Pairwise Preferences in Collaborative Filtering

Laura Blėdaitė

Faculty of Computer Science

Free University of Bozen - Bolzano

A thesis submitted for the degree of

Master of Science

March 2014

Acknowledgements

I want to thank all the people that helped me in writing this thesis. The biggest gratitude goes to my supervisor prof. Francesco Ricci. Beginning with the course of Information Search and Retrieval he introduced me to the topic and raised my curiosity in the field of Recommender Systems. During the period of writing the thesis he was always patient, kept me concentrated and motivated.

I am very grateful to Matthias Braunhofer, Mehdi Elahi and all the members of the IDSE group of the Computer Science faculty for the rich feedback and suggestions without which I would not have done the same job.

I would also like to express gratitude for all my family and friends who not only became the participants in the experiment but were also very encouraging throughout the thesis writing period.

Abstract

Collaborative Filtering (CF) techniques generate item recommendations for items that users have not rated, based on the ratings given by users to other items. Ratings are a form for expressing user preferences, but there is another way to state preferences. Users can express their preferences by comparing pairs of items, e.g., A is preferred to B. Our hypothesis is that exploiting pairwise preferences one can improve the quality of recommendations computed by CF techniques, because in many situations users are more inclined to give pairwise preferences than ratings and because they provide more useful data for computing better predictions. This may result from pairwise comparisons being easier to perceive than the numerical ratings in some limited scale. While in recommender systems this approach to model user preferences has not received much interest, some literature (not in Recommender Systems (RSs)) on ranking based on pairwise preferences exists. However, the existing techniques are not personalized. Therefore, we developed a personalized variant of a ranking algorithm which combines pairwise preferences and ratings with a measure of user-to-user similarity. We hypothesize that pairwise preference incorporation in the standard CF techniques can improve recommendation quality of RSs. In order to test our hypothesis we perform offline and online experiments. We show that the system using pairwise preferences makes users more aware of their options. Also, we show that the system that incorporates pairwise preferences has a higher recommendation accuracy in terms of $nDCG$ and precision in case a subset of users is restricted to the most involved ones. Moreover, it improves the recommendation accuracy in terms of precision after additional pairwise preferences, while MF does not. We noticed that, despite the fact that users enjoy entering pairwise preferences more, the system asking pairwise preferences still suffers from a more complex preference request generation. That is, the choice of the most informative pairs that are likely to be compared by the user is a more difficult task than the choice of the most informative items that are likely to be rated.

Contents

Nomenclature

General Notation

$\mathcal{D}_{train}$	the training set, i.e. the set of the user-item pairs (u, i) for which the rating r_{ui} is known
$\mathbf{e}$	column vector of all ones
e_{ui}	r^*_{ui} prediction error
$\|\mathbf{A}\|_F$	Frobenius norm of matrix $\mathbf{A}$
γ	parameter used in significance of similarity computation that must be cross-validated
$\mathcal{I}$	set of items
$\mathcal{I}_u$	set of items that have been rated by user u
$\mathcal{I}_{uv}$	set of items that have been rated by both users u and v
m	number of users in a set
m_i	number of users that have rated item i
m_{ij}	number of users that have rated items i and j
n	number of items in a set
$\mathcal{N}(u)$	set of neighbors of user u
$\mathcal{R}$	set of ratings recorded
r_i	average rating of item i
r_u	average rating of user u
r_{ui}	rating user u gave to item i
r^*_{ui}	predicted rating for user u and item i
$\mathcal{S}$	set of possible rating values

$\mathcal{U}$	set of users
$\mathcal{U}_i$	set of users that have rated item i
$\mathcal{U}_{ij}$	set of users that have rated both items i and j
w_{uv}	similarity between users u and v

Notation Used in Matrix Factorization Model

γ_{MF}	learning rate in MF models
λ_{MF}	regularization parameter in MF models
p_{ju}	value measuring the extent of interest u has in items that are high on the factor j
$\mathbf{p}_u$	factor vector associated with user u
$\mathbf{q}_i$	factor vector associated with item i
q_{ji}	value measuring the extent to which item i possesses factor j

