Available online at www.sciencedirect.com
SCIENCE DIRECT

Engineering Applications of Artificial Intelligence 18 (2005) 781–789

Engineering Applications of
ARTIFICIAL INTELLIGENCE

www.elsevier.com/locate/engappai

Qualitative analysis of user-based and item-based prediction algorithms for recommendation agents[☆]

Manos Papagelis[a,b,*], Dimitris Plexousakis[a,b]

[a]Institute of Computer Science, Foundation for Research and Technology—Hellas, P.O. Box 1385, GR-71110 Heraklion, Greece
[b]Department of Computer Science, University of Crete, P.O. Box 2208, GR-71409 Heraklion, Greece

Received 30 May 2005; accepted 5 June 2005

Abstract

Recommendation agents employ prediction algorithms to provide users with items that match their interests. In this paper, several prediction algorithms are described and evaluated, some of which are novel in that they combine *user-based* and *item-based* similarity measures derived from either *explicit* or *implicit* ratings. Both *statistical* and *decision-support* accuracy metrics of the algorithms are compared against different levels of data sparsity and different operational thresholds. The first metric evaluates the accuracy in terms of average absolute deviation, while the second evaluates how effectively predictions help users to select high-quality items. The experimental results indicate better performance of item-based predictions derived from explicit ratings in relation to both metrics. Category-boosted predictions lead to slightly better predictions when combined with explicit ratings, while implicit ratings, in the context that have been defined in this paper, perform much worse than explicit ratings.
© 2005 Elsevier Ltd. All rights reserved.

Keywords: Recommendation algorithms; Collaborative filtering; Similarity measures

1. Introduction

Recommendation systems (Resnick and Varian, 1997) have been a popular topic of research ever since the ubiquity of the web made it clear that people of hugely varying backgrounds would be able to access and query the same underlying data. The initial human–computer interaction challenge has been made even more challenging by the observation that customized services require sophisticated data structures and well thought-out architectures to be able to scale up to thousands of users and beyond.

In recent years, recommendation agents are extensively adopted by both research and e-commerce recommendation systems in order to provide an intelligent mechanism to filter out the excess of information available and to provide customers with the prospect to effortlessly find out items that they will probably like according to their logged history of prior transactions.

1.1. Background

Recommendation agents need to employ efficient prediction algorithms so as to provide accurate recommendations to users. If a *prediction* is defined as a value that expresses the predicted likelihood that a user will “like” an item, then a *recommendation* is defined as the list of *n* items with respect to the top-*n* predictions from

[☆]This paper is part of the special issue of selected best papers of the 9th international workshop on cooperative information agents (CIA 2004) organised by Matthias Klusch, Rainer Unland, and Sascha Ossowski.

[*]Corresponding author. Institute of Computer Science, Foundation for Research and Technology—Hellas, P.O. Box 1385, GR-71110, Heraklion, Greece.

E-mail addresses: papaggel@csd.uoc.gr, papaggel@ics.forth (M. Papagelis), dp@ics.forth.gr (D. Plexousakis).

0952-1976/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.engappai.2005.06.010

the set of items available. Improved prediction algorithms indicate better recommendations. This explains the essentiality of exploring and understanding the broad characteristics and potentials of prediction algorithms and the reason why this work concentrates on this research direction.

1.1.1. Approaches of recommendation algorithms

There are generally two methods to formulate recommendations both depending on the type of items to be recommended, as well as, on the way that user models (Allen, 1990) are constructed. The two different approaches are content-based (Balabanovic and Sholam, 1997; Kalles et al., 2003) and collaborative filtering (Herlocker et al., 2000; Hofmann, 2003), while additional hybrid techniques have been proposed as well (Balabanovic and Sholam, 1997).

Content based recommendation algorithms: Content-based algorithms are principally used when documents are to be recommended, such as web pages, publications, jokes or news. The agent maintains information about user preferences either by initial input about user's interests during the registration process or by rating documents. Recommendations are formed by taking into account the content of documents and by filtering in the ones that better match the user's preferences and logged profile.

Collaborative filtering based recommendation algorithms: Collaborative-filtering algorithms aim to identify users that have relevant interests and preferences by calculating similarities and dissimilarities between user profiles (Herlocker et al., 2004). The idea behind this method is that, it may be of benefit to one's search for information to consult the behavior of other users who share the same or relevant interests and whose opinion can be trusted.

1.1.2. Challenges of recommendation agents

The challenges for recommendation algorithms expand to three key dimensions, identified as *sparsity*, *scalability* and *cold-start*.

Sparsity: Even users that are very active, result in rating just a few of the total number of items available in a database. As the majority of the recommendation algorithms are based on similarity measures computed over the *co-rated* set of items, large levels of sparsity can be detrimental to recommendation agents. In Huang et al. (2004), authors propose to deal with sparsity problem by applying an associative retrieval framework and related spreading activation algorithms to explore transitive associations among consumers through their past transactions and feedback.

