Expert Systems With Applications 66 (2016) 234–244

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

A collaborative filtering method for music recommendation using playing coefficients for artists and users

Diego Sánchez-Moreno, Ana B. Gil González, M. Dolores Muñoz Vicente, Vivian F. López Batista, María N. Moreno García*

Department of Computing and Automation, University of Salamanca, Salamanca, Spain

ARTICLE INFO

Article history:
Received 22 June 2016
Revised 10 September 2016
Accepted 11 September 2016
Available online 13 September 2016

Keywords:
Collaborative filtering
Music recommendation
Data mining
Sparsity
Gray-sheep

ABSTRACT

The great quantity of music content available online has increased interest in music recommender systems. However, some important problems must be addressed in order to give reliable recommendations. Many approaches have been proposed to deal with cold-start and first-rater drawbacks; however, the problem of generating recommendations for gray-sheep users has been less studied. Most of the methods that address this problem are content-based, hence they require item information that is not always available. Another significant drawback is the difficulty in obtaining explicit feedback from users, necessary for inducing recommendation models, which causes the well-known sparsity problem. In this work, a recommendation method based on playing coefficients is proposed for addressing the above-mentioned shortcomings of recommender systems when little information is available. The results prove that this proposal outperforms other collaborative filtering methods, including those that make use of user attributes.

© 2016 Elsevier Ltd. All rights reserved.

1. Introduction

Current music platforms make available to users a large number of songs through web sites and mobile apps. These abilities have led to the extension of the overload problem, which has its origin in the context of information retrieval, to this kind of applications, since users have difficulties in finding out the music they like. In order to make this task easier, many of the platforms have searching services and some of them are endowed with recommendation mechanisms. However, in the last case, it is necessary to address some of the usual problems of recommender systems.

Collaborative filtering (CF) methods are widely used in recommender systems. They provide recommendations based on ratings that users give to items. The results of these techniques are quite good; however, the difficulty in obtaining explicit feedback in the form of ratings from the users causes the sparsity problem, which takes place when the number of available ratings for the items to be recommended is small. This is the main drawback that prevents the application of this approach in many systems. A way to address this problem is to derive implicit ratings from user behavior.

There are two approaches for collaborative filtering: memory-based and model-based algorithms. Memory-based algorithms, also known as nearest-neighbor methods, were the earliest used. They treat all user items in order to find users with similar preferences (neighbors). The advantage of these algorithms is the quick incorporation of the most recent information, but they can present scalability problems given that the search for neighbors in large databases is slow. Model-based CF algorithms deal with scalability problems by using the ratings from users for computing item similarity instead of user similarity, considering that items are similar if they are liked/disliked by the same users. Thus, recommendations can be provided to users, given that users are expected to have similar preferences for similar items.

Additional drawbacks presented by CF methods are early-rater (first-rater) and cold-start problems. The first one takes place when new products are introduced in the system. These items have never been rated, therefore they cannot be recommended. Cold-start problem affect new users, who cannot receive recommendations since they have no evaluations about products.

Content-based algorithms have been proposed as an alternative to CF methods in order to deal with the shortcomings discussed previously. These methods can be used for recommending any kind of item by making use of its features. Thus, new items can be recommended according to their similarity to other items for which the user showed interest in the past. Gray-sheep users also generate a problem in recommender systems that has sometimes been addressed with content-based methods. These are users who have unusual preferences; thus they do not have enough neighbors for computing reliable recommendations.

* Corresponding author.
E-mail addresses: sanchezhh@gmail.com (D. Sánchez-Moreno), abg@usal.es (A.B. Gil González), mariado@usal.es (M.D. Muñoz Vicente), vivian@usal.es (V.F. López Batista), mmg@usal.es (M.N. Moreno García).

http://dx.doi.org/10.1016/j.eswa.2016.09.019
0957-4174/© 2016 Elsevier Ltd. All rights reserved.

Most of the current recommender systems use hybrid techniques aiming at taking advantage of the strengths of both approaches and avoiding their drawbacks. These methods take into account the preferences of other users as well as the characteristics of items and users (age, gender, occupation…). Therefore, new users can receive recommendations depending on their characteristics.

Although there are many proposals in the literature for dealing with the weaknesses of recommender systems, the gray-sheep problem has received less attention, and is mainly addressed by means of hybrid approaches that involve some content-based technique. Their results are usually good but these methods require information about items and users that often is not available. In the context of music recommender systems, content-based filtering algorithms use musical content for inducing the models; therefore, a complex extraction task of music features is necessary. In this work, the proposed recommendation methodology addresses the above-mentioned drawback when little information is available. The recommendation process could be incorporated into any music platform as long as it stores user and artist identification and the number of times the user plays a song in the platform, without the need for collecting rating data.

The rest of the paper is organized as follows: Section 2 includes a description of the state of the art of recommendation methods with special focus on collaborative filtering. The proposed methodology is described in Section 3 and the empirical study conducted for its validation is reported in Section 4. Finally, the conclusions and future work are given in Section 5.

2. Related work

Most of the current recommender systems use some CF based approach. The aim of CF is to predict the rating that a target user would give to an item taking into account users having similar preferences to this target user regarding previously rated items. The GroupLens research system for Usenet news (Resnick, Iacovou, Suchack, Bergstrom, & Riedl, 1994) was the first recommender system using CF, and Ringo (Sarwar, Karypis, Konstan, & Riedl, 2001) was one of the first and most popular music recommender systems based on CF.

CF requires user-explicit expression of personal preferences for items in the form of ratings, which are usually difficult to obtain. This fact is at the root of one of the main drawbacks of this approach, the sparsity problem, which arises when the number of ratings needed for prediction is greater than the number of the ratings obtained from the users. The time that users spend examining the items is an alternative way to obtain implicit user preferences (Sarwar et al., 2001) but it requires processing of log files and this implicit information about user preferences is not as reliable as the explicit ratings. In the music recommender area several ways of dealing with this problem have been proposed. The access history of users is taken as an implicit way of obtaining user interests in a music recommendation system based on music and user grouping (Chen & Chen, 2005). In several works where the last.fm database is used, the times that the users play the songs (play counts) are converted to ratings by means of different functions (Lee & Lee, 2015; Vargas & Castells, 2011). The ratings, whether implicit or explicit, are arranged in a user-items rating matrix. Empty elements of the matrix represent items not rated by the corresponding users.

In memory-based (user-based or user-user) CF methods (Resnick et al., 1994) the predictions for a given user, called the active user, are based on that person's nearest neighbors. Neighbors are users who have similar preferences to the active user since they have rated items in common with a similar score. These methods need to use the entire rating matrix to compute the similarity between users. In consequence, the computation time grows linearly with both the number of customers and the number of items in the system. This drop in performance, known as the scalability problem, has a direct impact on the user response time since similarity is computed at recommendation time. There are different measures for obtaining the similarity; however, the most extended ones are the Pearson correlation coefficient and cosine similarity (Breese, Heckerman, & Kadie, 1998). The Pearson correlation usually provides better results than cosine similarity but its computational cost is higher.

Model-based (item-based or item-item) CF was proposed in Sarwar et al. (2001) with the aim of avoiding the scalability problem, has a direct impact on the user response time since similarity is computed at recommender time the scalability problems (Schafer, Konstant, & Riedl, 2001) associated with memory-based methods by precomputing the similarities between items. This can be done since it is expected that new ratings given to items in large rating databases do not significantly change between-item similarity, especially for much-rated items. On the contrary, precomputing the similarities between users would not be effective given that the neighborhood of a user is obtained from both his ratings and the ratings of other users, which undergo continuous changes and additions (Ekstrand, Riedl, & Konstan, 2010). There are several procedures for computing item similarity; however, cosine similarity is the most extended one because of its simplicity, efficiency and better results regarding accuracy than the Pearson coefficient. Recommendations provided by item-based methods usually have less quality than those provided by user-based approaches, but they can be suitable to be applied in large-scale systems where scalability is a serious problem. For example, they have been used in popular systems like Amazon (Lucas et al., 2013).

