, L.-C. Cheng, and C.-N. Chuang, "A group recommendation system with consideration of interactions among group members," *Expert systems with applications*, vol. 34, no. 3, pp. 2082–2090, 2008.

[19] I. Cantador, I. Fernández-Tobías, S. Berkovsky, and P. Cremonesi, "Cross-domain recommender systems," in *Recommender systems handbook*. Springer, 2015, pp. 919–959.

[20] P. Vermeir, D. Vandijck, S. Degroote, R. Peleman, R. Verhaeghe, E. Mortier, G. Hallaert, S. Van Daele, W. Buylaert, and D. Vogelaers, "Communication in healthcare: a narrative review of the literature and practical recommendations," *International journal of clinical practice*, vol. 69, no. 11, pp. 1257–1267, 2015.

[21] Y. Himeur, A. Sayed, A. Alsalemi, F. Bensaali, A. Amira, I. Varlamis, M. Eirinaki, C. Sardianos, and G. Dimitrakopoulos, "Blockchain-based recommender systems: Applications, challenges and future opportunities," *Computer Science Review*, vol. 43, p. 100439, 2022.

[22] F. Karimova, "A survey of e-commerce recommender systems," *European Scientific Journal*, vol. 12, no. 34, pp. 75–89, 2016.

[23] L. Wu, X. He, X. Wang, K. Zhang, and M. Wang, "A survey on accuracy-oriented neural recommendation: From collaborative filtering to information-rich recommendation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 5, pp. 4425–4445, 2022.

[24] J. Liu, C. Shi, C. Yang, Z. Lu, and S. Y. Philip, "A survey on heterogeneous information network based recommender systems: Concepts, methods, applications and resources," *AI Open*, vol. 3, pp. 40–57, 2022.

[25] U. Javed, K. Shaukat, I. A. Hameed, F. Iqbal, T. M. Alam, and S. Luo, "A review of content-based and context-based recommendation systems," *International Journal of Emerging Technologies in Learning (iJET)*, vol. 16, no. 3, pp. 274–306, 2021.

[26] K. Kundegraber and S. Pletzl, "Basic approaches in recommendation systems," EasyChair, Tech. Rep., 2022.

[27] S. Chen, S. Owusu, and L. Zhou, "Social network based recommendation systems: A short survey," in *2013 international conference on social computing*. IEEE, 2013, pp. 882–885.

[28] S. Reddy, S. Nalluri, S. Kunisetti, S. Ashok, and B. Venkatesh, "Content-based movie recommendation system using genre correlation," in *Smart Intelligent Computing and Applications: Proceedings of the Second International Conference on SCI 2018, Volume 2*. Springer, 2019, pp. 391–397.

[29] N. Y. Asabere, "Review of recommender systems for learners in mobile social/collaborative learning," *International Journal of Information*, vol. 2, no. 5, 2012.

[30] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, "Evaluating collaborative filtering recommender systems," *ACM Transactions on Information Systems*, vol. 22, no. 1, pp. 5–53, 2004.

[31] M. D. Ekstrand, J. T. Riedl, J. A. Konstan *et al.*, "Collaborative filtering recommender systems," *Foundations and Trends® in Human–Computer Interaction*, vol. 4, no. 2, pp. 81–173, 2011.

[32] M. A. Hameed, O. Al Jadaan, and S. Ramachandram, "Collaborative filtering based recommendation system: A survey," *International Journal on Computer Science and Engineering*, vol. 4, no. 5, p. 859, 2012.

[33] S. A. Amin, J. Philips, and N. Tabrizi, "Current trends in collaborative filtering recommendation systems," in *World Congress on Services*. Springer, 2019, pp. 46–60.

[34] M. Jalili, S. Ahmadian, M. Izadi, P. Moradi, and M. Salehi, "Evaluating collaborative filtering recommender algorithms: a survey," *IEEE access*, vol. 6, pp. 74 003–74 024, 2018.

[35] S. Tokala, M. K. Enduri, T. J. Lakshmi, and H. Sharma, "Community-based matrix factorization (cbmf) approach for enhancing quality of recommendations," *Entropy*, vol. 25, no. 9, p. 1360, 2023.

[36] S. Gong, "A collaborative filtering recommendation algorithm based on user clustering and item clustering." *J. Softw.*, vol. 5, no. 7, pp. 745–752, 2010.

[37] J. Wang, A. P. De Vries, and M. J. Reinders, "Unifying user-based and item-based collaborative filtering approaches by similarity fusion," in *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, 2006, pp. 501–508.

[38] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th international conference on World Wide Web*, 2001, pp. 285–295.

[39] M.-P. T. Do, D. Nguyen, and L. Nguyen, "Model-based approach for collaborative filtering," in *6th International conference on information technology for education*, 2010, pp. 217–228.