Scalability: Recommendation algorithms seem to be efficient in filtering in items that are interesting to users. However, they require computations that are very expensive and grow non-linearly with the number of users and items in a database. Therefore, in order to bring recommendation algorithms successfully on the web, and succeed in providing recommendations with acceptable delay, sophisticated data structures and advanced, scalable architectures are required. In Cosley et al. (2002), authors describe an open framework for practical testing of recommendation systems in an attempt to provide a standard, public testbed to evaluate recommendation algorithms in real-world conditions.

Cold-start: An item cannot be recommended unless it has been rated by a substantial number of users. This problem applies to new and obscure items and is particularly detrimental to users with eclectic taste (Schein et al., 2002; Melville et al., 2002). Likewise, a new user has to rate a sufficient number of items before the recommendation algorithm be able to provide reliable and accurate recommendations.

1.2. Contributions

The primary contributions of this work are:

- The utilization of explicit ratings in an "implicit" sense so as to enrich a user's model, without actually prompting users to express their preference to categories.
- The description of item-based and user-based similarity measures derived from either explicit or implicit ratings.
- The formation of a range of item-based and user-based prediction algorithms according to item-based and user-based similarity measures.
- The qualitative analysis and experimental evaluation of presented prediction algorithms.

1.3. Organization

Section 2 describes a set of similarity measures to compare the relevance between users or items. Section 3 describes a set of existing and newly introduced prediction algorithms that integrate the similarity measures. Section 4 presents the experimental evaluation metrics that are employed in order to compare the algorithms and the results of the evaluation are discussed. Section 5 summarizes the contributions of this work and draws directions for further research.

2. Similarity measures

In this section, a set of similarity measures are presented based on the Pearson correlation coefficient, a metric of relevance between two vectors (Pearson, 1900). When the values of these vectors are associated with a user's model then the similarity is called *user-based similarity*, whereas when they are associated with an item's model then it is called *item-based similarity*. The similarity measure can be effectively used to balance the ratings significance in a prediction algorithm and therefore to improve accuracy.

There are several similarity algorithms that have been used: cosine vector similarity, Pearson correlation, Spearman correlation, entropy-based uncertainty measure and mean-squared difference. Breese et al. (1998) suggest that Pearson correlation performs better than cosine vector similarity, while Herlock et al. (1999) suggest that Pearson's correlation performs better than Spearman's correlation, entropy-based uncertainty and mean-squared difference for collaborative filtering. According to these remarks Pearson correlation is selected to compute item-based and user-based similarities taking advantage of both explicit and implicit ratings.

An explicit rating identifies the preference of a user to a specific item. An user is prompted by the agent's interface to provide ratings for items so as to improve user's model. The more ratings the user provides, the more accurate the recommendations provided are. Ratings range from 1 to 10 with 1 expressing greatest aversion to the item and 10 expressing greatest liking to the item. Explicit ratings are logged by the system and are employed to construct the user's model.

An implicit rating (Nichols, 1997; Kleinberg et al., 2001) identifies the preference of a user to specific categories.[1] The term "implicit" is used here somewhat excessively, so as to express that a user is never actually prompted to express a degree of preference to categories. Taking advantage of the fact that an item belongs to a number of categories, it is possible to develop a user model based on category preferences. If the explicit rating of a user to a specific item that belongs to a set of categories is considered "good" then user's model is updated so as to include the preference and vice versa. A rating is considered as "good" when it is greater than or equal to a threshold.

Before describing the algorithms the following definitions are introduced to facilitate the explanation process:

- A set of m users $U = \{u_x : x = 1, 2, \ldots, m\}$;
- A set of n items $I = \{i_x : x = 1, 2, \ldots, n\}$;
- A set of p categories $C = \{c_x : x = 1, 2, \ldots, p\}$;
- A set of q explicit ratings $R = \{r_x : x : 1, 2, ..., q \wedge q \leqslant m^* n\}$;
- A set of t implicit ratings $R = \{r'_x : x = 1, 2, ..., t \wedge t \leqslant m^* p\}$;
- The explicit rating of a user u_x with reference to an item i_h as r_{u_x, i_h};
- The average explicit rating of a user u_x as $\overline{r_{u_x}}$.

In the sequence, three matrices are defined that derive from user's rating activity: the user-item matrix, the user-category matrix and the item-category bitmap matrix.

- *User-item matrix* is a matrix of users against items that have as elements the explicit ratings of users to items. Some of the user-matrix cells are not filled, as there are items that are not rated by any user.
- *User-category matrix* is a matrix of users against item categories that have as elements, values that show the number of times a user has rated positively or negatively for a category. For each category two columns are kept, one for positive ratings and one for negative ratings.
- *Item-category bitmap matrix* is a matrix of items against categories that have as elements the value 1 if the item belongs to the specific category and the value 0 otherwise.

Similarity is computed over the parts of the two vectors that derive from one of these matrices, as it is depicted by the shadowed parts of Fig. 1.

