Alternative Formulas for Rating Prediction Using Collaborative Filtering

Amar Saric, Mirsad Hadzikadic, David Wilson
College of Computing and Informatics
The University of North Carolina at Charlotte, 9201 University City Blvd, Charlotte, NC 28223, USA
{asaric, mirsad, davils}@uncc.edu

Abstract. This paper proposes and evaluates several alternate design choices for common prediction metrics employed by neighborhood-based collaborative filtering approach. It first explores the role of different baseline user averages as the foundation of similarity weighting and rating normalization in prediction, evaluating the results in comparison to traditional neighborhood-based metrics using the MovieLens data set. The approach is further evaluated on the Netflix movie data set, using a baseline correlation formula between movies, without meta-knowledge. For the Netflix domain, the approach is augmented with a significance weighting variant that results in an improvement over the original metric. The resulting approach is shown to improve accuracy for neighborhood-based collaborative filtering, and it is general and applicable to establishing relationships among agents with a common list of items which establish their preferences.

Keywords: Collaborative filtering, Personalized Recommendation, Rating Prediction, Similarity measure.

1 Introduction

Collaborative filtering recommender systems employ ratings-based user profiles in order to make item recommendations or predictions about user ratings for items. Suitable items for recommendation, such as suggested movies to watch, are identified not because their description matches them with a target user, but rather because these items have been liked by users who are similar to the target user in terms of how they have rated other items. Collaborative filtering can employ a variety of foundational algorithms, but the most prevalent are the so-called neighborhood-based methods. Neighborhood-based methods first locate a subset of the user population, based on their similarity to the current or active user. Typically, then a weighted combination of the neighbors' ratings are employed as the basis for rating-prediction or recommendation for the active user. Herlocker et al. [4] identified and tested several main aspects of the design space for neighborhood-based collaborative filtering, including: similarity weighting, significance weighting, variance weighting, neighborhood selection, rating normalization, and neighbor contribution weighting. In

this paper, we focus on improving the accuracy in neighborhood-based collaborative filtering along the dimensions of computing the neighborhood and computing the prediction (see for instance [3], [6] or [9]). This paper proposes and evaluates several alternate design choices for common prediction metrics employed by neighborhood-based collaborative filtering approaches. It first explores the role of different baseline user averages as the foundation of similarity weighting and rating normalization in prediction, evaluating the results in comparison to traditional neighborhood-based metrics using the MovieLens data set. The approach is further evaluated on the Netflix movie data set, using a baseline correlation formula between movies, without meta-knowledge. For the Netflix prize [8] domain, the approach is augmented with a significance weighting variant that results in an improvement over the original Netflix metric. Evaluation results show that our approach improves accuracy for neighborhood-based collaborative filtering. The approach is general and applicable to establishing relationships among agents with a common list of items that establish their preferences.

2 Collaborative filtering formulas

The baseline algorithm in the literature for neighborhood-based collaborative filtering uses the Pearson correlation [4]. For raters M and N it is calculated using

$$r_{MN} = \frac{\sum_k (M_k - \overline{M})(N_k - \overline{N})}{\sqrt{\sum_k (M_k - \overline{M})^2} \sqrt{\sum_k (N_k - \overline{N})^2}},$$

where M_k and N_k respectively are ratings for the item k, $\overline{M} = \frac{1}{\dim_{MN}} \sum_k M_k$ and $\overline{N} = \frac{1}{\dim_{MN}} \sum_k N_k$ the mean values, and $\dim_{MN}$ denotes the total number of items rated by both users. If $\dim(M, N) = 0$, then r_{MN} is simply set to zero. All sums in the above formula are computed over the ratings which both users have in common. $\overline{M}$ is strictly speaking a function of N, and vice versa, but writing it down explicitly would unnecessarily complicate the formulas. The predictions are then computed using the following formula

$$M_j = \overline{M} + \frac{\sum_{N \in Raters \setminus \{M\}} (N_j - \overline{N}) r_{MN}}{\sum_{N \in Raters \setminus \{M\}} |r_{MN}|},$$

where $\overline{M}$ is the mean of all ratings for a given rater. This is the standard GroupLens approach for the so-called user-to-user ratings prediction. We considered the raters/users to be 'agents' and compare them among each other to establish links