Notation Used in Differential Matrix Model

δ	preference function
δ^u	pairwise preference average of user u
$\mathbf{K}$	score difference matrix
$\mathcal{P}$	set of item pairs
$\mathcal{P}_{uv}$	set of item pairs that have been compared by both users u and v
S_{ij}	number of points scored by team i when the team i meets team j in sports team ranking scenario and average of ratings that item i received from all the users who have rated both items i and j in case of ranking items by user preferences
S_u	a set of all sessions of user u
$\mathbf{V}$	evaluation difference matrix
v_i	optimal evaluation of item (or team) i

List of Figures

Chapter 1

Introduction

1.1 Motivation

Recommender systems (RS) try to make recommendations of items for the target user. They use a model of the known user preferences to infer the unknown preferences. Collaborative filtering (CF) methods for making recommendations have acquired more and more interest recently. The idea behind these methods is to make recommendations for items based on the predicted ratings derived from the ratings acquired in the training set. The vast majority of the CF methods derived recently focus on the computation of a total order of the items. For example, Wang et al. [2012] adapts the TF-IDF scheme used in Information Retrieval (IR) and uses it for ranking-based CF.

In CF the preference model is based on a set of known ratings and the system predicts the unknown ratings. Based on the predicted ratings of the unrated items the system produces a ranking and recommends the top ranked items. While the approach using the numerical ratings is natural to consider, another way to express preferences exists. Preferences can also reflect choices about alternative options, e.g. A is preferred to B, and C to D. A possible specification of a choice about options is a *pairwise preference* which states how much one option is preferred to the other, e.g. A is 3 times more preferred to B, C is 3 points better (in some arbitrary scale) than D, etc. We conjecture that expressing user preferences as pairwise preferences has certain advantages over ratings. Pairwise preferences are easier to formulate and reason about than ratings. Namely, it is easier to decide which item (and how much more) you prefer than the other than to rate both of the items in some limited scale, e.g. 5-star.

We hypothesize that such an advantage may make users more inclined to give pairwise preferences than ratings and thus result in more preference data gathered during preference elicitation process. Consequently, we expect that RSs using pairwise preferences to reflect user preferences would produce a better ranking of recommended items and so improve the recommendation quality in terms of standard RS prediction accuracy metrics such as precision and Normalized Discounted Cumulative Gain Herlocker et al. [2004]; Shani and Gunawardana [2011].

In recommender systems the approach to express user preferences in terms of pairwise preferences instead of ratings has not received much interest. One of the reasons is that most of the current interfaces for evaluating items are based on ratings, e.g., 5 stars, thumb up/down, etc. As a consequence, the recent research in the field of CF using pairwise preferences used the approach of transforming ratings to pairwise preferences, which is in fact a loss of information Brun et al. [2010].

However, ranking using pairwise preferences has been analyzed in a different research area (not in RS) Langville and Meyer [2012]. The core problem discussed there is: having data about options that describes how much an option is better than another one, generate a ranked list of the options. The majority of the examples used in the book are taken from team sports, where the data consists of the results of matches between teams, with the aim of generating a ranking of the teams.

The case of recommender systems is considered by Langville and Meyer [2012] as well. The techniques mentioned above are adapted to address the RS problem by interpreting either (1) item rating differences or (2) item pairwise preferences (they call them direct preferences) as match score differences in a case of sport teams. However, the main problem of ranking the sport teams based on their scores is in fact different form the core concept of recommender systems. The methods described produce a global (non-personalized) ranking list, which is somewhat marginal with respect to RS literature and problems. Indeed, the power of a recommender system comes from generating *personalized* recommendations, that can be derived only from personalized ranked lists.

Another issue that has not received much activity in RS research is that usually the users are not stable in their rating behavior. People use some benchmark or reference items while they are rating and these references may change. That is, if a person has rated only good items during the current session, i.e., while he is logged in the system, and he finds an average item, he will probably rate it lower than the good items rated in the same session before. In contrast, if he has rated (in the same session) many bad items and then he finds the same average item, he will most likely rate it higher than the bad items rated in the same session before.

Based on the reasoning explained above, we conjecture that the more close in time two ratings are the more dependent they are. This concept is not new and there exist time-aware matrix factorization models Koren [2009]. One of the temporal effects used in time-aware matrix factorization models allows users to change their baseline ratings over time. This may reflect several factors such as a natural drift in a users rating scale or the fact that ratings are given in relationship to other ratings Koren and Bell [2011].

