Context-Based Splitting of Item Ratings in Collaborative Filtering

Linas Baltrunas
Free University of Bozen-Bolzano,
Via Sernesi 1,
Bolzano, Italy
lbaltrunas@unibz.it

Francesco Ricci
Free University of Bozen-Bolzano,
Via Sernesi 1,
Bolzano, Italy
fricci@unibz.it

ABSTRACT

Collaborative Filtering (CF) recommendations are computed by leveraging a historical data set of users' ratings for items. It assumes that the users' previously recorded ratings can help in predicting future ratings. This has been validated extensively, but in some domains item ratings can be influenced by contextual conditions, such as the time or the goal of the item consumption. This type of information is not exploited by standard CF models. This paper introduces and analyzes a novel pre-filtering technique for context-aware CF called item splitting. In this approach, the ratings of certain items are split, according to the value of an item-dependent contextual condition. Each split item generates two fictitious items that are used in the prediction algorithm instead of the original one. We evaluated this approach on real world and semi-synthetic data sets using matrix-factorization and nearest neighbor CF algorithms. We show that item splitting can be beneficial and its performance depends on the item selection method and on the influence of the contextual variables on the item ratings.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information filtering*

General Terms

Algorithms, Experimentation

1. INTRODUCTION

Collaborative Filtering (CF) recommendations are computed by leveraging historical log data of users' online behavior [2]. CF assumes that the user's previously recorded ratings for items can help in predicting the ratings of like-minded users. This assumption is valid only to some extent. In fact, the user's general interests can be relatively stable, but the exact evaluation of an item can be influenced by many additional and varying factors. In certain domains the consumption of the same item can lead to extremely different experiences when the context changes [1, 4]. For instance, in a tourism application the visiting experience to a beach in summer is strikingly different from the same visit in winter (e.g., during a conference meeting). However, most CF recommender system would not distinguish between these two experiences, thus providing a poor recommendation in certain situations.

Context-aware recommender systems is a new area of research [1, 4], and context-aware approaches can be classified into three groups: pre-filtering, post-filtering and contextual modelling [3]. [8] presents a Case Based Reasoning approach; this music recommender falls into the category of contextual post-filtering, it uses a cascade architecture and re-ranks the recommendation list depending on the current genre and artist information. A pre-filtering approach, as that proposed in this paper, was explored by [4], where contextual information is used to alter the user model. The authors do not use a fixed set of contextual attributes but extract "contextual cues" from the data that are later used to pre-filter the user's rating data. [1] extends the classical CF method adding to the standard dimensions of users and items new ones representing contextual information. Here recommendations are computed using only the ratings made in the same context as the target one. The authors use a hierarchical representation of context, therefore, the exact granularity of the used context is searched (optimized) among those that improve the accuracy of the prediction. Similarly, in our approach we enrich the simple 2-dimensional CF matrix with a context model comprising a dynamic set of user, item, and evaluation features. We adopt the definition of context introduced by Dey, where "Context is any information that can be used to characterize the situation of an entity" [7]. Here, the entity is the experience of an item that can be influenced by contextual variables describing the state of the user and the item. In this paper we propose a new approach for using these contextual dimensions to pre-filter the item ratings (evaluation of a user for an item). In practice, the set of ratings for an item is not filtered but it is split into two subsets according to the value of a contextual variable, e.g., ratings collected in "winter" or in "summer" (the contextual variable is the season of the rating/evaluation). These two sets of ratings are then assigned to two new fictitious items (e.g. beach in winter and in summer). This split is performed only if there is the statistical evidence that under these two contextual conditions the item's ratings were different, i.e., users evaluate differently the item.

RecSys'09, October 23–25, 2009, New York, New York, USA.
Copyright 2009 ACM 978-1-60558-435-5/09/10 ...$10.00.

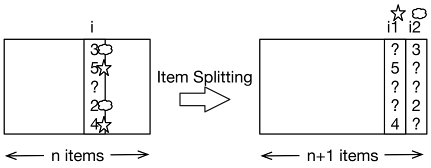


Figure 1: Item splitting

This study shows that standard neighborhood and matrix factorization based CF models cannot cope with rating data influenced by contextual conditions. In fact, we show that if the contextual condition does influence the item ratings, item splitting techniques can help to improve the accuracy of CF, especially with matrix factorization techniques.

