Context-Dependent Items Generation in Collaborative Filtering

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

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

ABSTRACT

Collaborative Filtering (CF) exploits users' recorded ratings for predicting ratings on items not evaluated yet. In classical CF each item is modelled by a set of users' ratings not specifying in which contextual conditions the ratings were obtained (e.g., the time when the item was rated or the goal of the consumption). In some domains the context could heavily influence the rating values. Therefore, a single rating for each user and item combination could be insufficient for making accurate predictions. This paper introduces and analyzes a technique, item splitting, for dealing with context by generating new items. In this approach, the ratings' vectors of some items are split in two vectors containing the ratings collected in two alternative contextual conditions. Hence, each split 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 also compared our approach to the classical reduction based context-aware CF approach. We show that item splitting can be beneficial and its performance depends on the splitting criteria and on the influence of the contextual variables on the item ratings. Moreover, we show that item splitting can perform better than the reduction based approach.

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 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 systems 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], and context-aware approaches can be classified into three groups: pre-filtering, post-filtering and contextual modelling [3]. Reduction based approach [1] extended 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-dim. CF matrix with a model of the context comprising a set of features either of the user, or the item, or the evaluation. 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 target item ratings (the item whose rating prediction is sought). Actually, to be precise, 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 statistical evidence that under these two contextual conditions the item's ratings were different, i.e., users evaluate the item differently.

This study also 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 rat-

CARS-2009, October 25, 2009, New York, NY, USA.

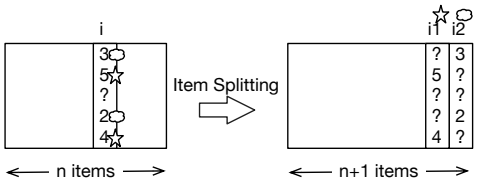


Figure 1: Item splitting

ings, item splitting techniques can help to improve the accuracy of CF, especially with matrix factorization techniques.

2. ITEM SPLITTING

In our approach we extend the traditional CF data model by assuming that each rating r_{ui} in a $m \times n$ users-items matrix, is stored together with some 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 conjecture that the splitting could be beneficial if the ratings within each newly obtained item are more homogenous, or if they are significantly different in the new items coming from a split. One way to accomplish this task is to define an impurity criteria t [6]. So, if there are some candidate splits $s \in S$, which divide i into i_1 and i_2, we choose the split s that maximizes $t(i,s)$ over all possible splits in S. A split is determined by selecting a contextual variable and a partition of its values in two sets. Thus, the space of all possible splits of item i is defined by the context model C. We considered five impurity criteria: t_{mean}, t_{prop}, t_{size}, t_{IG} and t_{random}.

$\boldsymbol{t_{mean}(i,s)}$ impurity criteria is defined using the two-sample t-test and computes how different are the means of the ratings in two rating subsets, when the split s is used. The bigger the t value of the test, the more significant the difference of the means in two partitions is: $t_{mean} = \left| \frac{\mu_{i1} - \mu_{i2}}{\sqrt{s_{i1}/n_{i1} + s_{i2}/n_{i2}}} \right|$ where μ_i is the mean rating of the item i, s_i is the rating variance of item i and n_i is the number of ratings that item i contains.

$\boldsymbol{t_{prop}(i,s)}$ uses the two-proportion z-test and determines whether there is a significant difference between the proportions of high and low ratings in $i1$ and $i2$, when s is used. We consider a rating to be high if its value is 4 or 5, and low if it is 1, 2 or 3 [8]. To test the difference between proportions we use the two-proportion z-test computed as: $t_{prop} = \frac{p_{i1} - p_{i2}}{\sqrt{p(1-p)(1/n_{i1} + 1/n_{i2})}}$ where $p = (p_{i1} n_{i1} + p_{i2} n_{i2})/(n_{i1} + n_{i2})$, p_{i1} (p_{i2}) is the proportion of high ratings in $i1$ ($i2$), and n_{i1} (n_{i2}) is the number of ratings $i1$ (n_{i2}).

$\boldsymbol{t_{size}(i,s)}$ 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 (IG), also known as Kullback-Leibler divergence [9], given by s to the knowledge of the item i rating: $t_{IG} = H(i) - H(i_1)P_{i_1} + H(i_2)P_{i_2}$ where H(i) is the Shannon Entropy of the item i rating distribution and P_{i_1} is the proportion of ratings that i_1 receives from item i.

