Accuracy Improvements for Multi-criteria Recommender Systems

DIETMAR JANNACH, TU Dortmund, Germany
ZEYNEP KARAKAYA, TU Dortmund, Germany
FATIH GEDIKLI, TU Dortmund, Germany

Recommender systems (RS) have shown to be valuable tools on e-commerce sites which help the customers identify the most relevant items within large product catalogs. In systems that rely on collaborative filtering, the generation of the product recommendations is based on ratings provided by the user community. While in many domains users are only allowed to attach an overall rating to the items, increasingly more online platforms allow their customers to evaluate the available items along different dimensions. Previous work has shown that these criteria ratings contain valuable information that can be exploited in the recommendation process.

In this work we present new methods to leverage information derived from multi-dimensional ratings to improve the predictive accuracy of such multi-criteria recommender systems. In particular, we propose to use Support Vector regression to determine the relative importance of the individual criteria ratings and suggest to combine user- and item-based regression models in a weighted approach. Beside the automatic adjustment and optimization of the combination weights, we also explore different feature selection strategies to further improve the quality of the recommendations.

An experimental analysis on two real-world rating datasets reveals that our method outperforms both recent single-rating algorithms based on matrix factorization as well as previous methods based on multi-criteria ratings in terms of the predictive accuracy. We therefore see the usage of multi-criteria customer ratings as a promising opportunity for e-commerce sites to improve the quality and precision of their online recommendation services.

Categories and Subject Descriptors: H.3.3 [**Information Search and Retrieval**]: Information filtering

Additional Key Words and Phrases: Recommender systems, Multi-criteria Ratings, Machine Learning

1. INTRODUCTION

The provision of personalized shopping recommendations in online stores has shown to be a valuable way to help customers find interesting items and to consequently increase customer loyalty and sales. Recent studies, for example, showed that using a recommender system (RS) can lead to increased sales volumes in the short term and in the long term or help to increase sales diversity by directing customers to other parts of the available product catalog [Senecal and Nantel 2004; Zanker et al. 2006; Fleder and Hosanagar 2007; Dias et al. 2008; Jannach and Hegelich 2009].

Over the last two decades, various approaches have been explored to build effective recommender systems [Jannach et al. 2010]. Among them, systems based on collaborative filtering (CF) are particularly popular and used by large online retailers such

Author's addresses: D. Jannach, Computer Science Department, TU Dortmund, Germany; Z. Karakaya, Computer Science Department, TU Dortmund, Germany; F. Gedikli, Computer Science Department, TU Dortmund, Germany.

EC'12, June 4–8, 2012, Valencia, Spain. Copyright 2012 ACM 978-1-4503-1415-2/12/06...$10.00.

as Amazon.com [Linden et al. 2003] or by the DVD-rental service Netflix[1]. Pure CF-based recommender systems rely solely on product ratings provided by a large user community to generate personalized recommendation lists for each individual online visitor.

In traditional CF systems the assumption is that customers provide an *overall rating* for the items which they have purchased, for example, using a 5-star rating system. However, given the value of customer feedback to the business, customers in some domains are nowadays given the opportunity to provide more fine-grained feedback and to rate products and services along various dimensions. A typical example is the tourism domain, where on most large booking platforms customers can rate hotels with respect to cleanliness, breakfast service, value for money or staff friendliness. Recently, also Amazon.com introduced an evaluation dimension ("fun") for video games and eBay now lets customers rate sellers in various dimensions.

In the context of product recommendation, the question arises how this more fine-grained information about customer preferences can be exploited to generate more accurate product suggestions. One of the first proposals in that direction was by Adomavicius and Kwon. In [Adomavicius and Kwon 2007] they introduced two basic schemes of incorporating multi-criteria rating information in the recommendation process. Their first experiments in the movie domain showed that their approaches led to more precise recommendations when compared with traditional, single-rating nearest-neighbor approaches. Our paper continues this line of work and introduces further techniques to substantially improve the predictive accuracy of multi-criteria recommender systems. The contributions of our paper can be summarized as follows.

(1) In contrast to previous work, we propose to use Support Vector (SV) regression for automatically detecting the existing relationships between detailed item ratings and the overall ratings. According to our experiments, SV regression not only leads to higher predictive accuracy but also allows us to handle very sparse datasets.

(2) We propose to learn such regression models not only per item but also per user and to combine the individual predictions in a weighted approach. In particular, we also propose to derive individual weights per user and item using a fast optimization procedure.

(3) We show that different feature-selection strategies can help to further improve the predictive accuracy and that it can be better *not* to include all available ratings in the process as some features may contain noise.

(4) We conduct an in-depth experimental evaluation, which is based on a multi-criteria rating dataset obtained from a commercial hotel booking platform. We compare the accuracy of different algorithms and show that our new method is capable of outperforming recent single-rating approaches based on matrix factorization and existing algorithms that rely on multi-criteria ratings. In order to validate our findings in a second domain, we conduct additional experiments with a multi-criteria movie-ratings dataset.

In the next section we will describe algorithmic improvements for the multi-criteria RS and compare our work with previous approaches. Afterwards, we present the results of our empirical evaluation.

2. RECOMMENDING BASED ON MULTI-CRITERIA RATINGS

Traditional (single-rating) CF recommenders are characterized in [Adomavicius and Kwon 2007] as systems that try to estimate a rating function R: $Users \times Items \rightarrow R_0$

[1] http://www.netflix.com

for predicting a rating for a given user-item pair. R_0 is a totally ordered set, typically consisting of real-valued numbers in a certain range, e.g., from 0.5 to 5.

In multi-criteria RS, in contrast, the rating function R has the form $Users \times Items \rightarrow R_0 \times R_1 \times .... \times R_k$. Therefore we have to predict an overall rating R_0 as well as k additional criteria ratings. In the single-rating case the rating function R is estimated based on a sparse user-item rating matrix. In the multi-criteria case, the rating database contains a sparse matrix of the overall ratings, and the detailed criteria ratings of the user community. Table I shows an example of a multi-criteria rating database for the movie domain.

Table I. Multi-criteria rating database fragment for Yahoo!Movies.

