On the Enhancement of Collaborative Filtering by Demographic Data

Article *in* Web Intelligence and Agent Systems · January 2006

Source: DBLP

CITATIONS
46

READS
202

2 authors:

Manolis G. Vozalis
University of Macedonia
32 PUBLICATIONS **546** CITATIONS

SEE PROFILE

Konstantinos G. Margaritis
University of Macedonia
412 PUBLICATIONS **2,005** CITATIONS

SEE PROFILE

Web Intelligence and Agent Systems: An international journal 4 (2006) 117–138
IOS Press

On the enhancement of collaborative filtering by demographic data

Manolis Vozalis* and Konstantinos G. Margaritis
Parallel Distributed Processing Laboratory, Department of Applied Informatics, University of Macedonia, Egnatia 156, PO Box 1591, 54006, Thessaloniki, Greece

Abstract. Demographic data regarding users and items exist in most available recommender systems data sets. Still, there has been limited research involving such data. This work sets the foundations for a novel filtering technique which relies on information of that kind. It starts by providing a general, step-by-step description of an approach which combines demographic information with existing filtering algorithms, via a weighted sum, in order to generate more accurate predictions. U-Demog and I-Demog are presented as an application of that general approach specifically on User-based and Item-based Collaborative Filtering. Several experiments involving different settings of the proposed approach support its utility and prove that it shows enough promise in generating predictions of improved quality.

Keywords: Collaborative filtering, memory-based filtering, demographic data, personalization, prediction, recommender systems

1. Introduction

Recommender Systems were proposed as a computer-based intelligent technique whose purpose was to assist with the problem of information and product overload. By providing efficient personalized solutions in various e-business domains, they benefit both the customer and the retailer.

Two basic entities are featured in all Recommender Systems: the *user* and the *item*, where m is the number of users, $U = \{u_1, u_2, \ldots, u_m\}$ and n is the number of items, $I = \{i_1, i_2, \ldots, i_n\}$. A user who wishes to take advantage of the Recommender System should provide his opinion about a variety of items. The *goal* of the Recommender System would be to generate suggestions about new items for that particular user. The process is based on a *filtering algorithm*, applied on some kind of *input*, which, in the majority of cases, consists of ratings on past items. An alternative kind of input is the demographic data regarding the user in mind. *Demographic data*, which will be the focal part of this paper, refers to information such as the age, the gender and the education of the user. This kind of data is usually difficult to obtain and is normally collected directly from the user.

Recommender systems are usually distinguished as belonging to one of two wide categories: *Memory-based* or *Model-based Systems*. Memory-based Systems are more efficient as a result of their producing recommendations without a need for any preprocessing. Still, they suffer from serious scalability problems. User-based Collaborative Filtering [11,20, 26] generates its predictions by constructing neighborhoods as collections of similar users. Content-based Filtering draws its main ideas from the Information Retrieval/Information Filtering literature [21] and generates its predictions by analyzing the content related to items in mind. Both techniques are examples of Memory-based filtering. A different approach is taken by Model-based Systems [3,4]. The algorithms used in this approach develop a model of user ratings in order to produce their predictions. The construction of that model requires time, but once created, it speeds up considerably the generation of the recommendations. Model-based algorithms often approach the filtering process from a probabilistic perspective [6,14]. A recent representative of this class, utilized in this paper, is

*Corresponding author. E-mail: mans@uom.gr.

Item-based Collaborative Filtering [9,23]. This method bases its predictions on the construction of neighborhoods of similar items.

Demographic Recommender Systems rely on user or item attributes, classified as demographic data, in order to produce their recommendations. Their functionality is sometimes enhanced by the utilization of pregenerated demographic clusters. Demographic Generalization, the approach developed by Krulwich [17] and applied in his Lifestyle Finder, suggested items and web pages to users, after assigning them to the appropriate clusters. The classification was made possible with the help of a commercially available database of demographic data and the input provided by the user, which constituted his profile. Pazzani [19] attempted to find regularities among users, by applying on them the Winnow algorithm, an algorithm originally designed for text classification. The user profiles utilized for similarity calculations had the form of user demographic vectors.

Hybrid systems [5] combine different filtering techniques in order to produce improved recommendations. Fab [1] utilizes collaborative filtering techniques to compare user profiles, previously generated by content analysis. Content-Boosted Collaborative Filtering [18] improves on user-based collaborative filtering predictions by enhancing the initial matrix of ratings through the application of a content-based predictor. P-Tango [7] combines different filtering methods, by first relating each of them to a distinct component and then basing its predictions on the *weighted average* of the predictions generated by those components. GroupLens introduces the use of filterbots [10,22,24], which are automated rating agents that evaluate new items based on the rating algorithm that they carry. Smyth and Cotter implement their PTV system [27] as a mixed hybrid, by presenting together recommendations, generated by a collaborative filtering and a content-based component. Basu, Hirsh and Cohen present Ripper [2] as an inductive learning system that learns rules by utilizing both content and collaborative features. Condliff et al. propose a two stage mixed-effects Bayesian Recommender [8], which first fits a Bayes classifier to each user, utilizing item features, and then links those classifiers across different users through a regression model.

In this paper, we present the Demog algorithm, a novel filtering approach which can utilize any common filtering algorithm as its base and extend it by exploring the usefulness of demographic data as an enhancing factor. According to the Burke's taxonomy [5], our algorithm can be classified as a *Feature combination* hybrid, since features from different recommendation data sources are blended into a single recommendation algorithm. The base filtering algorithms are User-based and Item-based Collaborative Filtering. The user and item neighborhoods, as generated by these algorithms, are further refined, by the help of demographic correlations, before final predictions are produced.

This paper can be outlined as follows. Section 2 provides information about the utilized data set and the evaluation metrics, two issues closely related to the success of the implemented experiments. Section 3 presents an overview of the two filtering algorithms involved in our experiments. Section 4 starts by describing the general structure of the Demog Algorithm. It does so in a formal, step-by-step manner. Section 4 proceeds by applying the Demog Algorithm on User-based and Item-based Collaborative Filtering. The utility of the resulting algorithms, U-Demog and I-Demog, is supported by the execution of several experiments and a presentation of their results. In Section 5, the proposed algorithms are compared with a couple of past demographic approaches. The paper is concluded in Section 6 where an overall evaluation of the Demographic algorithm is attempted, while pointers for future research are outlined. Three appendices accompany the main body of the text: Appendix A focuses on the structure of the MovieLens Data Set, Appendix B details the structure of the user and item demographic vectors, which are utilized by our approach, and Appendix C includes segments of pseudo-code, also providing the corresponding analysis of computational complexity.

2. Experimental methodology

2.1. MovieLens: A GroupLens data set

For the execution of our subsequent experiments we utilized the data publicly available from the GroupLens movie recommender system. The MovieLens data set, used by several researchers [12,25,28], consists of 100.000 ratings which were assigned by 943 users on 1682 movies. Users should have stated their opinions for at least 20 movies in order to be included. Ratings follow the 1(bad)-5(excellent) numerical scale. Starting from the initial data set, five distinct splits of training and test data were generated (*u1.base, u2.base, u3.base, u4.base, u5.base and u1.test, u2.test, u3.test, u4.test, u5.test*). For each data split, 80% of the original set was included in the training and 20% of it was

included in the test data. The test sets in all cases were disjoint.

The complete data set includes in random order 100.000 vectors of the following form:

```
user id | item id | rating | time stamp
```

Obviously, users are enumerated from 1 to 943, items from 1 to 1682, while `ratings` take values between 1 and 5. The time stamps are unix seconds since 1/1/1970 UTC. An actual sample from the GroupLens data set can be found in Appendix A.

Except for ratings awarded by users on items, the MovieLens data set includes information regarding specifically the users and the items. Such information proved to be crucial for the subsequent algorithmic methods. Regarding users, the included data consists of a sequential list, with 943 vectors of the following form:

```
user id | age | gender | occupation | zip code
```

The `user ids` are the ones also used in the main data file. The `gender` can be either 'M', for male, or 'F', for female. The `occupation` takes a value from a list of 21 distinct possibilities. An actual sample from the demographic information about the users, which are included in the MovieLens data set, can be found in Appendix A.

Finally, regarding the items, which in the case of the MovieLens data set correspond to movies, there is another sequential list, with 1682 vectors of the following form:

```
movie id | movie title | release date | video release date | IMDb URL | unknown | Action | Adventure | Animation | Children's | Comedy | Crime | Documentary | Drama | Fantasy | Film-Noir | Horror | Musical | Mystery | Romance | Sci-Fi | Thriller | War | Western
```

The `movie ids` are the ones used in the main data set. The `movie title` is a string with the title of the movie. The `release dates` are of the form *dd-mmm-yyyy*, eg. 14-Jan-1967. The `IMDb URL` is a web link leading to the Internet Movie Database page of the corresponding movie. The last 19 fields are the film genres. Items can belong to more than one genres at the same time. An actual sample from the item-related information which are part of the MovieLens data set, can be found in Appendix A.

Both user and item specific information, incorporated in the MovieLens data set, fall under the "demographic data" category. They will be used extensively, as part of the algorithmic approaches which will be proposed in the following sections.