[40] J. Hintz, "Matrix factorization for collaborative filtering recommender systems," 2015.

[41] D. kumar Bokde, S. Girase, and D. Mukhopadhyay, "Role of matrix factorization model in collaborative filtering algorithm: A survey," *Clinical Orthopaedics and Related Research, abs/1503.07475*, vol. 1, no. 12, 2015.

[42] J. Ivarsson and M. Lindgren, "Movie recommendations using matrix factorization," 2016.

[43] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[44] J. Liu, C. Wu, Y. Xiong, and W. Liu, "List-wise probabilistic matrix factorization for recommendation," *Information Sciences*, vol. 278, pp. 434–447, 2014.

[45] A. Mnih and R. R. Salakhutdinov, "Probabilistic matrix factorization," *Advances in neural information processing systems*, vol. 20, 2007.

[46] Z. Huang, A. Zhou, and G. Zhang, "Non-negative matrix factorization: a short survey on methods and applications," in *International Symposium on Intelligence Computation and Applications*. Springer, 2012, pp. 331–340.

[47] Y.-X. Wang and Y.-J. Zhang, "Nonnegative matrix factorization: A comprehensive review," *IEEE Transactions on knowledge and data engineering*, vol. 25, no. 6, pp. 1336–1353, 2012.

[48] P. Paatero and U. Tapper, "Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values," *Environmetrics*, vol. 5, no. 2, pp. 111–126, 1994.

[49] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[50] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Application of dimensionality reduction in recommender system-a case study," Minnesota Univ Minneapolis Dept of Computer Science, Tech. Rep., 2000.

[51] E. J. Ientilucci, "Using the singular value decomposition," *Rochester Institute of Technology, Rochester, New York, United States, Technical Report*, 2003.

[52] R. Mehta and K. Rana, "A review on matrix factorization techniques in recommender systems," in *2017 2nd International Conference on Communication Systems, Computing and IT Applications*. IEEE, 2017, pp. 269–274.

[53] J. Jiao, X. Zhang, F. Li, and Y. Wang, "A novel learning rate function and its application on the svd++ recommendation algorithm," *IEEE Access*, vol. 8, pp. 14 112–14 122, 2019.

[54] R. Kumar, B. Verma, and S. S. Rastogi, "Social popularity based svd++ recommender system," *International Journal of Computer Applications*, vol. 87, no. 14, 2014.

[55] "Kaggle," https://www.kaggle.com/datasets/prajitdatta/movielens-100k-dataset (accessed on 31 July 2023).

[56] "Kaggle," https://www.kaggle.com/datasets/odedgolden/movielens-1m-dataset (accessed on 31 July 2023).

[57] "Konect," http://konect.cc/networks/librec-filmtrust-ratings/ (accessed on 31 July 2023).

[58] "Konect," http://konect.cc/networks/wang-tripadvisor/ (accessed on 31 July 2023).

[59] "Konect," http://konect.cc/networks/jester2/ (accessed on 31 July 2023).

[60] "Github," https://github.com/MengtingWan/marketBias (accessed on 31 July 2023).

[61] S. Dara, C. R. Chowdary, and C. Kumar, "A survey on group recommender systems," *Journal of Intelligent Information Systems*, vol. 54, no. 2, pp. 271–295, 2020.

[62] M. M. Khan, R. Ibrahim, and I. Ghani, "Cross domain recommender systems: a systematic literature review," *Association for Computing Machinery Computing Surveys*, vol. 50, no. 3, pp. 1–34, 2017.

[63] M. Enrich, M. Braunhofer, and F. Ricci, "Cold-start management with cross-domain collaborative filtering and tags," in *International Conference on Electronic Commerce and Web Technologies*. Springer, 2013, pp. 101–112.

[64] I. Fernández-Tobías, I. Cantador, M. Kaminskas, and F. Ricci, "Cross-domain recommender systems: A survey of the state of the art," in *Spanish conference on information retrieval*, vol. 24. ACM Valencia, Spain, 2012.

[65] S. Bouraga, I. Jureta, S. Faulkner, and C. Herssens, "Knowledge-based recommendation systems: A survey," *International Journal of Intelligent Information Technologies*, vol. 10, no. 2, pp. 1–19, 2014.

[66] R. Burke, "Knowledge-based recommender systems," *Encyclopedia of library and information systems*, vol. 69, no. Supplement 32, pp. 175–186, 2000.

SUPPLEMENTARY INFORMATION

In the supplementary information section, RSE graphs for MF at k = 10, 50, 100, 200, 300, 400 at 10 steps and 100 steps at 10 latent features are shown and RMSE graphs for MF, PMF, NMF, and SVD methods are provided at 100 steps and 10 latent features and MAE graphs for the SVD++ method at $k = 10, 50, 100, 200, 300, 400$ at 10 steps are shown.