Row	User	Item	**Overall**	Acting	Story	Visuals	Directing
1	u1	i1	**4**	3	3	5	5
2	u1	i2	**3**	2	2	4	2
3	u2	i1	**4**	5	5	3	4
4	u2	i2	**5**	5	4	3	5

The two basic schemes proposed in [Adomavicius and Kwon 2007] are called *similarity based* and *aggregation function based*. The general idea of these schemes can be summarized as follows.

2.1. Similarity-Based Approaches

In traditional neighborhood-based CF approaches, the rating prediction for a user u and a target item i is based on the ratings for i provided by users which are similar to u. The similarity between users is determined based on their past rating behavior and a measure such as the Pearson correlation coefficient. The idea of similarity-based *multi-criteria* recommendation approaches is to rely on more fine-grained similarity measures which are based on the users' preferences for different aspects of an item. If we consider the data given in Table I, both user $u1$ and user $u2$ gave an overall rating of 4 to item $i1$. A closer look however reveals that they liked different aspects of the movie so they might actually not be as similar as one would suspect if only the overall ratings were available.

In [Adomavicius and Kwon 2007], different potential ways were proposed to measure the similarity between users based on their detailed ratings. The user's ratings for an item can for example be viewed as k-dimensional vectors so that standard multidimensional distance metrics such as the Euclidian or the Chebyshev distance can be calculated. Alternatively, one could calculate the Pearson correlation coefficient for each rating dimension individually and take the average or smallest value as an overall similarity metric.

For computing the final rating prediction, a standard nearest-neighbor approach can be used. The only difference between a single-rating approach and similarity based multi-criteria recommendation approaches therefore lies in the usage of a different similarity metric.

2.2. Aggregation Function Based Approaches

In this second scheme, the general and intuitive assumption is that there exists a relationship between the overall item ratings and the individual criteria ratings. Technically, the overall rating r_0 can therefore be seen as being determined by a function f of individual criteria ratings:

$$r_0 = f(r_1, ..., r_k) \tag{1}$$

The prediction of r_0 for a given user u and a target item i can be accomplished in a multi-step process. First, in an offline phase, the function f has to be determined. One option could be to define f based on domain expertise or by averaging the criteria ratings. A more promising approach, however, is to apply statistical or machine learning techniques to automatically detect the hidden relationship between the overall rating and the criteria ratings. For example, in [Adomavicius and Kwon 2007] it is proposed to approximate a function f for each item in the catalog using multiple linear regression techniques. Thus, the overall rating r_0 can be viewed to be dependent on a linear combination of the criteria ratings, where each criterion is assigned a weight w_i, that is

$$r_0 = w_1 r_1 + ... + w_k r_k + c \tag{2}$$

where the weights w_i and the constant c are estimated from the data.

In the online phase, first the criteria ratings r_1 to r_k for i have to be estimated. Afterwards, the overall rating can be calculated using f. For estimating r_1 to r_k, any standard CF algorithm can be used separately for each criteria. In the movie domain, for example, we can view the prediction of the rating for the *story* aspect to constitute its own CF problem, which we compute based on the ratings for this aspect of the neighbors of the target user.

Figure 1 summarizes the different computation phases of the aggregation function based approach. The methods proposed later on in this paper follow this aggregation function based approach. The improvements suggested in this paper affect the first and the third step shown in the figure.

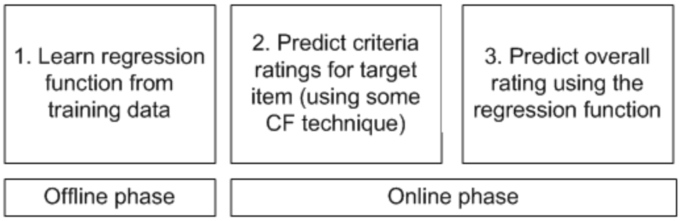
1. Learn regression function from training data
2. Predict criteria ratings for target item (using some CF technique)
3. Predict overall rating using the regression function
Offline phase
Online phase

Fig. 1. Overview of aggregation function based approaches.

2.3. Improving the accuracy of multi-criteria recommender systems

An evaluation of the above-mentioned two multi-criteria recommendation schemes on a comparably small dataset from Yahoo!Movies in [Adomavicius and Kwon 2007] showed that at least for a relatively dense dataset[2] both the similarity-based as well as an aggregation function based approach using linear regression outperformed a classical user-based CF algorithm.

In our work, we propose new methods to substantially improve the predictive accuracy of multi-criteria RS. We will compare these methods with recent top-performing matrix factorization techniques and analyze the algorithm's performance also on lower-density datasets.

[2] Each user had at least rated 10 items and each item was rated by at least 10 users.

2.3.1. Using Support Vector regression. As mentioned above we will follow an aggregation function based approach. The first improvement we suggest is related to the choice of the method to learn the regression functions. In contrast to [Adomavicius and Kwon 2007], who used a multiple linear least squares regression method which is based on Gentleman's algorithm [Gentleman 1974], we propose to rely on Support Vector (SV) regression [Drucker et al. 1997] to learn the regression functions as shown in Equation 2. One reason for this choice is that SVM-based regression has shown higher predictive accuracy and a smaller tendency for over-fitting in other recommendation scenarios, see [Sen et al. 2009] or [Gedikli and Jannach 2012]. In addition, SV-regression has the advantage that it works when only a few data points, and many rating dimensions, are given. This is typical, for example, in the tourism domain.

2.3.2. Combining user- and item-based models. While the works described in [Adomavicius and Kwon 2007] and [Sen et al. 2009] rely on learning a regression function for every item, we propose to additionally learn a regression function for each user and combine the two predictions in the online phase in a weighted approach.

The difference between the two models can be illustrated using the data in Table I. In an *item-based approach*, the multi-criteria rating database is split according to the items. A regression function for item $i1$ is therefore learned from rows 1 and 3 of the table and one for item $i2$ is estimated based on rows 2 and 4. In other words, we try to learn how strong the individual aspects of a movie influence the overall movie ratings across all users. In a *user-based* regression approach, models are learned per user, that is, we use rows 1 and 2 for learning a model for user $u1$ and rows 3 and 4 for user $u2$. Thus we can try to learn for each user which aspects of a movie are of particular importance across all the movies the user has rated.