2.2. The evaluation metric

Several techniques have been used to evaluate Recommender Systems. Those techniques have been divided by Herlocker et al. [12] into three categories. The first category includes *Predictive Accuracy Metrics*, such as mean absolute error, mean squared error and normalized mean absolute error. These metrics measure how close the recommender's predictions are to the true user ratings. The second category, *Classification Accuracy Metrics*, includes methods such as Receiver Operating Characteristic curves (ROC curves) and the F1 metric, and measures how often a recommender system can decide correctly whether an item is beneficial for the user and therefore should be suggested to him. Those metrics require a binary classification of items into useful and not useful. The last category of metrics, called *Rank Accuracy Metrics*, measure the proximity of a predicted ordering of items, as generated by a recommender system, to the actual user ordering of the same items.

The choice among those metrics should be based on the selected user tasks and the nature of the data sets. P-Tango [7] calculates its efficiency via the deviation of generated predictions from the user-specified ratings. As a result the selected evaluation metric is *inaccuracy*, an alternative name for mean absolute error. On the other hand, Breese et al. [4] want to evaluate a ranked list of recommended items and therefore they calculate the expected utility of that list to the user and use that amount as their metric. We wanted our proposed algorithms to derive a predicted score for already rated items rather than generate a top-N recommendation list. Based on that specific task we proceeded in the selection of the initial evaluation metric for our experiments. That metric was *Mean Absolute Error* (*MAE*) [26].

$$mae = \frac{\sum_{i=1}^{k} pr_i - r_i}{k} \quad (1)$$

It is a statistical accuracy metric which measures the deviation of predictions, pr_i, generated by the Recommender System, from the true rating values, r_i, as they were specified by the user, for k {*user, item*} pairs. MAE is measured only for those items, for which a user has expressed his opinion.

3. The base algorithms

In this section we will briefly discuss the two filtering algorithms which will be utilized by our proposed algorithm. These algorithms are User-based and Item-based Collaborative Filtering.

3.1. User-based collaborative filtering

User-based Collaborative Filtering methods are based on the assumption that people who agreed in their subjective evaluation of past items are likely to agree again in the future [20].

The execution steps of the algorithm are:

(a) *Data Representation* of the ratings provided by m users on n items. That is achieved by the construction of an $m \times n$ user-item matrix, R, with elements r_{ij} (where $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$) representing the ratings of user u_i on item i_j.

(b) *Neighborhood Formation*, which concludes by the construction of the active user's neighborhood. First, the similarities for each possible couple of users are calculated, by exclusively utilizing past ratings. The similarity computations are implemented by the use of the corresponding similarity metrics on those ratings. Possible similarity metrics include the *Pearson Correlation*:

$$sim_{ik} = corr_{ik} = \frac{\sum_j (r_{ij} - \bar{r_i})(r_{kj} - \bar{r_k})}{\sqrt{\sum_j (r_{ij} - \bar{r_i})^2 \sum_j^l (r_{kj} - \bar{r_k})^2}} \quad (2)$$

or the *Cosine/Vector Correlation*:

$$sim_{ik} = cos_{ik} = \sum_j \frac{r_{ij}}{\sqrt{\sum_j r_{ij}^2}} \frac{r_{kj}}{\sqrt{\sum_j r_{kj}^2}} \quad (3)$$

In both cases we are evaluating the proximity of users u_i and u_k (sim_{ik}), based on their opinions (r_{ij} and r_{kj}) on all items that they have rated in common (i_j). $\bar{r_i}$ and $\bar{r_k}$ correspond to the mean ratings of users u_i and u_k respectively. Experiments described in [4,13] have shown that Pearson Correlation yields more accurate predictions, and thus, it will be our metric of choice, when talking about User-based CF.

Once the similarities among all users are calculated and stored in a proximity matrix of dimensions $m \times m$, the algorithm proceeds and selects those users that appear to be most similar to the active user, u_a, i.e. the user for whom we wish to generate predictions. Those users will become the active user's neighborhood.

(c) *Prediction Generation*, where final predictions regarding the active user's, u_a, opinion on item i_j are calculated, as the average deviation of a neighbor's rating on that item, r_{ij}, from the same neighbor's mean rating $\bar{r_i}$, converted to fit the active user's rating distribution by adding it to the active user's mean rating, $\bar{r_a}$.

$$pr_{aj} = \bar{r_a} + \frac{\sum_i (r_{ij} - \bar{r_i}) * pears\_cor_{ai}}{\sum_i |pears\_cor_{ai}|} \quad (4)$$

$pears\_cor_{ai}$ represents the Pearson correlation between the active user, u_a, and all users, u_i, belonging to his neighborhood, as defined solely from past ratings.

3.2. Item-based collaborative filtering

Similarly to User-based Collaborative Filtering, Item-based Filtering is based on the creation of neighborhoods. Yet, unlike the User-based Collaborative Filtering approach, these neighborhoods consist of similar items rather than similar users [23].

The execution steps of the algorithm are:

(a) *Data Representation* of the ratings provided by m users on n items. Once again this step is based on the construction of an $m \times n$ user-item matrix, R, which includes the ratings, r_{ij}, of all users, u_i for $i = 1, 2, \ldots, m$, on all items, i_j for $j = 1, 2, \ldots, n$.

(b) *Neighborhood Formation*, which concludes by the construction of the active item's neighborhood. Similarities for all possible pairs of items existing in the data set are computed by the application of the selected similarity metrics. Except for Pearson Correlation (Eq. (2)) and Cosine/Vector Correlation (Eq. (3)), another metric preferred in the case of Item-based CF is *Adjusted Cosine Similarity*:

$$sim_{jk} = adjcorr_{jk} = \frac{\sum_i (r_{ij} - \bar{r_i})(r_{ik} - \bar{r_i})}{\sqrt{\sum_i (r_{ij} - \bar{r_i})^2 \sum_i (r_{ik} - \bar{r_i})^2}} \quad (5)$$

Similarly to User-based CF, the calculations regarding the proximity of items i_j and i_k (sim_{jk}) are based on the ratings (r_{ij} and r_{ik}) that those items have received by all users (u_i) who have expressed their opinion on both. $\bar{r_i}$ corresponds to the mean ratings of user u_i.

Among the proposed similarity metrics, past experiments [23] have shown that the accuracy achieved by utilizing the Adjusted Cosine Similarity is superior when compared to that of alternative approaches. Thus, we will prefer it in all future examples involving the use of Item-based CF.

The current step is concluded when items most similar to the active item, as judged by their calculated correlation value, are selected for its neighborhood. Active item, i_a, refers to the item for which predictions should be generated.

(c) *Prediction Generation*, where final predictions regarding the opinion of user u_a on item i_j are calculated as a weighted sum of ratings, r_{ak}, given by that user on all k items included in the active item's neighborhood.

$$pr_{aj} = \frac{\sum_k r_{ak} * adj\_cor_{jk}}{\sum_k |adj\_cor_{jk}|} \quad (6)$$

adj_cor_{jk} represents the Adjusted Cosine Similarity between the active item, i_j, and all items, i_k, belonging to its neighborhood, as defined solely from past ratings.

4. The demographic algorithm

We will present a hybrid algorithm that keeps the core ideas of two existing recommender systems and enhances them with relevant information extracted from demographic data. User-based and Item-based Collaborative Filtering will be our base filtering algorithms. In the approach taken by Pazzani [19], user profiles were expressed as vectors constructed solely from demographic data and similarities among those user profiles were calculated in order for final predictions to be generated. In other words, the only source of user data utilized for recommendations was the demographic information available for them, while the ratings awarded by the same users for past items were disregarded.

In our proposal, similarly to what is known from plain User-based and Item-based Collaborative Filtering, user and item correlations, based exclusively on past ratings, lead to the construction of neighborhoods. Still, before these neighborhoods should be utilized for the generation of final predictions, the correlations between neighborhood members and active users or items are re-evaluated, this time by also taking into account existing demographic correlations. An outline of our approach is the following:

- *Step 1*: Construct demographic vectors for the m users and n items that participate in the recommendation process. The information required for these vectors is usually collected explicitly and can be found in most collaborative filtering data sets, like MovieLens and EachMovie.
  As mentioned in Section 2.1, our data set of choice is MovieLens, which already includes user and item demographic data, and as a result there is no need for their collection and preprocessing.
- *Step 2*: Execute the first two steps of the selected filtering algorithm, that is Data Representation and Neighborhood Formation. Data Representation consists of the construction of an $m \times n$ user-item matrix, with elements r_{ij}, corresponding to the ratings of user u_i (for $i = 1, 2, \ldots, m$) on item i_j (for $j = 1, 2, \ldots, n$). Neighborhood Formation will utilize the selected correlation metrics on the elements of that matrix, in order to calculate the ratings-based correlations between users or items. Based on these correlations, it will generate a neighborhood of users/items most similar to the active user/item, ui_a. The ratings-based correlation between the active user/item and a random user/item, ui_i, will be described by rat_cor_{ai}.
- *Step 3*: Calculate the demographic correlation between the active user/item, ui_a, and each of the members of its neighborhood. Demographic correlation is defined by the similarity of the vectors which represent specific users or items, as expressed by the cosine of the angle between them.
  There is a number of possible vector similarity metrics discussed in the Information Retrieval literature [16], which we could have used instead. We selected the cosine measure not only because it appears to be one of the successful metrics [15], but also for compatibility reasons with past experiments we have executed, where it was also used for the comparison of vectors. Still, as future work, it would be interesting to test and compare the performance of a system utilizing alternative vector similarity metrics.
  If we assume that active user/item ui_a has a corresponding feature vector $\overrightarrow{ui_a}$ with k features values $ui_{a,j}$ ($j = 1, 2, \ldots k$), and a member of its neighborhood, ui_i, has a corresponding feature vector $\overrightarrow{ui_i}$ with features values $ui_{i,j}$ ($j = 1, 2, \ldots k$), then their demographic correlation, dem_cor_{ai}, can be calculated as follows:

$$\begin{aligned} dem\_cor_{ai} &= vect\_sim(\overrightarrow{ui_a}, \overrightarrow{ui_i}) \\ &= \frac{\overrightarrow{ui_a} \cdot \overrightarrow{ui_i}}{||\overrightarrow{ui_a}|| * ||\overrightarrow{ui_i}||} \end{aligned} \quad (7)$$

  The numerator includes the inner product of vectors $\overrightarrow{ui_a}$ and $\overrightarrow{ui_i}$, calculated by the use of the "$\cdot$" operator. $||\overrightarrow{ui_a}||$ and $||\overrightarrow{ui_i}||$ correspond to the lengths of vectors ui_a and ui_i. The vectorial notation of Eq. (7) is equivalent to the following expression:

$$dem\_cor_{ai} = \frac{\sum_j (ui_{aj}\ ui_{ij})}{\sqrt{\sum_j ui_{aj}^2\ \sum_j ui_{ij}^2}} \quad (8)$$

where the feature values, ui_{aj} and ui_{ij}, corresponding to user/item vectors ui_a and ui_i, have been utilized.