$\boldsymbol{t_{random}(i,s)}$ 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 one real-world and three 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 data set is enriched with the user age and gender features, here used as contextual variables. We used 3 age groups: users below 18 (u18), between 18 and 50 (18to50), and above 50 (a50). The semi-synthetic data sets were used to analyze item splitting when varying the influence of the context on the user ratings. We modified the original Yahoo! data set by replacing the gender feature with a new artificial feature $c \in \{0, 1\}$ that was randomly assigned to the value 1 or 0 for each rating. 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 already 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 generated three synthetic data sets with $\alpha = 0.1$, $\alpha = 0.5$, $\alpha = 0.9$.

For computing the rating predictions we used three different techniques: user-based CF (KNN), matrix factorization ($FACT$) and a non-personalized recommendation computed

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

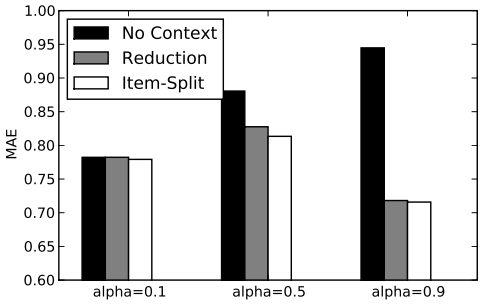


Figure 2: Comparison of contextual pre-filtering methods

as the item-average (AVG). In KNN we used Pearson Correlation Coefficient as user-to-user similarity metric. When making a rating prediction for a user we took into account only the neighbors that have rated the target item and have co-rated a minimum number of items with the target user (six in our case) [5]. Matrix factorization uses the gradient descent based matrix-factorization algorithm implemented and provided by Timely Development[2]. 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).

3.1 Comparison on Semi-Synthetical Data

To understanding the potential of item split in a context-dependent set of ratings we first tested this approach on the semi-synthetical data sets described earlier, i.e., replacing the gender feature with a new contextual variable that does influence the ratings. We compared the results with a classical contextual pre-filtering approach called reduction based [1]. Figure 2 shows comparison of three methods for three semi-synthetic data sets. For each data set ($\alpha = 0.1, \alpha = 0.5, \alpha = 0.9$) we computed MAE for the three prediction methods. The baseline method is FACT that does not take into account any contextual information. It is compared with reduction based and our item splitting technique. Item splitting here uses t_{IG} splitting criteria. For all the three data sets the algorithm splits an item if any split leads to an IG bigger than 0.08. In the three context-dependent data sets, increasing the value of α, i.e., increasing the number of ratings that are correlated to the value of the context feature, caused an increase of the overall MAE of baseline method. Thus, the contextual condition plays the role of noise added to the data, even if this is clearly not noise but a simple functional dependency from a hidden variable. In fact, $FACT$ cannot exploit the additional information brought by this feature and cannot effectively deal with the influence of this variable.

Conversely, using both context-aware CF methods, we can improve the performance of $FACT$ CF. The performance increases substantially when more information is covered by contextual feature c. The biggest improvement is observed when $\alpha = 0.9$. In fact, for this data set reduction based approach improved MAE by 23.9%, whereas, item splitting improved by 24.2%. As we expected, the smaller is the impact of the contextual feature c, the smaller is the increase of the performance. For data set with $\alpha = 0.5$, reduction based approach improved MAE by 6% and item splitting by 7.6%. Reducing α to 0.1, reduction based approach did not find any contextual segment which could increase the performance of the prediction, using the full data set. The item splitting approach improved the performance by 0.4%. Note that both context-aware methods get the largest MAE for $\alpha = 0.5$, and the smallest for $\alpha = 0.9$. We would rather expect that MAE should be linearly dependent on the context effect (α). In fact, there are two factors influencing the rating of an item: the users' profiles and the context. We conjecture that the current solutions may not be the optimal in fitting and balancing these two factors. Hence, we believe, that there could be more effective algorithms for discovering the rating dependency from the context.