Overall, we try to estimate two types of dependencies in the rating database. Note that in most recommendation domains, the rating databases can be very sparse, which means that models have to be learned from very few samples as users often only rate a few items and items are only rated by a few people. The intuition behind our hybridization and weighting approach is that both types of information can be valuable. Consider a case where the target item has not been rated by many users and the item-based regression model is therefore based on only a couple of ratings. If at the same time we know much about the target user, we could rely more on this user's general rating behavior (based on the user-based regression model) instead of using the few ratings that we have for the item. We therefore assume in general that more accurate predictions are possible when we combine the two predictions that we can calculate for a particular user-item pair.

In principle, various ways of combining the two predictions are possible. A general weighting scheme is shown in Equation 3, where $\hat{r}^{user}$ corresponds to a prediction depending on a user-based regression function, $\hat{r}^{item}$ to an item-based one, and $\hat{r}$ corresponds to the overall prediction.

$$\hat{r}_{u,i} = w_u * \hat{r}_{u,i}^{user} + w_i * \hat{r}_{u,i}^{item} \qquad (3)$$

In such a scheme, w_u and w_i could take values from 0 to 1 and we could set w_i to $(1 - w_u)$ in order to ensure that the predicted ratings remain within the allowed range. Alternatives to this scheme could use the (harmonic) mean of the user- and item-based predictions as an overall prediction.

However, instead of relying on global single value for w_u and w_i (which could be determined empirically or through optimization) or a static weighting scheme, we propose to determine individual and optimized weights for each single user and item. An optimal weight in that context is one that minimizes the prediction error, which is calculated as the difference between the predicted and the actual rating.

Similar to the work in [Koren 2010], [Gedikli et al. 2011] and other recent approaches in the field, we suggest to use a fast, heuristic gradient descent procedure to efficiently estimate parameters for each user and item. Algorithm 1 sketches the general scheme that is used to compute w_u and w_i for all users and items.

The algorithm works as follows. Before the iterative optimization process begins, the user and item weights are initialized with values around 0.5. In each iteration, we compute a prediction $\hat{r}_{u,i}$ using the user- and item based predictors and their current weights for each user-item rating pair. Next, the prediction error $e_{u,i}$ is determined. This prediction error is then used to slightly change the values for w_u and w_i in the direction in which we expect an improvement. The parameters γ determines the size of the correcting step and λ is used for regularization and to avoid overfitting. Suitable values for our experiments were determined based on values from literature and were fine-tuned manually.

Algorithm 1 Gradient descent calculation of weights.

```
Require: $\#iterations$, $\gamma$, $\lambda$
  // Gradient descent iterations:
  for $1$ to $\#iterations$ do
    for each user $u$ do
      for each rated item $i$ of user $u$ do
        // compute prediction with current weights
        $\hat{r}_{u,i} \leftarrow w_u \cdot \hat{r}^{user}_{u,i} + w_i \cdot \hat{r}^{item}_{u,i}$
        // compare with real rating $r_{u,i}$ and determine the error $e_{u,i}$
        $e_{u,i} \leftarrow r_{u,i} - \hat{r}_{u,i}$
        // Adjust $w_u$ in gradient step
        $w_u \leftarrow w_u + \gamma \cdot (e_{u,i} - \lambda \cdot w_u)$
        // Adjust $w_i$ in gradient step
        $w_i \leftarrow w_i + \gamma \cdot (e_{u,i} - \lambda \cdot w_i)$
      end for
    end for
  end for
  return $w_u$ for each user $u$ and $w_i$ for each item $i$
```

2.3.3. Feature selection. Beside the described algorithmic improvements we also propose to apply feature selection strategies to improve the prediction accuracy in cases where there are many rating dimensions. On modern tourism platforms, for example, customers can rate holiday packages or hotels along quite a number of different dimensions. In our experimental analysis, we will use a dataset obtained from HRS.com, one of the largest European hotel booking platforms. In this dataset, customer ratings for up to twenty different criteria are available. While all these ratings may carry relevant preference information, we consider that this comparably large number of rating criteria may also lead to noise in the data, for example, when customers do not understand the meaning of an individual criterion or simply pick arbitrary values. For that reason, we propose to take only a subset of the available rating dimensions into account for the prediction task. Following the usual terminology in machine learning, we call this process feature selection.

We designed and evaluated the following heuristic strategies to select the optimal set of features (rating dimensions):

— ST1: This strategy is based on the incremental addition of features based on a relevance metric. In a first step, we therefore calculate the relevance for each feature e.g. based on the chi-square statistic [Liu and Setiono 1995]. Starting from the feature with the highest relevance, we incrementally add features to be taken into consideration in order of decreasing relevance. In each step, we measure the prediction error (using the RMSE) and finally choose the feature set which led to the minimal prediction error[3].
— ST2: The strategy is similar to ST1 but is different from it insofar as we remove a feature from the feature set when we observe that it led to a deterioration of the RMSE in the last step.
— ST3: Beside the basic and greedy schemes ST1 and ST2, we also ran experiments with an "optimal" feature set that was determined based on an evolutionary algorithm (EA). We used a genetic feature selection algorithm in which the mutation step corresponds to the inclusion or exclusion of a feature and the crossover step reflects the exchange of features. We applied an SVM learner with a radial kernel type to learn the target function based on the selected features and evaluated the model using five-fold cross-validation using the RMSE metric. The RMSE values were iteratively used as new input for the EA and thus influenced the feature selection process. We used the standard implementation from the RapidMiner[4] data-mining system and default parameters for the evolutionary process [Mierswa 2009].

2.4. Related approaches

Before giving the details of our experimental evaluation, we will shortly summarize other existing approaches to building multi-criteria RS and describe their relation to our work. A recent overview on research in multi-criteria RS can be found in [Adomavicius et al. 2011]. A systematic classification of such multi-criteria recommender systems and, more generally, systems for multi-criteria decision making and optimization is provided by [Manouselis and Costopoulou 2007].