– *Step 4*: Calculate the **Enhanced Correlation**, enh_cor_{ai}, for every pair of the form $\{ui_a, ui_i\}$, where ui_a is the active user/item and ui_i is a member of its neighborhood. Enhanced Correlation can be thought as incorporating the contributions of the ratings-based correlation and the newly acquired demographic correlation. It can be calculated via a weighted sum of the following form:

$$enh\_cor_{ai} = \alpha * rat\_cor_{ai} + \beta * dem\_cor_{ai} + \gamma * (rat\_cor_{ai} * dem\_cor_{ai}) \quad (9)$$

where rat_cor_{ai} and dem_cor_{ai} represent the ratings-based and the demographic correlation between active user/item ui_a and neighborhood member ui_i, while α, β and γ are flags that define the participation of each of the three components.

– *Step 5*: Proceed with the final step of the recommendation procedure, which is Prediction Generation. Regardless of the selected filtering algorithm (User-based or Item-based Collaborative Filtering), which is reflected by the prediction generation formula to be applied, our demographic approach differs from the one set by the base algorithms.
Specifically, while in plain User or Item-based CF algorithms, prediction generation is a function of known *ratings*, r, assigned by users to items, and *rating-based correlations*, rat_cor, between couples of users or items:

$$pr\_plain = \mathbf{f}(r, rat\_cor) \quad (10)$$

in the prediction generation formula of our demographic algorithm, we replace the ratings-based correlation, rat_cor, by the enhanced correlation, enh_cor, which was calculated in the previous step. Thus, the general form of the prediction equation would be:

$$pr\_demog = \mathbf{f}(r, \mathbf{enh\_cor}) \quad (11)$$

Details about the actual prediction generation equations which are utilized by the specific implementations of our algorithm will be discussed in the subsequent sections.

In the following paragraphs we will describe how this general approach can be applied specifically in the cases of User-based and Item-based Collaborative Filtering, enhancing their predictions, and, depending on the various parameter settings, lead to more accurate recommendations.

4.1. U-Demog: Enhancing user-based collaborative filtering with demographic correlations

The subsequent paragraphs will detail how the general description of our demographic approach can be applied on User-based Collaborative Filtering. The execution steps of *U-Demog* can be outlined as follows:

– *Step 1*: Construct the demographic vectors for the m users who participate in the recommendation process.
To achieve that goal we need to take advantage of the user demographic information, incorporated in the MovieLens data set, and explained in Section 2.1 and Appendix A. In brief, this information includes the age, the gender, the profession, one possible selection from a set of 21 distinct occupations, and the zip code for each single user who provided his ratings. A user demographic vector, $usdemog$, was defined as a vector with 27 features. Its structure is explained in Table 1.
In Information and Text Retrieval it is common for users and documents to be represented with the help of vectors. Through the years, a great variety of weighting approaches regarding the vector features have been investigated or proposed [21]. Still, in the initial stages of our experiments with the Demog algorithm, which are described in this work, we have selected to utilize the simplest possible solution, that is a binary representation of the features, as explained in Table 1, and ignore possible weighting schemes. It would be interesting, though, to observe the effects of such weighting schemes and we plan to experiment with them in the near future.
Regarding the grouping we selected in order to encode the ages of the users, we based our decision on distinguishing between non-adults (*age* $\leqslant$*18*), young adults (*18* $<$ *age* $\leqslant$ *29*), prime-age adults (*29* $<$ *age* $\leqslant$ *49*) and mature adults (*age* $>$ *49*). Finally, in the current set of experiments we did not take into account the users' zip codes.
Sample user demographic vectors, resulting from the application of the rules of Table 1, on the MovieLens user demographic data, can be found in Appendix B.

Table 1
Structure of the User Demographic Vector, $usdemog$

feature#	feature contents	comments
1	age $\leqslant 18$	
2	$18 <$ age $\leqslant 29$	each user belongs to a single age grouping
3	$29 <$ age $\leqslant 49$	the corresponding slot takes value 1 (true)
4	age > 49	the rest of the features remain 0 (false)
5	male	the slot describing the user gender is 1
6	female	the other slot takes a value of 0
7–27	occupation	a single slot describing the user occupation is 1 the rest of the slots remain 0

– *Step 2*: Execute the first two steps of User-based Collaborative Filtering, that is Data Representation and Neighborhood Formation.
In the case of User-based Filtering, the conclusion of the Neighborhood Formation step should lead to the construction of the active user's neighborhood. This neighborhood includes the l users, u_i, for $i = 1, 2, \ldots, l$, who seem to be closer to the active user, u_a, according to the Pearson Correlation, $pears\_cor_{ai}$, calculated exclusively on their past ratings on common items.
– *Step 3*: Calculate the demographic correlation between the active user, u_a, and each of the members of its neighborhood.
Once the members of the active user's neighborhood are selected, and since each user's demographic vector is already known from Step 1, we can proceed and calculate the proximity between the active user, u_a, and the users u_i, for $i = 1, 2, \ldots, l$, belonging to his neighborhood, as it is defined by their registered demographic data. Their demographic correlation, dem_cor_{ai}, will be calculated by applying the vector similarity formula (Eq. (7)) on the corresponding demographic vectors.
– *Step 4*: Calculate the Enhanced Correlation, enh_cor_{ai}, for every pair of the form $\{u_a, u_i\}$, where u_a is the active user and u_i is a member of its neighborhood.
The Enhanced Correlation for User-based Collaborative Filtering, which unites the outcomes from Pearson Correlation and Demographic Correlation for each $\{u_a, u_i\}$ pair, can be calculated by the following formula:

$$enh\_cor_{u,ai} = \alpha * pears\_cor_{ai} + \beta * dem\_cor_{ai} + \gamma * (pears\_cor_{ai} * dem\_cor_{ai}) \quad (12)$$

where $pears\_cor_{ai}$ is the Pearson Correlation for users u_a and u_i, and dem_cor_{ai} is their demographic correlation. α, β and γ are flags that define the participation of each of the three components in the final result.
– *Step 5*: Proceed with the final step of the recommendation procedure, which is Prediction Generation.
Plain User-based Collaborative Filtering computes the predicted rating of user u_a on item i_j as described in Eq. (4).
In U-Demog, the modified formula used for prediction generation has the following form:

$$udem\_pr_{aj} = \bar{r_a} + \frac{\sum_i (r_{ij} - \bar{r_i}) * enh\_cor_{ai}}{\sum_i |enh\_cor_{ai}|} \quad (13)$$

The only difference from prediction generation, as executed in plain User-based Collaborative Filtering, is that the enhanced correlation, enh_cor_{ai}, between active user u_a and all his neighbors, u_i, which was defined in Step 4, has replaced simple Pearson Correlation, $pears\_cor_{ai}$, between them.

In the worst case, the computational complexity of U-Demog is equal to $O(m^2 n)$, owed to the user-user similarity calculations, which evaluate the correlation of m users, with the remaining $m - 1$ users, based on their ratings on n items. A detailed analysis of the algorithm's computational complexity, along with fragments of pseudo-code corresponding to each step of its execution, can be found in Appendix C.

4.1.1. Comparing 4 implementations of U-Demog with User-based collaborative filtering

In this section we report the results from applying distinct combinations of values to the enhanced correlation flags (α, β and γ) of formula 12, which corresponds to testing different ways of combining ratings-based and demographic correlations. The results of these experiments will reveal part of the utility of our proposed filtering approach. The basic characteristic of the four implementations we experimented with, as

Table 2
Brief Description of U-Demog Implementations

	flags	enhanced correlation
U-Demog1	$\alpha = 0, \beta = 0, \gamma = 1$	$enh\_cor = pears\_cor * dem\_cor$
U-Demog2	$\alpha = 1, \beta = 0, \gamma = 1$	$enh\_cor = pears\_cor + pears\_cor * dem\_cor$
U-Demog3	$\alpha = 1, \beta = 1, \gamma = 1$	$enh\_cor = pears\_cor + dem\_cor + pears\_cor * dem\_cor$
U-Demog4	$\alpha = 1, \beta = 1, \gamma = 0$	$enh\_cor = pears\_cor + dem\_cor$

expressed by the values of the flags and the corresponding enhanced correlation formulas, are summarized in Table 2.