Fig. 14 describes the RSE value on six different datasets for the MF method on 10 steps and 400 latent features. In all datasets, it is observed that there is an increase in RSE value as the k-value increases. Compared to all the datasets less RMSE value is given by film trust and more RMSE value is given by the jester dataset.

Fig. 15 describes the RSE value on six different datasets for the MF method on 100 steps and 10 latent features. In all datasets, except for jester, it is observed that if the k-value is increasing then RSE decreases. Compared to all datasets, the MF method gives less RSE value for the market dataset and more RSE value for the jester.

Fig. 16 describes the RMSE value on six different datasets for the MF method at different steps. In all datasets, if the k-value is increasing then RMSE also increases. Compared to all datasets, the MF method gives less RMSE value for film trust, movie lens-1M datasets, and more RMSE value for movie lens-100K, trip advisor, film trust, and market.

Fig. 17 describes RMSE value on six different datasets for the PMF method on 100 steps and 10 latent features. In the movie lens-100K dataset, at $k = 1$ there is a constant behavior maintained. For all the remaining k values there are many alterations in RMSE value. In movie lens-1M, for $k = 2$, there is a major deviation in the RMSE value as compared to all remaining k values. In the film trust dataset, for all the k values between 1 to 9, there are fluctuations as they are altered. For $k = 10$ the RMSE value is high as compared to the remaining k values. In the trip advisor dataset, all the k values exhibit different behavior. In the jester dataset, as compared to the remaining datasets, the RMSE value is too high as there is a decrease in the RMSE value with different steps. In the market dataset, all the k values have different behavior with an increase in steps.

Fig. 18 describes RMSE value on six different datasets for the NMF method on 100 steps and 10 latent features. In all datasets, if the k-value is increasing then RMSE also decreases. Compared to all the datasets, the NMF method gives less RMSE value for trip advisor, movie lens-1M, and market datasets and some more RMSE value for movie lens-100K, and film trust datasets. More RMSE value is given by the jester dataset due to less items.

Fig. 19 (a), (b), and (c) describes RMSE value of SVD method with various k values for pair of datasets movie lens-100K, jester, movie lens-1M, trip advisor, and film trust, market, respectively. In all datasets, it is observed that as the k value increases there is a decrease in RMSE value. Compared to all the datasets, the SVD method gives less RMSE value for trip advisor, movie lens-1M, and market datasets. More RMSE value for movie lens-100K and jester datasets.

Fig. 20 describes the MAE value on six different datasets for the SVD++ at different steps. In movie lens-100K, movie lens-1M, trip advisor, and market datasets, it is observed that there is an increase in RMSE value as the step increases. In the jester dataset, a similar constant RMSE value is maintained at different steps.