between them (at least conceptually). The viewpoint can be changed so that the items become 'agents', the mechanics and, more importantly, also the underlying logic, stay unchanged – it all depends on what is considered the 'agent' the user or the rated item. Here we compare the items to each other, with M and N in the above formulas denoting two items and summation performed over the common raters for these items. Also the summation in the prediction formulas is performed over similar items not raters. This is commonly referred to as item-to-item collaborative filtering. In order to disambiguate the notation in what follows, we define two different correlation coefficients as

$$\overline{r_{MN}} = \frac{\sum_k (M_k - \overline{M})(N_k - \overline{N})}{\sqrt{\sum_k (M_k - \overline{M})^2}\sqrt{\sum_k (N_k - \overline{N})^2}}, \quad \overline{\overline{r_{MN}}} = \frac{\sum_k (M_k - \overline{\overline{M}})(N_k - \overline{\overline{N}})}{\sqrt{\sum_k (M_k - \overline{\overline{M}})^2}\sqrt{\sum_k (N_k - \overline{\overline{N}})^2}}.$$

Here $\overline{\overline{M}}$ denotes the mean values over all ratings of the user M, and $\overline{M}$ the mean value only over the ratings that the user has in common with the user N. The summation in both formulas goes over all common ratings. In what follows we will discuss collaborative filtering and show how additional formulas can be obtained. Our main goal will be to see what changes the basic ideas of collaborative filtering allow. We start with

$$M_j = \overline{\overline{M}} + \frac{\sum_{N \in Raters \setminus \{M\}} (N_j - \overline{N})\overline{r_{MN}}}{\sum_{N \in Raters \setminus \{M\}} \left|\overline{r_{MN}}\right|}, \tag{1}$$

which is the original GroupLens formula as presented in the paper by Resnick, Iacovou, Suchak, Bergstrom and Riedl [9], restated in our notation. Towards the end of the paper we will also move away from using only linear elements in the prediction formula. There are several different possibilities to alter this formula, for example using normalization. Our main criticism of the formula, however, is that it utilizes the mean over all ratings $\overline{\overline{M}}$ to offset the predictions, while computing the Pearson correlation over only the property values (i.e. ratings) that both agents, say users, have in common. Therefore we test several formulas using the mean calculated over all the values with other agents $\overline{\overline{M}}$ as well as only over the common ratings $\overline{M}$ for a given user. Using the mean averages over all the ratings the equivalent of the above

$$M_j = \overline{\overline{M}} + \frac{\sum_{N \in Raters \setminus \{M\}} (N_j - \overline{\overline{N}})\overline{\overline{r_{MN}}}}{\sum_{N \in Raters \setminus \{M\}} \overline{\overline{r_{MN}}}}. \tag{2}$$

However, some raters might be reluctant to give the best or worst possible ratings on the Likert scale. Therefore, at least for user-to-user comparisons, a possible change from which we might expect some improvement is to adjust the offset from the mean in the prediction formula, which is in fact based on the ratings given by other users. If we try to scale these contributions using the same norm as in the Pearson correlation (see for instance [5]), the GroupLens formula (1) becomes

$$M_j = \overline{\overline{M}} + \frac{\sum_{N \in Raters \setminus \{M\}} \frac{\sqrt{\sum_k (M_k - \overline{M})^2}}{\sqrt{\sum_k (N_k - \overline{N})^2}} (N_j - \overline{N}) \overline{r_{MN}}}{\sum_{N \in Raters \setminus \{M\}} \left| \overline{r_{MN}} \right|} \quad (3)$$

Normalization makes little sense if we are comparing items, since in this case we cannot talk of rating 'tendencies', although the variance itself might of course be useful. By using the averages over all the ratings instead, the above formula is transformed to

$$M_j = \overline{\overline{M}} + \frac{\sqrt{\sum_k (M_k - \overline{\overline{M}})^2}}{\sum_{N \in Raters \setminus \{M\}} \left| \overline{\overline{r_{MN}}} \right|} \sum_{N \in Raters \setminus \{M\}} \frac{\overline{\overline{r_{MN}}}}{\sqrt{\sum_k (N_k - \overline{\overline{N}})^2}} (N_j - \overline{\overline{N}}) \quad (4)$$