Another kind of model-based algorithms builds a predictive model of user preferences by means of data mining techniques. Besides the ratings, other attributes of items and/or users can be used for inducing the model, so this process also involves a content-based approach. Data mining methods usually behave better against sparsity, especially association-based methods (Lucas, Laurent, Moreno, & Teisseire, 2012). Moreover, scalability problems are avoided since predictive models are already built when the user requests recommendations; thus, the building time has no impact on the user response time. The main inconvenience of these techniques is the need for frequent updating of the models in order to incorporate the most recent information generated by users. Moreover, data mining methods require more information than the simple rating data.

Common shortcomings of both user-based and item-based CF are the cold-start and early-rater (first-rater) problems, described in the previous section. In these cases, when recommendations cannot be provided to new users or new products cannot be recommended, respectively, then content-based methods can be applied. They were first used to recommend text documents by comparing their contents and the contents of other documents associated with the user profile but without taking into account the opinion of other users (Lee, Kim, & Rhee, 2001). Currently, they have been extended to other domains by replacing document contents by other characteristics of the items (Billsus & Pazzani, 1999; Krulwich & Burkey, 1996). They also take advantage of the similarity between items but they do not need rating data since they make use of other features of the items for computing the similarity. Some content-based approaches use distance metrics such as cosine similarity but others resort to data mining methods. In the music field, metadata of the items, such as title, artist, genre and lyrics, can be exploited as content attributes, but also audio features like timbre, melody, rhythm or harmony. In Tzanetakis (2002), similarity was determined from chord structure (spectrum, rhythm and harmony). Melody style is the music feature used in Kuo and Shan (2002) for music recommendation. A

content-based method is proposed where a classification of music objects in melody styles is performed and users' music preferences are learned by mining the melody patterns from the music access behavior of the users. Clustering of similar songs according to different features of audio content is performed in Cataltepe (2007) in order to provide users with recommendations of music from the appropriate clusters. The listening behavior of the user is taken to determine the best cluster for that person. In Chen and Chen (2005) music recommendation is based on the classification of musical objects according to the pitch, tempo, loudness and entropy features. The content used is the metadata of data items, which includes the title, artist, genre and lyrics of a musical piece. Although these kinds of methods are good for solving cold-start and early-rater problems, they are not as good as the CF methods since content similarity does not reflect user preferences well.

Currently, hybrid techniques are the ones most extensively implemented in recommender systems, in an attempt to address the limitations of CF and content-based approaches. These methods combine either different categories of CF methods or CF with other recommendation techniques such as content-based schemes (Su & Khoshgoftaar, 2009). The combination of memory-based and model-based CF approaches is a common way of building hybrid CF approaches that usually yields better recommendations than the single methods applied separately (Yu, Schwaighofer, Tresp, Xu, & Kriegel, 2004). Moreover, many proposals of CF and content-based hybrid methods have been made with the aim of improving prediction performance (Melville, Mooney, & Nagarajan, 2002) as well as for dealing with the sparsity and cold-start problems (Lucas et al., 2013; Moreno, Segrera, López, & Muñoz, 2016; Su, Greiner, Khoshgoftaar, & Zhu, 2007), among others. Hybrid strategies have also been adopted in the development of music recommender systems. In Yoshii, Goto, Komatani, Ogata, and Okuno (2006), the authors associate rating and content data with latent variables that directly describe unobserved user preferences. They adapt to music recommendation a Bayesian network called a three-way aspect model that was originally designed for document recommendation. Unobservable user preferences are represented as a set of latent variables that are statistically estimated and introduced in the Bayesian network. Another hybrid music recommender system is presented in Lu and Tseng (2009). Its authors propose a content-based scheme for recommending unrated music, a collaboration algorithm for recommendations based on other users' suggestions and an emotion-based recommendation procedure that determines interesting music for users by computing the differences between the interests of users and musical emotions. A weighting system based on user listening behavior is used to combine the three methods. This proposal requires users to fill in a questionnaire so that their interests can be discovered, which is not always possible. Recently, many hybrid recommender systems exploit social media and other web sources in order to gather information that can be useful in the recommendation process (Deng, Wang, Li, & Xu, 2015; Hyung, Lee, & Lee, 2014).

The described categories of recommender methods have not been specially formulated to deal with the less addressed but significant drawback that gray sheep users suffer from. This group of users with unusual preferences usually receives poor recommendations since they do not have many neighbors (Claypool et al., 1999). Furthermore, it is not only gray sheep users that are affected by this problem since it has been proved that the existence of a large number of gray sheep users might have an important impact on the recommendation quality of the entire community (Ghazanfar & Prügel-Bennett, 2014). Content-based methods can help to alleviate this limitation but they are not the proper solution. Semantic Web Mining is another approach that can be used to solve gray sheep and other typical problems of recommender systems. Semantic information is added to the available data in order to formalize and classify product and user features. In this way it is able to generate more reliable content-based models that can be combined with other approaches in order to improve recommendations (Kim, Alkhaldi, El Saddik, & Jo, 2011; Moreno et al., 2016). In Cantador, Bellogín, and Castells (2008) the authors make use of domain ontologies to classify users and items in a multi-layered community of interests prior to the similarity computation. These types of methods are not easily extendible given that every application domain would involve the time-consuming task of defining a specific ontology.

Most of the ways to face the gray sheep problem proposed in the literature are very complex and require additional information that sometimes is unavailable. However, a more extended and simpler approach of improving recommendations for gray sheep users is the application of clustering methods (Ghorbani & Novin, 2016). A comprehensive review about the use of diverse clustering techniques in recommender systems is carried out in Ghazanfar and Prügel-Bennett (2014). In addition, these authors provide their own proposal to deal with the gray sheep drawback. They use the k-means algorithm to generate clusters in order to detect the gray sheep users and provide them with recommendations based on their profiles, while the recommendations for the remaining users are obtained by a clustering-based CF algorithm. They also analyze the effect of different distance metrics in the quality of the recommendations. In some works, the clustering technique is used to address the sparsity and gray sheep problems at the same time since some authors consider that both problems are related. In Lucas et al. (2012, 2013) fuzzy class association rules are induced from previously clustered data in order to assign more than one cluster to each user with different degrees of belonging. A simulated scenario for gray sheep users proved the effectiveness of the method. The process, implemented in a tourist system, is not simple and requires user and item features. The last.fm dataset is used in Shepitsen, Gemmell, Mobasher, and Burke (2008) to validate a hierarchical agglomerative clustering method for recommending resources in folksonomies, which takes into account the user's current navigation context in cluster selection. As far as we know, all of the methods proposed for dealing with the gray sheep problem make use of user and/or item attributes.

3. Recommendation method

The proposed procedure aims at providing reliable recommendations when content and rating information is not available. The improvement of the recommendations is mainly achieved by focusing on the gray sheep users, given that their presence has a negative impact on the recommendations for them as well as on those for the other users. However, in contrast to most of the methods in the literature, gray sheep users are neither treated differently from the rest of the users nor are they separated into a different group; Our strategy involves the determination for every user of a coefficient representing the degree to which they are gray sheep users.