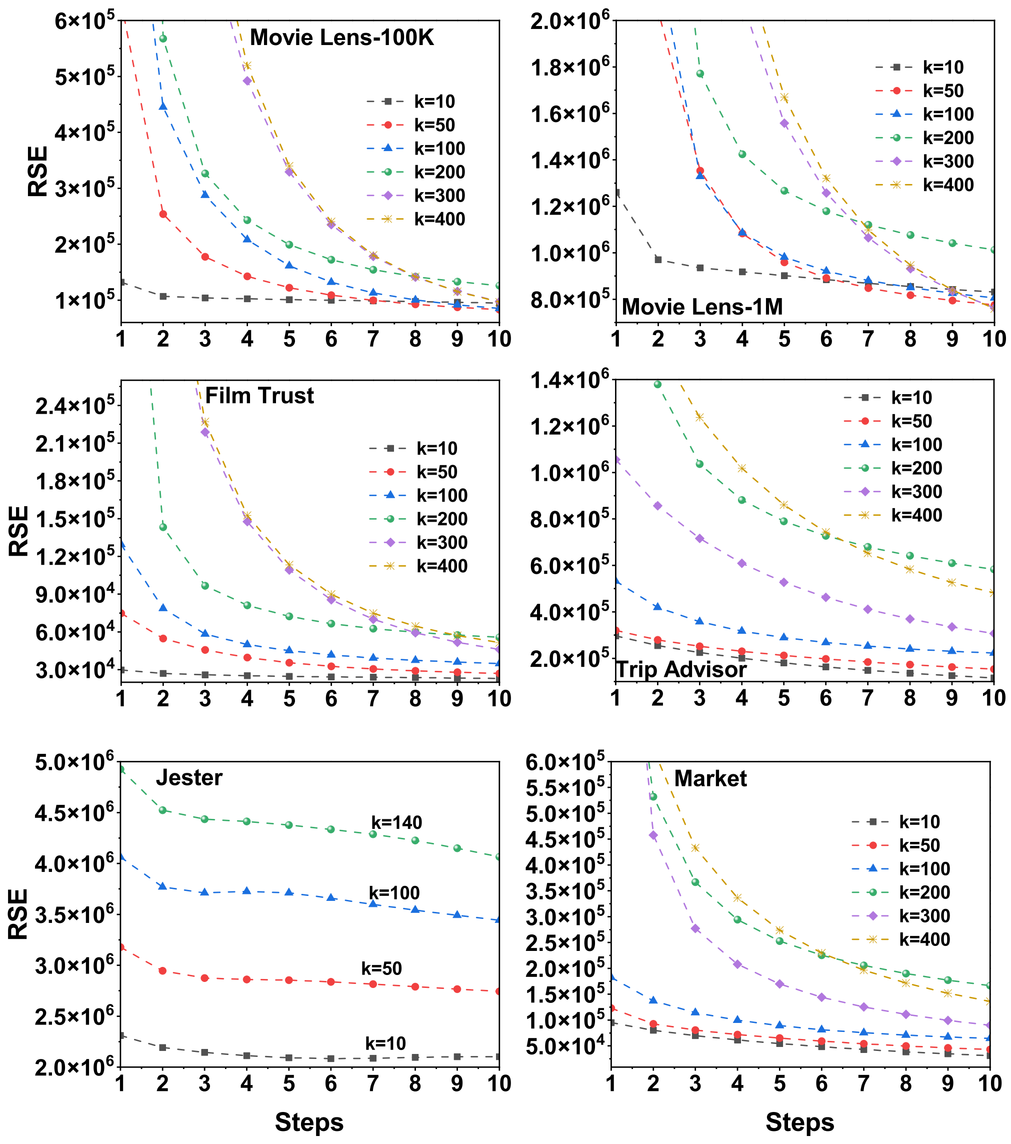


Fig. 14. Regularized Square Error (RSE) graphs for basic Matrix Factorization (MF) on six datasets, namely, Movie Lens-100K, Movie Lens-1M, Film Trust, Trip Advisor, Jester, and Market datasets for 10 steps and 400 latent features.