In the context of more traditional recommender systems, [Adomavicius et al. 2011] identify the following categories of existing systems which have some form of a multi-criteria nature:

(1) Systems such as classical content-based recommenders, which try to learn content-based preferences based, for example, on the given overall ratings for the items. In classical information retrieval (IR) scenarios, the learned user profile for instance consists of a vector containing a relevance-weighted list of the terms appearing in the documents.
(2) Systems for content or item retrieval that allow users to state their *general* preferences using a set of (predefined) categories. Such approaches are common in knowledge-based [Jannach and Kreutler 2005; Jannach 2006; Jannach et al. 2009] or critique-based recommendation systems [Burke 2002].
(3) Multi-criteria rating recommenders, in which users are allowed to specify their preferences (ratings) *for individual products* along different dimensions as described in the previous sections.

The work presented in this paper falls into the third category. In the following, we will therefore limit our discussion to previous works which also belong to this category.

In [Sahoo et al. 2011] and [Sahoo et al. 2006], Sahoo et al. propose to extend the Flexible Mixture Model (FMM) for collaborative filtering proposed in [Si and Jin

[3] Note that also other relevance metrics based on correlation or the Gini impurity index are possible. Our experiments however showed that the differences between the various metrics are minimal.
[4] http://www.rapidminer.com

2003] for multi-criteria rating[5] scenarios. The main idea of their approach is to detect existing dependency structures within the criteria ratings, embed these dependencies in the probability calculations and estimate the distribution parameters using the Expectation-Maximization algorithm. An analysis of their method on the Yahoo!Movies dataset revealed that the incorporation of the criteria ratings led to better MAE (Mean Absolute Error) values for low data density levels when compared with an FMM approach, which only considers the overall rating. An improvement of the predictive accuracy on the precision/recall metric could also be observed for low-density configurations.

A related probabilistic approach was later on presented by Zhang et al. in [Zhang et al. 2009], who extend the Probabilistic Latent Semantic Analysis (PLSA) model [Hofmann 2004] to the multi-criteria rating case in two different variations. An evaluation also performed on a dataset with special characteristics retrieved from Yahoo!Movies showed that their method outperforms a single-rating item-based algorithm using Pearson correlation as a similarity metric. Since we evaluate our algorithm also on a dataset crawled from Yahoo!Movies, we will compare our results also with the ones observed by Zhang et al.

In [Li et al. 2008], Li et al. propose to apply multi-linear singular value decomposition (SVD) to exploit context information about the user as well as multi-criteria ratings in the recommendation process. An experimental evaluation in a restaurant recommendation scenario indicates that higher precision and recall values can be achieved when compared with a comparably weak baseline method that does not rely on multi-criteria ratings. While we cannot directly compare their results with our experimental findings as the dataset is not publicly available, we will compare our methods with state-of-the-art single-rating algorithms based on matrix factorization which in our view represent a harder baseline than traditional user- or item based nearest-neighbor approaches.

Recently, Liu et al. presented a multi-criteria recommendation approach which is based on the clustering of users [Liu et al. 2011]. Their idea is that for each user one of the criteria is “dominant” and users are grouped according to their criteria preferences. A prediction for a user is then based on the ratings of other users belonging to the same cluster. To determine the importance of the different criteria, they apply linear least squares regression, assign each user to one cluster, and evaluate different schemes for the generation of predictions. An evaluation on a hotel-booking dataset from TripAdvisor shows that significant accuracy improvements with respect to the MAE can be achieved.

In some sense, our work shares the idea with [Liu et al. 2011] and take the user-specific ranking of the different rating criteria into account. Generally, we view the clustering approach to be complementary to our method. An opportunity for future work is therefore to combine the predictions of such a clustering-based approach with our method in a hybrid approach. With respect to the achieved improvements in [Liu et al. 2011], note that their non-public dataset was preprocessed such that all users rated at least 20 hotels. In real-world datasets, however, only a tiny percentage of users have rated more than 10 hotels. In our work we will show that accuracy improvements can also be achieved using our method for very sparse datasets. In addition, while in [Liu et al. 2011] the baseline method is a traditional single-rating CF method, we compare our work with more recent algorithms.

[5] Sahoo et al. call them “multi-component” ratings.

3. EVALUATION

In order to analyze the effectiveness of the proposed methods, we have conducted several experiments on two datasets using different algorithms and quality metrics.

3.1. Datasets

We used multi-criteria rating datasets from two different domains in our evaluation:

- The hotel rating dataset (HRS): The first dataset was provided by the hotel booking platform HRS. As mentioned above, it contains criteria ratings in more than a dozen dimensions (using a 1-to-10 scale) and an overall comfortableness rating on a 1-to-3 scale. We can observe that the data is very sparse and that only very few ratings per user and item are available. The reason for that lies in the nature of the domain where the majority of customers do not book and rate many different hotels.
- The Yahoo!Movie dataset (YM): In absence of a public benchmark dataset for the movie domain, we collected rating data from the Yahoo!Movies website with a web crawler. All considered movies are part of the single-rating dataset provided by Yahoo!Research[6]. On the Yahoo!Movies platform, users can rate movies in 4 dimensions (Story, Acting, Direction, Visuals) and also assign an overall rating. A 13-level rating scale (from A+ to F) is used.

  We transformed this 13-level rating scale into the typical 5-point scale for most experiments since our aim is to make the accuracy results comparable with previous work and other experiments from literature[7].

In most previous research (not only on multi-criteria RS), such raw and "noisy" datasets are usually pre-processed. Items for which only few ratings are available are filtered out along with users, who have only issued very few ratings. In [Adomavicius and Kwon 2007], for example, only movies are considered for which at least 10 ratings exist; at the same time, users must have rated at least 10 items to be considered in the evaluation. We pre-processed our datasets accordingly and created the test datasets with different density and quality levels. The quality level was varied by adding constraints on the minimum number of ratings per user and item. The datasets are summarized in Table II. HRS-3-3, for example, stands for an HRS dataset where each user rated at least 3 items and each item was rated by at least 3 users. HRS-RAW stands for an un-preprocessed dataset.

Table II. Overview of the dataset characteristics.