Therefore, we hypothesize that weighting the two ratings (or pairwise preferences) that are closer together in time, stronger (we call it temporal slicing) may produce improved recommendations in terms of $nDCG$ - a state-of-the-art IR measure for measuring recommendation prediction accuracy.

One more motivating factor is that eliciting preference information from the user in a conversational way can be useful. Normally, CF is a one shot algorithm. But in practice people interact with these systems. It suggests that letting the user to better specify her preferences using pairwise preferences can lead to a certain number of positive effects compared to the standard approach. For example, if the pairs that the user is asked to compare are chosen by using constantly updated information about the user's taste, then the pairwise preferences gathered should be more valuable for the recommendation prediction accuracy. Therefore, we hypothesize that adding the additional step in a RS which asks the user to provide some more pairwise preferences results in a recommendation prediction accuracy improvement in terms of $nDCG$.

1.2 Research Hypotheses

Motivated by the issues described in Section 1.1 we raise several hypotheses. The main research hypothesis is whether incorporating pairwise preferences into recommender systems that use standard CF methods improves recommendation accuracy in terms of the normalized Discounted Cumulative Gain ($nDCG$) and precision - the standard IR metrics for measuring recommendation prediction accuracy. We try to prove it by developing a novel technique for producing recommendations using a mixture of ratings and pairwise preferences (we call it Personalized differential matrix with ratings and pairwise preferences (PDMRPP)) and comparing it to the Matrix Factorization (MF) model which is a state-of-the-art model in CF.

Besides the recommendation prediction accuracy, we think that the pairwise preference usage in RS improves the users' perceived recommendation quality. We test this hypothesis by asking the users to evaluate a set of statements related to the perceived recommendation quality in an online experiment.

There are several side hypotheses that we assume are the sources of the main hypothesis and which we are interested in as well. One of them is that the users are more inclined to give pairwise preferences than ratings because they get more satisfaction from comparing items than from giving ratings to them. We test this side assumption in an online experiment that we designed by asking the users to provide feedback about the preference elicitation process in a form of statements related to it.

Our strongest assumption is that the system where users provide pairwise preferences performs highly better than the one where users provide ratings in terms of making users more aware of their choice options, which we believe can serve as an incentive to provide more preferences and thus enable the system to produce more accurate recommendations.

Motivated by the reasoning that the conversational way of eliciting users' preferences is advantageous in certain ways, we raise another assumption that adding a step in a RS which asks the users to provide some more pairwise preferences results in a recommendation prediction accuracy improvement in terms of $nDCG$ and precision. We test this assumption in an online experiment.

As was mentioned before, most of the research about pairwise preferences in recommender systems that we are aware of apply the transformation step that creates pairwise preferences from the existing ratings, which is a loss of information. We assume that such a loss of information actually exists, but we do not expect the difference of the recommendation prediction accuracy caused by this loss of information to be extreme. We raise two hypotheses in relation to the loss of information. Firstly, the item ranking quality of the Non-Personalized Differential Matrix with Ratings (NPDMR) (Langville and Meyer [2012] variant) technique is expected to be lower than the ranking quality based on the item Rating Averages (RA). Secondly, the ranking quality of the Personalized Differential Matrix with Ratings (PDMR) (developed by us, but using rating transformation to pairwise preferences) is assumed to be lower than the item ranking quality of the matrix factorization model. We expect that disadvantages of NPDMR over RA and of PDMR over MF exist, but they are not extreme.

Another hypothesis originating from the fact that the users are not stable in their rating behavior is that using temporal slices to define rating (or preference) acquisition sessions may improve the recommender system. We test this assumption in an offline experiment by incorporating the predefined sessions in our developed NPDMR model and expect the item ranking quality increase compared to the MF algorithm.

1.3 Results

Based on the motivation described in Section 1.1 and driven by the research hypotheses presented in Section 1.2 we modified the techniques proposed in Langville and Meyer [2012] for solving the global ranking problem applied to the sport teams domain in order to compute a personalized ranking.

We conducted offline and online experiments to test the hypotheses presented in Section 1.2.