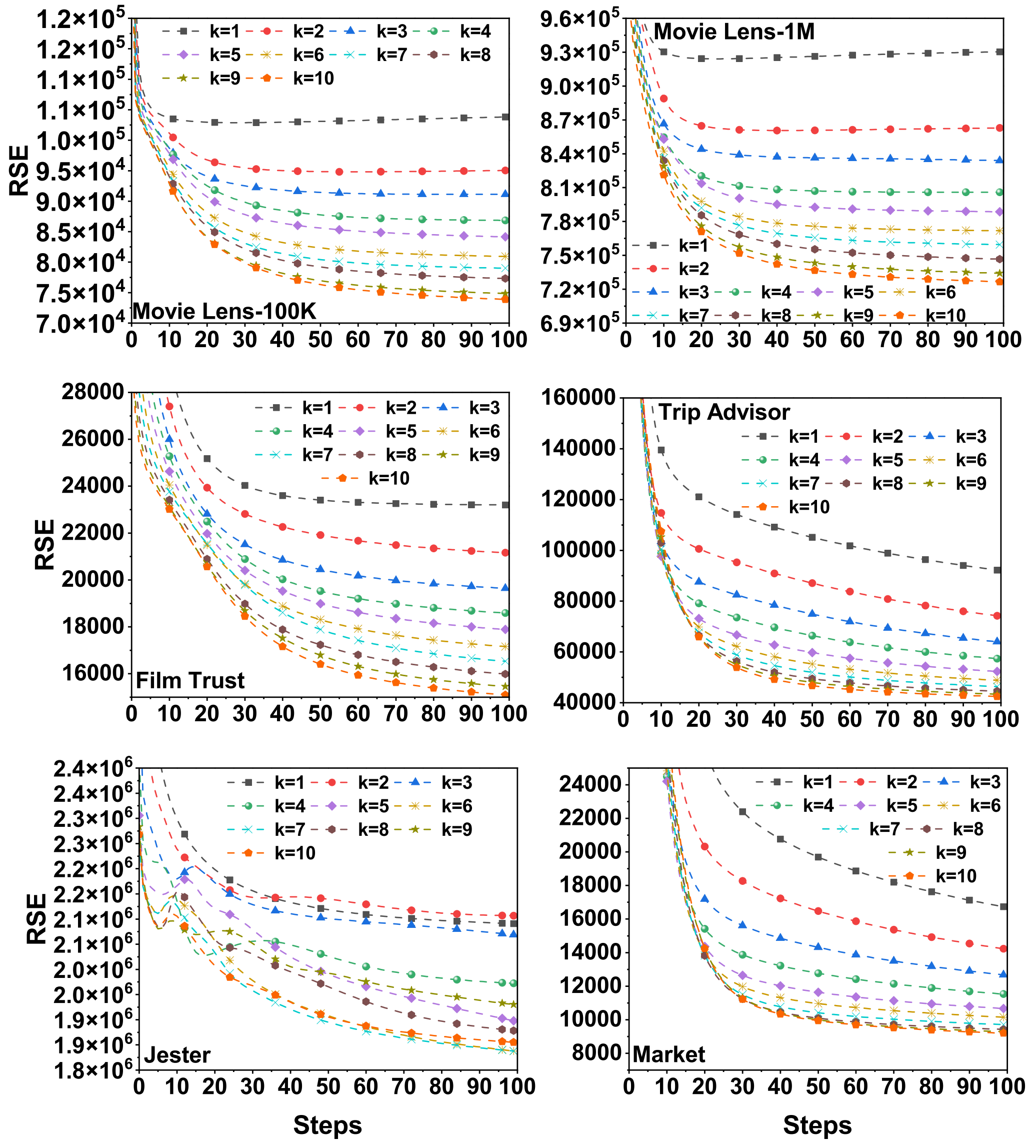


Fig. 15. Regularized Square Error (RSE) graphs for basic Matrix Factorization (MF) on six datasets, namely, Movie Lens-100K, Movie Lens-1M, Film Trust, Trip Advisor, Jester, and Market datasets for 100 steps and 10 latent features.