Figure 1 compares the Mean Absolute Errors (MAE) obtained from the U-Demog implementations (*u-demog1&2&3&4*) for neighborhoods with varying sizes of 50–300 users and contrasts them with the accuracy values achieved by User-based Collaborative Filtering (ub) under the same parameter settings. We tested the behavior of the algorithms for a single threshold value, $it = 10$, meaning that only users who rated at least 10 common items with the active user could be included in his neighborhood. The results were averaged over all 5 data splits of the MovieLens data set.

Table 3 collects the best MAE values among the neighborhood sizes tested for each of the 4 U-Demog implementations we experimented with and for the User-based Collaborative Filtering approach.

A careful review of the accuracy figure and the best MAE values achieved by the tested implementations leads to the following conclusions:

- The error values from *U-Demog2*, *U-Demog3* and *U-Demog4* seem to practically coincide, with *U-Demog3* being slightly better overall. The worst accuracy of the tested methods belongs to *U-Demog1*, while User-based Collaborative Filtering displays MAE values which are really close to the best MAE values observed in our experiments, trailing slightly the results from *U-Demog2*, *U-Demog3* and *U-Demog4*.
- *U-Demog1* is the single U-Demog implementation which allows demographic correlation to assume such a decisive role in the prediction generation. Specifically, it is possible for users to be initially included in the neighborhood of the active user, based on their ratings-based correlation with him, and then to see their role in prediction generation diminish, because the value of the demographic correlation disagrees, implying that the same users are not highly correlated with the active user. In the extreme but not improbable case, neighbors who display no common demographic information with the active user will have their involvement in prediction generation completely cancelled out, which will lead to a considerable shrinking of the participating user neighborhood.
  The effect of possible contradictions between the ratings-based and demographic correlations for the same pair of users is reflected in the results, which reveal that the predictions generated by *U-Demog1* are poorer when compared to the other U-Demog implementations and also when compared to plain User-based Filtering. We can conclude that this quality loss is owed to a combination of two factors: a) the shrinking of active user's neighborhood, and b) the inability of demographic correlations to improve on recommendations when playing such a decisive role in prediction generation.
- The remaining U-Demog implementations (*U-Demog2*, *U-Demog3* and *U-Demog4*) assign to demographic correlation a role which can be described as complementary. While in the case of *U-Demog1*, demographic correlation could *reduce* or even cancel out the participation of a neighborhood user in the recommendation process, the demographic correlation in these implementations can merely *strengthen* the already established, via the pearson correlation, participation of a neighborhood user. When a member of the active user's neighborhood appears to have no demographic resemblance with the active user, his participation in the recommendation procedure remains the same, being expressed simply by the ratings-based correlation between the two. On the other hand, a neighbor, u_i, who is highly correlated to the active user, u_a, in terms of demographic information, will see his role in the recommendation procedure increase by a factor, which grows bigger depending on their calculated demographic correlation. The impact of that factor varies among the three U-Demog implementations. The average accuracy results of *U-Demog2*, *U-Demog3* and *U-Demog4* are so close that we cannot express a clear preference for any of the three alternatives.
  Conclusively, in these U-Demog implementations the size of the participating user neighborhood is not affected, as is in the case of *U-Demog1*, while at the same time users who display common de-

Table 3
Best MAE values of U-Demog implementations and user-based filtering

	ub	u-demog1	u-demog2	u-demog3	u-demog4
MAE	0,7668	0,7705	0,7663	0,7661	0,7662

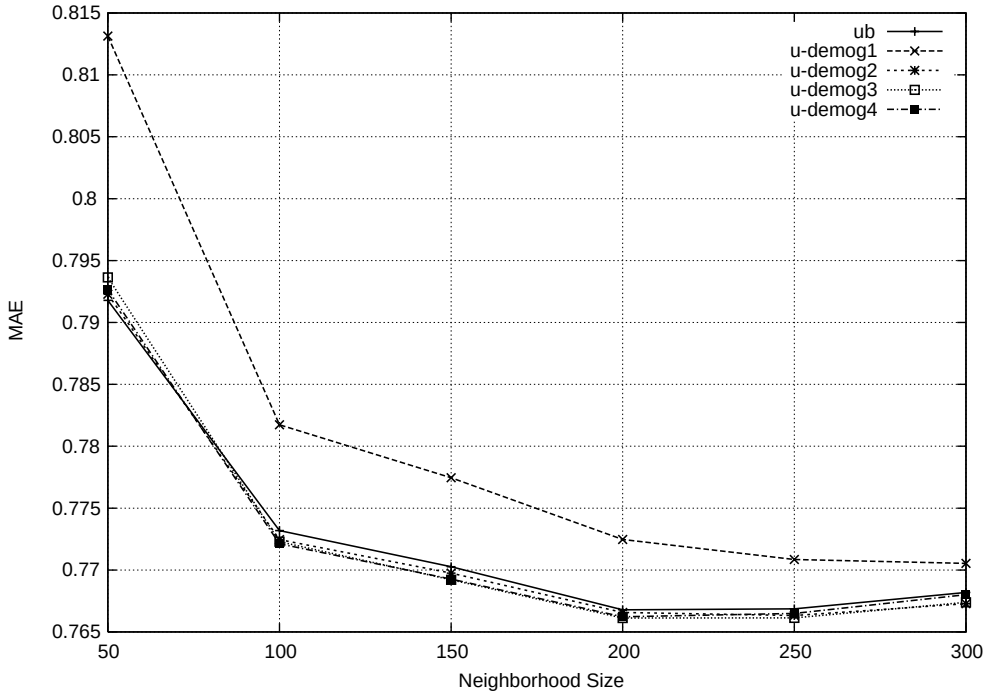

Fig. 1. U-Demog implementations: Average MAE for various neighborhood sizes.

mographic features with the active user do have an increased impact in the recommendation process.

– As documented by the performance comparison between U-Demog implementations and plain User-based Collaborative Filtering, the involvement of demographic information regarding users can possibly enhance the recommendation procedure, if only to a limited degree. Nevertheless, it is imperative for demographic correlations, as proven by the performance difference between *U-Demog1* and the rest of U-Demog implementations, to be assigned just an assisting role in the prediction generation process.

4.2. *I-Demog: Enhancing item-based collaborative filtering with demographic correlations*

Having completed our discussion on U-Demog, we will now proceed with a detailed description on how our demographic approach can be applied on Item-based Collaborative Filtering. The execution steps of *I-Demog* can be outlined as follows:

– *Step 1*: Construct the demographic vectors for the n items which participate in the recommendation process.
For this step we are taking into account the demographic information available from the MovieLens data set for items, which in our case corresponds to films rated by users. The MovieLens data set distinguishes 18 distinct film genres, ranging from Children's to Horror. This results to a vector of 19 features, $itdemog[19]$, if we utilize an additional slot, rightfully called "unknown", especially for films that cannot be categorized under any of the existing genres. It is important to point out that a film can belong to more than one genres at the same time. For example, it can be a Comedy, a Children's flick and a Musical. In that case, the slots which correspond to each of these categories should take a value of 1 (True), with the rest staying fixed at 0 (False).
Sample item demographic vectors can be found in Appendix B.

– *Step 2*: Execute the first two steps of Item-based Collaborative Filtering, that is Data Representation and Neighborhood Formation.
The Neighborhood Formation step of the Item-based Filtering algorithm concludes with the construction of the active item's neighborhood. This neighborhood includes the l items, i_k, with $k = 1, 2, \ldots, l$, which are most similar to the active item, i_j, according to the selected ratings-based correlation metric. The correlation metric of our choice is the *Adjusted Cosine Similarity*, which performs better than other proposed metrics, based on previously described experiments [23,29].
– *Step 3*: Calculate the demographic correlation between the active item, i_j, and each of the members of its neighborhood.
For the execution of this step we are required to isolate the demographic vectors only for those items included in the active item's neighborhood. We can now compute the correlation between the active item, i_j, and its neighborhood items, i_k, with $k = 1, 2, \ldots, l$. Their demographic correlation, dem_cor_{jk}, should be calculated by applying the vector similarity formula (Eq. (7)) on the corresponding vectors.
– *Step 4*: Calculate the Enhanced Correlation, enh_cor_{jk}, for every pair of the form $\{i_j, i_k\}$, where i_j is the active item and i_k is a member of its neighborhood.
The Enhanced Correlation for Item-based Collaborative Filtering, which unites the outcomes from Adjusted Cosine Similarity and Demographic Correlation for each $\{i_j, i_k\}$ pair, can be calculated by the following formula:

$$enh\_cor_{i,jk} = \alpha * adj\_cor_{jk} + \beta * dem\_cor_{jk} + \gamma * (adj\_cor_{jk} * dem\_cor_{jk}) \quad (14)$$

where adj_cor_{jk} is the Adjusted Cosine Similarity for items i_j and i_k, and dem_cor_{jk} is their demographic correlation. α, β and γ are flags that define the participation of each of the three components in the final result. In the sections that follow, we report the results from testing 4 distinct combinations of values for these flags, which correspond to different ways of combining ratings-based and demographic correlations.
– *Step 5*: Proceed with the final step of the recommendation procedure, which is Prediction Generation.
Plain Item-based Filtering computes the predicted rating of user u_a on item i_j as described in Eq. (6). Our approach in I-Demog modifies the prediction generation formula as follows:

$$idem\_pr_{aj} = \frac{\sum_k r_{ak} * enh\_cor_{jk}}{\sum_k |enh\_cor_{jk}|} \quad (15)$$

Clearly, the only difference between the two prediction generation formulas lies in the use of the enhanced correlation, enh_cor_{jk}, between the active item, i_j, and all its neighbors, i_k, instead of the ratings-based, adjusted cosine similarity, adj_cor_{jk}, which is utilized in the original algorithm. The enhanced correlation, as defined in Step 4, incorporates the contributions from both ratings-based and demographic correlations.