2.1. User-based similarity

2.1.1. Based on explicit ratings

If the set of items that users u_x and u_y have co-rated is defined as $I' = \{i_x : x = 1, 2, ..., n' \wedge n' \leqslant n\}$, where n is the total number of items in the database, then the similarity between two users is defined as the Pearson correlation coefficient of their associated rows in the user-item matrix and is given by

$$\kappa_{x,y} = \mathrm{sim}(u_x, u_y) = \frac{\sum_{h=1}^{n'} \left(r_{u_x,i_h} - \overline{r_{u_x}}\right) - \left(r_{u_y,i_h} - \overline{r_{u_y}}\right)}{\sqrt{\sum_{h=1}^{n'} \left(r_{u_x,i_h} - \overline{r_{u_x}}\right)^2} \sqrt{\sum_{h=1}^{n'} \left(r_{u_y,i_h} - \overline{r_{u_y}}\right)^2}}. \quad (1)$$

2.1.2. Based on implicit ratings

Whenever an explicit rating is submitted by a user for a specific item, the respective values of the user category matrix elements are incremented to include the new rating. Thus, it is possible to infer the preference of a user $u_x \in U$ to the category $c_x \in C$ by the user-category matrix. This preference, which is considered as an implicit rating $r'_{u_x,c_x} \in R'$ to that category is computed as $r'_{u_x,c_x} = \left(c_{x_{\mathrm{pos}}} / c_{x_{\mathrm{pos}}} + c_{x_{\mathrm{neg}}}\right) * 10$, where $c_{x_{\mathrm{pos}}}$, $c_{x_{\mathrm{neg}}}$ are, respectively, the number of positive and negative ratings[2] that user u_x has implicitly given to category x. Implicit ratings range from 1 to 10, with 1 expressing greatest aversion to the category and 10 expressing greatest liking to the category. The similarity between

[1]Items in the database belong to categories.

[2]Ratings are considered as positive or negative when they are greater or lower than a threshold, respectively.

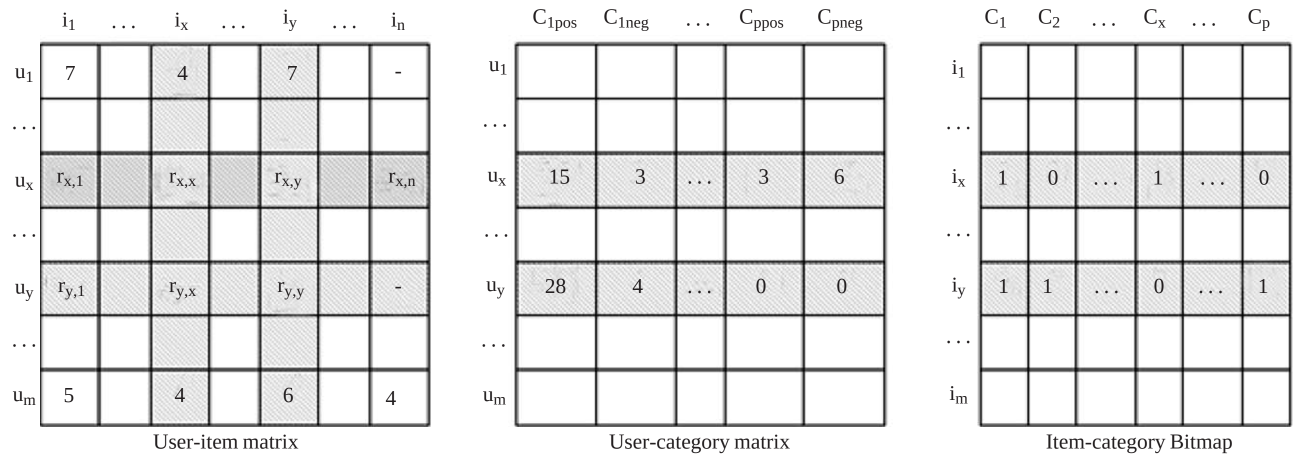

Fig. 1. User similarities and Item similarities derive from one of the user-item matrix, the user-category matrix or the item-category matrix by applying vector similarity measures.

the two users is defined as the Pearson correlation coefficient of their implicit ratings to all categories $c \in C$ and is given by Eq. (2), where p is the number of available categories.

$$\lambda_{x,y} = \text{sim}(u_x, u_y) = \frac{\sum_{h=1}^{p} \left(r'_{u_x,c_h} - \overline{r'_{u_x}}\right) - \left(r'_{u_y,c_h} - \overline{r'_{u_y}}\right)}{\sqrt{\sum_{h=1}^{p} \left(r'_{u_x,c_h} - \overline{r'_{u_x}}\right)^2} \sqrt{\sum_{h=1}^{p} \left(r'_{u_y,c_h} - \overline{r'_{u_y}}\right)^2}} \quad (2)$$