Our proposal is based on the use of implicit information to obtain user preferences as well as to characterize users and items to be recommended. Specifically, we have designed an algorithm for recommending artists by using information obtained from the last.fm database. The number of plays, that is, the number of times that every user listens to a specific artist, is taken as implicit information to know user preferences. The count of plays is also used as input to compute a listening coefficient for the artists in order to characterize their popularity. The listening coefficient and users' behavior regarding the artists they play are used to characterize them according to the degree of uncommonness of their preferences. In this way, gray sheep users are identified by considering the artists they listen to, differently from some authors who take

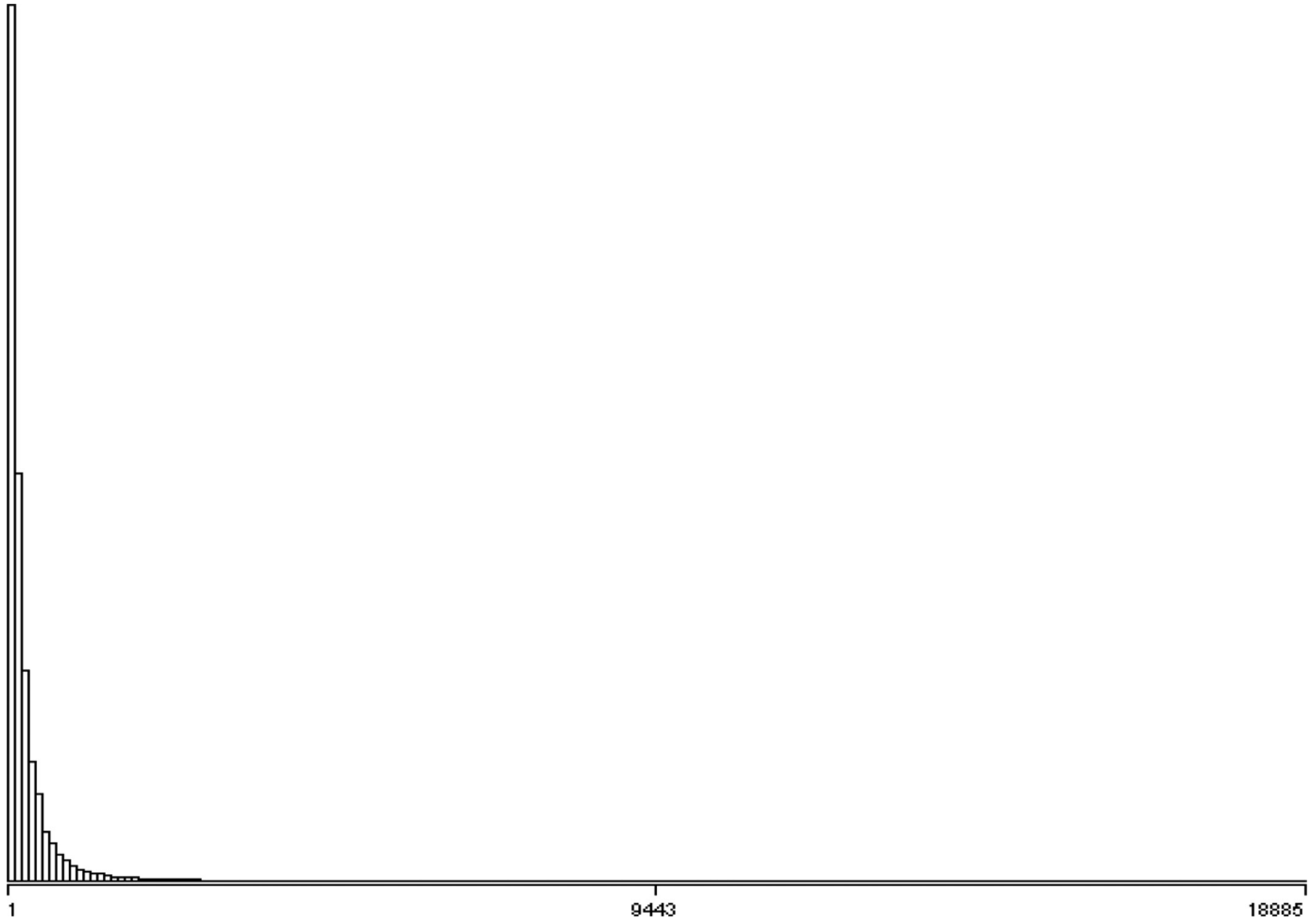

Fig. 1. Power law distribution of play frequencies.

into account only the number of ratings provided by that user. The work of Ghazanfar and Prügel-Bennett (2014) proves that there is no potential correlation between the playing count of a user and their belonging to the gray sheep group.

3.1. Deriving ratings from plays

An empirical study was conducted in order to validate the proposed method using a dataset collected by Oscar Celma (http://mtg.upf.edu/node/1671) from the last.fm database (http://www.last.fm). The data are distributed in two files, one of them containing user profile information (genre, age and country) and other one containing identifiers of artists and users as well as the number of plays representing how many times each user has listened to each artist.

The dataset does not contain any information about user preferences; thus, to estimate the ratings it is necessary to resort to the counts of plays, which is the only available information encompassing the implicit feedback of the users. We followed the method by Pacula (2009), which is indicated for the play frequencies that have a clear power law distribution since there are few highly played artists and most of them have few plays. This does not occur when ratings are given in an explicit way. Fig. 1 shows this frequency distribution in the dataset used in this study.

The play frequency for a given artist i and a user j is defined as follow:

$$Freq_{i,j} = \frac{p_{i,j}}{\sum_{i'} p_{i',j}} \tag{1}$$

Where $p_{i,j}$ is the number of times that a user j plays an artist i.

On the other hand, $Freq_k(j)$ denotes the kth most listened to artist for user j. Then, a rating for an artist with rank k is computed as a linear function of the frequency percentile:

$$r_{i,j} = 4\left(1 - \sum_{k'=1}^{k-1} Freq_{k'}(j)\right) \tag{2}$$

Once the ratings are calculated, CF methods can be applied in the way done for datasets containing explicit user preferences.

3.2. Recommendation algorithm

The key feature of the recommendation algorithm is the characterization of artists depending on their popularity as well as the characterization of users according to the artists they listen to. We consider that gray sheep users, who have unusual preferences, are those who mostly play unpopular artists while standard users have similar preferences to other users since they listen to popular artists. However, we neither consider a division line between the two types of users nor create a group for gray sheep users in order to manage them in a different way. The aim of the proposed procedure is to compute a coefficient for all users that characterizes them depending on the popularity of the artists they play most. Popularity of the artists is based on a listening coefficient that involves the number of users who play them and the number of plays they have. The process can be formalized as follows.

Given a set of artists A and a set of users U where $a_i \in A,\ i = 1,\ldots,n$ and $u_j \in U,\ j = 1,\ldots,m$ represent an artist and a user respectively, the number of times that a user j plays an artist i is characterized as $p_{i,j}$.

A listening coefficient (l_{a_i}) for artist a_i is computed in order to establish his popularity (Eq. (3)).

$$l_{a_i} = \left(TU_{a_i} / \overline{TU}\right) \frac{\sum_j \left(p_{i,j} / \overline{p_j}\right)}{\sum_i \sum_j \left(p_{i,j} / \overline{p_j}\right)/n} \tag{3}$$

Where n is the overall number of artists, TU_{a_i} is the number of users who play the artist a_i, $\overline{TU}$ is the average number of users per artist, and $\overline{p_j}$ the average number of plays per artist of user j.The listening coefficient for a given artist takes into account the number of users who listen to him with respect to the average number of users per artist as well as the relation between the behavior of the users who listen to the artist and the average behaviors for all artists. The behavior of the user is quantified as the number of plays for the given artist with respect to the average number of plays of this user.In order to obtain values between 0 and 1, a normalized listening coefficient L_{a_i} was computed (Eq. (4)).

$$L_{a_i} = \frac{l_{a_i} - min\ l_{a_i}}{max\ l_{a_i} - min\ l_{a_i}} \Rightarrow L_{a_i} \in [0,1] \tag{4}$$

Therefore, the higher the L_{a_i} is, the more popular the artist is.

