KYBERNETIKA — VOLUME *58* (2022), NUMBER 3, PAGES 440–455

A NEW CURVE FITTING BASED RATING PREDICTION ALGORITHM FOR RECOMMENDER SYSTEMS

Yilmaz Ar, Şahin Emrah Amrahov, Nizami A. Gasilov
and Sevgi Yigit-Sert

The most algorithms for Recommender Systems (RSs) are based on a Collaborative Filtering (CF) approach, in particular on the Probabilistic Matrix Factorization (PMF) method. It is known that the PMF method is quite successful for the rating prediction. In this study, we consider the problem of rating prediction in RSs. We propose a new algorithm which is also in the CF framework; however, it is completely different from the PMF-based algorithms. There are studies in the literature that can increase the accuracy of rating prediction by using additional information. However, we seek the answer to the question that if the input data does not contain additional information, how we can increase the accuracy of rating prediction. In the proposed algorithm, we construct a curve (a low-degree polynomial) for each user using the sparse input data and by this curve, we predict the unknown ratings of items. The proposed algorithm is easy to implement. The main advantage of the algorithm is that the running time is polynomial, namely it is $\theta(n^2)$, for sparse matrices. Moreover, in the experiments we get slightly more accurate results compared to the known rating prediction algorithms.

Keywords: recommender systems, collaborative filtering, curve fitting

Classification: 68Q25, 68T01, 65D10

1. INTRODUCTION

Based on the exact but partial information, predicting unknown values may be encountered in many real-life problems and engineering applications. To tackle this problem, Collaborative Filtering (CF) approach which is widely used in Recommender Systems (RSs) applications is offered in the literature. This approach is based on the assumption that users' future behaviours are connected with their past behaviours. Thus, users' preferences that are not stated can be estimated by their past evaluations. For example, suppose that a user is searching a movie to watch, but he/she has not yet made a choice. If this user trusts an RS, he/she will follow its recommendation. Therefore, the aim is to create a reliable RS based on the data already obtained. From the movie perspective, the input data and the problem can be described as follows: some of m users watched some of n movies, and gave scores them ranging from 1 to 5. Hereafter 0 means that no score is given. Our goal is to predict the score that a user would give to a movie

DOI: 10.14736/kyb-2022-3-0440

after watching it, provided that he/she did not watch it before. This problem can be mathematically formulated as follows: $m \times n$ matrix A is given partially: some of its entries are known while others are unknown. Our goal is to estimate unknown entries of the matrix by using the known values. The dimensions of the matrix A in the real world applications would be large enough, because there are lot of users and lot of movies. However, the total number of users who watch movies and give scores after watching movies is not so much. Therefore the rating matrix is sparse: a relatively few number of entries are known.

Many valuable studies related to RSs have been performed up to now, but the subject has not lost its importance. Furthermore, the researches on RSs have been increasing significantly in recent years. Early studies on RSs were performed towards the end of the 1990's. The first study was done by [32]. In this study, the authors first provided the scientific concept of "recommender systems" and created RSs for a particular case. The most widely used approach for RSs is CF which was developed by [31]. Various research projects have produced different approaches for RSs. These approaches are Content-Based Recommendation Systems (CBRS), Collaborative Filtering Recommendation Systems (CFRS), Trust Based Recommendation Systems (TBRS), Hybrid Recommendation Systems (HRS) and Social Networks Based Recommendation Systems (SNBRS). In fact, the main idea of CF: "the past behaviors of the users determines their future behaviors" exists in all these systems [22]. In the CBRS, newly introduced products are generally not recommended to users. [33] developed an item-based collaborative filtering method and this method generalized by [6] for a group of users. These methods in which the similarity indices between the users were utilized [3] are types of CFRS. There are many studies on TBRS. Some of them used given trust information between users [17], the others performed the trust computation using rating values [38]. HRSs were created by combining RSs with different techniques such as fuzzy logic or merging different types of RSs, to overcome the limitation of particular RSs such as CBRS and TBRS [11, 16]. In recent years, with the wide use of social networking platforms, SNBRSs have been created and in these systems the needed data on users preferences are mined from social networks [26]. Meo et al. [12] proposed combining CF-based matrix factorization and social friendship information to provide more accurate recommendations. As we mentioned above, generally the CF approach is utilized to create RSs. The most used CF approaches are neighbor-based CF and Probabilistic Matrix Factorization (PMF). The most typical examples of neighbor-based CFs are user-based and item-based CF approaches. They utilize the similarity indices between the users and the items, respectively [33].

Hoffman developed PMF that is based on the computing user preferences by involving hidden factors [19]. Although this method gives acceptable results, it has some shortcomings:

- It is very sensitive to small changes in the input data;
- It cannot produce reliable predictions for new users and new products;
- It does not perform well on very sparse datasets.

Several studies have been performed to overcome some of these shortcomings [4, 23, 27]. Since there are uncertainties in the formulation of the problem, some researchers suggest to use fuzzy logic in building RSs [10].

Besides the above-mentioned methods; the graph-based clustering algorithm [13], CF method based on artificial neural network [1], a higher-order sparse linear method [9] were used for RSs (see [40] for an overview). As a new stage in this direction, new RSs can be developed by the in-depth study of properties of the products. Several practical examples illustrating this trend can be found in [24].