2. ITEM SPLITTING

Our approach extends the traditional CF data model by assuming that each rating r_{ui} in a $m \times n$ users-items matrix, is stored together with a contextual information $c(u, i) = (c_1, \ldots, c_n), c_j \in C_j$, describing the conditions under which the user experience was collected (c_j is a nominal variable). The proposed method identifies items having significant differences in the ratings (see later the exact test criteria). For each one of these items, our algorithm splits its ratings into two subsets, creating two new artificial items with ratings belonging to these two subsets. The split is determined by the value of one contextual variable c_j, i.e., all the ratings in a subset have been acquired in a context where the contextual feature c_j took a certain value. So, for each item the algorithm seeks for a contextual feature c_j that can be used to split the item. Then it checks if the two subsets of ratings have some statistical significant difference, e.g., in the mean. If this is the case, the split is done and the original item in the ratings matrix is replaced by the two newly generated items. In the testing phase, the rating predictions for the split item are computed for one of the newly generated item. For example, assume that an item i has generated two new items i_1 and i_2, where i_1 (i_2) contains ratings for item i acquired in the contextual condition $c_j = v$ ($c_j \neq v$). Now assume that the system needs to compute a rating prediction for the item i and user u in a context where $c_j = x$. Then the prediction is computed for the item i_1 if $x = v$, or i_2 if $x \neq v$, and is returned as the prediction for i.

Figure 1 illustrates the splitting of one item. As input, item splitting step takes a $m \times n$ rating matrix of m users and n items and outputs a $m \times n+1$ matrix. The total number of ratings in the matrix does not change. This step can be repeated for all the items that show a dependency of their ratings from the value of one contextual variable. In this paper we focus on a simple application of this method where an item is split only into two items, using only one selected contextual variable. A more aggressive split of an item into several items, using a combination of features, could produce even more "specialized" items, but potentially increasing data sparsity.

We conjectured that splitting is beneficial if the ratings in the newly obtained items are more homogenous, or if the ratings in the two newly generated items are statistically different. One way to accomplish this task (also used in the decision tree theory [6]) is to define an impurity criteria t. So, if there are some candidate splits $s \in S$ we choose the split s that maximizes $t(i, s)$ in S. A split is determined by selecting a contextual variable and a partition of its values in two sets.

We considered five impurity criteria: t_{mean}, t_{prop}, t_{size}, t_{IG} and t_{random}. $\boldsymbol{t_{mean}(i, s)}$ uses t-test to estimate how different the means of the ratings in two ratings' subsets are, when s is used. $\boldsymbol{t_{prop}(i, s)}$ uses two-proportion z-test and determines whether there is a significant difference between the proportions of high (> 3) and low (< 4) ratings in the generated subsets of ratings [9]. $\boldsymbol{t_{size}(i, s)}$ measures the number of ratings for i and does not depend on s. We use this measure to determine which items to split first. We hypothesized that an item is worth splitting if it contains enough (or many) ratings. $\boldsymbol{t_{IG}(i, s)}$ measures the information gain (Kullback-Leibler divergence) given by s to the knowledge of the item i rating. Finally, $\boldsymbol{t_{random}(i, s)}$ is used as a reference for comparing the behavior of the other methods. It returns a random score for each split s.

3. EXPERIMENTAL EVALUATION

We tested the proposed method on two real-world and one semi-synthetic data sets with ratings in {1, 2, 3, 4, 5}. The Yahoo![1] Webscope movies data set contains 221K ratings, for 11,915 movies by 7,642 users. The MovieLens[2] data set contains 1M ratings, for 3,706 movies by 6,040 users, who rated 20 and more items. In both data sets the user age and gender features were used as contextual variables. We used 3 age groups: users below 18 (u18), between 18 and 50 (18to50), and above 50 (a50). Because of lack of space, we cannot provide detailed results for MovieLens data.