2.2. *Item-based similarity*

2.2.1. *Based on explicit ratings*

If the set of users that have rated both items i_x and i_y is defined as $U' = \{u_x : x = 1, 2, ..., m' \wedge m' \leqslant m\}$, where m is the total number of users in database, then the similarity between two items is defined as the Pearson correlation coefficient of their associated columns in the user-item matrix and is given by

$$\mu_{x,y} = \text{sim}(i_x, i_y) = \frac{\sum_{h=1}^{m'} \left(r_{u_h,i_x} - \overline{r_{i_x}}\right) - \left(r_{u_h,i_y} - \overline{r_{i_y}}\right)}{\sqrt{\sum_{h=1}^{m'} \left(r_{u_h,i_x} - \overline{r_{i_x}}\right)^2} \sqrt{\sum_{h=1}^{m'} \left(r_{u_h,i_y} - \overline{r_{i_y}}\right)^2}}. \quad (3)$$

2.2.2. *Based on item-category bitmap*

It is also possible to compute the correlation between two items by taking into account the categories in which they belong. In this case, the similarity between two items is defined as the Pearson correlation coefficient of their associated rows in the item-category bitmap matrix and is given by Eq. (4), where p is the number of categories and v_{c_h,i_x} is a Boolean value that equals to 1 if the item x belongs to the category h or equals to 0 otherwise

$$v_{x,y} = \text{sim}(i_x, i_y) = \frac{\sum_{h=1}^{p} \left(v_{c_h,i_x} - \overline{v_{i_x}}\right) - \left(v_{c_h,i_y} - \overline{v_{i_y}}\right)}{\sqrt{\sum_{h=1}^{p} \left(v_{u_h,i_x} - \overline{v_{i_x}}\right)^2} \sqrt{\sum_{h=1}^{p} \left(v_{u_h,i_y} - \overline{v_{i_y}}\right)^2}}. \quad (4)$$

3. Prediction algorithms

Prediction algorithms (Breese et al., 1998) try to guess the rating that a user is going to provide for an item. This user will be referred as *active user* u_a and this item as *active item* i_a. These algorithms take advantage of the logged history of ratings and of content associated with users and items in order to provide predictions.

3.1. *Random prediction algorithms*

The random prediction algorithm represents the worst case of prediction algorithm,[3] since instead of applying a sophisticated technique to produce a prediction it generates a random one. The random prediction algorithm serves as a reference point that helps to define how much better results are obtained by the utilization of more sophisticated algorithms.

3.2. *User-based prediction algorithms description*

User-based prediction algorithms are based on user's average rating and an adjustment to it, as given by

$$\text{prediction} = \text{user\_average} + \text{adjustment}. \quad (5)$$

[3]Actually, this is not absolutely true. Worse prediction algorithms than the random-based one can be artificially produced but in order to do this some kind of logged information is needed.

The adjustment is most often a weighted sum that integrates user-based or item-based similarity measures. Since prediction arises as the sum of the two, improvements can be considered in both operators. Next, the classic user-based collaborative filtering prediction algorithm is presented and some improvements are suggested that take advantage of the different user-based and item-based similarity measures described in the earlier section.

3.2.1. *User-based with explicit ratings* ($\mathrm{CF_{UB-ER}}$)

This prediction algorithm represents the classic user-based collaborative filtering prediction algorithm and comes up as the sum of the active user's average rating, regarding the whole set of items that the active user has rated, and an adjustment. The adjustment is a weighted sum of the other users' ratings concerning the active item and their similarity with the active user. The prediction algorithm is given by Eq. (6), where m' is the number of users that have rated the item i_a and $\bar{r}_{u_a}$is the user's average rating over the set of items that the active user has rated

$$\mathrm{CF_{UB-ER}} = p_{u_a,i_a} = \bar{r}_{u_a} + \frac{\sum_{h=1}^{m'} \kappa_{a,h}(r_{u_h,i_a} - \bar{r}_{u_h})}{\sum_{h=1}^{m'} |\kappa_{a,h}|}. \quad (6)$$

3.2.2. *User-based with explicit ratings and category boosted* ($\mathrm{CF_{UB-ER-CB}}$)

Instead of computing the active user's average rating over the total number of rated items, it may be preferable to take into account the active user's average rating over the subset of rated items that belong to the same categories as the active item. This seems reasonable, since user's ratings may be higher for specific item categories and lower for others. The prediction algorithm is given by Eq. (7), where m' is the number of users that have rated the active item i_a and $\bar{r}_{u_a}$ is user's average rating over the set of items that have been rated by the active user and belong to at least one of the categories that active item i_a belongs to

$$\mathrm{CF_{UB-ER-CB}} = p_{u_a,i_a} = \bar{r}_{u_a} + \frac{\sum_{h=1}^{m'} \kappa_{a,h}(r_{u_h,i_a} - \bar{r}_{u_h})}{\sum_{h=1}^{m'} |\kappa_{a,h}|}. \quad (7)$$

3.2.3. *User-based with implicit ratings* ($\mathrm{CF_{UB-IR}}$)