[14] proposed a hybrid approach that uses both the Radial Basis Function Network (RBFN) and CF methods. [15] reviewed different aspects of large-scale social RSs. They summarized the difficulties encountered in such RSs and discuss ways to overcome some of them. [20], based on multi-party random masking and polynomial aggregation, proposed confidentiality preserving CF scheme on arbitrary distributed data of healthcare services. [21] constructed a new genetic-based CFRS. Their RS combines both the neighborhood models and the latent factor models. The authors only dealt with active users and their corresponding items. [29] built hybrid RSs based on CF and ontology. It is known that all RSs have two main drawbacks: sparsity and scalability. To avoid these drawbacks the authors used dimensionality reduction and ontology methods. [8] also employed dimensionality reduction coupled with normalization to obtain denser data. They introduced a CFRS algorithm that constructs a dynamic evolutionary clustering model and make predictions based on correlation between users. [36] proposed hybrid knowledge-based RS to learners for choosing an e-learning resource. Their RS is based on sequential pattern mining and ontology. [30] constructed an emotion-aware RS, which is based on the fusion of hybrid information. They used three types of information (users' rating data, users' social network data and sentiment data from user reviews) to analyze the users' features. It is clear that user rating data and user social network data are explicit information, however sentiment data is emotional information. [39] considered a PMF model for multiple social networks. The authors used joint recommendation framework which is based on joint probability distribution. They applied the model to different kinds of social networks with the various distribution functions of the similarity of user preferences. [2] proposed a method which predicts the success of a movie in terms of ratings and temporal popularity. [35] proposed a new measure to calculate user–user similarities and used these similarities as a weight in item-based CF. [41] provided a method using the advantage of involving the item title, which contains important information about the product and plays an important role in attracting users' attention. [18] developed a rating prediction system based on reviews.

As we explained above, the approaches, proposed so far to solve the rating prediction problem, are based on either the similar users, or the spectral characteristics of the input matrix. In most approaches that utilize similarity between users, in the process of making a decision, for the user under consideration, k users are determined that are most similar to him. These users are interpreted as neighbors in some algorithms. The choice of the value of k is a critical issue. If it is chosen large, then the accuracy of the approach may become low: Involving unrelated neighbors preferences may decrease the accuracy of the prediction. On the other hand, if the value of k is chosen to be smaller than it should be, the method may miss some useful information. Therefore,

deciding on the number of nearest neighbors is not an easy task. To be able to determine the nearest neighbors we need to compute similarity values between all users. This computation requires a significant amount of time. The other approaches based on the spectral characteristics of the input matrix may produce predictions regardless of the order of users and products. In the CF approaches and PMF techniques with which we compare our proposed method, the only information available is the user ratings for placed products. We also use only rating information in our proposed method. In this paper, we create a function with the help of the known entries of the input matrix, and we transform the problem to the classical curve fitting problem. We test our proposed algorithm on MovieLens100K, MovieLens1M and LastFM datasets. We prove that the complexity of the proposed algorithm is $\theta(n^3)$ in worst case and it is $\theta(n^2)$ for a sparse input matrix.

2. BACKGROUND

2.1. CF Techniques

RSs can basically be classified into two main classes: collaborative filtering RSs (CFRS) and content-based RSs (CBRS). CFRSs assume that there is a strong connection between the past preferences and the future choices of a user. CF methods can be divided in two main subgroups. Memory-based methods are also known as k-neighbor methods and are based on similarity values between users or items. Many different approaches to find the similarity were proposed. In order to evaluate similarity weights, some of them use only the user-item rating matrix, others may use an additional information, such as trust. In this study, we will only use user-item rating matrix and utilize the following similarity metrics: Pearson's Correlation Coefficient (PCC), Extended Jaccard Coefficient (EJC), and Vector Cosine Similarity (VCS). These similarity metrics are used to measure the proximity of rating vectors of two users, e. g. user a and user b, and to find k close neighbors of user a.

PCC is a measure of the degree of linear relationship between rating vectors of two users (a and b). It is computed as:

$$PCC(a,b) = \frac{\sum_{i \in I_{ab}} (r_{a,i} - \bar{r}_a)(r_{b,i} - \bar{r}_b)}{\sqrt{\sum_{i \in I_{ab}} (r_{a,i} - \bar{r}_a)^2} \cdot \sqrt{\sum_{i \in I_{ab}} (r_{b,i} - \bar{r}_b)^2}} \tag{1}$$

where $\bar{r}_a$ and $\bar{r}_b$ are the mean of ratings that are given by the users a and b, respectively. I_{ab} is the set of items that are rated by both users.

EJC computes the degree of overlap between the ratings given on commonly rated items by two users. EJC is defined as:

$$EJC(a,b) = \frac{\sum_{i \in I_{ab}} (r_{a,i} r_{b,i})}{\sum_{i \in I_{ab}} (r_{a,i})^2 + \sum_{i \in I_{ab}} (r_{b,i})^2 - \sum_{i \in I_{ab}} (r_{a,i} r_{b,i})} \tag{2}$$

VCS measures the cosine of angle between rating vectors of two users:

$$VCS(a,b) = \frac{\sum_{i \in I_{ab}} (r_{a,i} r_{b,i})}{\sqrt{\sum_{i \in I_{ab}} (r_{a,i})^2} \cdot \sqrt{\sum_{i \in I_{ab}} (r_{b,i})^2}} \tag{3}$$

Along with the above similarity metrics, we also consider a special case, namely, Random Similarity (RND). For two users a and b, RND assigns a random value in the range from 0.0 to 1.0 as the similarity value between them.