Nevertheless, the behavior of agents outside the range of values common to both users cannot be implied by their behavior within this range. This is also the most likely reason why the GroupLens formula uses $\overline{M}$ instead of $\overline{\overline{M}}$. But this itself is not consistent, since the mean value calculated over all the ratings is used as the starting point to which the contributions from the other users are added. At first glance, it might seem that there is no other way to do this but to use $\overline{\overline{M}}$. However, the following formula overcomes this "imperfection"

$$M_j = \frac{\sum_{N \in Raters \setminus \{M\}} \left| \overline{r_{MN}} \right| \overline{M}}{\sum_{N \in Raters \setminus \{M\}} \left| \overline{r_{MN}} \right|} + \frac{\sum_{N \in Raters \setminus \{M\}} (N_j - \overline{N}) \overline{r_{MN}}}{\sum_{N \in Raters \setminus \{M\}} \left| \overline{r_{MN}} \right|} \quad (5)$$

by using a weighted average also in the first term of the formula. The contribution of the other users is added to the average of user M for the same range and the total result scaled by the value of their Pearson correlation. Thus, every user contributes a value to the total estimate, which is proportional to the absolute value of the correlation coefficient. Alternatively, we can also rewrite this formula as

$$M_j = \frac{\sum_{N \in Movies \setminus \{M\}} \overline{|r_{MN}|}(\overline{M} + \mathrm{sgn}(\overline{r_{MN}})(N_j - \overline{N}))}{\sum_{N \in Movies \setminus \{M\}} \overline{|r_{MN}|}}.$$

The prediction formula is therefore a weighted average of single predictions based on other agents. This is also valid for item-to item collaborative filtering. After normalizing the ratings in the formula (6) we obtain

$$M_j = \frac{\sum_{N \in Raters \setminus \{M\}} \overline{|r_{MN}|}\overline{M}}{\sum_{N \in Raters \setminus \{M\}} \overline{|r_{MN}|}} + \frac{\sum_{N \in Raters \setminus \{M\}} \frac{\sqrt{\sum_k (M_k - \overline{M})^2}}{\sqrt{\sum_k (N_k - \overline{N})^2}} (N_j - \overline{N})\overline{r_{MN}}}{\sum_{N \in Raters \setminus \{M\}} \overline{|r_{MN}|}}. \quad (6)$$

All of the above formulas are reasonable alternatives to the GroupLens formula (1). One could think that the alternative formulas are computationally more expensive, but the use of $\overline{M}$ the means in Formula (1), which is computed only over the common values, actually requires the same number of passes through data as the Formula (5) – only the additional means have to be stored along with the correlation values. The Formulas (2) and (4) are in fact easier to compute since the values for the means can be pre-computed (for all user ratings), stored, and used in all subsequent calculations. Possibly one formula will calculate a prediction, while the other one will not, in which case we use the average for prediction. This does not happen frequently.

2.1 Experimental results

The evaluation was performed using the MovieLens data containing 100,000 ratings, which provides data splits suitable for use as training and test sets. The data can be obtained from the GroupLens webpage [1]. The data was randomly split into 5 base and test sets, of 80,000 and 20,000 ratings respectively, in order to be able to empirically evaluate formulas. We used mean absolute error (MEA) as performance measure for comparisons.

The graph in figure 1 indicates that best predictions are obtained for Formulas (5) and (6). The evaluation of theses formulas was performed using the mean absolute error as a measure. All users for which the absolute value of correlation with the current user was below 0.1 were ignored in the computation, as well as all the users having only one rating in common. After checking whether the predicted value is out of range, i.e. less than 1 or greater than 5 in the case of MovieLens data, and correcting it to the closest possible value, we conclude that the normalized prediction formulas are favored again over other alternatives. This is shown in Figure 2, which brings only minimal gains, possibly because it does not occur very often.

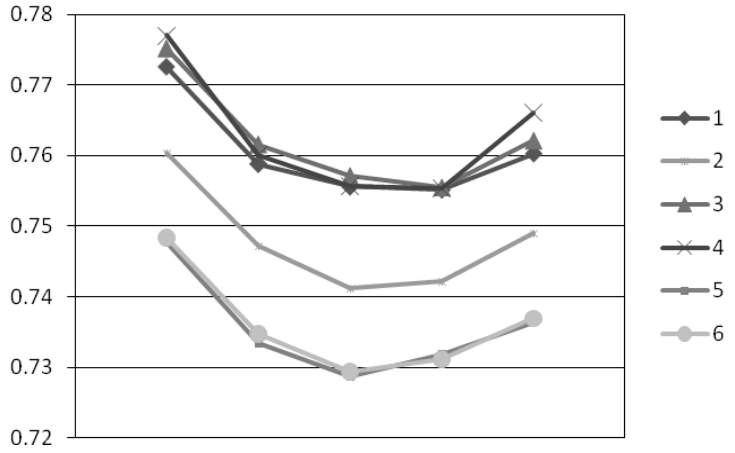

Fig. 1. Mean absolute error using cross-validation on 5 different splits for formulas (1) - (6)