A second coefficient related to users, the User Playing Coefficient (UPC), is proposed, aimed at characterizing them according to the popularity of the artists they play. First, the $\alpha_{i,j}$ parameter is computed to retain the played artists by user j, then, the UPC_{u_j} coefficient for ranking the users is computed (Eq. (5)).

$$\alpha_{i,j} = \begin{cases} 0, & p_{i,j} = 0 \\ 1, & p_{i,j} > 0 \end{cases} \quad UPC_{u_j} = \frac{\sum_i \alpha_{i,j}\ L_{a_i}}{TA_{u_j}} \tag{5}$$

TA_{u_j} is the number of artists played by user j.

High values of UPC_{u_j} represent users that like popular artists and low values correspond to gray sheep users.

L_{a_i} and UPC_{u_j}coefficients are used in the recommendation procedure described below. The first step of the algorithm requires us to obtain the number of times that user j plays an artist i, $p_{i,j}$, for all users and artists. They are represented by means of matrix $\mathbf{P} := p_{i,j}$ where $\mathbf{P} \in M_{n\times m}(\mathbb{N})$:

$$\boldsymbol{P} = \begin{bmatrix} p_{1,1} & \cdots & p_{1,m} \\ \vdots & \ddots & \vdots \\ p_{n,1} & \cdots & p_{n,m} \end{bmatrix}$$

The algorithm presented in Fig. 2 describes the complete sequence of steps required for inducing user playing coefficients and ratings. TP_{a_i} is the total number of plays for artist a_i.

The computed coefficient UPC is used as a user attribute of two kinds of user-based CF methods: Rating prediction and item recommendations. The first provides users with predicted ratings for items that they have not rated, while the second one, also called top-N recommendations, provides them with a list of N top-ranked items that could match their preferences. Both approaches make use of similarity measures computed either for users or items.

The introduction of the UPC coefficients in the CF process implies that our proposal is a hybrid method since, besides the ratings, another attribute is involved. That is, two users are similar not only if they give similar rates to the same items but also if they have similar playing coefficients. Given that the Pearson Coefficient is only defined for ratings, it is necessary to use an alternative similarity measure that can be applied also for other attributes. We propose the cosine similarity, which is widely used in CF and hybrid approaches. This metric has a lower computational cost than the Pearson coefficient, although sometimes its results are slightly worse.

```
1:  Create matrix P := p_{i,j}
2:  for i = 1 to n do
3:      TU_{a_i} = Σ_j u_j  ∀ j | p_{i,j} > 0
4:      TP_{a_i} = Σ_j p_{i,j}
5:      PpU_{a_i} = TP_{a_i} / TU_{a_i}
6:  end for
7:  TU = Σ_i TU_{a_i} / n
8:  α_{i,j} = 0
9:  for j = 1 to m do
10:     p_j = Σ_i p_{i,j} / TA_{u_j}
11:     Compute α_{i,j}
12: end for
13: for i = 1 to n do
14:     Compute l_{a_i} using eq. 3
15:     Compute L_{a_i} using eq. 4
16: end for
17: for j = 1 to m do
18:     Compute UPC_{u_j} using eq. 5
19:     Compute rating r_{i,j} from p_{i,j} using eq. 1 and 2
20: end for
```

Fig. 2. Algorithm for computing User Playing Coefficients (UPC) and ratings.

UPC can be used either as the only attribute if there is no more user information available or jointly with other user attributes. This is one of the main advantages of the proposed method since an improvement in the recommendations is achieved even when no user and item information is available.

In order to validate the methodology, we have carried out a study using data from the last.fm database. The following section is devoted to describing this study.

4. Experimental study

This study was carried out with a dataset obtained from the last.fm database that was described in Section 3.1. We preprocessed the two original files to join data in a single file with a subset containing 18,698 records about 12,293 users and 320 artists. The data available in the file were the identifiers of users and artists, the attributes genre, age and country of the users, as well as the number of plays indicating how many times each user has listened to each artist. In order to apply the recommendation method described in the preceding section, listening coefficients for artists, user playing coefficients (UPC), and ratings were computed.

A comparative study was conducted in which the results of our proposal were analyzed against those of traditional CF approaches and variants of them using user attributes such as age, gender and country. The introduction of these attributes affects only to the similarity measure, which has to include them. As mentioned before, cosine similarity metric is used since it can be applied in any n dimensional space, where n is the number of attributes.

K-nearest neighbor (K-NN) was the method used in the study since it is the most common in the implementation of CF-based recommender systems. K-NN builds a neighborhood of K users similar to the active user according to a given similarity measure which can be computed from user ratings as well as from other user attributes. It can also be applied to find similarities between items. We tested user-based and item-based K-NN using both cosine and Pearson similarity measures. In addition to user ratings, the user-based K-NN algorithm was also tested making use of user attributes. All the available user attributes were used, including the user playing coefficients (UPC), specific to our proposal. The number of K neighbors was set to 60 although the results were quite similar for both higher and lower values.

The comparative study was carried out for two types of recommendations, rating prediction and item recommendation. The metrics used for validating the proposal are described in the next

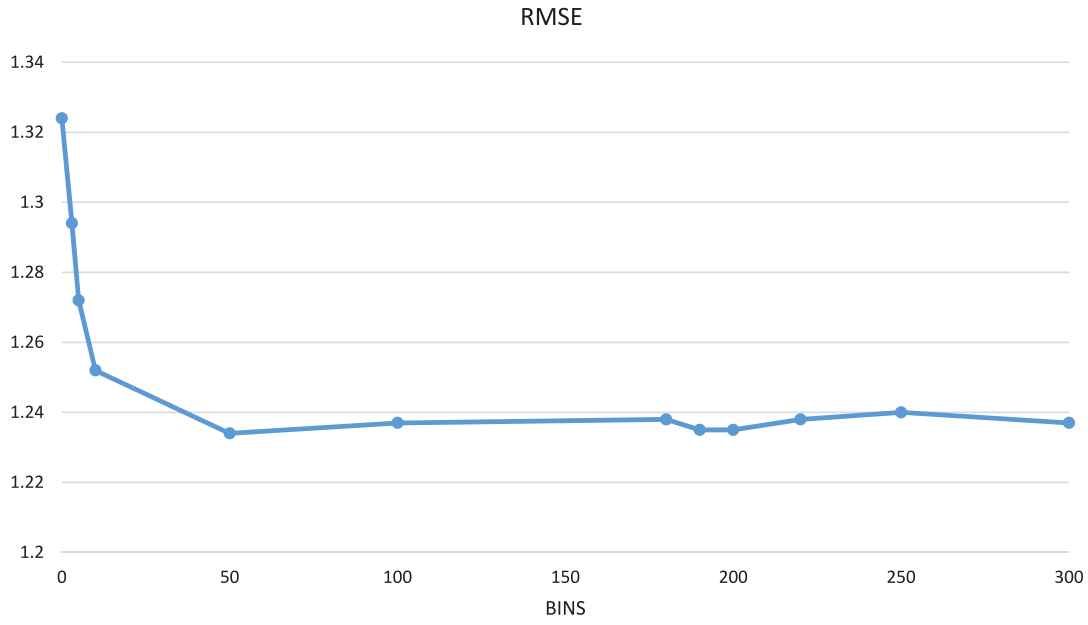

Fig. 3. RMSE of rating prediction for different number of bins.

subsection. Ten-fold cross validation was performed in the evaluation of all the algorithms.

4.1. Validation metrics

Metrics used to validate the quality of the recommendations are different depending on the type of recommendations supplied to the active user, rating prediction or item recommendation. The usual way of validating the results for rating prediction is to compute the deviation between actual and predicted ratings. The most extended metrics are MAE (Mean Absolute Error), NMAE (Normalized Mean Absolute Error) and RMSE (Root-Mean-Square Error) (Herlocker, Konstan, Terveen, & Riedl, 2004).