The semi-synthetic data sets were used to analyze item splitting when varying the influence of the context on the user ratings. We extended the original Yahoo! data set adding to the gender and age features a new artificial and random feature $c \in \{0, 1\}$. This feature c is representing a contextual condition that could affect the rating. We then randomly chose $\alpha * 100\%$ of the ratings and we increased (decreased) the rating value by one if $c = 1$ ($c = 0$) and if the rating value was not 5 (1). For example, if $\alpha = 0.5$ the synthetic data set has half of the ratings increased or decreased according to the value of c.

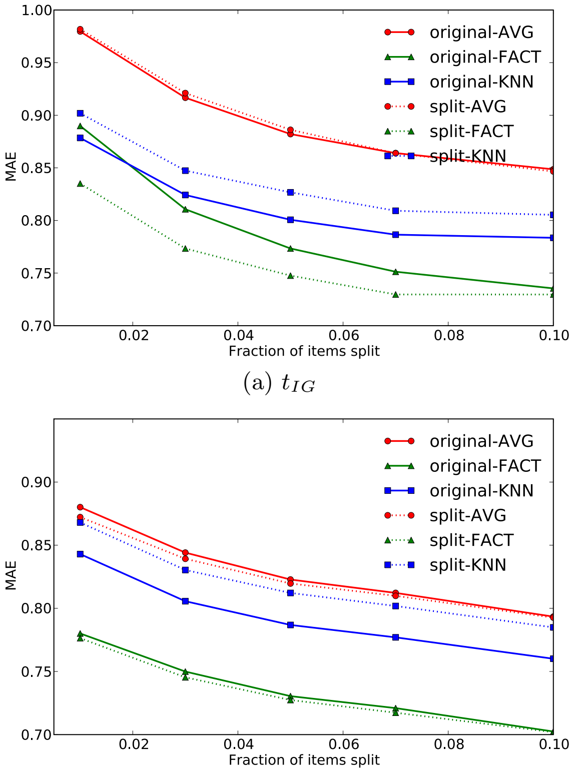
We computed the rating predictions with three techniques: user-based CF (*KNN*), matrix factorization (*FACT*) and a non-personalized recommendation computed as the item-average (*AVG*). In *KNN* we used Pearson Correlation as user-to-user similarity metric and when making a rating prediction for a user we consider only the neighbors that have rated the target item and have co-rated a minimum of 6 items with the target user [5]. Matrix factorization uses the gradient descent based matrix-factorization algorithm implemented and provided by Timely Development[3]. We used a single validation set to find the best parameters for the two CF methods. *KNN* uses k=30 nearest neighbors for the Yahoo! and synthetic data sets, whereas, *FACT* uses 60 factors and the other parameters are set to the same values optimized for the Netflix data set. To evaluate the described methods we used 5-fold cross-validation and measured the Mean Absolute Error (MAE).

We made an initial statistical analysis of Yahoo! data us-

[1] Webscope v1.0, http://research.yahoo.com/

[2] http://www.grouplens.org

[3] http://www.timelydevelopment.com



(a) t_{IG}

(b) t_{prop}

Figure 2: MAE for item splitting criteria

ing two-proportion z-test. We searched for a single item with the largest variations of ratings. In Yahoo! data when considering the gender feature, the biggest difference (p-value) was found for a romantic story “Chocolate”: the average male rating was 4.2 (60 ratings), and average female rating was 4.8 (83 ratings). The action movie “2 fast, 2 furious” was rated higher by users under 18 years, average 3.9 over 312 ratings, compared to the average of 3.5 over 1026 ratings in the group 18to50, and the average of 3.0 over 42 ratings in the a50 group. This shows that items are rated differently in different demographic groups, however, the differences are not big when considering absolute values. These differences are even smaller for the MovieLens data set.