Also, there is the possibility of varying the "cut-off" value for the correlation with a small absolute value, as well as discarding those users who have less than a fixed number of ratings in common with the user for whom we are trying to make the prediction in order to remove noise. Consequently, the interesting question is how the different formulas would behave in these cases. In the rest of the text we look into these two issues in more detail. Figure 3, for example, shows the values obtained after discarding all the users with a correlation of less than 0.5, and clipping the predicted value if it falls outside the boundaries. Obviously, increasing this value does not necessarily decrease the overall mean error.

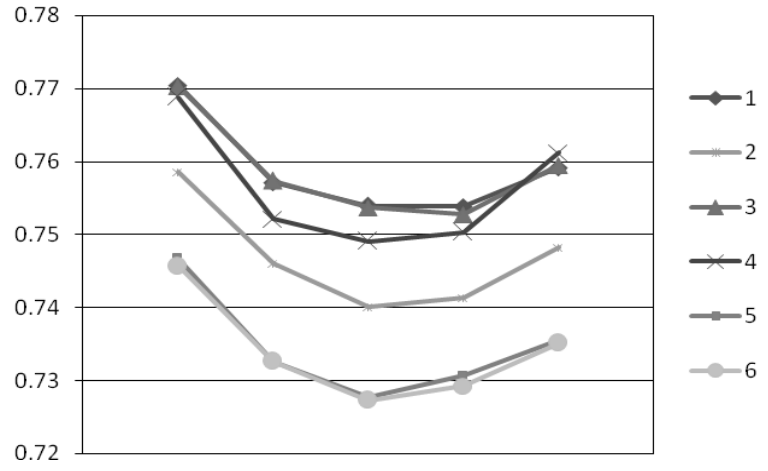

Fig. 2. Clipping values in case out of range predictions for formulas (1) - (6)

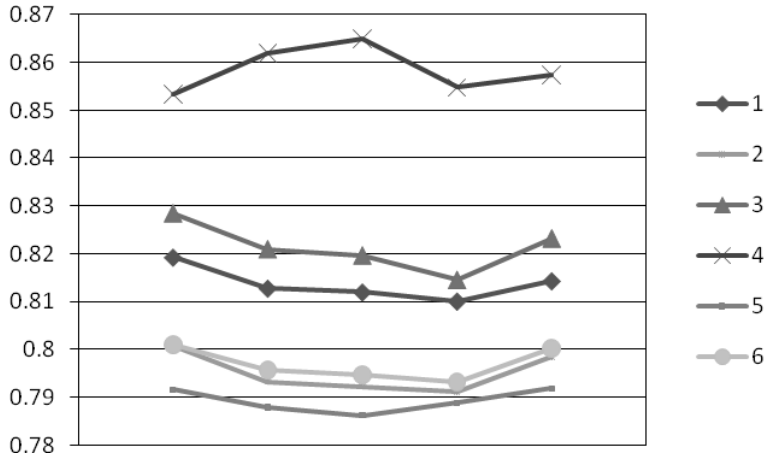

Fig. 3. Excluding small correlation values from contributing to the prediction, does not necessarily improve the MAE

Finally, let us examine what happens when we exclude all the correlations that are based on 10 or fewer common ratings, and disallow any contribution from these users. The important thing to note is that this has to do with the trust in the similarity measure and that it does improve results (Figure 4). This will also, obviously,