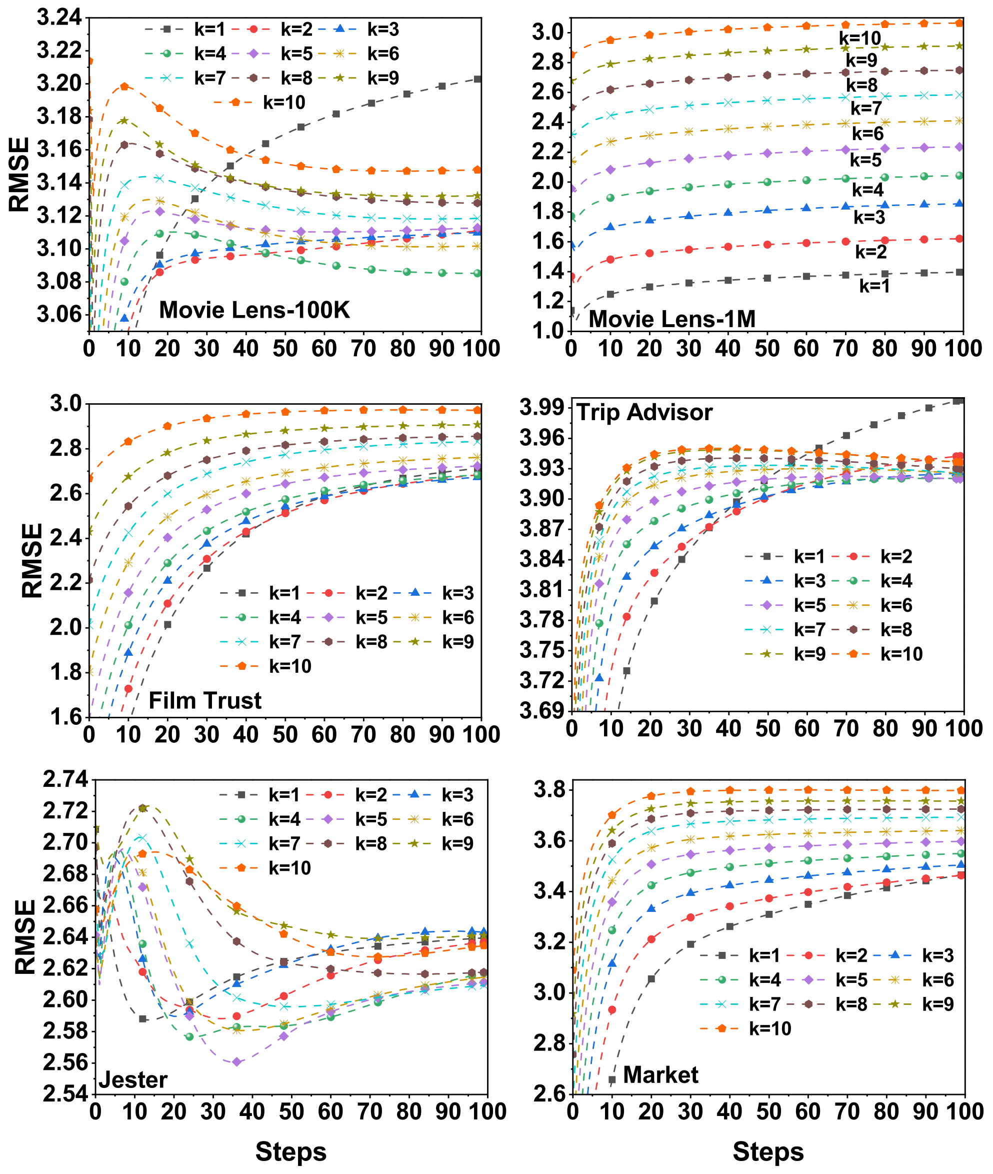


Fig. 16. Root Mean Square Error (RMSE) graphs for basic Matrix Factorization (MF) on six datasets, namely, Movie Lens-100K, Movie Lens-1M, Film Trust, Trip Advisor, Jester, and Market datasets for 100 steps and 10 latent features.

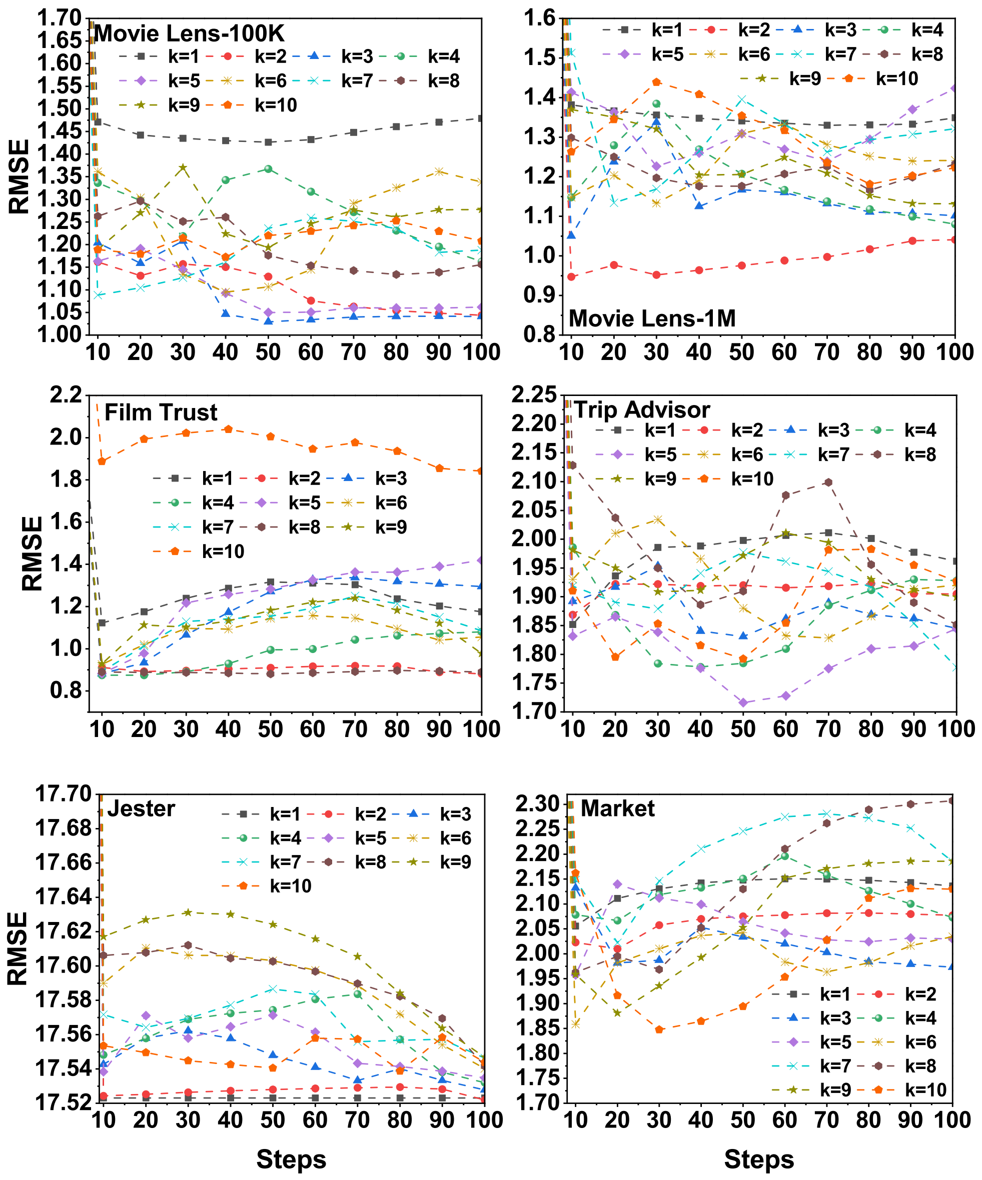


Fig. 17. Root Mean Square Error (RMSE) graphs for probabilistic Matrix Factorization (PMF) on six datasets, namely, Movie Lens-100K, Movie Lens-1M, Film Trust, Trip Advisor, Jester, and Market datasets for 100 steps and 10 latent features.