3.1 Evaluating Item Splitting

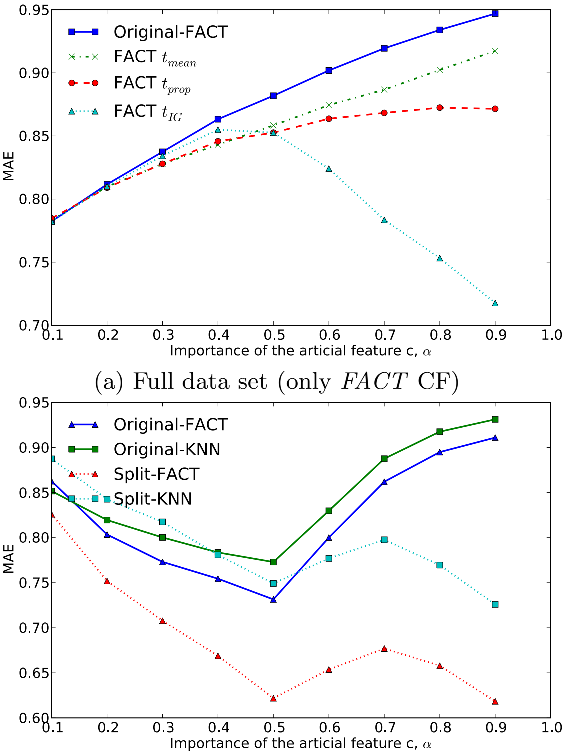
The first experiment investigates the effect of different impurity criteria for choosing the item to split and the contextual variable to use in the splitting. We split an increasing percentage of the items, selecting those with the highest impurity. In order to better show the effect of item splitting we computed the MAE for all the test ratings and those belonging to split items. We compared the rating prediction accuracy using the split data set with that obtained for the same ratings using the original data set (not split). The results for t_{IG} and t_{prop} impurity criteria on the Yahoo! data set are shown in Figure 2. With the exception of t_{IG}, item splitting improves the performance of the non-personalized AVG method (original-AVG vs split-AVG in the figures). When 1% of the items (with highest impurity) are split the improvements are as follows: -0.2% for t_{IG}, 1.1% for t_{prop} 0.8% for t_{size}, 1.0% for t_{mean} and 0.4% t_{random}. The improvements are small and basically indicate that gender and age do not significantly influence the prediction based on the mean of the ratings. In Movielens data AVG predictor is not improved by item splitting. We also observed that KNN was negatively affected by item splitting (both, t_{IG} and t_{prop}), and MAE increased (original-KNN vs split-KNN). We can explain this by observing that each split of the item reduces the number of ratings in the target item profile. We initially optimized the number of nearest neighbors (k parameter) to 30. But, after the split, the target item will have a smaller number of ratings (the average size of an item profile is 19 ratings) and KNN will tend to use all the users that have rated the target (without making any user selection). In fact, to avoid such effect, we should optimize k for each data set separately (pre-processed and original).

Conversely item splitting is strongly beneficial for $FACT$; when splitting 1% of the items with the highest impurity the improvements are as follows: 5.6% for t_{IG}, 0.4% for t_{prop}, 0.6% for t_{size}, 0.7% for t_{mean}, 0.9% for t_{random}. Here, notably the best performance is achieved by t_{IG}. t_{IG} measures the information brought by the contextual variable and is very different from all the other criteria used. Naturally, item splitting also affects the prediction performance for all the other ratings in the data set. We measured the performance of these split criteria in the full data sets. MAE of $FACT$ increased: 0.3% for t_{prop}, 0.3% for t_{size}, 0.3% for t_{mean}, 0.05% for t_{random}. Conversely, when t_{IG} is used and 1% of the items are split MAE for whole data set is decreased by 0.1%. These changes in performance are small, because most of the predictions in the test data set are not affected by pre-filtering when a small amount of items are split.

Thus, the conclusion of this experiment is that item splitting applied to the contextual features (gender, age), in these data sets, has a small effect. Besides, the best performing method is t_{IG}; it can sensibly improve the accuracy of the prediction for ratings belonging to split items and also for the others. We conjectured that these improvements are small because there is not a strong dependency between these contextual features and the rating behavior of the users. Moreover, strictly speaking gender and age are not contextual conditions but features of a user. The splitting according to these features always put the ratings of a user in only one of the artificial items. Thus, the final effect is to discard the opinion of many users while making a prediction. This negative effect was observed in *KNN*.