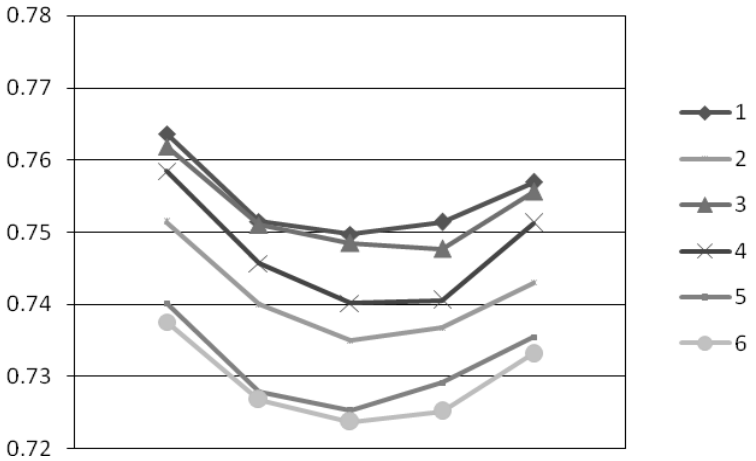

Fig. 4. Predictions calculated using correlation values based on at least 10 common ratings for formulas (1) - (6)

decrease the number of cases in which we are able to make a prediction. Again, the predicted values in this example were corrected if they were out of range, and correlation values were discarded if they were below 0.1. In the rest of the text we look in more detail into these two issues.

3 A more reliable similarity measure

Based on our initial experiments, we wanted to test for increased scale on real-world data. We used item-to-item collaborative filtering on the Netflix data [2] to verify that the approach can result in improvements over the standard formula on real problems, other than that we are following the original approach by Resnick et al. [9]. No clustering or transformation of data was performed or additional information used (such are the dates when the rating was made, the genre of the movie etc.). Formula (5) based item-to-item collaborative filtering did not improve the score, but it also performed just as good as Netflix's original algorithm. Better results for the dataset were obtained by replacing absolute values by squares, which effectively amounts to using a different similarity measure: Specifically, we insert $\mathrm{sgn}(r_{MN})\left|\overline{r_{MN}}\right|^2$ into the equation (5), which replaces $\overline{r_{MN}}$, and also use item-to-item comparisons. The resulting formula is

$$M_j = \frac{\sum_{N \in Movies \setminus \{M\}} \left|\overline{r_{MN}}\right|^2 (\overline{M} + \mathrm{sgn}(\overline{r_{MN}})(N_j - \overline{N}))}{\sum_{N \in Movies \setminus \{M\}} \left|\overline{r_{MN}}\right|^2} \tag{7}$$

which is a weighted average of predictions based on other items. The ratings were not normalized, mostly because we were using item-to-item comparisons, since in that case (i.e. for a given movie) one cannot expect that ratings provided by different users will vary from the mean by the same average amount. Rather, one would expect such rating tendencies to be valid for raters and not the items. Adjusting the value of the weights by taking into account the number of common ratings yields the following slightly involved but otherwise straight-forward formula

$$M_j = \frac{\sum_{N \in Movies \setminus \{M\}} \beta(M,N)^2 (\overline{M} + \mathrm{sgn}(\overline{r_{MN}})(N_j - \overline{N}))}{\sum_{N \in Movies \setminus \{M\}} \beta(M,N)^2} \tag{8}$$

where $\beta(M,N) = \overline{r_{MN}} \tanh(\lambda \dim_{MN})$, the number of users who have rated movies M and N is given by $\dim_{MN}$, and λ is a parameter to be determined. $\beta(M,N)^2$, which replaces $\left|\overline{r_{MN}}\right|$ in formula (5), includes a penalty function used to scale the similarity

measure (here the square of the Pearson correlation) based on the number of ratings the two movies have in common, to construct a new similarity measure. The value λ determines, roughly speaking, the minimal number of users who rated both movies. The contributions coming from other movies with only few raters in common are penalized. This is, in principle, only a minor modification of discarding only loosely (anti-)correlated movies. Appropriate λ can be found empirically. For the Netflix data values between 0.001 and 0.003 seems to work well. Figure 5 shows the contribution of the $\tanh^2(\lambda \dim(M,N))$ term for $\lambda = 0.002$ .This approach resulted in a 2.9% improvement over the original root mean squared error of 0.9514 achieved by the "Cinematch" algorithm.