In the worst case, the computational complexity of I-Demog is equal to $O(mn^2)$, owed to the item-item similarity calculations, which evaluate the correlation of n items, with the remaining $n - 1$ items, based on their received ratings by m users. A detailed analysis of the algorithm's computational complexity, along with fragments of pseudo-code corresponding to each step of its execution, can be found in Appendix C.

4.2.1. Comparing 4 implementations of I-Demog with Item-based collaborative filtering

In the following paragraphs we will discuss 4 alternative I-Demog implementations. Each implementation tests different values for the flags (α, β and γ) of the enhanced correlation formula 14. By these experiments we intend to evaluate distinct combinations of the ratings-based and demographic correlations, while documenting the impact of the demographic correlations in the final predictions. The basic attributes of the implementations involved in our experiments, as expressed by the values of the flags and the corresponding enhanced correlation formulas, are summarized in Table 4.

Figure 2 compares the Mean Absolute Errors (MAE) collected from our I-Demog implementations (*i-demog1&2&3&4*) and Item-based Collaborative Filtering (ib), for neighborhoods with varying sizes of 20-140 items. In both cases, we tested the behavior of the algorithms for a single threshold value, *ut= 10*, meaning that at least 10 users should have rated the active item and a random item, in order for the latter to be included in the active item's neighborhood. The results reported here were averaged over all 5 data splits.

Table 5 collects the lowest MAE values, among all neighborhood sizes tested, for the 4 I-Demog imple-

Table 4
Brief Description of I-Demog Implementations

	flags	enhanced correlation
I-Demog1	$\alpha = 0, \beta = 0, \gamma = 1$	$enh\_cor = adj\_cor * dem\_cor$
I-Demog2	$\alpha = 1, \beta = 0, \gamma = 1$	$enh\_cor = adj\_cor + adj\_cor * dem\_cor$
I-Demog3	$\alpha = 1, \beta = 1, \gamma = 1$	$enh\_cor = adj\_cor + dem\_cor + adj\_cor * dem\_cor$
I-Demog4	$\alpha = 1, \beta = 1, \gamma = 0$	$enh\_cor = adj\_cor + dem\_cor$

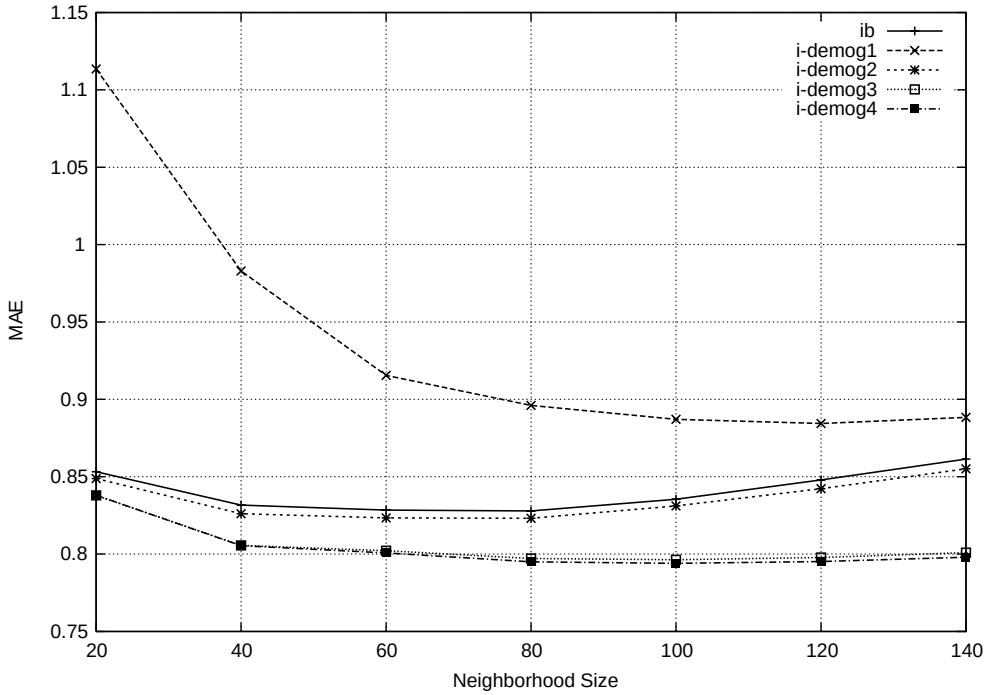

Fig. 2. I-Demog implementations: Average MAE for various neighborhood sizes.

mentations and for the Item-based Collaborative Filtering approach.

A careful look at Fig. 2 and the best accuracy values achieved by the tested methods leads us to the following conclusions:

– The error values from the tested approaches can be thought as belonging to 3 distinct clusters whose members display similar accuracy behavior. Specifically: The first cluster includes *I-Demog3* and *I-Demog4*, the two implementations with the best overall accuracy results. The second cluster includes *I-Demog2* and Item-based Filtering, with MAE values worse than those of the previous cluster. Within the cluster, *I-Demog2* displays a slightly better behavior when compared to Item-based Filtering. The final cluster includes only *I-Demog1*, which generated the worse accuracy results in our experiments.

– Similarly to what discussed for U-Demog, the poor performance of *I-Demog1* can be attributed to the decisive role assigned to demographic correlation by the specific approach. Still, the accuracy difference between *I-Demog1* and the remaining I-Demog implementations is wider than the corresponding difference in the case of U-Demog. Bearing these observations in mind, we have two important notes to make:

1. While in the case of user vectors, two individuals have a 50% chance of belonging to the same gender and therefore display a demographic correlation greater than 0, the situation regarding item vectors is different. Such demographic vectors, constructed by the distinct film genres that apply to a specific item, are more difficult to coincide, making it more common for the corresponding items to have a demographic correlation of 0. Based on the definition of en-

Table 5
Best MAE values of I-Demog implementations and item-based filtering

	ib	i-demog1	i-demog2	i-demog3	i-demog4
MAE	0,8279	0,8844	0,8231	0,7964	0,7939

hanced correlation for *I-Demog1*, we can easily conclude that demographic correlations equal to 0 will result to a considerable shrinking of the participating item neighborhood.

2. Our experiments with I-Demog implementations start with a neighborhood that includes 20 items. This neighborhood size is small and will become even smaller if we take into account all the couples of items with demographic correlation equal to 0. This leads to the poor performance of *I-Demog1* for small neighborhoods and the time it takes for demographic correlations to have an actual impact on the final predictions.

– The remaining I-Demog implementations (*I-Demog2*, *I-Demog3* and *I-Demog4*) assign to demographic correlation a complementary role, reducing its impact in the prediction generation process. The improvement in the accuracy results proves that such a role fits perfectly to demographic correlation. Furthermore, while the improvement of *I-Demog2* over Item-based Filtering is small, the MAE values achieved by *I-Demog3* and *I-Demog4* are significantly better than those of Item-based Filtering. Based on the experimental results for *I-Demog3* and *I-Demog4* we can attribute the performance improvement observed in both cases to the existence of the component $adj\_cor + dem\_cor$ in the enhanced correlation prediction formula.
– As documented by the performance comparison between I-Demog implementations and plain Item-based Collaborative Filtering, the involvement of demographic information about items in the recommendation process can possibly lead to a measurable accuracy improvement. Still, as proven by the implementations we experimented with, there should be a careful selection regarding the role of demographic correlation, which in all cases should merely complement ratings-based correlation.

5. Comparison with past demographic approaches

Having given a detailed description of our general filtering approach, and for two of its possible implementations on User and Item-based Collaborative Filtering, we are now able to provide a comparison of our approach to a couple of past Demographic Recommender Systems, which were mentioned in Section 1.

Krulwich describes *Lifestyle Finder* [17] as an intelligent agents system that interacts with users via the World Wide Web, constructs their profiles through a short series of predefined questions, and utilizes those profiles in order to recommend Web pages to them. The core of the system is a technique for intelligent user profiling, developed by Krulwich and his colleagues, and called *demographic generalization*. Demographic generalization uses Prizm, a commercially available, large-scale, database of demographic data. Prizm contains more than 600 variables, each referring to specific lifestyle characteristics, and utilizes them to divide the population into 62 predefined demographic clusters. Demographic generalization handles the answers provided by the users on the questions set by the system, as constraints on the values of a set of Prizm variables, and tries to match each user to a specific demographic cluster. If more than one clusters seem to match with the user's answers, it either keeps a partial profile, including parts of all the matching clusters, or singles out those variables that differentiate the matching clusters, and prompts the user for further information, in order to refine his profile. If the user data are more abstract than those included in the database, there is a need to compile the demographic data into a more abstract form, that would correspond to the form of the provided user answers. The evaluation of the system, as it was presented in the paper, was performed explicitly (the users were asked to tell if they liked the suggested URLs and if they considered the generated profile precise enough) and implicitly (by the percentage of sites, out of the total proposed, that the users actually visited).