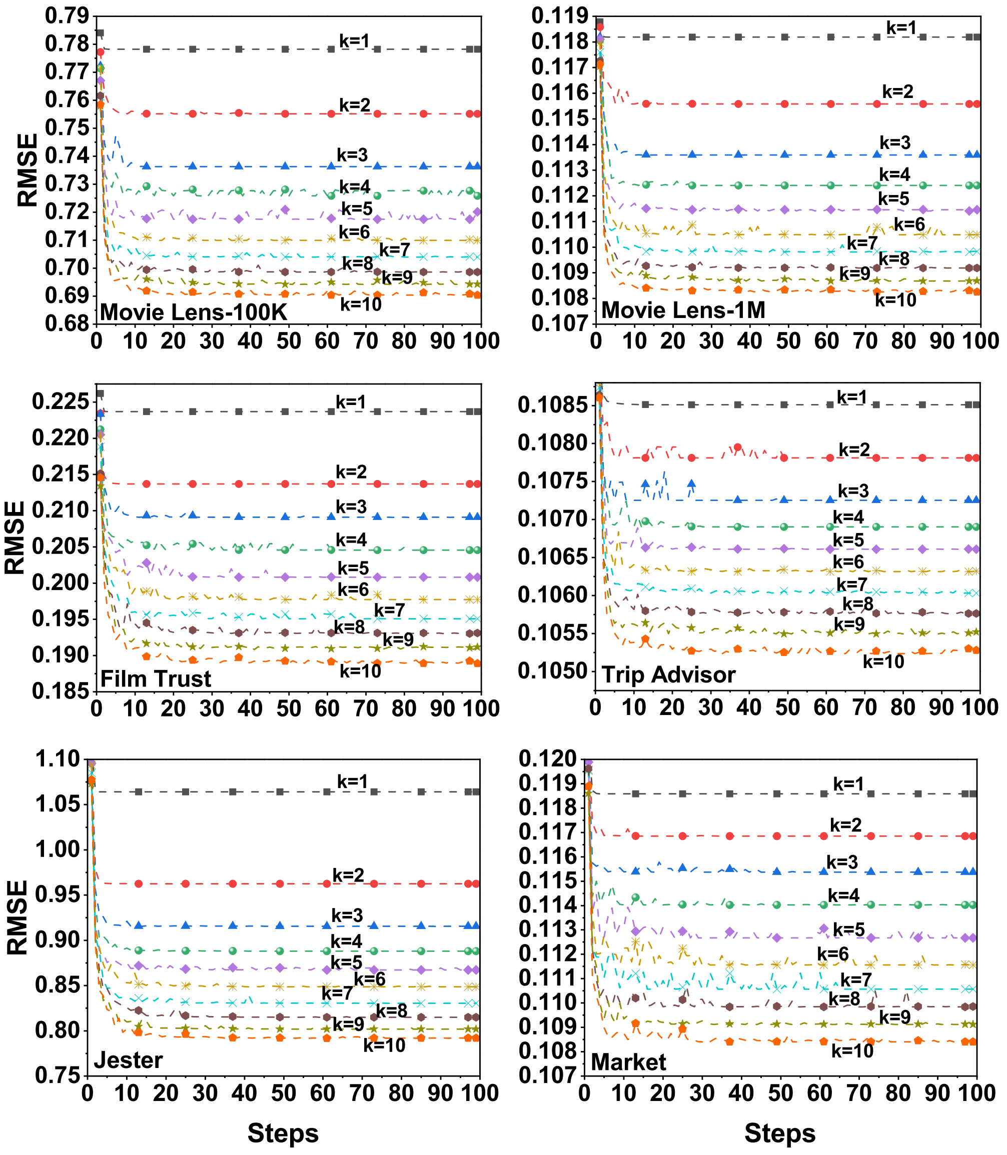


Fig. 18. Root Mean Square Error (RMSE) graphs for Non-Negative Matrix Factorization (NMF) on six datasets, namely, Movie Lens-100K, Movie Lens-1M, Film Trust, Trip Advisor, Jester, and Market datasets for 100 steps and 10 latent features.