MAE is defined as the average of the absolute error or difference between the actual rating (r) and the predicted rating (pr).

$$MAE = \frac{1}{c}\sum_{i=1}^{c}|r_{i,j} - pr_{i,j}|$$

Where c is the number of items that user j has evaluated, $r_{i,j}$ is the rating that user j has given to item i and $pr_{i,j}$ is the predicted rating for user j and item i.

NMAE is a measure of the precision independent of the evaluation range. It is the absolute error normalized with respect to the range of rating values

$$NMAE = \frac{MAE}{r_{max} - r_{min}}$$

RMSE is a good measure to compare the errors of different predictive models. It represents the sample standard deviation of the differences between predicted values and observed values.

$$RMSE = \sqrt{\frac{1}{c}\sum_{i=1}^{c}\left(r_{i,j} - pr_{i,j}\right)^2}$$

The evaluation of item recommendations requires different metrics to apply to the ranked list provided to the user. Some measures that were defined in the information retrieval field (Jarvelin & Kekalainen, 2002) are commonly used. Two usual metrics are MAP (Mean Average Precision) and NDCG (Normalized Discounted Cumulative Gain).

MAE is the average of the precision value obtained for each item in the top-N list.

NDCG is a measure based on the assumption that the lower the ranked position of a relevant item, the less useful it is for the user. When applying this metric, gain is accumulated starting at the top of the ranking. The graded relevance value of items in lower positions is reduced in a quantity logarithmically proportional to the position of the result. DCG is the discounted cumulative gain accumulated at a particular rank k.

The well-known metric AUC (Area Under the ROC Curve) was also used in the item recommendation evaluation.

4.2. Validation of the proposal for rating prediction

We first evaluated the validity of the proposed CF method in the prediction of ratings. However, before applying the methods involved in the study, user playing coefficients (UPC) were discretized in order to improve efficiency. The results of the proposal for different numbers of bins were examined with the aim of obtaining the optimal partition. In Figs. 3–5, which shows the values of the error metrics, we can see that their values tend to stabilize for 50 bins. Thus, this was the number of bins chosen to conduct the experiments.

Once the attribute UPC was discretized, the proposed procedure was applied making use of the intervals obtained in the discretization process instead of using the continuous values.

Table 1 shows the results yielded by the algorithms tested in the study, including our proposal: *user attribute KNN-UPC*. Three categories of methods were used: user-based (*user KNN*), item-based (*item KNN*) and variants of user-based methods, which include different user attributes (*user attribute KNN*). The values of the error metrics and their standard deviation are presented in the table. The lowest values of error were provided by the method proposed in this work. Another positive observation is the narrower width of the confidence intervals in the results of all of the metrics for this method as opposed to the rest of the tested approaches. The only exception is the value of MAE standard deviation of *item KNN-Pearson*, which is slightly lower than the value of our proposal. Figs. 6–8 show the values of the metrics RMSE, MAE and NMAE respectively. These figures allow us appreciate in a better way the significant improvement achieved by means of the user-

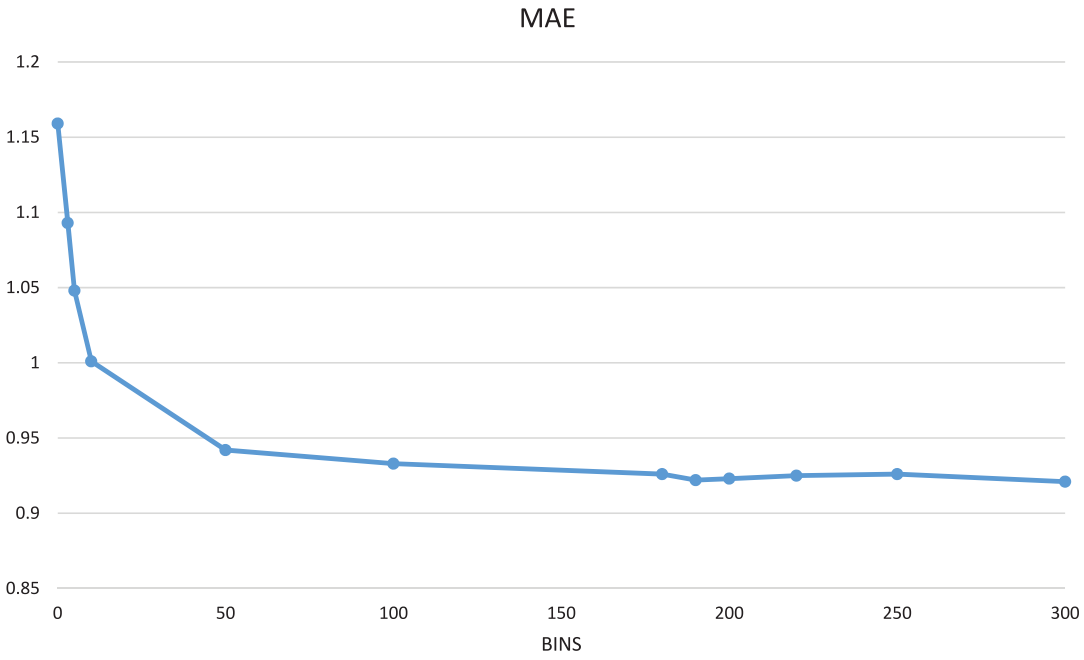

Fig. 4. MAE of rating prediction for different number of bins.

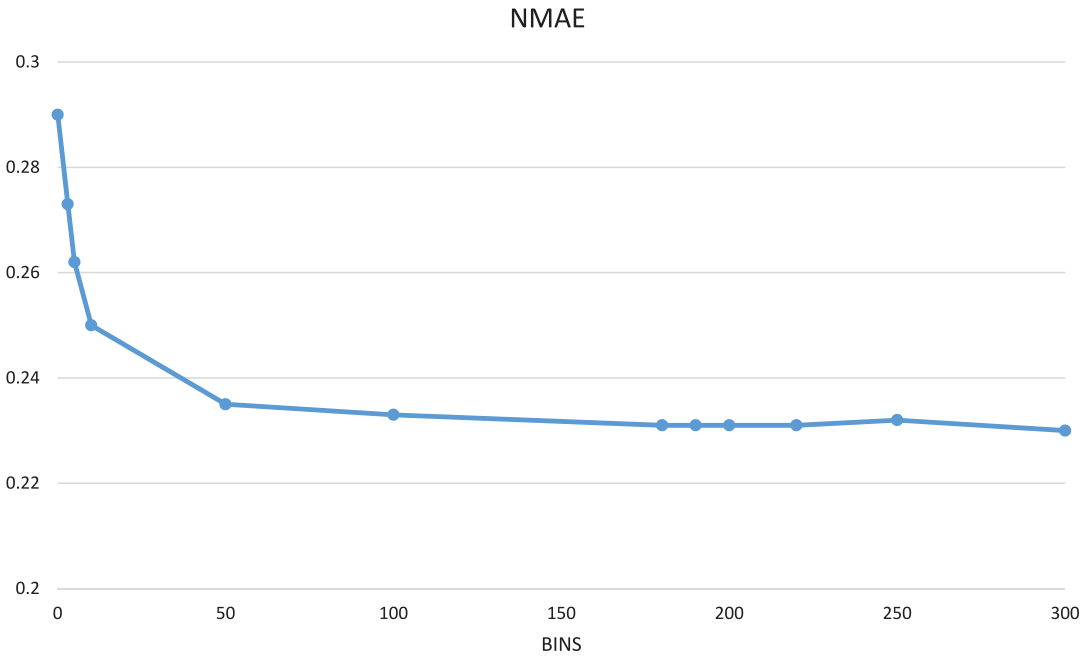

Fig. 5. NMAE of rating prediction for different number of bins.

Table 1
Results for rating prediction.