The basic advantage of Lifestyle Finder is that it utilizes the large pool of available variables included in Prizm, and by a small number of carefully selected questions, it manages to locate one or just a few clusters that match the user most. It is important that the user is not required to pass through an extensive sequence of questions, and his profile can be built by the smallest possible contribution by him. On the other hand, the existence of 62 predefined Prizm demographic clusters is pretty limiting, as the resulting user profile usually takes the form of *one* of these clusters.

In their described form, a direct comparison between Lifestyle Finder and our system is not possible. Most significantly, the nature of the generated output is different. Furthermore, the applied evaluation methods,

the selection of which is forced by the type of the corresponding problems, do not lead to comparable results. Still, we can note the following: Similarly to Lifestyle Finder, our system utilizes a small number of demographic information about both the users (age, gender, occupation) and the items (film genre). Thus the burden for the demographic data collection placed on the user is similar. Additionally, we believe that our system has a general enough structure, which allows it to be combined with a new data set, including different demographic data – such a decision would require only specific changes, like the mapping of the new demographic data to new demographic vectors. We are not certain that such a database change would be equally feasible for Lifestyle Finder and Prizm. Added to that, the output, in the form of predictions, generated by our system, is not restricted to a predefined number of results, as in the case of Lifestyle Finder, but follows a continuous numerical scale between 1 and 5, seemingly capturing the user's profile in a more flexible manner.

Pazzani gives a framework for demographic filtering in [19]. In his approach, he uses pure demographic information, similar to those included in the GroupLens data set (gender, age, employed etc), to identify the types of users that like a specific object. For each user he constructs a distinct user demographic vector, based on the values of his demographic data, collected exclusively from the users' home pages on the World Wide Web. If a user has no home page, his demographic profile remains blank. Pazzani's approach of choice is to apply the Winnow algorithm on demographic vectors of users that belong to the training set, in order to classify users from the test set in a binary manner, as liking or not liking a previously unknown item. According to the Winnow algorithm, each user vector from the training set is treated as either a positive or a negative training example, with an initial weight assigned to each demographic feature. Through a training procedure, where those weights are adjusted based on the users' expressed opinions on known items, Winnow converges on a final set of weights, which are then used to classify unknown items. The system's performance was evaluated by the precision achieved by the top three recommendations, which were generated for each user - a selection partly owed to the binary nature of the system's output. The results which are reported, show that this demographic approach generates poorer recommendations compared to those of plain User-based Collaborative Filtering or Content-based Filtering, while the system's performance is improved when the demographic recommendations are combined with recommendations from multiple profiles (User-based CF, Item-based CF, Content-based and Collaboration via content).

Again, a direct comparison between Pazzani's demographic approach and our method is not possible in their discussed form, mainly because of the different scales utilized for recommendation generation. A problem existing in Pazzani's approach concerns the recommendations for users without a home page. He selects to put no burden on the users, by avoiding to require any explicit information from them, and by only using data existing in their home pages instead. This decision puts users without home pages in disadvantage. Yet, it is not difficult to change that part of Pazzani's method and choose an approach similar to the one applied by our algorithm, where a short list of demographic data are required for all users and items. Regarding the selection of a binary output, probably connected to the application of the Winnow algorithm, which in the past was used extensively in problems of text classification, the resulting user profiling can be characterized as more rigid, when contrasted to a continuous rating scale, like the one chosen for our approach. Finally, the results reported from the application of Pazzani's approach, show that the demographic data alone, hold weaker information regarding user profiling, leading to worse recommendations compared to those of the traditional techniques (User-based, Item-based filtering). In contrast, as proven by our experiments, an algorithm that uses demographic information as an enhancing factor to the traditional techniques, can improve the overall system performance.

6. Conclusions

In this work we have presented a unique filtering approach which draws ideas from existing algorithms and combines them with demographic information available in recommender systems data sets. That general filtering approach was used to demographically enhance User-based and Item-based Collaborative Filtering, leading to U-Demog and I-Demog, respectively. We tested several degrees of demographic data involvement in the recommendation process, expressed by varying parameter settings for U-Demog and I-Demog.

The experimental results supported the ability of our demographically enhanced approach to generate predictions which are more accurate than those of the base filtering algorithms. There were only two cases where our technique showed a performance degradation when compared to User-based or Item-based Filtering. In

these cases, demographic information did assume a major role in the recommendation process. In all other implementations, with demographic information assuming just a complementary role, the accuracy of our proposed approach displayed an improvement over the base filtering algorithms. The extent of this improvement was significant for specific parameter settings of the I-Demog algorithm.

These results prove that the demographic information existing in the MovieLens data set does not hold enough data about the users or the items in order to capture their distinguishing features and generate accurate predictions, when utilized just by themselves. Still, when combined appropriately with other forms of filtering, such as collaborative filtering, they can enhance the recommendation process and lead to improved predictions.

As a possible future extension of our approach we are thinking of experimenting with various weighting schemes involving the features of the demographic vectors. Such weighting schemes have proved to be particularly helpful in Information and Text Retrieval, and it would be interesting to examine if the same is true in our case. Similarly, we would like to move beyond the binary values for flags α, β and γ which define the role of ratings-based and demographic correlations in the enhanced correlation factor. Techniques involving different ways of assigning values to these flags, and thus further adjusting the involvement of ratings-based or demographic correlations in the filtering process, could be tested. We can also imagine a Recommender System which operates under the rules of natural selection, and involves the presence of numerous ratings-based and demographic filtering agents, each of them with distinct parameter settings. During the filtering procedure, strong agents, representing components whose recommendations are of high quality, will persevere, while weak agents will gradually evolve, in order to survive, or disappear. Finally, in order to tackle the problem of sparsity, one of the most persistent in filtering systems, we intend to test a couple of known matrix factorization techniques, such as Singular Value Decomposition, and evaluate how they can be combined with our method.

References

[1] M. Balabanovic and Y. Shoham, Fab: Content-based, collaborative recommendation, *Communications of the ACM* **40** (1997).

[2] C. Basu, H. Hirsh and W. Cohen, *Recommendation as classification: Using social and content-based information in recommendation*, In Proceedings of the 15th National Conference on Artificial Intelligence, Madison, WI, 1998.

[3] D. Billsus and M.J. Pazzani, *Learning collaborative information filters*, In 15th International Conference on Machine Learning, Madison, WI, 1998.

[4] J.S. Breese, D. Heckerman and C. Kadie, *Empirical analysis of predictive algorithms for collaborative filtering*, in Fourteenth Conference on Uncertainty in Artificial Intelligence, Madison, WI, 1998.

[5] R. Burke, Hybrid recommender systems: Survey and experiments, *User Modeling and User-Adapted Interaction* **12** (2002), 331–370.

[6] Y.-H. Chen and E.I. George, *A bayesian model for collaborative filtering*, In Proceedings of the Seventh International Workshop on Artificial Intelligence and Statistics, 1999.

[7] M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes and M. Sartin, *Combining content-based and collaborative filters in an online newspaper*, In ACM SIGIR Workshop on Recommender Systems- Implementation and Evaluation, Berkeley, CA, 1999.

[8] M.K. Condliff, D.D. Lewis, D. Madigan and C. Posse, *Bayesian mixed-effects models for recommender systems*, In ACM SIGIR '99 Workshop on Recommender Systems: Algorithms and Evaluation, Berkeley, CA, 1999.

[9] M. Deshpande and G. Karypis, Item-based top-n recommendation algorithms, *ACM Transactions on Information Systems* **22** (2004), 143–177.

[10] N. Good, J.B. Schafer, J.A. Konstan, A. Borchers, B. M. Sarwar, J. Herlocker and J.T. Riedl, *Combining collaborative filtering with personal agents for better recommendations*, In Conference of the American Association of Artifical Intelligence (AAAI-99), 1999, 439-446.

[11] J. Herlocker, J.A. Konstan, A. Borchers and J.T. Riedl, *An algorithmic frameworkd for performing collaborative filtering*, In The 1999 Conference on Research and Development in Information Retrieval, 1999.

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

[13] J.L. Herlocker, *Understanding and Improving Automated Collaborative Filtering Systems*, PhD thesis, University of Minnesota, 2000.

[14] T. Hofmann and J. Puzicha, *Latent class models for collaborative filtering*, In Proceedings of the International Joint Conference in Artificial Intelligence, 1999.

[15] W.P. Jones and G.W. Furnas, Pictures of relevance: a geometrical analysis of similarity measures, *Journal of the American Society for Information Science* **38** (1987), 420–442.

[16] Y. Jung, H. Park and D.Z. Du, *An effective term-weighting scheme for information retrieval*, Technical report, University of Minnesota, 2000.

[17] B. Krulwich, Lifestyle finder: Intelligent user profiling using large-scale demographic data, *Artificial Intelligence Magazine* **18** (1997), 37–45.

[18] P. Melville, R.J. Mooney and R. Nagarajan, *Content-boosted collaborative filtering*, In ACM SIGIR Workshop on Recommender Systems, New Orleans, LA, 2001.

[19] M.J. Pazzani, A framework for collaborative, content-based and demographic filtering, *Artificial Intelligence Review* **13** (1999), 393–408.

[20] P. Resnick, N. Iacovou, M. Sushak, P. Bergstrom and J. Riedl, *Grouplens: An open architecture for collaborative filtering of*

netnews, In ACM 1994 Conference on Computer Supported Cooperative Work, New York, NY, 1994, 175–186.

[21] G. Salton and C. Buckley, Term weighting approaches in automatic text retrieval, *Information Processing and Management* **24**(5) (1988), 513–523, 1988.