These experiments show that both context-aware pre-filtering approaches outperform the base line $FACT$ CF method, especially when the context strongly influences the ratings. It is worth noting that item splitting is computationally cheaper and performed slightly better than reduction based. The time complexity of item splitting is linear to the number of items in the data set and depends on the space of all possible splits S. On the contrary, reduction based approach searches through all the (exponentially growing) segmentations of the data that improve the accuracy of the prediction. Moreover, item splitting method stores only a single model to generate the predictions. Whereas, in reduction based each significantly better segment (and its model) must be stored separately and loaded when needed to make a prediction. Besides, the differences in MAE between these two approaches are small (up to 1.6 percent points of improvement), and could depend on the particular baseline prediction algorithm, i.e., FACT in our experiments. However, we choose FACT as it is currently largely used since it normally outperforms traditional user-based or item-based CF methods.

To better understand both methods we further analyzed the prediction processes. Reduction based approach searches all the possible contextual segments to find out where the prediction using segmentation data is beneficial. It generates rules showing which segment should be used for the prediction. The rule set for $\alpha = 0.1$ data set is empty implying that the algorithm should always use the full data to generate any prediction. It means that there is no contextual segment that improves the accuracy of the prediction. The rules for $\alpha = 0.5$ data set suggests that for all the tuples where the attribute c has value "1" (increased rating) the corresponding contextual segment should be used. The average MAE (for 5 folds) of prediction using full data set within that segment was 0.864 comparing to MAE of 0.758 if only the segment data would be used. When the contextual feature provides a stronger influence, then reduction based partitions the data into segments according to this feature. In fact, when $\alpha = 0.9$ reduction based partitions the data using the artificial feature c, and always uses segmented data for the prediction. For item splitting we looked at the number of items the algorithm splits and also on which attribute

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

	Male	Female	u18	18to50	a50
#ratings	158,507	52,224	45,084	157,844	7,803
mean	4.03	4.23	4.17	4.06	3.99

Table 1: Rating statistics for different demographic groups

the split was performed. When $\alpha = 0.1$, the algorithm splits 475.1 (5%) items (on average in 5 folds). The age attribute was chosen 157.3 times and the artificial attribute was chosen 318.2 times. When $\alpha = 0.9$, the algorithm splits 1183.2 (10%) items on average. For this data set the age attribute was chosen only 59.6 times and artificial attribute was chosen 1124.2 times. Note, that despite IG favors attributes with many possible values [9] item splitting chooses the attribute having larger influence on the rating. We further observe that the number of split items is not large. However, each split of an item affects also the prediction for the items that are not split. Splitting an item is equivalent to create two new items and deleting one, therefore, it causes a modification of the data set. When CF generates a prediction for a target user-item pair all the other items' ratings, including those in the new items coming from some split, are used to build that prediction. In conclusion, we could regard item split as a more dynamical version of reduction based. Here the split is done for each item separately and using an external measure (such as IG) to decide if the split is needed.

3.2 Comparison on Original Yahoo! Data

Initially, we made a general statistical analysis of the Yahoo! data. We used the two-proportion z-test (as described above) and observed if the ratings between two genders and different age groups are significantly different. For the Yahoo! data set the biggest statistical difference was obtained for the gender attribute (z-score 25.8). The summary of the Yahoo! data statistics is showed in the Table 1. The two-proportion z-test statistics shows that different demographic groups rate movies differently. However, the difference in the means of the ratings are small and most of the time can be captured by the underlying CF algorithm. In fact, in this case we observed that reduction based could not find any segment that improves the prediction accuracy of the baseline approach. Therefore, the prediction accuracy of the reduction based approach for this data set is the same as any standard method without contextual pre-filtering.