Name	#Users	#Items	#Overall ratings
HRS-5-5	1,162	1,203	9,712
HRS-3-3	1,768	1,762	10,347
HRS-RAW	1,582	2,277	4,564
YM-20-20	429	491	18,504
YM-10-10	1,827	1,471	48,026
YM-5-5	5,978	3,079	82,599

Note that the datasets are not equally large. We used random subsampling to create subsets of comparable sizes from the massive raw dataset to measure the effect of different density levels in isolation. Otherwise the raw dataset would have contained much more training data than the others, which would represent a different measurement. Furthermore, we must not directly compare the observed accuracy numbers across different datasets as the performance of most algorithms improves simply when

[6] http://webscope.sandbox.yahoo.com/

[7] The transformation is straightforward: A-ratings (A+,A,A-) correspond to 5 stars, B-ratings to 4 stars etc.

there is more data. Regarding the absolute size of the datasets, we decided to run our various tests on comparably small subsets of the overall data for a faster evaluation process. Experiments with larger subsets lead to slightly different absolute numbers but no differences in the ranking of the algorithms or the significance of the observed differences.

3.2. Algorithms

We compared the following algorithms in our experiments.

- **SlopeOne:** We use this single-rating CF algorithm from [Lemire and Maclachlan 2005] as the assumed lowest baseline. Its performance is comparable to traditional user-based nearest-neighbor approaches, however, the computations can be done more efficiently.
- **Funk-SVD:** This is a relatively recent, single-rating CF algorithm based on matrix factorization (Singular Value Decomposition) which has shown to lead to highly accurate results in the Netflix competition[8]. We also ran experiments with Koren's recent factorized neighborhood algorithm (item-based version) [Koren 2010] and obtained results which were comparable with those achieved with FunkSVD. However, Koren's algorithm requires that the step-size and regulation parameters are fine-tuned for each dataset to achieve good results. We therefore relied on Funk-SVD, which appeared to be more stable across the datasets. We used the following algorithm parameters which we determined empirically and which led to good results across the different settings: number of neighbors: 30, initial steps: 5.
- **MC-Similarity:** A similarity-based multi-criteria rating approach as described in Section 2.1. We used the worst-case similarity as discussed in [Adomavicius and Kwon 2007] to measure the similarity between users because this variant exhibited the best performance in our experiments. Note that in [Adomavicius and Kwon 2007], the Chebyshev distance metric performed best, however, we could not reproduce this result in our experiments. We used $n = 50$ as the number of neighbors.
- **LS-regress-*:** An aggregation function based approach using Ordinary Least Squares regression, which roughly corresponds to the item-based linear regression method from [Adomavicius and Kwon 2007]. We used an implementation that applies QR-decomposition and is implemented in the Apache Commons Math Java library[9]. We experimented with both user-based (LS-regress-U) and item-based regression models (LS-regress-I).
- **SV-regress-*:** The Support Vector regression approach based either on users (-U) or items (-I). We used the Java version of the `libsvm` library[10] in our experiments and $c = 0.15$ as the error penalty parameter in all experiments.
- **WeightedSVM:** This is our new method which combines the estimates of SV-regress-I and SV-regress-U in a weighted approach. We used the following empirically determined meta-parameters for the heuristic optimization of the weights across all datasets. Number of iterations: 20, γ: 0.002, λ: 0.08.

For predicting the unknown criteria ratings for the target item in all aggregation function based methods (LS-regress-*, SV-regress-*, WeightedSVM), we relied on a nearest-neighbor CF method with the Pearson correlation coefficient as a similarity measure. Interestingly, experiments using a matrix factorization method performed poorly in this task. When individual criteria ratings were missing for some items, we used the average of the other criteria ratings as an estimate of the user's rating.

[8] http://sifter.org/˜simon/journal/20061211.html

[9] http://commons.apache.org/math/

[10] http://www.csie.ntu.edu.tw/ cjlin/libsvm/

3.3. Evaluation method / metrics

Beside the usual *Root Mean Squared Error* (RMSE) metric, we will report the *F1* measure as the harmonic mean of *Precision* and *Recall* and use the evaluation procedure used also in [Nakagawa and Mobasher 2003] or [Gedikli and Jannach 2010]. To that purpose, we transform the rating predictions into "like" and "dislike" statements. Ratings above the user's mean rating are interpreted as "like" statements. We then determine the set of Existing Like Statements (ELS) and compare them to the set of Predicted Like Statements (PLS) returned by the recommender, where $|PLS| \leq |ELS|$. *Precision* is defined as $\frac{|PLS \cap ELS|}{|PLS|}$ and *Recall* is measured as $\frac{|PLS \cap ELS|}{|ELS|}$. We use five-fold cross-validation to determine these metrics. For the RMSE measure we split the data into 95% training and 5% test data, used random subsampling and repeated the experiments appropriately to factor out effects of randomness. The reported RMSE values correspond to the average result achieved in 30 test runs[11]. Finally, we report a *coverage* metric, which describes for how many of the user/item pairs the system could generate a prediction.

3.4. Accuracy results without feature selection

Table III shows the average RMSE values for the different quality levels of the datasets after 30 runs. Note again that the achieved RMSE values cannot be directly compared across the datasets because of the different dataset characteristics and densities. However, the predictive accuracy increases when the data quality increases. We furthermore do not report the numbers for the LS-regress-* methods for the HRS-dataset. The LS-regress-* methods were basically not applicable for this dataset as they require that there are at least as many data points as there are dimensions. In the hotel booking domain we, however, have up to twenty dimensions but there are almost no users who have rated 20 hotels.

Table III. RMSE results for the different dataset; coverage numbers given in parentheses.