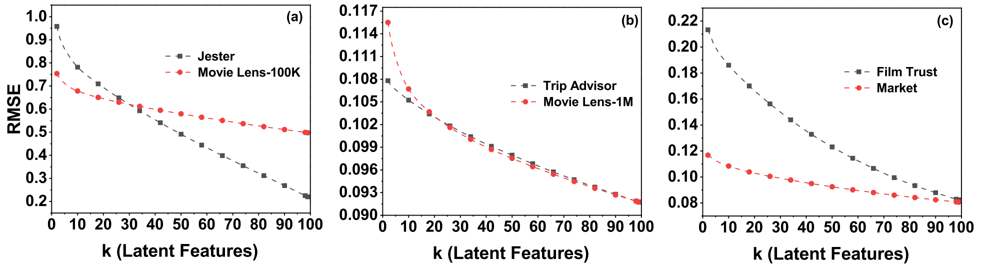


Fig. 19. Root Mean Square Error (RMSE) graphs for Singular Value Decomposition (SVD) on six datasets, namely, Movie Lens-100K, Movie Lens-1M, Film Trust, Trip Advisor, Jester, and Market datasets for 100 steps and 100 latent features.

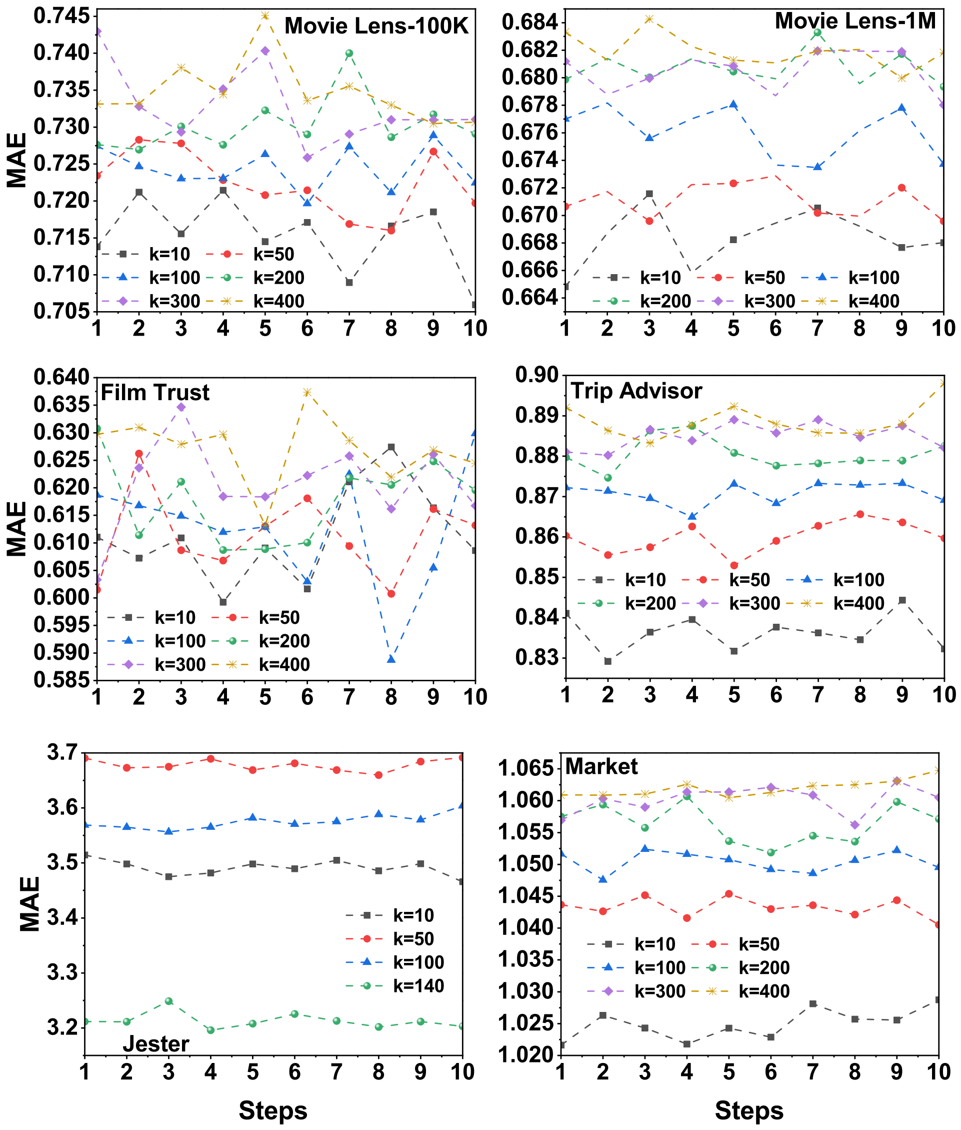


Fig. 20. Mean Absolute Error (MAE) graphs for SVD++ on six datasets, namely, Movie Lens-100K, Movie Lens-1M, Film Trust, Trip Advisor, Jester, and Market datasets at 10 steps and 400 latent features.