Each of those four metrics actually computes $w_{a,b}$ that is the similarity weight between user a and user b. Since $r_{a,i} \geq 0$ and $r_{b,i} \geq 0$, the range of this weight value is 0.0 - 1.0 (For PCC, the range is [-1, 1]). High weights mean that user a and user b are quite similar with respect to item preferences. The computed similarity weights between users can be used to generate predictions. CF's main goal is to predict user a's preference on item i (i. e. the value $p_{a,i}$) based on the preferences of his neighbors. The k most similar neighbors are used in the prediction procedure (where k is a specified parameter). The following formula will be used to compute $p_{a,i}$.

$$p_{a,i} = \frac{\sum_{b \in U_a} (w_{a,b} r_{b,i})}{\sum_{b \in U_a} w_{a,b}} \tag{4}$$

where U_a is the set of k users most similar to user a. $r_{b,i}$ represents the rating of user b on item i. $w_{a,b}$ shows the similarity weight between user a and user b. There are other memory-based approaches that aim to improve the prediction (4). Some of them use different similarity metrics that includes additional information like trust ratings or review ratings. The step by step algorithm for a user-to-user CF is as follows:

- Step 0: Specify the number of neighbors, k. Initialize variables.
- Step 1: In order to compute similarity values between each pair of users (say, a and b), for example in the frame of PCC, we have to do the following substeps:
  - Find the set I_{ab} of commonly rated items
  - Compute user's average ratings $\bar{r}_a$ and $\bar{r}_b$
  - Compute the similarity value $w_{a,b}$ by formula (1)
- Step 2: For each user a, find the set U_a of his most nearest k neighbors with respect to similarity value.
- Step 3: Compute the predicted rating $p_{a,i}$ of the user a on a given item i by formula 4 .

As stated above, in this study, the $m \times n$ user-item rating matrix A is assumed to be very sparse. In this matrix, each row vector $\mathbf{u}_a$ represents user a's ratings on n items, each column vector $\mathbf{v}_i$ shows m users' ratings on item i. The values of many of the components of these vectors are missing. Due to this circumstance, rating spaces (for users and for items) can be reduced in dimensions. For this purpose, each of the sets of users and items can be divided into k subgroups consisting of members highly similar to each other [25]. As a result, the rating spaces for both users and items can be reduced to a smaller dimension k. Therefore, each user and each item can be represented by k-dimensional vectors. There are several dimensionality reduction algorithms such as Singular Value Decomposition (SVD), Principal Component Analysis (PCA) and Matrix Factorization (MF). They are considered as possible solutions for the prediction of user preferences.

MF is a model-based CF approach and basically represents users and items by latent vectors inferred from user-item rating matrix. MF models map users and items into a latent factor space with a specified dimension (k). Each user in the dataset has its own latent factor vector with k elements and each item has a latent vector with k factors. The inner product of those vectors actually produces the predicted preference $p_{a,i}$ of given user a on a given item i [7]:

$$p_{a,i} = (\mathbf{u}_a)^T \mathbf{v}_i \tag{5}$$

where $\mathbf{u}_a$ is user a's latent-factor vector and $\mathbf{v}_i$ is item i's latent-factor vector. To learn those vectors, MF aims to minimize the regularized square error on the set of ratings of training dataset. There are several matrix factorization techniques that aim to improve the accuracy of predictions compared to the base method. In recent years, some studies have been carried out to improve the method of non-negative matrix factorization. The study of [37] is one of the best examples of these kinds of studies. In the study, the authors proposed the distributed non-negative matrix factorization algorithm instead of the classical non-negative matrix factorization algorithm and obtained better results in terms of both processing time and accuracy.

SVD factorizes the rating matrix R into three matrices as follows:

$$R = U\Sigma V^T \tag{6}$$

where U is an $m \times m$ orthonormal matrix, Σ is an $m \times n$ rectangular diagonal matrix whose diagonal entries are non-negative real numbers and V is an $n \times n$ orthonormal matrix. The diagonal entries σ_i of Σ are called the singular values of R.

SVD is well-defined when the matrix is complete. In our problem, ratings matrix is very sparse. To use SVD, we need to provide default values for missing ratings. Average rating of a user (or average rating for an item) may be used as a default value and that approach is named as imputation. Another solution is to compute SVD using normalized ratings matrix; in that method missing values are considered as 0. Alternatively, several approaches have been proposed to estimate SVD using only known ratings like least squares or gradient descent.

2.2. Accuracy metrics for RSs

The accuracy of the RSs is measured by several metrics. The most used metrics are Root Mean Square Error (RMSE) and Mean Absolute Error (MAE). The formulas of these metrics are as follows:

$$RMSE = \sqrt{\frac{1}{|S|} \sum_{(a,i)\in S} (\hat{r}_{a,i} - r_{a,i})^2} \tag{7}$$

$$MAE = \frac{1}{|S|} \sum_{(a,i)\in S} |\hat{r}_{a,i} - r_{a,i}| \tag{8}$$

where $S \subset \{(a,i)|r_{a,i} > 0\}$ is the test dataset, $|S|$ denotes the number of elements (ratings) in this set. $r_{a,i}$ represents original rating value that user a gave to item i, $\hat{r}_{a,i}$ represents predicted rating value that is produced by RSs.