Algorithm	HRS-5-5	HRS-3-3	HRS-RAW	YM-20-20	YM-10-10	YM-5-5
WeightedSVM	0.52 (1.0)	0.56 (0.99)	0.61 (0.73)	0.57 (1.0)	0.60 (1.0)	0.63 (1.0)
SV-Regress-I	0.59 (1.0)	0.62 (0.99)	0.72 (0.73)	0.66 (1.0)	0.69 (1.0)	0.72 (1.0)
Funk-SVD	0.60 (1.0)	0.64 (0.99)	0.66 (0.73)	0.83 (1.0)	0.87 (1.0)	0.91 (1.0)
SV-Regress-U	0.61 (1.0)	0.66 (0.99)	0.66 (0.72)	0.60 (1.0)	0.65 (1.0)	0.73 (1.0)
SlopeOne	0.68 (1.0)	0.71 (0.99)	0.77 (0.72)	0.81 (1.0)	0.89 (1.0)	0.97 (1.0)
LS-Regress-U	-	-	-	0.65 (1.0)	0.72 (1.0)	0.83 (0.97)
LS-Regress-I	-	-	-	0.70 (1.0)	0.79 (1.0)	0.82 (0.97)
MC-Similarity	0.65 (0.32)	0.71 (0.12)	0.77 (0.31)	0.87 (0.99)	0.93 (0.56)	0.99 (0.24)

We can see in Table III that our proposed method outperforms the other algorithms on all datasets, densities and quality levels in terms of the RMSE. The differences between WeightedSVM and the next best performing algorithms are statistically significant ($p < 0.01$) for all datasets according to a two-tailed paired t-test.

As for the HRS dataset, we can also observe that the matrix factorization approach works comparably well and is as expected clearly better than methods such as SlopeOne. The individual multi-criteria Support Vector regression methods are often on a par with the single-rating matrix factorization methods for the hotel dataset. The performance of the similarity-based method MC-Similarity is comparable or even slightly better than SlopeOne. This supports the findings of [Adomavicius and Kwon

[11] Additional experiments on some datasets with up to 100 runs showed that after 30 runs the average RMSE remains stable.

2007] with respect to the accuracy; a closer look on the prediction coverage, however, reveals that MC-Similarity can only generate recommendations for a third of the users.

On the denser Yahoo!Movies datasets, we see that both the user-based and the item-based Support Vector regression methods work comparably well. For the datasets with the highest qualities, the linear regression methods which are similar to those proposed in [Adomavicius and Kwon 2007] lead to good results. Quite interestingly, on the very dense YM-20-20 dataset, SlopeOne worked even better than the matrix factorization approach Funk-SVD, even when we tried to optimize the parameters of the latter algorithm. Koren's factorized neighborhood method did not lead to better results.

Table IV reports the numbers for the F1 measure. We do not report all detailed precision and recall numbers here for better readability. In our analysis, we observed no strong differences between the different algorithms and datasets with respect to precision and recall values.

Table IV. F1 values for evaluated datasets.

Algorithm	HRS-5-5	HRS-3-3	HRS-RAW	YM-20-20	YM-10-10	YM-5-5
WeightedSVM	90.39	91.67	71.00	88.70	91.53	94.32
SV-Regress-I	88.82	90.60	69.72	87.64	89.93	93.37
SV-Regress-U	87.69	89.50	71.05	86.35	88.18	91.42
Funk-SVD	85.33	88.36	69.13	78.62	83.30	89.07
LS-Regress-I	-	-	-	86.35	88.14	90.07
LS-Regress-U	-	-	-	87.04	88.43	73.66
SlopeOne	68.40	46.40	8.31	78.47	82.64	87.39
MC-Similarity	26.99	13.55	5.91	75.97	52.47	32.87

This said, we observe that our method is slightly better than the other algorithms. The ranking of the other algorithms is also relatively consistent to the RMSE metric. The performance of the similarity-based approach and of SlopeOne drops strongly when the data quality gets lower, while the performance of the matrix factorization method FunkSVD remains relatively stable.

In order to put our observations in relation with previous work, we evaluated our approach on the Yahoo!Movies dataset using an additional set of metrics. We report Precision@5 and Precision@7 values as used in [Adomavicius and Kwon 2007] as well as the Mean Absolute Error (MAE) as used in [Zhang et al. 2009]. We additionally included an algorithm called UB-Pearson, which corresponds to a traditional kNN approach and used a neighborhood size of 200. The results are shown in Table V[12].

We are aware that a comparison with numbers reported in the literature has to be done with care as the data sets are not fully identical. However, the absolute precision values for the top 5 list (Precision@5) for the nearest-neighbor approaches UB-Pearson and "standard CF" are similar to those reported in [Adomavicius and Kwon 2007]. We see this as an indicator that the datasets are to some extent comparable. The comparison of the absolute values for MC-Similarity and their similarity-based "cos-min" method further confirms that.

Furthermore, similar to the findings in [Adomavicius and Kwon 2007], we see that the item-based regression approaches (LS-Regress-* and "total-reg") help to improve precision over the classical kNN methods[13]. Overall, the differences in the precision values for all regression-based methods are comparably small and the WeightedSVM method is only marginally better than the others in this recommendation task.

[12] Empty entries in Table V indicate numbers not reported in literature.

[13] In [Adomavicius and Kwon 2007] an additional precision value of 74.00 for a method "movie-reg95" is given, which can, however, only be achieved for a subset movies with a very high regression fit.

Table V. Precision at top 5 and top 7 and MAE at original 1-13 scale for YM-10-10.

Algorithm	**Precision@5**	**Precision@7**	**MAE**
WeightedSVM	75.62	73.26	1.05
SV-Regress-U	75.15	72.93	0.97
SV-Regress-I	75.48	73.12	1.05
LS-Regress-U	75.29	73.04	1.25
LS-Regress-I	74.98	72.87	1.20
Funk-SVD	73.15	71.65	1.89
MC-Similarity	67.69	71.12	2.06
SlopeOne	72.95	71.49	1.89
UB-Pearson	71.68	70.38	2.07
standard CF[Adomavicius and Kwon 2007]	68.70	69.00	–
total-reg[Adomavicius and Kwon 2007]	70.90	70.40	–
cos-min[Adomavicius and Kwon 2007]	68.80	69.10	–
FPLSA[Zhang et al. 2009]	–	–	2.06
PCC[Zhang et al. 2009]	–	–	2.18

When looking into the MAE values, we see that the value for UB-Pearson is comparable to the kNN method called PCC used in [Zhang et al. 2009], which is another indicator that the datasets are comparable. Generally, the MAE results also confirm that the aggregation function based methods are more accurate than the single-rating techniques and that using Support Vector regression can be advantageous. Quite interestingly, when using the MAE metric, which does not penalize larger errors as much as the RMSE metric, the user-based method SV-Regress-U is more accurate than the weighted method. Note that the user-based regression method also worked quite well on the RMSE measure on the denser movie datasets (Table III). Compared with the weighted scheme, the user-based regression seems to produce larger errors than the weighted method, leading to a higher RMSE value.