Algorithm	RMSE	MAE	NMAE
User KNN-Cosine	1.309 +/− 0.018	1.165 +/− 0.014	0.291 +/− 0.004
User KNN-Pearson	1.331 +/− 0.021	1.182 +/− 0.015	0.295 +/− 0.004
User attribute KNN-UPC	**1.231 +/−** 0.014	**0.940 +/−** 0.012	**0.235 +/−** 0.003
User attribute KNN-Age	1.368 +/− 0.017	1.201 +/− 0.015	0.300 +/− 0.004
User attribute KNN-Gender	1.352 +/− 0.018	1.200 +/− 0.015	0.300 +/− 0.004
User attribute KNN-Country	1.391 +/− 0.018	1.195 +/− 0.013	0.299 +/− 0.003
User attribute KNN- Country, Gender, Age	1.368 +/− 0.017	1.201 +/− 0.015	0.300 +/− 0.004
User attribute KNN- UPC, Country, Gender, Age	1.352 +/− 0.018	1.200 +/− 0.015	0.300 +/− 0.004
Item KNN-Cosine	1.854 +/− 0.014	1.598 +/− 0.012	0.400 +/− 0.003
Item KNN-Pearson	1.361 +/− 0.016	1.214 +/− 0.011	0.304 +/− 0.003

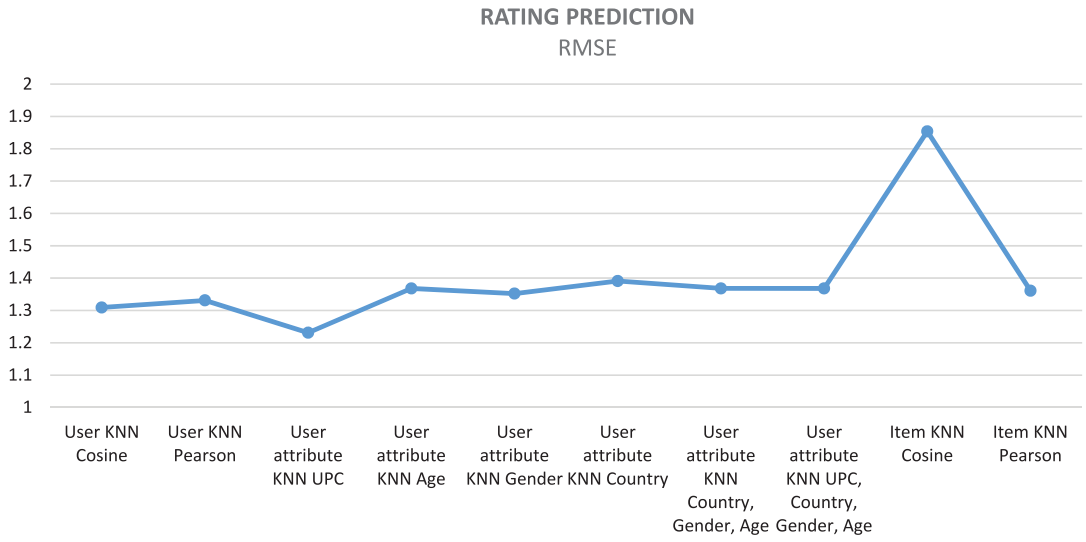

Fig. 6. Values of RMSE for rating prediction.

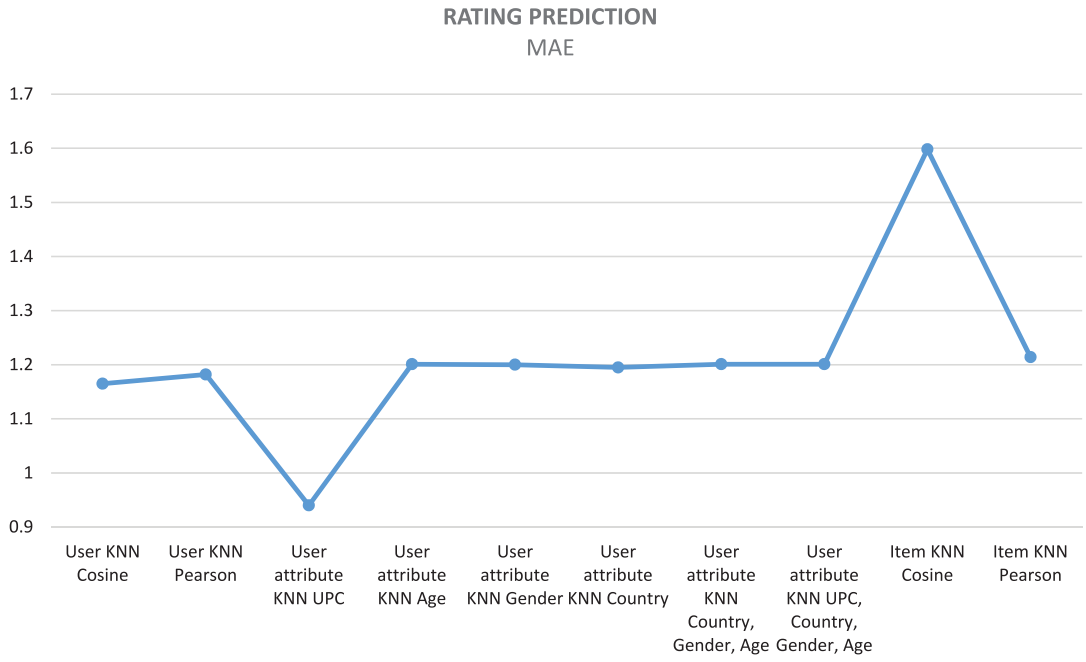

Fig. 7. Values of MAE for rating prediction.

based K-NN algorithm with the lone UPC attribute, regarding the results of the other algorithms, even with respect to the outcome of the same method with other attributes. The use of UPC jointly with all the other user attributes also gave worse results.

4.3. Validation of the proposal for item recommendation

The same dataset used for rating prediction was the input of the item recommendation algorithms. The same algorithms were also tested, but the similarity measure used in all the algorithms was cosine since it is the most suitable one for this kind of recommendation. The results obtained in this study are presented in Table 2. In Figs. 9–11 the values of the quality metrics are presented. The best values of these measures were yielded by our approach. We can appreciate in the figures that the difference was especially significant for Mean Average Precision (MAP) and Normalized Discounted Cumulative Gain (NDCG) while it was lower for the area under the ROC curve (AUC). Another observation to highlight is the extremely poor results of the user based K-NN and item-based K-NN algorithms, which compute the similarity only from user ratings. The use of other user attributes improves the recommendations, especially the use of UPC alone.

Table 2
Results for item recommendation.

Algorithm	MAP	NDCG	AUC
User KNN	0.088	0.206	0.224
User attribute KNN-UPC	**0.532**	**0.622**	**0.865**
User attribute KNN-Age	0.258	0.404	0.827
User attribute KNN-Gender	0.268	0.412	0.825
User attribute KNN-Country	0.306	0.444	0.844
User attribute KNN-Country, Gender, Age	0.258	0.404	0.827
User attribute KNN-UPC, Country, Gender, Age	0.258	0.404	0.827
Item KNN	0.010	0.132	0.113