In the offline experiments we were not able to show that the Non-Personalized Differential Matrix with Ratings (NPDMR) technique performs significantly worse than simple rating average (RA) used for ranking the items in terms of Kendall's τ - a ranking correlation measure. The decrease of the ranking precision was expected because of the loss of information introduced by the transformations of the ratings to the pairwise preferences.

Similarly the Personalized Differential Matrix with Ratings (PDMR) was not shown to fall significantly behind MF in rating accuracy in terms of Kendall's τ computed between the technique of interest (PDMR or MF) and the real ranking produced by sorting the items in the test set according to their known ratings. The decrease of the ranking precision was expected because of the loss of information introduced by the transformation of the ratings to the pairwise preferences.

In the online experiments we were able to prove the important expected conclusion that the system eliciting pairwise preferences over movies makes users more aware of their choices compared to the system that asks users to rate movies. This is a natural and expected conclusion as the former system encourages one to compare the movies and thus to discern their own tastes.

The overall users' satisfaction in preference elicitation process in terms of a pooled score was slightly higher in case the users were asked to compare movies than in case they had to rate them, but the difference was not significant.

We showed that PDMRPP outperforms MF in recommendation precision accuracy in terms of *nDCG* and precision when a subset of more knowledgeable and/or interested users is used. We also found out the difficulty that the PDMRPP suffers from. It is the information loss that is incurred by the rating transformation to pairwise preferences, since the model in our experiment used a mixture of data where the number of pairwise preferences is highly lower than the number of ratings. Another difficulty that PDMRPP model faces is a low probability of finding a pair that the user will compare which is much lower than the probability of finding the movie that the user will rate.

Furthermore, we were able to prove significant improvement of recommendation precision after the additional preferences were added in a case of PDMRPP technique. The *nDCG* improvement was also present, though, not significant.

Chapter 2

State of the Art

This chapter introduces the basic concepts that are needed in the thesis. In Section 2.1 the state-of-the-art collaborative filtering techniques are overviewed. In Section 2.2 related work in ranking using pairwise preferences is discussed. In Section 2.3 the problem of preference acquisition is described and the existing solutions are presented.

2.1 Collaborative Filtering

Recommender Systems (RSs) are software tools and techniques providing suggestions for items to be of use to a user Ricci et al. [2011]. The domain of the items may vary from books and CDs Linden et al. [2003]; Mooney and Roy [2000], music Shardanand and Maes [1995], movies Bell and Koren [2007]; Hill et al. [1995]; Miller et al. [2003] to news Billsus et al. [2002]; Konstan et al. [1997]; Terveen et al. [1997], jokes Goldberg et al. [2001], places Ricci and Missier [2004], web pages Billsus and Pazzani [1998]; Mobasher et al. [2002] and so on. Nowadays almost every business offering products or services has its own e-commerce website. The number of the items offered by a company may reach millions, not to speak about the number of users. Therefore, there is an obvious necessity to assist users while making decisions and match the right users with the right products. Being sophisticated tools, recommender systems enable users to tame the information overload and suggest only the most suitable items for a particular user.

Recommendations can be made by exploiting the user's personal information and the item's content information, by making use of the opinions of the users with similar tastes, or by using a mixture of both. The two main categories of RSs are *content-based filtering* and *collaborative filtering*. Content-based recommendation systems try to recommend items similar to those a given user has liked in the past Lops et al. [2011]. **Collaborative Filtering (CF)** methods produce user specific recommendations of items based on patterns of ratings or usage without need for exogenous information about either items or users Koren and Bell [2011].

The main objective of CF is to predict the opinion the target user, i.e., the user that is requesting the recommendation, has on items listed in the system and be able to recommend the best items to her. It is a recommendation technique based on the analysis of user's previous likings and the opinions of other like minded users. Collaborative filtering methods can further be divided in two general groups of *neighborhood-based* and *model-based* methods. In neighborhood-based collaborative filtering Desrosiers and Karypis [2011], the user-item ratings stored in the system are directly used to predict ratings for new items, while the model-based approaches use these ratings to learn a predictive model Koren and Bell [2011].

Rating prediction

The neighborhood based collaborative filtering recommendation technique proceeds in these steps:

1. For a target/active user (the user to whom a recommendation has to be produced) the set of his ratings is identified.
2. The users more similar to the target/active user (according to a similarity function) are identified (neighbor formation).
3. The products evaluated by these similar users are identified.
4. For each one of these products a prediction of the rating that would be given by the target user to the product is generated.
5. Top N items with the largest predicted ratings are recommended.