Overall, we can observe that the proposed approach based on SV-regression is comparable with and has a slight edge over previous approaches when compared using the MAE and Precision metrics used in the literature.

3.5. Effects of feature selection

In the following, we report the results of our analysis of the effects of different feature selection strategies, where the goal is to find out if we can further improve the prediction accuracy by taking only a subset of the rating dimensions into account.

Let us first consider the HRS-5-5 dataset, in which every hotel was rated by at least 5 users and each user rated at least 5 hotels. As shown in Table III in the previous section, the RMSE achieved with the WeightedSVM method is 0.52. According to our selection strategy ST1 in Section 2.3.3, we first determined the relevance of each rating dimension for the overall rating using the chi-square statistic. The analysis for the HRS-RAW dataset, for example, revealed that the rating describing the “value for money” aspect is the most relevant one[14]. The general hotel ambiance was the most important factor in the other HRS datasets.

Following strategy ST1, we start with a setting, where we only use one single feature to predict the overall rating. The somewhat surprising result is that an RMSE value can be obtained for the HRS-5-5 dataset which is the same as if all features would have been taken into account (0.52). Continuing with strategy ST1 we then made measurements in which we incrementally added more and more features in order of their relevance. The RMSE values for the different HRS datasets are shown in Table VI

[14] Our partners and domain experts from HRS confirmed the importance of this factor in the highly competitive and transparent hotel market.

Table VI. RMSE for incremental feature selection strategy for the HRS-RAW and HRS-5-5 datasets.

Nb. of features	HRS-RAW	HRS-5-5
1	0.61	0.52
2	0.61	0.49
3	**0.59**	0.49
4	**0.59**	0.49
5	0.61	**0.48**
6	0.60	0.49
7	0.60	0.50
8	0.61	0.50
9	0.61	0.49
10	0.60	0.49
11	0.60	0.50
12	0.61	0.50
13	0.62	0.50
14	0.62	0.51

The numbers in Table VI show that we can further improve the RMSE values by taking only a very small subset of the features into account. The effect is a bit stronger for the higher-quality dataset. Note that we only considered those 14 features in our analysis, which were the most relevant according to the chi-square statistic having a value higher than 0.4. Features beyond rank 14 had a considerably lower relevance value and we assume that they will not contribute to the improvement of the RMSE anymore.

In feature selection strategy ST2, we also start with the rating dimension with the highest relevance and incrementally add more features. This time, however, we remove features once we notice a decrease in the RMSE value, before we add the next feature. The design rationale behind strategy ST2 is that there might be a "valuable" feature which for some reason has a lower rank according to the chi-square statistic.

When we applied this strategy, we obtained slight RMSE improvements or at least no deterioration when the first four (respectively five for the HRS-5-5) features were added. After that, however, adding any other individual feature to this set did not lead to a further improvement of the RMSE. We therefore conclude that the chi-square rank is actually a good indicator for the value of an individual feature and that it is sufficient to apply the more efficient greedy feature selection strategy ST1.

We finally applied the feature selection strategies also on the Yahoo!Movies YM-10-10 dataset, even though the dataset comprises only four different rating dimensions. The measurements showed that in this setting all rating criteria are relevant and helpful to decrease the RMSE. In other words, leaving out individual dimensions leads to a deterioration of the predictive accuracy.

The evolutionary optimization procedure used in strategy ST3 lead to a set of 14 relevant features for the HRS-5-5 dataset and 8 for the HRS-raw out of a total of 22 features. This strategy achieves an RMSE value of 0.61 on the HRS-raw dataset and 0.52 on HRS-5-5 dataset. These RMSE results are similar to those reported in Table III, which were achieved without any feature selection. The optimization based on a single global regression function therefore did not lead to a further improvement on the RMSE measure.

Overall, our results indicate that particularly in situations when many different criteria ratings are available, it can be advantageous to focus on the most relevant ones in order to improve the predictive accuracy of the recommendation system. The experiments show that using domain-independent metrics based, e.g., on the chi-square statistics and comparably simple procedures can lead to better accuracy results. More elaborate optimization schemes or the usage of domain-specific knowledge are also possible. Further evaluations with different parameter settings for the evolutionary process are part of our ongoing work.

4. SUMMARY

We have proposed new methods to improve the predictive accuracy of multi-criteria recommender systems and have conducted a detailed analysis of our approaches on different datasets. Our results confirmed the value of detailed customer-provided rating information. Furthermore, we could demonstrate that the quality of recommendations can be further improved when user-specific and item-specific dependencies in the data are taken into account. Finally, we have applied the concept of feature selection to multi-criteria recommender systems and analyzed different basic strategies of how to focus on the most relevant rating dimensions. Our future work includes an analysis whether other hybrid strategies (e.g., combining single-rating recommendation approaches and multi-criteria based ones as, for example, suggested in [Sen et al. 2009]) can help to further increase the predictive accuracy.

Overall, our work shows that sites can expect to see an increase in recommendation quality when online customers can express their preferences using multiple attributes, and when these attributes are incorporated in the sites' recommendation system.

REFERENCES

ADOMAVICIUS, G. AND KWON, Y. 2007. New recommendation techniques for multicriteria rating systems. *IEEE Intelligent Systems 22*, 48–55.

ADOMAVICIUS, G., MANOUSELIS, N., AND KWON, Y. 2011. Multi-criteria recommender systems. In *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. Springer US, 769–803.

BURKE, R. 2002. Interactive critiquing forcatalog navigation in e-commerce. *Artificial Intelligence Review 18,* 3-4, 245–267.

DIAS, M. B., LOCHER, D., LI, M., EL-DEREDY, W., AND LISBOA, P. J. 2008. The value of personalised recommender systems to e-business: a case study. In *Proceedings of the 2008 ACM Conference on Recommender Systems (RecSys'08)*. Lausanne, Switzerland, 291–294.