RSs aim to minimize the values of RMSE and MAE. Based on the results of our observation, we concluded that improvement of existing algorithms cannot reduce these values significantly. We believe that in order to solve the problem, a completely new algorithm should be developed. In this paper, we propose such a new algorithm.

3. RAPAL: RATING PREDICTION ALGORITHM

In this study, we convert the recommendation problem to a curve fitting problem. The details can be found in Algorithm 1. We name our proposed algorithm as RAPAL. This name is formed by using the first one or two letters of the expression "Rating Prediction Algorithm".

We can briefly describe the algorithm as follows. We pass rows of the rating matrix one by one. Suppose the current passed row is the ith row. If there is no empty cell (zero element) in this row, then we do not perform any actions and continue with the next row. Otherwise, we pass the elements of the ith row one by one. For each non-zero element, say $y = A[i,j]$, we determine the average x of non-zero elements in the jth column, except $y = A[i,j]$ itself, and create a pair (x,y). The meaning of this pair for the considered problem is as follows: while the average score given by other users for the jth item is x, the score of the ith user for this item is y. In this manner, we generate all (x,y)-pairs for the current ith row. Let them be (x_j, y_j), $j = 1, 2, \ldots, l$. Thereafter, we determine a function $y = f(x)$, which fits all these (x_j, y_j)-pairs. (The function $y = f(x)$ expresses how critical or tolerant the current jth user is when compared to the "average" user). We use 1st, 2nd and 3rd order polynomials as the $y = f(x)$ function in our algorithm. Thus, $y = f(x) = c_0 + c_1x + c_2x^2 + \ldots + c_px^p$ (where $p = 1$, 2, or 3), and the task is to find the coefficients $c_0, c_1, c_2, \ldots, c_p$. We use the least squares method to determine the values of coefficients.

$$\text{Let } C = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_p \end{bmatrix}, Y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_l \end{bmatrix} \text{ and } W = \begin{pmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^p \\ 1 & x_2 & x_2^2 & \cdots & x_2^p \\ 1 & x_3 & x_3^2 & \cdots & x_3^p \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_l & x_l^2 & \cdots & x_l^p \end{pmatrix}.$$

Then, the coefficients are found as $C = (W^TW)^{-1}W^TY$, where W^T denotes the transpose of the matrix W, and $(W^TW)^{-1}$ is the inverse matrix of W^TW.

Finally, for each empty cell in the current ith row, the average value of non-zero elements in the column of this empty cell is computed (aC, in the algorithm). If the number l of pairs produced for the given row is greater than the degree p of fitting polynomial and the number kC of non-zero elements in the considered column is greater than 0, then the empty cell is filled with the value $round(f(aC))$. If the above condition is not satisfied, then the empty cell is filled using the average rating given by the current user and the average rating for the considered item.

Let us compute the time complexity of the proposed algorithm RAPAL. The processing time of the lines from 1 to 15 is $\theta(mn)$. However, the processing time of the rest of the algorithm (lines 16-63) depends on the curve fitting function (line 37). We use polynomial fitting and apply the least squares method to determine the coefficients of the approximating polynomial. It is well known that the time complexity of pth order polynomial fitting algorithm for l pairs is $\theta(pl^2)$. In this study, p is chosen as 1, 2 and 3. Therefore, $\theta(pl^2) = \theta(l^2)$. Hence, the processing time of lines from 16 to 63 is $\theta(m)\theta(n+l^2+n)$. In the end, the complexity of RAPAL is $\theta(mn) + \theta(m)\theta(n + l^2 + n) = \theta(m(n + l^2))$. If $m \sim n$ (i. e., if m and n are of the same order), then the time complexity is $\theta(n^2 + nl^2)$. From the point of view of complexity, the worst case occurs when A is dense, in which case the complexity of the algorithm is $\theta(n^3)$, since $l = \theta(n)$. However, the user-item matrix, A, is usually sparse for RSs, it yields that l is very much smaller than m and n. For this reason, the complexity of RAPAL is $\theta(n^2)$ for RSs.