Instead of using the user-based explicit ratings similarity κ, it is possible to use the user-based implicit ratings similarity λ in order to compute the similarity between the active user and the other users. The prediction algorithm is given by Eq. (8), where m' is the number of users that have rated the item i_a and $\bar{r}_{u_a}$ is the user's average rating over the set of items that the active user has rated.

$$\mathrm{CF_{UB-IR}} = p_{u_a,i_a} = \bar{r}_{u_a} + \frac{\sum_{h=1}^{m'} \lambda_{a,h}(r_{u_h,i_a} - \bar{r}_{u_h})}{\sum_{h=1}^{m'} |\lambda_{a,h}|}. \quad (8)$$

3.3. *Item-based prediction algorithms description*

Item-based prediction algorithms refer to algorithms that are based on item's average rating and an adjustment to it, as given by

$$\text{prediction} = \text{item\_average} + \text{adjustment}. \quad (9)$$

The adjustment is most often a weighted sum that integrates user-based or item-based similarity measures. Since prediction arises as the sum of the two, improvements can be considered in both operators. Next, two item-based algorithms are suggested; an item-based collaborative filtering prediction algorithm based on explicit ratings and an item-based collaborative filtering prediction algorithm based on implicit ratings. Both cases employ the item-based similarity measures described in the earlier section.

3.3.1. *Item-based with explicit ratings*

This algorithm can be considered as the reverse of the classic user-based collaborative filtering. First, the item's average rating is computed and then an adjustment is added. The item-based collaborative filtering prediction algorithm comes up as the sum of the active item's average rating, regarding the whole set of users that have rated it, and an adjustment. The adjustment is a weighted sum of the ratings that the active user has given to other items and their similarity with the active item. The prediction algorithm is given by Eq. (10), where n' is the number of items that the active user u_a has rated and $\bar{r}_{i_a}$ is the item's average rating based on all the ratings that have been submitted for it

$$\mathrm{CF_{IB-ER}} = p_{u_a,i_a} = \bar{r}_{i_a} + \frac{\sum_{h=1}^{n'} \mu_{a,h}(r_{u_a,i_h} - \bar{r}_{u_a})}{\sum_{h=1}^{n'} |\mu_{a,h}|}. \quad (10)$$

3.3.2. *Item-based with implicit ratings*

Instead of using the item-based explicit ratings similarity μ, it is possible to use the item-based implicit ratings similarity v in order to compute the similarity between the active item and the other items. The prediction algorithm is given by Eq. (11), where n' is the number of items that the active user u_a has rated and $\bar{r}_{i_a}$ is the item's average rating based on all the ratings that have been submitted for it

$$\mathrm{CF}_{\mathrm{IB-IR}} = p_{u_\mathrm{a},i_\mathrm{a}} = \bar{r}_{i_\mathrm{a}} + \frac{\sum_{h=1}^{n'} v_{\mathrm{a},h}(r_{u_\mathrm{a},i_h} - \bar{r}_{u_\mathrm{a}})}{\sum_{h=1}^{n'} |v_{\mathrm{a},h}|}. \tag{11}$$

4. Experimental evaluation and results

4.1. Data set

The experimental data comes from an in-house movie recommendation system named Movie Recommendation System (MRS). The MRS database currently consists of 2068 ratings provided by 114 users to 641 movies, which belong to at least 1 of 21 categories. Therefore the lowest level of sparsity for the tests is defined as $114 \times 641 - 2068/114 \times 641 \simeq 0.9717$. The prediction algorithms are tested over a pre-selected 300-ratings set extracted randomly by the set of 2068 actual ratings. The interested user is strongly encouraged to visit the website of the system and obtain a more detailed view.

4.2. Metrics

Coverage and *accuracy* are two key dimensions on which the quality of a prediction algorithm is usually evaluated. The metrics that are employed to evaluate coverage and accuracy are discussed below.

4.2.1. Coverage metric

Coverage is a measure of the percentage of items for which a recommendation agent can provide predictions. A basic coverage metric is the percentage of items for which predictions are available. Coverage can be reduced by defining small neighborhood sizes or by sampling users to compute predictions. All experimental results demonstrated in this paper had coverage slightly less than perfect for typical level of sparsity (Coverage: 99%, Sparsity: 97.17%, in these experiments). Obviously a prediction cannot be computed in case that the active user has zero correlations with other users.

4.2.2. Accuracy metrics

Several metrics have been proposed for assessing the accuracy of collaborative filtering methods. They are divided into two main categories: *statistical accuracy metrics* and *decision-support accuracy metrics*.

Statistical accuracy metrics: Statistical accuracy metrics evaluate the accuracy of a prediction algorithm by comparing the numerical deviation of the predicted ratings from the respective actual user ratings. Some of them frequently used are *mean absolute error (MAE)*, *root mean squared error (RMSE)* and *correlation* between ratings and predictions (Herlocker et al., 1999). All of the above metrics were computed on result data and generally provided the same conclusions.