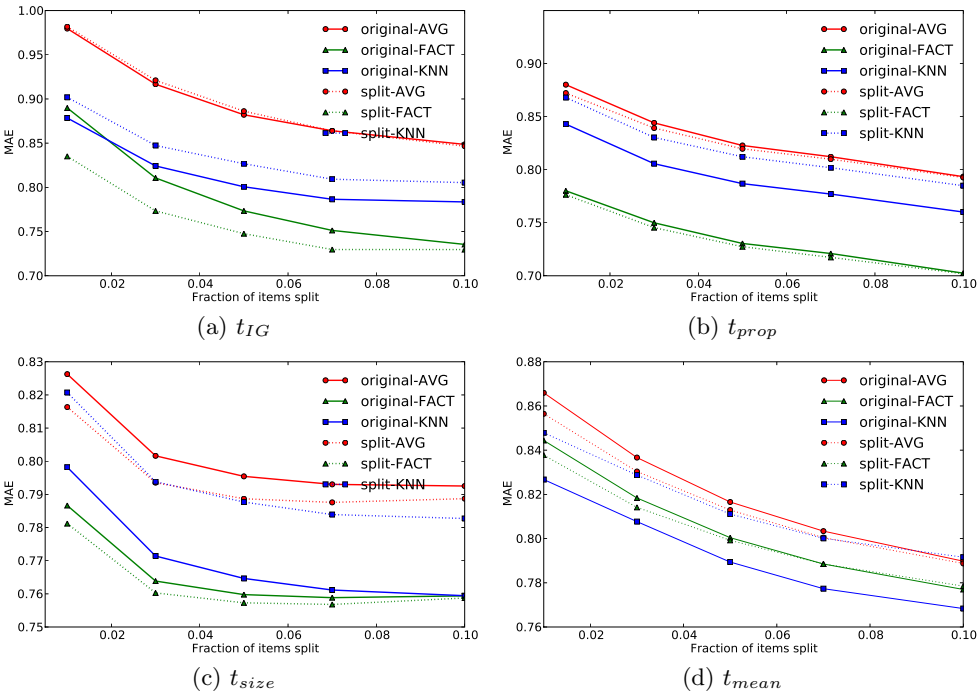
Conversely, when using item splitting with t_{IG} criteria pre-filtering slightly improves the performance of the *FACT* CF algorithm. When splitting 1% of the items with the highest impurity the MAE computed for the whole data set is decreased by 0.1%. The change is small, because most of the predictions in the test data set are not affected by pre-filtering since a small amount of items are split. When splitting more items we observe a decrease in the overall performance. This can be explained by the fact that there is no strong functional dependency between gender, age and the rating. Therefore, splitting the items using an arbitrary split (if there is no strong evidence to support the split) is not beneficial. Moreover, splitting items makes data more sparse and the computation of item-to-item correlation could become unreliable. We also measured the performance of the other proposed split criteria on the full data set. MAE of *FACT* increased: 0.3% for t_{prop}, 0.3% for t_{size}, 0.3% for t_{mean}, 0.05% for t_{random}.

Thus, item splitting has a small effect when applied to the contextual features (gender, age) in the Yahoo! data set. Besides, the only splitting criteria that improved the accuracy in the full data set is t_{IG}; it can sensibly improve the accuracy of the prediction for ratings belonging to split items (as it will be shown later) and also for the others. These improvements could be small because there is not a significant dependency between these contextual features and the rating behavior of the users. In fact, as we have shown in the previous section, when the dependency from the context is stronger then item splitting is more effective. 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.

To emphasize the effect of item splitting using different split criteria, we computed MAE only on the items that were split. We split an increasing percentage of the items, selecting those with the highest impurity. 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 various impurity criteria on the Yahoo! data set are shown in Figure 3. 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 this should basically depends on the fact that gender and age do not significantly influence the prediction. 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.

4. CONCLUSIONS AND FUTURE WORK

This paper evaluates a contextual pre-filtering technique for CF, called item splitting. Based on the assumption that certain items may have different evaluations in different contexts, we proposed to use item splitting to cope with this. The method is compared with a classical context-aware pre-filtering approach [1] which uses extensive searching to find contextual segments that improve baseline prediction. As a



(a) t_{IG} (b) t_{prop}

(c) t_{size} (d) t_{mean}

Figure 3: Item splitting for various criteria

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 influential this splitting 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. Item-splitting outperforms reduction based context-aware approach when *FACT* CF method is used. Moreover, the method is more time and space efficient and could be used with large context-enriched data bases.

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 extended in several ways. For instance one can try to split the users (not the items) according to the contextual features in order to represent the preferences of a user in different contexts by using various parts of the user profile. Another interesting problem is to find a meaningful item splitting in continuous contextual domains such as time or temperature. Here, the splitting is not easily predefined but have to be searched in the continuous space. Finally, item splitting could ease the task of explaining recommendations. The recommendation can be made for the same item in different context. The contextual condition on which the item was split could be mentioned 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] 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.

[9] J. R. Quinlan. *C4.5: Programs for Machine Learning (Morgan Kaufmann Series in Machine Learning)*. Morgan Kaufmann, 1 edition, January 1993.