Algorithm 1: Rating Prediction Algorithm based on Curve Fitting

Input: $m \times n$ matrix A, some (in fact, most) elements of which are zero, other elements are positive numbers in the range from 1 to 5. The element $A[i, j]$ represents the rating given by user i to item j.
p is the degree of fitting polynomial.
Output: $m \times n$ matrix A, whose values are in the range [1, 5].
procedure RAPAL(A, m, n, p)

```
1  for j ← 1 to n do
2      C[j] ←0
                                   /* C[j] is the sum of elements in j^th column of A */
3      kC[j] ←0
                                /* kC[j] is the number of non-zero elements in j^th column */
4      for i ← 1 to m do
5          if A[i, j] > 0 then
6              C[j] ← C[j] + A[i, j]
7              kC[j] ← kC[j] + 1
8          end
9      end
10     if kC[j] > 0 then
11         aC[j] ← C[j]/kC[j]                    /* the average of ratings for item j */
12     else
13         aC[j] ←0
14     end
15 end
```

```
for i ← 1 to m do
    R ←0                                   /* R is the sum of elements in i^th row of A */
    kR ←0                          /* kR is the number of non-zero elements in i^th row */
    l ←0          /* l represents the count of pairs that are generated for i^th row */
    for j ← 1 to n do
        if A[i, j] > 0 then
            R ← R + A[i, j]
            kR ← kR + 1
            if kC[j] > 1 then
                l ← l + 1
                Y[l] ← A[i, j]
                X[l] ← (C[j] − Y[l])/(kC[j] − 1)      /* the average of ratings for item j
                 given by all users except user i */
            end
        end
    end
    if kR > 0 then
        aR ← R/kR                          /* the average of ratings given by user i */
    else
        aR ←0;
    end
    if l > p then
        call ConstructFittingFunction(X, Y, l, p, f)              /* constructing a fitting
        polynomial f of pth order */
    end
    for j ← 1 to n do
        if A[i, j] = 0 then
            if l > p and kC[j] > 0 then
                A[i, j] ← f(aC[j])              /* predicting by the fitting polynomial */
            else
                /* the case that there is no enough pairs for fitting or there is no
                   rating for item j                                                  */
                if kR > 0 and kC[j] > 0 then
                    A[i, j] ← (aR + aC[j])/2       /* the mean of averages is assigned */
                else
                    if kR = 0 and kC[j] = 0 then
                        A[i, j] ← 3
                    else
                        A[i, j] ← aR + aC[j]
                    end
                end
            end
        end
        A[i, j] ← round(A[i, j])
        if A[i, j] < 1 then
            A[i, j] ←1
        end
        if A[i, j] > 5 then
            A[i, j] ←5
        end
    end
end
end procedure
```

Method	**MAE**
RND Original	0.8306
RND Evolved (GA)	0.7938
RND Evolved (ABC)	0.8039
PCC Original	0.8174
PCC Evolved (GA)	0.7815
PCC Evolved (ABC)	0.7846
VCS Original	0.8174
VCS Evolved (GA)	0.7649
VCS Evolved (ABC)	0.7597
EJC Original	0.8038
EJC Evolved (GA)	0.7779
EJC Evolved (ABC)	0.7886
RAPAL - Cubic	0.7082 (%6.8)
RAPAL - Quadratic	0.7026 (%7.5)
RAPAL - Linear	**0.6995 (%7.9)**

Tab. 1. MAE results for different methods. Bold value indicates the smallest MAE. The values by [5] and [34] are averaged on different numbers of neighbors (k). In parentheses is the percentage change compared to the best score by [34], namely, compared to the method VCS Evolved (ABC).

4. DATASET

In order to evaluate RAPAL, numerical experiments are performed on three datasets. MovieLens100K contains 943 users that rate on 1682 movies, and provides 100,000 ratings in total. In this dataset, each user rated at least 20 movies. MovieLens1M, as the name implies, includes 1 million ratings that are given by 6040 users on 3900 movies. Finally, LastFM that contains 1892 users' musical tastes is used. The number of musical items in this dataset is 17632 and the total number of ratings is nearly 92K. These three datasets are benchmarks and often used to evaluate the performance of RSs algorithms. To examine an RS method, we first split the selected dataset into 10 equal parts. Each time using 8 randomly selected parts (80%) as the training set and the remaining 2 parts (20%) as the test set, we run the method 5 times. The results reported in Section 5 are the average of these 5 different 80/20 splits.

5. EXPERIMENTAL RESULTS

Table 1 represents results of RAPAL and the study by [5] and [34] (first 12 rows in the table), obtained over MovieLens100K dataset. [5] predicts ratings by CF using genetic algorithm with various similarity metrics such as Pearsons Correlation Coefficient (PCC), Extended Jaccard Coefficient (EJC), Vector Cosine Similarity (VCS) and Random Similarity (RND). [34] extended this study including artificial bee colony (ABC) algorithm to predict ratings by CF.