As statistical accuracy measure, mean absolute error (MAE) (Hofmann, 2003) is employed. Formally, if n is the number of actual ratings in an item set, then MAE is defined as the average absolute difference between the n pairs $<p_h, r_h>$ of predicted ratings p_h and the actual ratings r_h and is given by

$$\mathrm{MAE} = \frac{\sum_{h=1}^{n} |p_h - r_h|}{n}. \tag{12}$$

The lower the MAE, the more accurate the predictions would be, allowing for better recommendations to be formulated. MAE has been computed for different prediction algorithms and for different levels of sparsity. Table 1 provides values for the MAE of the different prediction algorithms presented, while Fig. 2 illustrates the sensitivity of the algorithms in relation to the different levels of sparsity applied.

As far as statistical accuracy is concerned, the following outcomes about the quality performance of the prediction algorithms are reached:

- Performance of item-based prediction algorithms is of superior quality than user-based prediction algorithms.
- Performance of implicit rating based algorithms, in the sense that they have been defined for these tests, is of inferior quality than explicit rating based algorithms.
- Item-based algorithm, based on explicit ratings ($\mathrm{CF}_{\mathrm{IB\text{-}ER}}$) seems to be very sensitive to sparsity levels. As sparsity reduces, the MAE of the algorithm decreases, which means that prediction accuracy is increased. $\mathrm{CF}_{\mathrm{IB\text{-}ER}}$ performs as much as 39.5% better than classic Collaborative Filtering prediction algorithm, $\mathrm{CF}_{\mathrm{UB\text{-}ER}}$, for Sparsity levels close to 97.2%.
- $\mathrm{CF}_{\mathrm{UB\text{-}ER\text{-}CB}}$ increases the accuracy precision of $\mathrm{CF}_{\mathrm{UB\text{-}ER}}$, as it calculates the user average based only on the subset of items that belong to the same categories as the active item. However, this increment is insignificant if taking into consideration the extra computation needed to include the category information.

Experimental results indicate that item-based algorithms provide more accurate recommendations than user-based algorithms. In particular, $\mathrm{CF}_{\mathrm{IB\text{-}ER}}$ behaves much better as data becomes more dense (i.e. sparsity level decreases) in comparison to all other algorithms presented. A prospective recommendation agent would provide predictions with a mean absolute error lower than 1 grade (i.e. 0.838). In MRS recommendation system, ratings range from 1 to 10, while in other common systems (e.g. GroupLens, EachMovie dataset)

Table 1
Statistical accuracy of the different prediction algorithms in terms of mean absolute error (MAE) with respect to different Sparsity levels

Prediction algorithms	Sparsity levels						
	0.972	0.975	0.98	0.985	0.99	0.995	0.999
$CF_{UB\text{-}ER}$	1.385	1.457	1.541	1.637	1.801	1.746	1.865
$CF_{UB\text{-}ER\text{-}CB}$	1.34	1.412	1.518	1.606	1.807	1.667	1.771
$CF_{UB\text{-}IR}$	1.703	1.739	1.796	1.781	1.755	1.863	2.147
$CF_{IB\text{-}ER}$	0.838	0.91	1.06	1.14	1.28	1.626	1.66
$CF_{IB\text{-}IR}$	1.35	1.38	1.445	1.45	1.521	1.804	1.665
Random	3.166	3.515	3.414	3.024	3.256	3.174	3.398

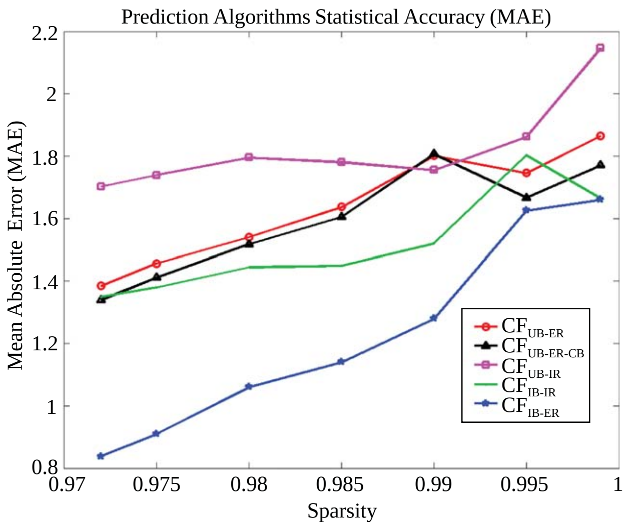

Fig. 2. Statistical accuracy of the different prediction algorithms in terms of mean absolute error (MAE) with respect to different sparsity levels.

ratings range from 1 to 5. In order to obtain a clear comparative view of presented MAE results, one needs to divide the results with a factor of 2. This consideration leads to MAE of 0.419 in the best case, which is particularly satisfactory for providing high-quality recommendations.