[22] B.M. Sarwar, *Sparsity, Scalability, and Distribution in Recommender Systems*, PhD thesis, University of Minnesota, 2001.

[23] B.M. Sarwar, G. Karypis, J.A. Konstan and J.T. Riedl, *Item-based collaborative filtering recommendation algorithms*, In 10th International World Wide Web Conference (WWW10), Hong Kong, 2001.

[24] B.M. Sarwar, J.A. Konstan, A. Borchers, J. Herlocker, B. Miller and J.T. Riedl, *Using filtering agents to improve prediction quality in the grouplens research collaborative filtering system*, In Conference on Computer Supported Cooperative Work, 1998.

[25] A.I. Schein, A. Popescul, L.H. Ungar and D.M. Pennock, *Methods and metrics for cold-start recommendations*, In ACM SIGIR-2002, Tampere, Finland, 2002.

[26] U. Shardanand and P. Maes, *Social information filtering: Algorithms for automating 'word of mouth'*, In Proceedings of Computer Human Interaction, 1995, 210–217.

[27] B. Smyth and P. Cotter, *Surfing the digital wave: Generation personalized tv listings using collaborative, case-based recommendation*, In Third International Conferece on Case-based Reasoning, Munich, Germany, 1999.

[28] S. Ujjin and P.J. Bentley, *Particle swarm optimization recommender system*, In Proceedings of the IEEE Swarm Intelligence Sympoisum 2003, Indianapolis, 2003.

[29] E.G. Vozalis and K.G. Margaritis, *Recommender systems: An experimental comparison of two filtering algorithms*, In Proceedings of the 9th Panhellenic Conference in Informatics – PCI 2003, 2003.

Appendix A: Examples from User, Item & Demographic Data

The MovieLens Data Set

The complete GroupLens Data Set includes 100.000 vectors of the following form, in random order:

`user id | item id | rating | time stamp`

A sample from the MovieLens Data file follows:

Listing 1: Sample from the MovieLens data file

```
...
194   181   3   879521396
125   478   4   879454628
110   688   1   886987605
299    14   4   877877775
151    10   5   879524921
269   127   4   891446165
  6    14   5   883599249
 54   106   3   880937882
303    69   5   879467542
 16   944   1   877727122
301   790   4   882078621
...
```

`User ids` take values between 1 and 943, `Item ids` between 1 and 1682, and `ratings` between 1 and 5. The `time stamps` are unix seconds since 1/1/1970 UTC.

User Demographic Data

The MovieLens user demographic data consists of a sequential list of 943 vectors, with elements delimited by '|'s. Each vector incorporates demographic information for the user with the corresponding `user id` and has the following form:

`user id | age | gender | occupation | zip code`

A sample of user demographics from the MovieLens data set follows:

Listing 2: Sample User Demographics from the MovieLens data set

```
...
138|46|M|doctor|53211
139|20|M|student|08904
140|30|F|student|32250
141|49|M|programmer|36117
142|13|M|other|48118
143|42|M|technician|08832
144|53|M|programmer|20910
145|31|M|entertainment|V3N4P
146|45|M|artist|83814
147|40|F|librarian|02143
148|33|M|engineer|97006
149|35|F|marketing|17325
150|20|F|artist|02139
151|38|F|administrator|48103
152|33|F|educator|68767
153|25|M|student|60641
154|25|M|student|53703
...
```

`User ids` are the same as in the GroupLens data set, taking values between 1 and 943. `Age` takes on integer values. `Gender` can be either 'M', for male, or 'F', for female. The `zip code` can be either numerical, for

users residing in the US, or non-numerical, for users residing in Canada. The `occupation` can take a value from the following set of 21 possibilities: {*administrator, artist, doctor, educator, engineer, entertainment, executive, healthcare, homemaker, lawyer, librarian, marketing, none, other, programmer, retired, salesman, scientist, student, technician, writer*}.

Item Demographic Data

The MovieLens item demographic data consists of a sequential list of 1682 vectors, with elements delimited by '|'s. Each vector incorporates 'demographic' information for the item (movie) with the corresponding `movie id` and has the following form:

```
movie id | movie title | release date | video release date | IMDb URL |
unknown | Action | Adventure | Animation | Children's | Comedy | Crime |
Documentary | Drama | Fantasy | Film-Noir | Horror | Musical | Mystery |
              Romance | Sci-Fi | Thriller | War | Western
```

A sample of item demographics from the MovieLens data set follows:

Listing 3: Sample Item Demographics from the MovieLens data set

```
...
291|Absolute Power (1997)|14-Feb-1997||
http://us.imdb.com/M/title-exact?Absolute|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0|1|0|0

292|Rosewood (1997)|21-Feb-1997||
http://us.imdb.com/M/title-exact?Rosewood|0|0|0|0|0|0|0|0|1|0|0|0|0|0|0|0|0|0

293|Donnie Brasco (1997)|28-Feb-1997||
http://us.imdb.com/M/title-exact?Donnie|0|0|0|0|0|0|1|0|1|0|0|0|0|0|0|0|0|0

294|Liar Liar(1997)|21-Mar-1997||
http://us.imdb.com/Title?Liar+Liar+(1997)
|0|0|0|0|0|1|0|0|0|0|0|0|0|0|0|0|0|0

295|Breakdown (1997)|02-May-1997||
http://us.imdb.com/M/title-exact?Breakdown|0|1|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0

296|Promesse, La (1996)|16-May-1997||
http://us.imdb.com/M/title-exact?Promesse|0|0|0|0|0|0|0|1|0|0|0|0|0|0|0|0|0

297|Ulee's Gold (1997)|01-Jan-1997||
http://us.imdb.com/M/title-exact?Ulee|0|0|0|0|0|0|0|1|0|0|0|0|0|0|0|0|0

...
```

The `movie id`s are the ones used in the main data set. The `movie title` is a string including the title of the movie, followed by the year of its release, in parenthesis. The `release date`s are of the form *dd-mmm-yyyy*, eg. 14-Jan-1967. The `IMDb URL` is a web link leading to the Internet Movie Database page of the corresponding movie. The last 19 fields represent the film genres: a value of 1 indicates that the movie belongs to that genre, while a value of 0 indicates the opposite. Movies can belong to more than one genres at the same time.

Appendix B: The Structure of User and Item Demographic Vectors

User Demographic Data

The user demographic data, included in the MovieLens Data Set and described extensively in Appendix A, should be converted to fit a specific vector structure, before they can be utilized by the proposed algorithmic methods. This

Table 6
The Outline of the User Demographic Vector, $usdemog$

feature#	feature contents	comments
age		
1	age $\leqslant 18$	
2	$18 <$ age $\leqslant 29$	each user belongs to a single age grouping
3	$29 <$ age $\leqslant 49$	the corresponding slot takes value 1 (true)
4	age > 49	the rest of the features remain 0 (false)
gender		
5	male	the slot describing the user gender is 1
6	female	the other slot takes a value of 0
occupation		
7	administrator	
8	artist	
9	doctor	
10	educator	
11	engineer	
12	entertainer	
13	executive	
14	healthcare	
15	homemaker	
16	lawyer	
17	librarian	
18	marketing	
19	none	
20	other	
21	programmer	
22	retired	
23	salesman	
24	scientist	
25	student	
26	technician	a single slot describing the user occupation is 1
27	writer	the rest of the slots remain 0

structure, outlined in Table 6, takes the data regarding each specific user from the data set, and transforms it into a 27-feature, binary vector.
Following, there are a couple of examples showing how user demographic data exported from the MovieLens data set, can be transformed to fit the appropriate user demographic vector structure.

Example 1

`148|33|M|engineer|97006`

Based on this sample from the MovieLens data set, user 148 corresponds to a male, 33-years old engineer, with a zip code of 97006.
After the transformation, the resulting user demographic vector is the following:

`0 0 1 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0`

Explanation:

- The '1' in the 3rd slot signifies an age in the $29 <$ age $\leqslant 49$ group.
- The '1' in the 5th slot informs us that the user is a 'male'.
- The '1' in the 11th slot states that the occupation of the user is 'engineer'.
- The rest of the slots are set to '0'

Example 2

`405|22|F|healthcare|10019`

Based on this sample from the MovieLens data set, user 405 corresponds to a female, 22-years old healthcare employer, with a zip code of 10019.
After the transformation, the resulting user demographic vector is the following:

```
0 1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Explanation:

- The '1' in the 2nd slot signifies an age in the $18 < \text{age} \leqslant 29$ group.
- The '1' in the 6th slot informs us that the user is a 'female'.
- The '1' in the 14th slot states that the occupation of the user can be placed under the 'healthcare' category.
- The rest of the slots are set to '0'

Item Demographic Data

Unlike user demographic data, the item demographic data, which is included in the MovieLens Data Set and analyzed in Appendix A, can easily take the form of a binary vector and be utilized by our methods, without any transformations similar to those described in the previous paragraphs.

Specifically, as mentioned in Appendix A, the MovieLens data set incorporates lists with information regarding each of the 1682 distinct items, which have the following form:

```
movie id | movie title | release date | video release date | IMDb URL |
unknown | Action | Adventure | Animation | Children's | Comedy | Crime |
Documentary | Drama | Fantasy | Film-Noir | Horror | Musical | Mystery |
Romance | Sci-Fi | Thriller | War | Western
```

The algorithmic methods proposed in this paper take advantage of each film's genre, which is expressed by the binary values (0 or 1) of the 19 final slots of the preceding list. As a result, except for the `movie id`, we only need to isolate these specific slots, and include them in the item demographic vectors to be utilized by our techniques. The outline of the resulting item demographic vectors is the following:

Table 7
The Outline of the Item Demographic Vector, $itdemog$

feature#	feature contents	comments
film genre		
1	unknown	
2	action	
3	adventure	
4	animation	
5	children's	
6	comedy	
7	crime	
8	documentary	
9	drama	
10	fantasy	
11	film-noir	
12	horror	
13	musical	
14	mystery	
15	romance	
16	sci-fi	
17	thriller	
18	war	a film can belong to more than 1 genres
19	western	thus, more than one slots can be 1

Following, there are a couple of examples showing how item demographic data, exported from the GroupLens data set, can be transformed to the appropriate item demographic vector structure.

Example 1

```
328|Conspiracy Theory (1997)|08-Aug-1997||http://us.imdb.com/M/title-exact?
Conspiracy+Theory+(1997)|0|1|0|0|0|0|0|0|0|0|0|0|1|1|0|1|0|0
```

Based on this sample from the MovieLens data set,
After the transformation, the resulting user demographic vector is the following:

```
0 1 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 0 0
```

Explanation: The film with `movie id` 328, titled 'Conspiracy Theory', has an item demographic vector with '1's for the 2nd, 14th, 15th and 17th features. Thus, it belongs to the following genres: *action*, *adventure*, *romance* and *thriller*.

Example 2

```
403|Batman (1989)|01-Jan-1989||http://us.imdb.com/M/title-exact?
Batman%20(1989)|0|1|1|0|0|0|1|0|1|0|0|0|0|0|0|0|0|0
```

Based on this sample from the MovieLens data set,
After the transformation, the resulting user demographic vector is the following:

```
0 1 1 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0
```

Explanation: The film with `movie id` 403, titled 'Batman', has an item demographic vector with '1's for the 2nd, 3rd, 7th and 9th features. Thus, it belongs to the following genres: *action*, *mystery*, *crime* and *drama*.

Appendix C: Pseudo-code for U-Demog and I-Demog

Pseudo-code for U-Demog

The following paragraphs include pseudo-code for the distinct steps of the U-Demog algorithm, which were detailed in Section 4.1.

Step 1: Construct the demographic vectors for the m users who participate in the recommendation process.

We read the user demographic data from the GroupLens data file. Since it follows the structure which was detailed in Appendix A, it needs to be transformed accordingly, in order to take the form discussed in Appendix B. Once the step is completed, the result would be a 943×27 user demographic matrix, where every line corresponds to a specific user, out of the 943, and includes the values for that user's 27 demographic features.

Step 2: Execute the first two steps of User-based Collaborative Filtering, that is Data Representation and Neighborhood Formation.

Regarding Data Representation, we read the complete GroupLens data set, whose structure was described in Appendix A, and fill the $m \times n$ user-item matrix accordingly. In the resulting array, each element of row i and column j will either include the rating awarded by user u_i on item u_j, if such a rating exists in the data set, or, it will be filled by a value of 0, denoting 'no-rating', if item i_j was not rated by user u_i.

Regarding Neighborhood Formation, we need to calculate the similarity, rat_cor, between each pair of users, u_i and u_j for $\{i,j\}$=*1,2,...,m*, based on their opinions on all items, i_k for *k=1,2,...l*, they have rated in common. Obviously, a couple of users may have rated in common at most n items, that is, $l \leqslant n$.

```
for user u_i with i = 1, 2, ..., m
    for user u_j with j = 1, 2, ..., m − 1
        for item i_k with k = 1, 2, ..., n
            calculate rat_cor(u_i,u_j,i_k)
        endfor
    endfor
endfor
```

In the worst case, all m users would have rated all n items. Thus, the complexity of computing the $m \times m$ user-user similarity matrix would be $O(m^2 n)$. Still, since the number of movies, l, rated in common are usually much less than the total, $l << n$, the average complexity is closer to $O(m^2 l)$=$O(m^2)$.

Step 3: Calculate the demographic correlation between a couple of users.

The calculation of the demographic correlation between every possible couple of users, u_i and u_j for $\{i,j\}$=*1,2,...,m*, is achieved by evaluating the vector similarity (Eq. (7)) of their corresponding user demographic vectors, on the 27 user demographic features (Appendix B).

```
for user u_i with i = 1, 2, ..., m
    for user u_j with j = 1, 2, ..., m − 1
        for user_demog_feature udf_k with k = 1, 2, ..., 27
            calculate dem_cor(u_i,u_j,udf_k)
        endfor
    endfor
endfor
```

Since the number of user demographic features is constant (27) and, in this specific case, this number is considerably lower than $m = 943$, the complexity of this step would be equal to $O(m^2)$.

Step 4: Calculate the Enhanced Correlation for every pair of users including the active user and a member of his neighborhood.

We apply the Enhanced Correlation formula (Eq. (12)) on the corresponding pair of users, utilizing the appropriate ratings-based and demographic correlation values which were calculated in Steps 2 and 3 respectively. In the worst case where the neighborhood of active user u_a for $a = 1, 2, \ldots, m$ includes all the remaining users, the Enhanced Correlation calculation would be repeated $m \times (m - 1)$ times.

Step 5: Proceed with the final step of the recommendation procedure, which is Prediction Generation.

For the calculation of the predicted rating of active user u_a on item i_j, we need to apply the prediction generation formula (Eq. (13)) on all the users, u_i, who were selected for the active user's neighborhood. In the worst case, the neighborhood users would be $m - 1$, for a complexity of $O(m)$.

```
for user u_i with i = 1, 2, ..., m − 1
    calculate upred_gen(u_a,i_j,u_i)
endfor
```

Still, the actual complexity could be significantly lower than that, since the number of neighborhood users is normally much less than $m - 1$.

Pseudo-code for I-Demog

The following paragraphs include pseudo-code for the distinct steps of the I-Demog algorithm, which were detailed in Section 4.2.

Step 1: Construct the demographic vectors for the n items which participate in the recommendation process.

We read the item demographic data from the GroupLens data file. Since it follows the structure which was detailed in Appendix A, it needs to be transformed accordingly, in order to take the form discussed in Appendix B. Once the step is completed, the result would be a 1682×19 item demographic matrix, where every line corresponds to a specific item, out of the 1682, and includes the values for that items's 19 demographic features.

Step 2: Execute the first two steps of Item-based Collaborative Filtering, that is Data Representation and Neighborhood Formation.

Regarding Data Representation, we read the complete GroupLens data set, whose structure was described in Appendix A, and fill the $m \times n$ user-item matrix accordingly. In the resulting array, each element of row i and column j will either include the rating awarded by user u_i on item u_j, if such a rating exists in the data set, or, it will be filled by a value of 0, denoting 'no-rating', if item i_j was not rated by user u_i.

```
for item i_j with j = 1, 2, ..., n
    for item i_k with k = 1, 2, ..., n − 1
        for user u_i with i = 1, 2, ..., m
            calculate rat_cor(i_j,i_k,u_i)
        endfor
    endfor
endfor
```

Regarding Neighborhood Formation, we need to calculate the similarity, rat_cor, between each pair of items, i_j and i_k for $\{j,k\} = 1, 2, \ldots, n$, based on the opinions expressed on them, by all users u_i for $i = 1, 2, \ldots l$, who have rated both.
In the worst case, all n items would have been rated by all m users. Thus, the complexity of computing the $n \times n$ item-item similarity matrix would be $O(mn^2)$. Still, since the number of users, l, who have rated both items are much less than the total, $l << m$, the average complexity is closer to $O(ln^2)$=$O(n^2)$.

Step 3: Calculate the demographic correlation between a couple of items.

The calculation of the demographic correlation between every possible couple of items, i_j and i_k for $\{j,k\} = 1, 2, \ldots, n$, is achieved by evaluating the vector similarity (Eq. (7)) of their corresponding item demographic vectors, on the 19 item demographic features (Appendix B).

```
for item i_j with j = 1, 2, ..., n
    for item i_k with k = 1, 2, ..., n − 1
        for item_demog_feature idf_i with i = 1, 2, ..., 19
            calculate dem_cor(i_j,i_k,idf_i)
        endfor
    endfor
endfor
```

Since the number of item demographic features is constant (19) and, in this specific case, this number is considerably lower than $n = 1682$, the complexity of this step would be equal to $O(n^2)$.

Step 4: Calculate the Enhanced Correlation for every pair of items including the active item and a member of its neighborhood.

We apply the Enhanced Correlation formula (Eq. (14)) on the corresponding pair of items, utilizing the appropriate ratings-based and demographic correlation values which were calculated in Steps 2 and 3 respectively. In the worst case where the neighborhood of active item i_a for $a = 1, 2, \ldots, n$ includes all the remaining items, the Enhanced Correlation calculation would be repeated $n \times (n-1)$ times.

Step 5: Proceed with the final step of the recommendation procedure, which is Prediction Generation.

For the calculation of the predicted rating of user u_a on active item i_j, we need to apply the prediction generation formula (Eq. (15)) on all the items, i_k, which were selected for the active item's neighborhood. In the worst case, the neighborhood items would be *n-1*, for a complexity of $O(n)$.

```
for item i_k with k = 1, 2, ..., n − 1
    calculate ipred_gen(u_a,i_j,i_k)
endfor
```

Still, the actual complexity could be significantly lower than that, since the number of neighborhood items is normally much less than *n-1*.