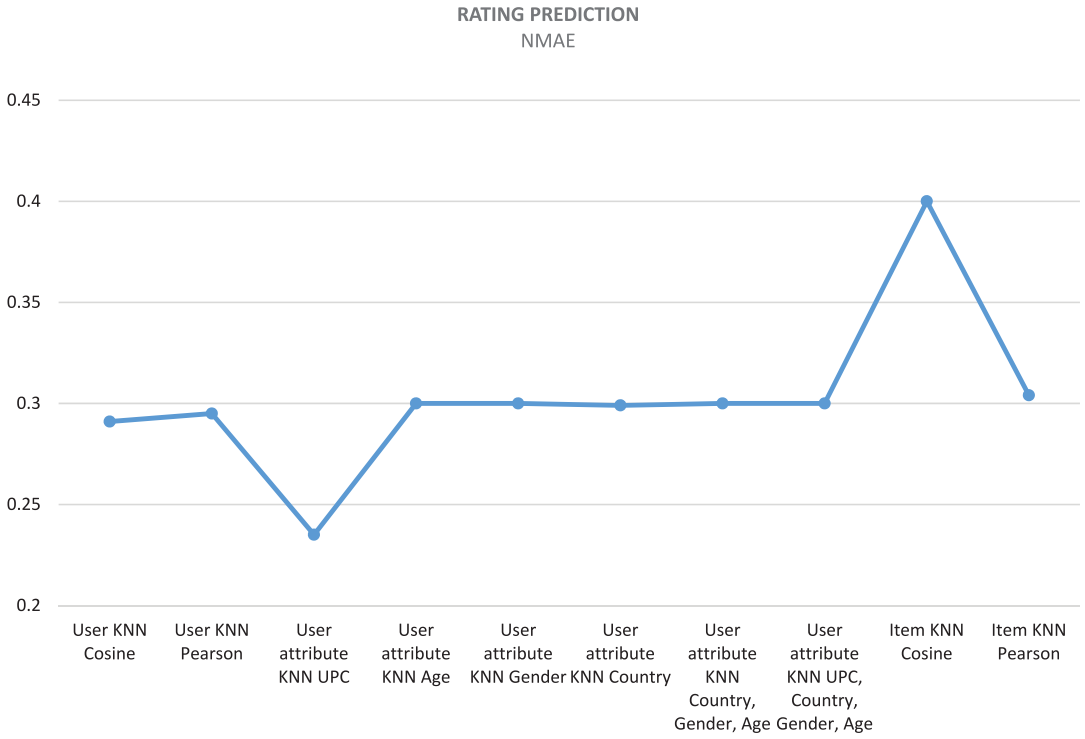

Fig. 8. Values of NMAE for rating prediction.

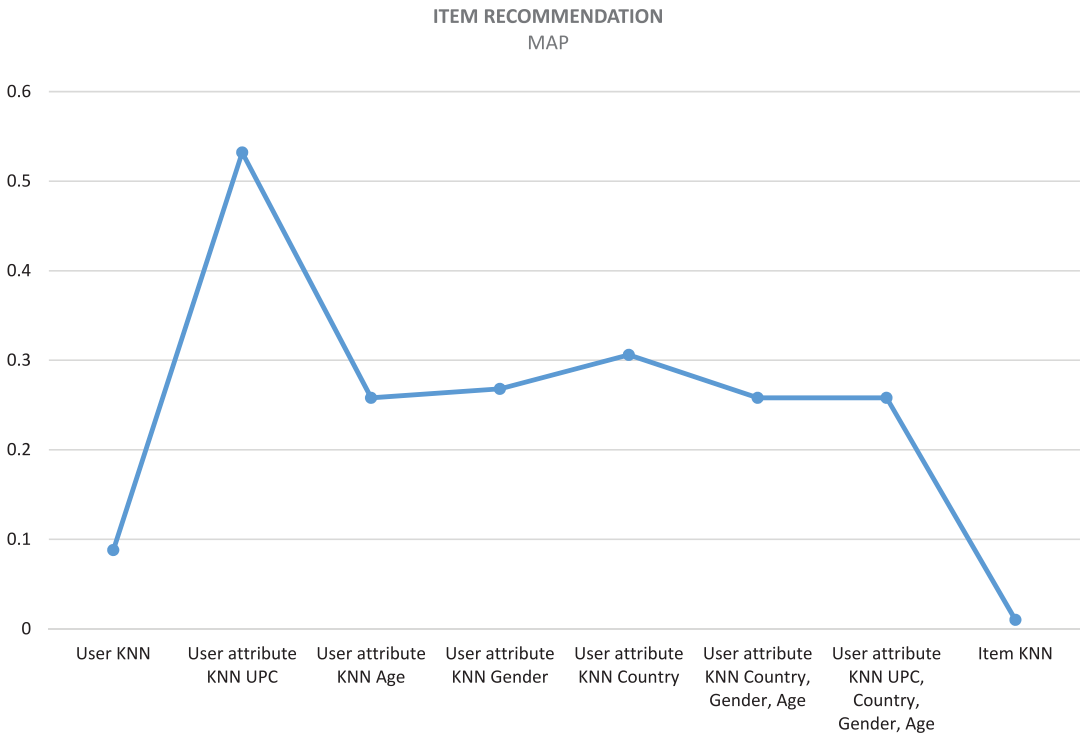

Fig. 9. Values of mean average precision for item recommendation.

5. Conclusions and future work

Music recommendation is a research topic of increasing interest since online music platforms have become popular. However, to provide reliable recommendations it is necessary to deal with the general drawbacks of recommender system as well as with those specific to this application domain, such as the difficulty of extracting content information from music. In this work some of these shortcomings are addressed by means of a recommendation methodology that is specially designed to deal with gray sheep and sparsity problems without needing user attributes, content data and explicit ratings from users. Thus, one of the main difficulties of the proposals for dealing with the gray sheep problem is avoided since most of these approaches involve some content-based tech-

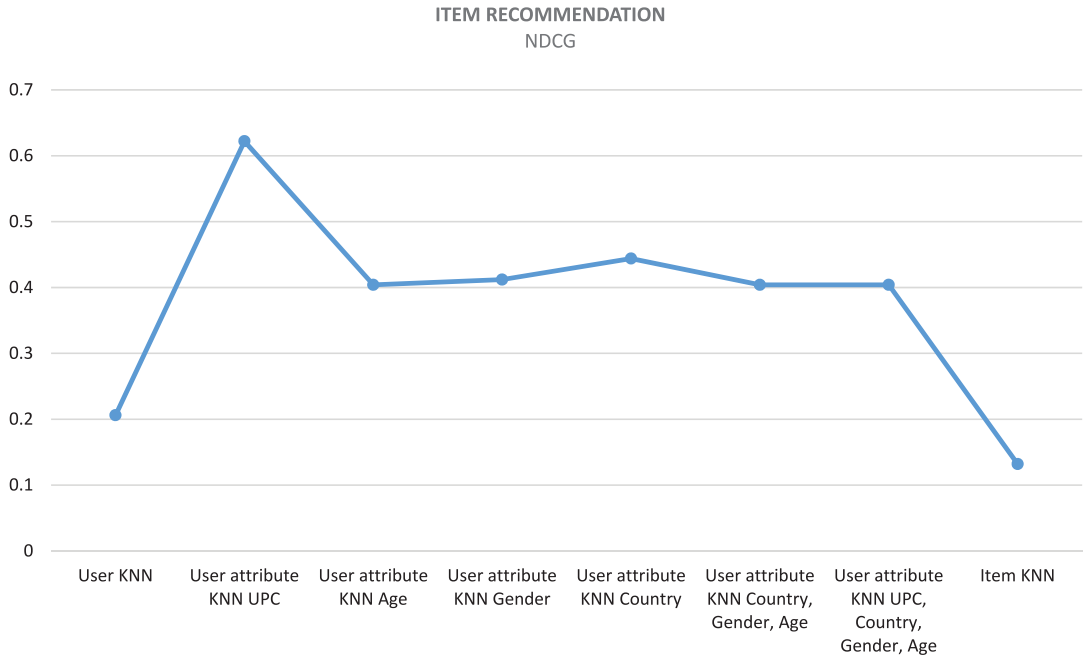

Fig. 10. Values of normalized discounted cumulative gain for item recommendation.

Fig. 11. Values of the area under de ROC curve for item recommendation.

niques that often requires a complex extraction task of music features.

The procedure proposed in this work makes use of artist and user playing coefficients to determine the degree to which a user is a gray sheep. The ratings needed to apply CF algorithms are derived from the counts of plays. The validation of the proposal was performed by means of a study where several CF algorithms were tested for both rating prediction and item recommendation. The results showed that our method significantly outperforms the other CF approaches.