3.2 Synthetic Data

For better understanding the potential of item split in a truly context-dependent set of ratings we tested this approach on the semi-synthetical data sets described earlier, i.e., adding to gender and age a new contextual feature that does influence the ratings. Figure 3(a) compares the MAE of $FACT$ for three different splitting criteria varying the impact of the context feature c on the ratings. Here, we split the item if the p-value of t_{mean} and t_{prop} tests are lower than 0.05, and when t_{IG} is greater than 0.18 (in the previous experiment these values gave a split of 5% of the items). Figure 3(a) shows that in these context-dependent data sets, increasing the value of α, i.e., increasing the number of ratings that are modified according to the value of the context feature, the overall MAE increases. So the dependency of the ratings from a contextual condition plays the role of noise added to the data, even if this is clearly not noise but a sim-



(a) Full data set (only *FACT* CF)

(b) Only split items (*FACT* and *KNN*)

Figure 3: Effect of synthetic contextual feature

ple functional dependency from a hidden variable. Hence, the conclusion is that this prediction method (*FACT*, but also the others) cannot exploit the additional information brought by this feature and cannot effectively deal with the influence of this variable. Conversely, using item splitting, we can improve the performance of *FACT* (if α>0.3) using all the three mentioned splitting criteria. Note that the three splitting criteria have different behaviors when α increases. t_{mean} and t_{prop} decrease the error also when α is small, however, when α>0.5 t_{IG} outperforms the other two splitting methods.

Finally, Figure 3(b) shows MAE computed only for test ratings belonging to split items. The figure shows that item split is more and more effective with increasing values of α, i.e., when more ratings are influenced by the feature c. We observed that *FACT* constantly benefits from the item splitting. When α is small the difference are small, however, when the feature c gets more important, then the preprocessing of the items pays off. Here, while using item splitting, *FACT* improves the performance even when a big fraction of ratings is modified. The behavior of *KNN* is different (as before for the original Yahoo! data). This method benefits from item splitting only when α>0.4.

4. CONCLUSIONS AND FUTURE WORK

This paper introduces and evaluates a new contextual prefiltering technique for CF, called item splitting. It is based on the assumption that certain items may have different evaluations in different contexts. As a result we observed that despite the increased data sparsity, item splitting is beneficial, when some contextual feature separates the item ratings into two more homogeneous rating groups. However, if the contextual feature is not really affecting the ratings then this technique resulted in just a minor decrease of the prediction error on the split items, and sometimes produced a minor increase of the error in the full data set.

We must observe that the experiments conducted on real world data are limited because they lack true contextually-tagged ratings and therefore we had to rely on demographically tagged data that have several limitations: they are classifying all the ratings of a user in one single context; and appear not be really dependent on these features. The method we proposed can be generalized in several ways. For instance one can try to split the users (not the items) according to the context, so basically to represent with different part of the user profile the preferences of a user in different contexts. Or one can mix these two approaches or search a better criteria for splitting, e.g., based on optimization or cross validation of the prediction error. We observe that splitting items and users can also create easy to explain recommendations since the ratings of a user or item profile could be better related to the target recommendation context, and therefore can be easier to mention as justifications of the recommendations.

5. REFERENCES

[1] G. Adomavicius, R. Sankaranarayanan, S. Sen, and A. Tuzhilin. Incorporating contextual information in recommender systems using a multidimensional approach. *ACM Transactins on Information Systems*, 23(1):103–145, 2005.

[2] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, 2005.

[3] G. Adomavicius and A. Tuzhilin. Context-aware recommender systems. In P. Pu, D. G. Bridge, B. Mobasher, and F. Ricci, editors, *RecSys*, pages 335–336. ACM, 2008.

[4] S. S. Anand and B. Mobasher. Contextual recommendation. In *Lecture Notes In Artificial Intelligence*, volume 4737, pages 142–160. Springer-Verlag, Berlin, Heidelberg, 2007.

[5] S. Berkovsky, T. Kuflik, and F. Ricci. Cross-domain mediation in collaborative filtering. In C. Conati, K. F. McCoy, and G. Paliouras, editors, *User Modeling*, volume 4511 of *Lecture Notes in Computer Science*, pages 355–359. Springer, 2007.

[6] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Statistics/Probability Series. Wadsworth Publishing Company, Belmont, California, U.S.A., 1984.

[7] A. K. Dey. Understanding and using context. *Personal Ubiquitous Comput.*, 5(1):4–7, February 2001.

[8] C. Hayes and P. Cunningham. Context boosting collaborative recommendations. In *In the Journal of Knowledge Based Systems, Volume 17, Issue*, pages 5–6. Elsevier, 2004.

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