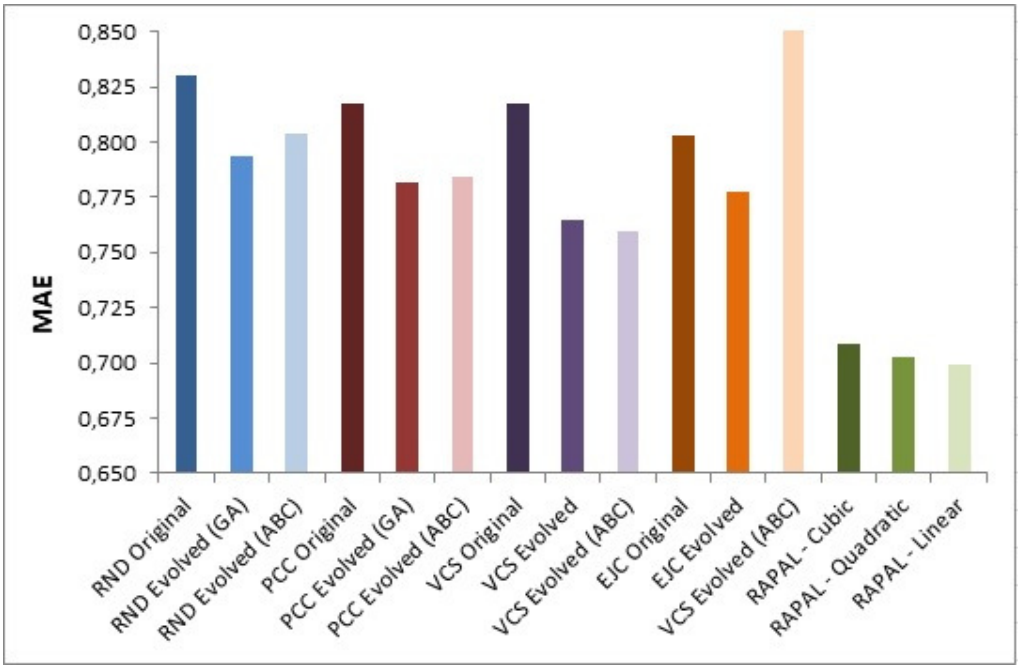

Fig. 1. Comparison of the performances of different methods in terms of MAE.

Hereafter, the results obtained by applying CF which uses a similarity metric are indicated as "Original" (after the name of the metric). In the same way, the results of [5] and [34], which are derived via CF trained by genetic algorithm, are labelled as "Evolved (GA)" and trained by artificial bee colony, are labelled as "Evolved (ABC)" with the corresponding metric. In RAPAL, for curve fitting, we use linear, quadratic and cubic polynomials. Therefore the results obtained by the proposed algorithm are named as "RAPAL - Linear", "RAPAL - Quadratic" and "RAPAL - Cubic". Table 1 shows that RAPAL yields better accuracy than all baseline methods which were experimented by [5] and [34]. Among curve fitting polynomials, RAPAL - Linear provides the best MAE. Figure 1 also clearly shows the improvement in MAE achieved by RAPAL.

Note that, since the rating values are between 1 and 5, the possible maximum value for MAE is $5 - 1 = 4$. (The minimum value is 0). Therefore, the MAE value of 0.6995 corresponds to an error of 17%.

Table 2 shows the RAPAL's performance on three benchmark datasets (MovieLens100K, MovieLens1M and LastFM) with respect to MAE and RMSE. In all three datasets, RAPAL with linear curve fitting produce slightly better results than quadratic and cubic curve fittings in both MAE and RMSE. The MAE and RMSE results show that our proposed method yield better or comparable predictions on benchmark datasets.

Dataset	RAPAL - Linear		RAPAL - Quadratic		RAPAL - Cubic		PMF	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
ML100K	0.6995	0.9798	0.7026	0.9859	0.7082	0.9954	0.7505	0.9538
	(6.8%)	(-2.7%)	(6.4%)	(-3.4%)	(5.6%)	(-4.4%)		
ML1M	0.6657	0.9419	0.6670	0.9451	0.6701	0.9507	0.6833	0.8695
	(2.6%)	(-8.3%)	(2.4%)	(-8.7%)	(1.9%)	(-9.3%)		
LastFM	0.2544	0.5184	0.2655	0.5382	0.2822	0.5711	0.3370	0.4668
	(24.5%)	(-11.1%)	(21.2%)	(-15.3%)	(16.3%)	(-22.3%)		

Tab. 2. MAE and RMSE values of RAPAL with linear, quadratic and cubic polynomials. In parentheses is the percentage change compared to the PMF method.

For all methods, presented in the literature, MAE and RMSE values on LastFM are lower in comparison to other datasets. The reason of this situation is the distribution of ratings in the data. In the LastFM dataset, the ratings are gathered around 3. In other words, the standard deviation in this dataset is small.

Table 2 also displays PMF algorithm's MAE and RMSE results. If we compare the two algorithms in terms of MAE, we can see that for the considered datasets, RAPAL provides improvement over PMF with the scores 0.6995, 0.6657, 0.2544 versus 0.7505, 0.6833, 0.3370, respectively. On all three datasets, as a whole, RAPAL's MAE and RMSE results are better in comparison with PMF. Namely, MAE values are essentially better, but RMSE values are a little worse than PMF. We can explain this case by the fact that RMSE is more susceptible to outliers in addition to bad predictions [28].

Unlike other algorithms, in RAPAL, we applied round operation to the computed predictions (See Line 55 of the pseudocode). We consider this operation as a part of our algorithm and believe that it is even necessary for the purity of the experiment, since in a real experiment the ratings can only be integers. The MAE and RMSE results of our experiments given in Table 2 were produced using round operation.

6. CONCLUSION

In this study, we proposed a new algorithm, RAPAL, for the rating prediction for RSs. The algorithm is based on the curve fitting method. As a curve we used linear, quadratic and cubic polynomials. Our algorithm is easy to implement and quicker than PMF. Numerical experiments, performed on three datasets (MovieLens100K, MovieLens1M, LastFM) showed the effectiveness of our algorithm with relative improvement of 6.8%, 2.6%, 24.5%, respectively, compared to PMF in MAE metric. For all three polynomials (linear, quadratic, cubic), RAPAL produces better results than existing methods, but the best results were obtained in the linear case. We proved that the complexity of RAPAL is $\theta(n^2)$ for a sparse input matrix, which takes place for the rating prediction problem in RSs. In the future, one can experiment with various modifications of RAPAL on different datasets. For example, first a curve can be fit for the most active user, and

then the estimates obtained for him can be involved when determining rating values from other users. As future work, we plan to apply our algorithm to much larger datasets in order to show the effectiveness of RAPAL.