Decision-support accuracy metrics: Decision-support accuracy metrics evaluate how effectively predictions help a user to select high-quality items. Some of them frequently used are *reversal rate*, *weighted errors*, *precision-recall curve (PRC) sensitivity* and *receiver operating characteristic (ROC) sensitivity* (Sarwar et al., 1998). They are based on the observation that, for many users, filtering is a binary process. Consequently, prediction algorithms can be treated as a filtering procedure, which distinguishes "good" items from "bad" items.

As decision support accuracy measure, ROC sensitivity is employed. ROC sensitivity is a measure of the diagnostic power of a filtering system. Operationally, it is the area under the receiver operating characteristic (ROC) curve-a curve that plots the sensitivity and the 1-specificity of the test. Sensitivity refers to the probability of a randomly selected "good" item being accepted by the filter. Specificity is the probability of a randomly selected "bad" item being rejected by the filter. The ROC curve plots sensitivity (from 0 to 1) and 1—specificity (from 0 to 1), obtaining a set of points by varying the quality threshold. The ROC sensitivity range between 0 and 1, where 0.5 is random and 1 is perfect.

If PR denotes the predicted rating, AR denotes the actual rating and the quality threshold as QT, then the following possible cases are defined by the filter for one item:

- True Positive (TP) when $PR \geqslant QT \wedge AR \geqslant QT$;
- False Positive (FP) when $PR \geqslant QT \wedge AR < QT$;
- True Negative (TN) when $PR < QT \wedge AR < QT$;
- False Negative (FN) when $PR < QT \wedge AR \geqslant QT$.

For a set of items sensitivity is defined as the true positive fraction (TPF) and the 1-specificity as the false positive fraction (FPF) where

- $\text{sensitivity} = \text{TPF} = \text{tp}/(\text{tp} + \text{fn})$, where tp, fn is the number of the true positive and the false negative occurrences over the set of items, respectively.
- $1 - \text{specificity} = \text{FPF} = \text{fp}/(\text{fp} + \text{tn})$, where tn, fp is the number of the true negative and the false positive occurrences over the set of items, respectively.

ROC curve has been computed for different prediction algorithms and for quality thresholds ranging between 1 and 9, while the sparsity level was equal to 0.972. Notation of the form ROC-threshold defines the discrete points on the ROC curve for the specific quality threshold value. The area under the curve represents how much sensitive the prediction algorithm is, so the more area it covers the better for the prediction algorithm. Fig. 3 illustrates the sensitivity of the different prediction algorithms, while Table 2 provides specific values for the ROC-6, ROC-7, ROC-8 and ROC-9 curves of the different prediction algorithms presented, which are of greatest interest. We consider

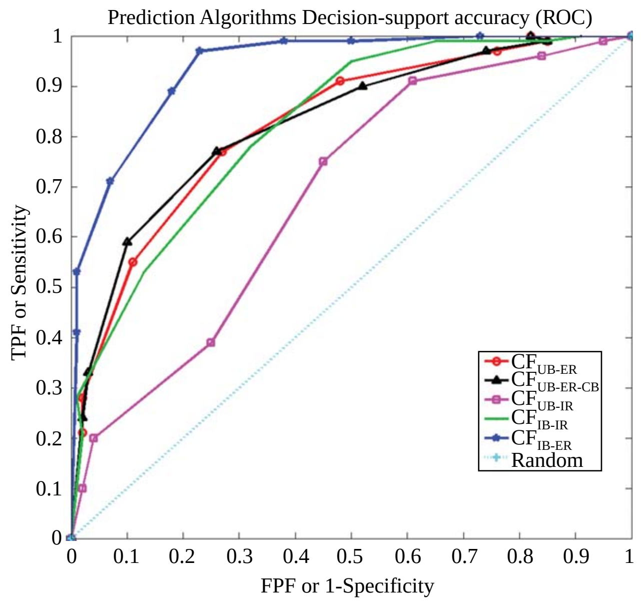

Fig. 3. Decision support accuracy of the different prediction algorithms in terms of receiver operating curve (ROC) with respect to different sparsity levels.

Table 2
Decision support accuracy of the different prediction algorithms in terms of receiver operating curve (ROC) with respect to different Sparsity levels

TPF or sensitivity	Quality threshold			
Prediction algorithms	ROC-6	ROC-7	ROC-8	ROC-9
$CF_{UB\text{-}ER}$	0.77	0.55	0.28	0.21
$CF_{UB\text{-}ER\text{-}CB}$	0.77	0.59	0.33	0.24
$CF_{UB\text{-}IR}$	0.75	0.39	0.2	0.1
$CF_{IB\text{-}ER}$	0.89	0.71	0.53	0.41
$CF_{IB\text{-}IR}$	0.78	0.53	0.28	0.21

these specific points in ROC curve of greatest interest, because typically an item is considered as "good" if its average rating is over 6, 7, 8, or 9 in a 1–10 rating scale.

The following remarks can be made about the quality of the prediction algorithms as far as decision-support accuracy is concerned.