DRUCKER, H., CHRIS, KAUFMAN, B. L., SMOLA, A., AND VAPNIK, V. 1997. Support vector regression machines. In *Advances in Neural Information Processing Systems 9*. Vol. 9. 155–161.

FLEDER, D. M. AND HOSANAGAR, K. 2007. Recommender systems and their impact on sales diversity. In *Proceedings of the 8th ACM Conference on Electronic Commerce (EC'07)*. San Diego, California, USA, 192–199.

GEDIKLI, F., BAGDAT, F., GE, M., AND JANNACH, D. 2011. RF-REC: Fast and accurate computation of recommendations based on rating frequencies. In *Proceedings of the IEEE Conference on Commerce and Enterprise Computing (CEC 2011)*. Luxembourg, Luxembourg, 50–57.

GEDIKLI, F. AND JANNACH, D. 2010. Neighborhood-restricted mining and weighted application of association rules for recommenders. In *Proceedings of the 11th International Conference on Web Information Systems Engineering (WISE 2010)*. Hong Kong, China, 157–165.

GEDIKLI, F. AND JANNACH, D. 2012. Improving recommendation accuracy based on item-specific tag preferences. *ACM Transactions on Intelligent Systems and Technology*, forthcoming.

GENTLEMAN, W. M. 1974. Algorithm as 75: Basic procedures for large, sparse or weighted linear least problems. *Journal of the Royal Statistical Society. Series C (Applied Statistics) 23,* 3, 448–454.

HOFMANN, T. 2004. Latent semantic models for collaborative filtering. *ACM Transactions on Information Systems 22*, 89–115.

JANNACH, D. 2006. Finding preferred query relaxations in content-based recommenders. In *3rd International IEEE Conference on Intelligent Systems*. London, 355–360.

JANNACH, D. AND HEGELICH, K. 2009. A case study on the effectiveness of recommendations in the mobile internet. In *Proceedings of the 2009 ACM Conference on Recommender Systems (RecSys'09)*. New York, NY, USA, 41–50.

JANNACH, D. AND KREUTLER, G. 2005. Personalized user preference elicitation for e-services. In *IEEE International Conference on e-Technology, e-Commerce,and e-Services (EEE 2005)*. 604–611.

JANNACH, D., ZANKER, M., FELFERNIG, A., AND FRIEDRICH, G. 2010. *Recommender Systems - An Introduction*. Cambridge University Press.

JANNACH, D., ZANKER, M., AND FUCHS, M. 2009. Constraint-based recommendation in tourism: A multiperspective case study. *Journal of Information Technology and Tourism 11,* 2, 139–155.

KOREN, Y. 2010. Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Transactions on Knowledge Discovery from Data 4*, 1–24.

LEMIRE, D. AND MACLACHLAN, A. 2005. Slope one predictors for online rating-based collaborative filtering. In *Proceedings of the 5th SIAM International Conference on Data Mining (SDM'05)*. Newport Beach, CA, 471–480.

LI, Q., WANG, C., AND GENG, G. 2008. Improving personalized services in mobile commerce by a novel multicriteria rating approach. In *Proceedings of the 17th International Conference on World Wide Web*. Beijing, China, 1235–1236.

LINDEN, G., SMITH, B., AND YORK, J. 2003. Amazon.com recommendations: item-to-item collaborative filtering. *Internet Computing, IEEE 7,* 1, 76–80.

LIU, H. AND SETIONO, R. 1995. Chi2: Feature selection and discretization of numeric attributes. In *Proceedings of the Seventh International Conference on Tools with Artificial Intelligence*. Vancouver, Canada, 388–391.

LIU, L., MEHANDJIEV, N., AND XU, D.-L. 2011. Multi-criteria service recommendation based on user criteria preferences. In *Proceedings of the fifth ACM Conference on Recommender Systems (RecSys 2011)*. Chicago, IL, USA, 77–84.

MANOUSELIS, N. AND COSTOPOULOU, C. 2007. Analysis and classification of multi-criteria recommender systems. *World Wide Web 10*, 415–441.

MIERSWA, I. 2009. Non-convex and multi-objective optimization in data mining, Doctoral thesis, Department of Computer Science, TU Dortmund, Germany.

NAKAGAWA, M. AND MOBASHER, B. 2003. A hybrid web personalization model based on site connectivity. In *Proceedings of the Workshop on Web Mining and Web Usage Analysis (WebKDD'03)*. Washington, DC, USA, 59–70.

SAHOO, N., KRISHNAN, R., DUNCAN, G., AND CALLAN, J. P. 2006. Collaborative filtering with multicomponent rating for recommender systems. In *Proceedings of the Sixteenth Annual Workshop on Information Technologies and Systems (WITS'06)*.

SAHOO, N., KRISHNAN, R., DUNCAN, G., AND CALLAN, J. P. 2011. The Halo Effect in multi-component ratings and its implications for recommender systems: The case of Yahoo! Movies. *Information Systems Research*.

SEN, S., VIG, J., AND RIEDL, J. 2009. Tagommenders: Connecting users to items through tags. In *Proceedings of the 18th International World Wide Web Conference (WWW'09)*. Madrid, Spain, 671–680.

SENECAL, S. AND NANTEL, J. 2004. The influence of online product recommendations on consumers' online choices. *Journal of Retailing 80,* 2, 159–169.

SI, L. AND JIN, R. 2003. Flexible mixture model for collaborative filtering. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML 2003)*. Washington, DC, 704–711.

ZANKER, M., BRICMAN, M., GORDEA, S., JANNACH, D., AND JESSENITSCHNIG, M. 2006. Persuasive online-selling in quality & taste domains. In *Proceedings of the 7th International Conference on Electronic Commerce and Web Technologies (EC-Web'06)*. Springer, Krakow, Poland, 51–60.

ZHANG, Y., ZHUANG, Y., WU, J., AND ZHANG, L. 2009. Applying probabilistic latent semantic analysis to multi-criteria recommender system. *AI Communications 22*, 97–107.