ACKNOWLEDGEMENT

We would like to thank Editor in Chief Prof. Sergej Čelikovský and the anonymous reviewers for taking the time and effort necessary to review the study. We sincerely appreciate all valuable comments and suggestions, which improved the quality of our paper.

(Received June 12, 2022)

REFERENCES

[1] A. M. Acilar and A. Arslan: A collaborative filtering method based on artificial immune network. Expert Systems with Applications *36* (2009), 8324–8332. DOI:10.1016/j.eswa.2008.10.029

[2] B. Alhijawi and A. Awajan: Prediction of movie success using Twitter temporal mining. In: 6th International Congress on Information and Communication Technology, Singapore 2022, pp. 105–116. DOI:10.1007/978-981-16-2377-6_12

[3] M. Y. H. Al-Shamri: Power coefficient as a similarity measure for memory-based collaborative recommender systems. Expert Systems Appl. *41* (2014), 5680–5688. DOI:10.1016/j.eswa.2014.03.025

[4] Y. Ar: An initialization method for the latent vectors in probabilistic matrix factorization for sparse datasets Evolution. Intell. *13* (2020), 269–281. DOI:10.1007/s12065-019-00299-2

[5] Y. Ar: A genetic algorithm solution to the collaborative filtering problem. Expert Systems Appl. *41* (2016), 122–128. DOI:10.1016/j.eswa.2016.05.021

[6] J. Bobadilla, F. Ortega, A. Hernando, and J. Bernal: Generalization of recommender systems: Collaborative filtering extended to groups of users and restricted to groups of items. Expert Systems Appl. *39* (2012), 172–186. DOI:10.1016/j.eswa.2011.07.005

[7] D. Bokde, S. Girase, and D. Mukhopadhyay: Matrix factorization model in collaborative filtering algorithms: A survey. Procedia Computer Sci. *49* (2015), 136–146. DOI:10.1016/j.procs.2015.04.237

[8] J. Chen, C. Zhao, and L. Chen: Collaborative filtering recommendation algorithm based on user correlation and evolutionary clustering. Complex Intell. Systems *6* (2020), 147–156. DOI:10.1007/s40747-019-00123-5

[9] E. Christakopoulou and G. Karypis: HOSLIM: Higher-order sparse linear method for top-N recommender systems. In: Advances in Knowledge Discovery and Data Mining, Taiwan 2014, pp. 38–49. DOI:10.1007/978-3-319-06605-9_4

[10] C. Cornelis, J. Lu, X. Guo, and G. Zhang: One-and-only item recommendation with fuzzy logic techniques. Inform. Sci. *177* (2007), 4906–4921. DOI:10.1016/j.ins.2007.07.001

[11] C. De Maio, G. Fenza, M. Gaeta, V. Loia, F. Orciuoli, and S. Senatore: RSS-based e-learning recommendations exploiting fuzzy FCA for knowledge modeling. Applied Soft Computing *12* (2012), 1, 113–124. DOI:10.1016/j.asoc.2011.09.004

[12] P. De Meo, E. Ferrara, G. Fiumara, and A. Provetti: Improving recommendation quality by merging collaborative filtering and social relationships. In: 11th International Conference on Intelligent Systems Design and Applications 2011, pp. 587–592. DOI:10.1109/ISDA.2011.6121719

[13] G. N. Demir, A. S. Uyar, and S. G. Ögüdücü: Graph-based sequence clustering through multiobjective evolutionary algorithms for web recommender systems. In: 9th Annual Conference on Genetic and Evolutionary Computation (GECCO'07), London 2007, pp. 1943–1950. DOI:10.1145/1276958.1277346

[14] M. K. Devi and P. Venkatesh: Smoothing approach to alleviate the meager rating problem in collaborative recommender systems. Future Generation Computer Systems *29* (2013), 262–270. DOI:10.1016/j.future.2011.05.011

[15] M. Eirinaki, J. Gao, I. Varlamis, and K. Tserpes: Recommender systems for large-scale social networks: A review of challenges and solutions. Future Generation Computer Systems *78* (2018), 413–418. DOI:10.1016/j.future.2017.09.015

[16] M. Göksedef and Ş. Gündüz-Öğüdücü: Combination of web page recommender systems. Expert Systems Appl. *37* (2010), 2911–2922. DOI:10.1016/j.eswa.2009.09.046

[17] J. Golbeck: Trust and nuanced profile similarity in online social networks. ACM Trans. Web *3* (2009), 12:1–12:33. DOI:10.1145/1594173.1594174

[18] S. Hasanzadeh, S. M. Fakhrahmad, and M. Taheri: Review based recommender systems: A proposed rating prediction scheme using word embedding representation of reviews. The Computer J. *65* (2022), 2, 345–354. DOI:10.1093/comjnl/bxaa044

[19] T. Hofmann: Probabilistic latent semantic indexing. In: 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99), pp. 50–57. DOI:10.1145/312624.312649

[20] H. Kaur, N. Kumar, and S. Batra: An efficient multi-party scheme for privacy preserving collaborative filtering for healthcare recommender system. Future Generation Computer Systems *86* (2018), 297–307. DOI:10.1016/j.future.2018.03.017