Although the scalability problem has not been addressed, in future work we intend to tackle it by searching for a reliable method of segmenting the users. Then CF methods can be separately applied to different groups generated from the training set in an attempt to provide recommendations in a more efficient way, since the searching for neighbors at recommender time would be performed in a more reduced space. Although several clustering techniques have been used in recommender systems for obtaining groups of user with similar preferences, we will try to address the problem from a different perspective, which is mainly focused on

the characterization of users according to their gray sheep degree. Thus, the playing coefficient will be one of the attributes involved in the generations of the clusters. In addition, given that one of the objectives of our work is not to separate sharply the gray sheep users from other users, our intention is to create fuzzy clusters, so that a user can belong to more than one cluster with different degrees of membership.

We are also working in the adaptation of the proposed method for recommending songs instead of artists. In this case, the degree to which a given user is a gray sheep will depend on the popularity of the songs that this user plays, thus new coefficients for songs and users must be defined. The next step in this line is to apply the recommendation procedure for generating personalized playlists. This is a more complex task where it is necessary to consider additional features such as song order and diversity. In this context, we intend to address the improvement of the procedure for obtaining implicit ratings taking into account not only the number of plays of every song but also the sessions of the users and the position of the songs in these sessions. This information is easily obtainable since most of the datasets from music platforms include the time when the users play each song in the platform.

References

Billsus, D., & Pazzani, M. (1999). A hybrid user model for news story classification. In *Proceedings of international conference on user modeling* (pp. 99–108).

Breese, J. S., Heckerman, D., & Kadie, C. (1998). *Technical report Microsoft research*. Morgan Kaufmann Publishers.

Cantador, I., Bellogín, A., & Castells, P. (2008). A multilayer ontology-based hybrid recommendation model. *AI Communications, 21*, 203–210.

Cataltepe, Z., & Altinel, B. (2007). Music recommendation based on adaptive feature and user grouping. In *22nd international symposium on computer and information sciences* (pp. 1–6).

Chen, H. C., & Chen, A. L. P. (2005). A music recommendation system based on music and user grouping. *Intelligent Information Systems, 24*(2/3), 113–132.

Claypool, M., Gokhale, A., Mir, T., Murnikov, P., Netes, D., & Sartin, M. (1999). Combining content-based and collaborative filters in an online newspaper. In *Proceedings of ACM SIGIR workshop on recommender systems*. ACM.

Deng, S., Wang, D., Li, X., & Xu, G. (2015). Exploring user emotion in microblogs for music recommendation. *Expert Systems with Applications, 42*(23), 9284–9293.

Ekstrand, M. D., Riedl, J. T., & Konstan, J. A. (2010). collaborative filtering recommender systems. *Foundations and Trends in Human–Computer Interaction, 4*(2), 81–173.

Ghazanfar, M. A., & Prügel-Bennett, A. (2014). Leveraging clustering approaches to solve the gray-sheep users problem in recommender systems. *Expert Systems with Applications, 41*, 3261–3275 (2014).

Ghorbani, H., & Novin, A. H. (2016). An introduction on separating gray-sheep users in personalized recommender systems using clustering solution. *International Journal of Computer Science and Software Engineering (IJCSSE), 5*(2), 14–18.

Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS), 22*(1), 5–53.

Hyung, Z., Lee, K., & Lee, K. (2014). Music recommendation using text analysis on song requests to radio stations. *Expert Systems with Applications, 41*(5), 2608–2618.

Jarvelin, K., & Kekalainen, J. (2002). Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems, 20*(4), 422–446.

Kim, H. N., Alkhaldi, A., El Saddik, A, & Jo, G. S. (2011). Collaborative user modeling with user-generated tags for social recommender systems. *Expert Systems with Applications, 38*(2011), 8488–8496.

Krulwich, B., & Burkey, C. (1996). Learning user information interests through extraction of semantically significant phrases. In *Proceedings of AAAI spring symposium on machine learning in information access* (pp. 110–112).

Kuo, F. F., & Shan, M. K. (2002). A personalized music filtering system based on melody style classification. In *Proceedings of the IEEE international conference on data mining* (pp. 649–652).

Lee, C. H., Kim, Y. H., & Rhee, P. K. (2001). Web personalization expert with combining collaborative filtering and association rule mining technique. *Expert Systems with Applications, 21*(2001), 131–137.

Lee, K., & Lee, K. (2015). Escaping your comfort zone: A graph-based recommender system for finding novel recommendations among relevant items. *Expert Systems with Applications, 42*(2015), 4851–4858.

Lu, C. C., & Tseng, V. S. (2009). A novel method for personalized music recommendation. *Expert Systems with Applications, 36*, 10035–10044.

Lucas, J. P., Laurent, A., Moreno, M. N., & Teisseire, M. (2012). A fuzzy associative classification approach for recommender systems. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 20*, 579–617.

Lucas, J. P., Luz, N., Moreno, M. N., Anacleto, R., Almeida, A., & Martins, C. (2013). A hybrid recommendation approach for a tourism system. *Expert Systems with Applications, 40*(9), 3532–3550.

Melville, P., Mooney, R. J., & Nagarajan, R. (2002). Content boosted collaborative filtering for improved recommendations. In Proceedings of the 18th national conference on artificial intelligence,*AAAI '02* (pp. 187–192).

Moreno, M. N., Segrera, S., López, V. F., & Muñoz, M. D. (2016). Web Mining based Framework for solving usual problems in recommender systems. A case study for movies' recommendation. *Neurocomputing, 176*, 72–80 (2016).

Pacula, M. (2009). *A matrix factorization algorithm for music recommendation using implicit user feedback* http://www.mpacula.com/publications/lastfm.pdf).

Resnick, P., Iacovou, N., Suchack, M., Bergstrom, P., & Riedl, J. (1994). Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of ACM CSW'94 conference on computer supported cooperative work* (pp. 175–186).

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithm. In *Proceedings of the tenth international World Wide Web conference* (pp. 285–295).

Schafer, J. B., Konstant, J. A., & Riedl, J. (2001). E-commerce recommendation applications. *Data Mining and Knowledge Discovery, 5*, 115–153.

Shepitsen, A., Gemmell, J., Mobasher, B., & Burke, R. (2008). Personalized recommendation in social tagging systems using hierarchical clustering. In *Proceedings of the 2008 ACM conference on recommender systems RecSys '08* (pp. 259–266). ACM.

Su, X., Greiner, R., Khoshgoftaar, T. M., & Zhu, X. (2007). Hybrid collaborative filtering algorithms using a mixture of experts. In Proceedings of the IEEE/WIC/ACM international conference on web intelligence *(WI '07)* (pp. 645–649). Silicon Valley.

Su, X., & Khoshgoftaar, T. M. (2009). A Survey of Collaborative Filtering Techniques. *Advances in Artificial Intelligence, 2009*, 1–19 Article ID 421425.

Tzanetakis, G. (2002). *Manipulation, analysis and retrieval systems for audio signals PhD thesis*.

Vargas, S., & Castells, P. (2011). Rank and relevance in novelty and diversity metrics for recommender systems. In Proceedings of the fifth ACM conference on recommender systems *RecSys '11* (pp. 109–116). ACM.

Yoshii, K., Goto, M., Komatani, K., Ogata, T., & Okuno, H. G. (2006). Hybrid collaborative and content-based music recommendation using probabilistic model with latent user preferences. In *Proceedings of the 7th international conference on music information retrieval*.

Yu, K., Schwaighofer, A., Tresp, V., Xu, X. &, & Kriegel, H. P. (2004). Probabilistic memory-based collaborative filtering. *IEEE Transactions on Knowledge and Data Engineering, 16*(1), 56–69.