- Performance of item-based prediction algorithms is of superior quality than user-based prediction algorithms.
- Performance of implicit rating based algorithms, in the context that have been defined in this paper, is of inferior quality than explicit rating based algorithms.
- $CF_{IB\text{-}ER}$ performs 95% better than classic collaborative filtering, $CF_{UB\text{-}ER}$, for ROC-9, 89% better for ROC-8 and 29% better for ROC-7. This means that if as "good" items are defined the ones that have average rating more than 9 or 8 or 7, respectively, and as "bad" items the ones that have average rating less than 9 or 8 or 7, respectively, then $CF_{IB\text{-}ER}$ predicts and therefore recommends items with 95% or 89% or 29%, respectively, more accuracy than classic collaborative filtering $CF_{UB\text{-}ER}$.
- To obtain a clear view of the overall performance of each algorithm one needs to compute the area under the ROC curve. It is clear from Fig. 3 that $CF_{IB\text{-}ER}$ performs much better than every other algorithm examined.

5. Conclusions and future work

The vast volume of information flowing on the web has given rise to the need for information filtering techniques. Recommendation agents are effectively used to filter out excess information and to provide personalized services to users by employing sophisticated, well thought-out prediction algorithms. This work described how explicit ratings can be utilized in order to implicitly obtain user's preference to specific categories. A number of prediction algorithms have been designed and implemented, based on either user or item similarity and have been thoroughly evaluated according to their statistical and decision-support accuracy performance. Experimental analysis showed that the performance of item-based prediction algorithms is of superior quality than user-based prediction algorithms. Category-boosted algorithms can lead to slightly better quality when combined with explicit ratings, while performance of prediction algorithms based on implicit ratings is of inferior quality than ones based on explicit ratings.

Directions for future research include incremental updates of the similarity measures and exploration of caching schemes for effectively bringing recommendation algorithms on the Web and dealing with the scalability problem, use of prediction algorithms to identify virtual, dynamically created communities based on user models' similarity, and identification of trust inferences between users, in a social network context, to alleviate the Sparsity problem of recommendation algorithms.

References

Allen, R.B., 1990. User models: theory, method and Practice. International Journal of Man–Machine Studies.

Balabanovic, M., Sholam, Y., 1997. Combining content-based and collaborative recommendation. Communications of the ACM 40 (3).

Breese, J.S., Heckerman, D., Kadie, C., 1998. Empirical analysis of predictive algorithms for collaborative filtering. Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence.

Cosley, D., Lawrence, S., Pennock, D. M., 2002. REFEREE: an open framework for practical testing of recommender systems using ResearchIndex. Proceedings of the 28th Very Large Data Bases Conference.

Herlocker, J.L., Konstan, J.A., Borchers, A., Riedl, J., 1999. An algorithmic framework for performing collaborative filtering. Proceedings of the 22nd ACM SIGIR Conference on Research and Development in Information Retrieval.

Herlocker, J.L., Konstan, J.A., Riedl, J., 2000. Explaining collaborative filtering recommendations. Proceedings of the ACM Conference on Computer Supported Cooperative Work.

Herlocker, J., Konstan, J.A., Terveen, L., Riedl, J., 2004. Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems (TOIS) 22 (1).

Hofmann, T., 2003. Collaborative filtering via Gaussian probabilistic latent semantic analysis. Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.

Huang, Z., Chen, H., Zeng, D., 2004. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. ACM Transactions on Information Systems 22 (1).

Kalles, D., Papagelis, A., Zaroliagis, C., 2003. Algorithmic aspects of web intelligent systems. In: Zhong, N., Liu J., Yao,Y. (Eds.), Web Intelligence. Springer, Berlin, pp. 323–345.

Kleinberg, J., Papadimitriou, C.H., Raghavan, P., 2001. On the value of private information. Proceedings of 8th Conference on Theoretical Aspects of Reasoning about Knowledge.

Melville, P., Mooney, R.J., Nagaragan, R., 2002. Content-boosted collaborative filtering for improved recommendations. Proceedings of the National Conference of the American Association Artificial Intelligence.

Nichols, D., 1997. Implicit rating and filtering. Proceedings of 5th DELOS Workshop on Filtering and Collaborative Filtering, pp. 31–36.

Pearson, K., 1900. Mathematical contribution to the theory of evolution: VII, on the correlation of characters not quantitatively measurable. Philosophical Transactions of the Royal Society of London A 195, 1–47.

Resnick, P., Varian Hal, R., 1997. Recommender systems (introduction to special section). Communications of the ACM 40 (3).

Sarwar B.M., Konstan, J.A., Borchers, A., Herlocker, J., Miller, B., Riedl, J., 1998. Using filtering agents to improve prediction quality in the groupLens research collaborative filtering system. Proceedings of the ACM Conference on Computer Supported Cooperative Work.

Schein, A.I., Popescul, A., Ungar, L.H., 2002. Methods and metrics for cold-start recommendations. Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.