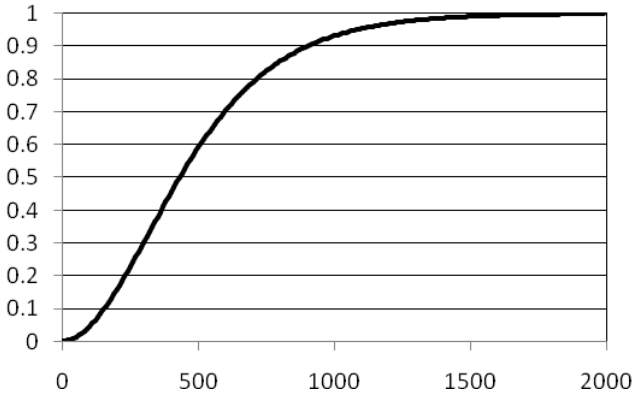

Fig. 5. Trust allocated based on the number of common ratings. The similarity measure is multiplied by this value to obtain the contribution of a movie in the prediction of rating for another movie

Additionally, we have also experimented with sorting the ratings for other movies in decreasing order based on the similarity values β and aborting summation after "enough" of the other movies were considered. We used $\sum_{N \in Movies \setminus \{M\}} r_{MN}{}^2 \tanh^2(\lambda \dim_{MN}) < 2.5$ as condition, but the effect was marginal – similar to that of rounding values when they are out of range – affecting only the third digit after decimal point in the RMSE. However, it is possible to do so, and, it did result in a small improvement albeit at the expense of having to determine a suitable cutoff value. We found the values between 2 and 3 to work well with the values of λ in the above range. The only drawback is that one has to change the logic of the application. The changes described previously consisted of storing additional values in calculations which already had to be performed in order to compute predictions based on formula (1). They could therefore be incorporated easily into any system using the approach described by [9].

4 Conclusion

Although additional tests should be performed, it seems realistic that the above formulas could be used successfully instead of the standard prediction formula [4]. It is noteworthy that the errors obtained for the formulas (5) and (6) in our test runs were consistently below those for the most commonly used prediction formula (1). We consider them also to be somewhat more appealing because of the way the averages are calculated. We therefore propose that formula (5) be the default formula for collaborative filtering, and one optionally use penalty functions as in formulas (7) and (8). Any system which is based on (1), can easily be modified to use (5), (6) or (7), and, if one is willing to empirically determine the additional parameter, also (8). There is no architectural reason not to simply replace the formula and leave the rest of the system unaltered. It is also somewhat surprising that, at least for the data sample used, the normalization seems to have had only a limited effect. Perhaps, one could obtain better results with other norms, for instance $\max_k \left|U_k - \overline{U}\right|$, or $\sum_k \left|U_k - \overline{U}\right|$.

Experiments with the Netflix database show that the modified formulas allow for improvements over the original collaborative filtering. The conclusion here is that one can improve the results by using penalty functions based on the number of common ratings in addition to the similarity measure.

References

1. GroupLens - http://www.grouplens.org/
2. Bennett, J. and Lanning, S.: The Netflix Prize, In Proceedings of the KDD Cup and Workshop (2007)
3. Herlocker, J. L., Konstan, J. A., Borchers, A., and Riedl, J.: An algorithmic framework for performing collaborative Filtering, Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, pp. 230-237, Berkeley (1999)
4. Herlocker, J., Konstan, J., and Riedl, J.: Empirical Analysis of Design Choices in Neighborhood-based Collaborative Filtering Algorithms. Informational Retrieval, Vol. 5, No. 4, pp.287-310. Springer. (2002)
5. Jin, R. and Si, L.: A Study of Methods for Normalizing User Ratings in Collaborative Filtering, The 27th Annual International ACM SIGIR Conference, pp. 568 – 569, Sheffield, (2004)
6. Konstan, J., Miller, B., Maltz, D., Herlocker, J., Gordon, L., and Riedl, J.: Grouplens: Applying collaborative filtering to usenet news, Communications of the ACM, Vol. 40, No. 3, pp. 77-87. ACM, New York (1997)
7. MovieLens - http://www.movielens.umn.edu/
8. Netflix prize - www.netflixprize.com
9. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P. and Riedl, J.: GroupLens: An Open Architecture for Collaborative Filtering of Netnews, Proceedings of ACM Conference on Computer Supported Cooperative Work, pp. 175–186, Chapel Hill (1994)
10. Sarwar, B., Karypis, G., Konstan, J. and Riedl, J.: Item-based collaborative filtering recommendation algorithms, WWW '01: Proceedings of the 10th international conference on World Wide Web, pp. 285–295.ACM, New York (2001)