[21] Y. Kilani, A. F. Otoom, A. Alsarhan, and M. Almaayah: A genetic algorithms-based hybrid recommender system of matrix factorization and neighborhood-based techniques. J. Comput. Sci. *28* (2018), 78–93. DOI:10.1016/j.jocs.2018.08.007

[22] Y. Koren and R. Bell: Advances in collaborative filtering. In: Recommender Systems Handbook 2011, pp. 77–118. DOI:10.1007/978-1-4899-7637-6_3

[23] Y. Koren, R, Bell, and C. Volinsky: Matrix factorization techniques for recommender systems. Computer *42* (2009), 30–37. DOI:10.1109/MC.2009.263

[24] J. Leskovec: New directions in recommender systems. In: 8th ACM International Conference on Web Search and Data Mining 2015, pp. 3–4. DOI:10.1145/2684822.2697044

[25] Q. Li and B. M. Kim: Constructing user profiles for collaborative recommender system. Advanced Web Technol. Appl. Lect. Notes Computer Sci. *3007* (2004), 100–110. DOI:10.1007/978-3-540-24655-8_11

[26] F. Liu and H. J. Lee: Use of social network information to enhance collaborative filtering performance. Expert Systems Appl. *37* (2010), 4772–4778. DOI:10.1016/j.eswa.2009.12.061

[27] J. Liu, C. Wu, and W. Liu: Bayesian probabilistic matrix factorization with social relations and item contents for recommendation. Decision Support Systems *55* (2013), 838–850. DOI:10.1016/j.dss.2013.04.002

[28] S. Najafi and Z. Salam: Evaluating prediction accuracy for collaborative filtering algorithms in recommender systems. `http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-186456`

[29] M. Nilashi, O. Ibrahim, and K. Bagherifard: A recommender system based on collaborative filtering using ontology and dimensionality reduction techniques. Expert Systems Appl. *92* (2018), 507–520. DOI:10.1016/j.eswa.2017.09.058

[30] Y. Qian, Y. Zhang, X. Ma, H. Yu, and L. Peng: EARS: Emotion-aware recommender system based on hybrid information fusion. Inform. Fusion *46* (2019), 141–146. DOI:10.1016/j.inffus.2018.06.004

[31] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl: GroupLens: An open architecture for collaborative filtering of netnews. In: 1994 ACM Conference on Computer Supported Cooperative Work (CSCW'94), pp. 175–186. DOI:10.1145/192844.192905

[32] P. Resnick and H. R. Varian: Recommender systems. Commun. ACM *40* (1997), 56–58. DOI:10.1145/245108.245121

[33] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl: Item-based collaborative filtering recommendation algorithms. In: 10th International Conference on World Wide Web (WWW'01), Hong Kong 2001, pp. 285–295. DOI:10.1145/371920.372071

[34] S. Y. Sert, Y. Ar, and G. E. Bostancı: Evolutionary approaches for weight optimization in collaborative filtering-based recommender systems. Turkish J. Electr. Engrg. Comput. Sci. *27* (2019), 3, 2121-2136. DOI:10.3906/elk-1812-175

[35] P. K. Singh, S. Sinha, and P. Choudhury: An improved item-based collaborative filtering using a modified Bhattacharyya coefficient and useruser similarity as weight. Knowledge Inform. Systems *64* (2022), 665–701. DOI:10.1007/s10115-021-01651-8

[36] J. K. Tarus, Z. Niu, and A. Yousif: A hybrid knowledge-based recommender system for e-learning based on ontology and sequential pattern mining. Future Generation Computer Systems *72* (2017), 37–48. DOI:10.1016/j.future.2017.02.049

[37] Z. Tu and W. Li: Multi-agent solver for non-negative matrix factorization based on optimization. Kybernetika *57* (2021), 60-77. DOI:10.14736/kyb-2021-1-0060

[38] P. Victor, C. Cornelis, M. D. Cock, and P. P. da Silva: Gradual trust and distrust in recommender systems. Fuzzy Sets Systems *160* (2009), 1367–1382. DOI:10.1016/j.fss.2008.11.014

[39] W. Yu and S.Li: Recommender systems based on multiple social networks correlation. Future Generation Computer Systems *87* (2018), 312–327. DOI:10.1016/j.future.2018.04.079

[40] Q. Zhang, J. Lu, and Y. Jin: Artificial intelligence in recommender systems. Complex Intell. Systems 7 (2021), 1, 439–457. DOI:10.1007/s40747-020-00212-w

[41] J. Zhu, Y. He, G. Zhao, X. Bo, and X. Qian: Joint reason generation and rating prediction for explainable recommendation. IEEE Trans. Knowledge Data Engrg. (2022). DOI:10.1109/TKDE.2022.3146178

Yilmaz Ar, Computer Engineering Department, Ankara University, 06830, Ankara. Turkey.
e-mail: ar@ankara.edu.tr

Şahin Emrah Amrahov, Computer Engineering Department, Ankara University, 06830, Ankara. Turkey.
e-mail: emrah@eng.ankara.edu.tr

Nizami A. Gasilov, Department of Computer Engineering, Baskent University, Ankara, 06790. Turkey.
e-mail: gasilov@baskent.edu.tr

Sevgi Yigit-Sert, Computer Engineering Department, Ankara University, 06830, Ankara. Turkey.
e-mail: syigit@ankara.edu.tr