Let's introduce some notation that will be used throughout the thesis. Very similar notation is used in Desrosiers and Karypis [2011].

$\mathcal{U}$ is the set of users, $\mathcal{I}$ is the set of items. $\mathcal{R}$ is the set of ratings recorded, $\mathcal{S}$ is the set of possible rating values. In most of the examples throughout the thesis we will use $\mathcal{S} = [1, 5]$ to reflect a 5-star scale. $\mathcal{U}_i$ is the set of users that have rated item i, $\mathcal{I}_u$ is the set of items that have been rated by user u, $\mathcal{U}_{ij}$ is the set of users that have rated both items i and j, $\mathcal{I}_{uv}$ - the set of items that have been rated by both users u and v. w_{uv} is a similarity measure of users u and v. $\mathcal{N}(u)$ is a set of neighbors of user u (the most similar). r_{ui} and r^*_{ui} are the real and predicted ratings the user u gave to the item i, respectively. r_u is the average rating of user u. r_i is the average rating of item i. n is the number of items and m is the number of users.

Using this notation, the prediction of the rating given by user u to item i is calculated as in Equation 2.1.

$$r^*_{uj} = r_u + K \sum_{v \in N_j(u)} w_{uv}(r_{vj} - r_v) \tag{2.1}$$

K is a normalization factor such that the absolute values of w_{uv} sum to 1.

Rating prediction measure given in Equation 2.1 combines several adjustments of the pure item rating average that would be the most naïve non-personalized prediction. Firstly, it weights stronger the ratings of the users that are more similar to the target user. Secondly, it normalizes the weights to sum up to 1. Thirdly, it takes into account the fact that different users tend to have different rating scale usage habits which necessitates the adjustment of the rating prediction to the user rating average. For example, some user may almost never give the highest rating (i.e., 5 in case of $\mathcal{S} = [1, 5]$) having in mind that it is reserved for the exceptionally good items. On the other hand, some other user may only use 4 and 5 thinking that the former reflects his/her bad opinion and the latter reflects the good opinion.

Similarity Computation

Similarity w_{uv} mentioned previously can be computed using metrics such as (adjusted-) cosine similarity and correlation-based similarity Sarwar et al. [2001]. The metrics proposed for measuring the similarity between users include Pearson and Spearman Correlation Resnick et al. [1994], the cosine angle distance Sarwar et al. [2000b], Entropy, Mean-squared difference and constrained Pearson correlation Shardanand and Maes [1995].

The cosine similarity can be computed as follows:

$$w_{uv} = \frac{\sum_{j \in I_{uv}} r_{uj} r_{vj}}{\sqrt{\sum_{j \in I_u} r_{uj}^2 \sum_{j \in I_v} r_{vj}^2}} \tag{2.2}$$

Pearson correlation is calculated similarly:

$$w_{uv} = \frac{\sum_{j \in I_{uv}} (r_{uj} - r_u)(r_{vj} - r_v)}{\sqrt{\sum_{j \in I_{uv}} (r_{uj} - r_u)^2 \sum_{j \in I_{uv}} (r_{vj} - r_v)^2}} \tag{2.3}$$

There are results proving better performance of each of these measures. Breese et al. [1998] shows that the cosine performs worse. Still, it is very wide used in Information Retrieval Sarwar et al. [2000a] and there exist experiments reporting cosine's better performance Anand and Mobasher [2005].

Matrix Factorization Methods

Even though the nearest-neighbor technique is very useful, there exist other more advanced approaches that bring competitive accuracy into neighborhood methods by utilizing temporal models and implicit feedback Koren and Bell [2011]. As the Netflix Prize competition has demonstrated, Matrix Factorization (MF) models are superior to classic nearest-neighbor techniques in terms of root mean squared error (RMSE) for producing product recommendations.

Besides the explicit feedback in a form of numerical ratings, they allow the incorporation of additional information such as implicit feedback (e.g. purchase history), temporal effects, and confidence levels Koren et al. [2009].

In the basic matrix factorization model each item i and user u is associated with an f-dimensional real vectors $\mathbf{q}_i$ and $\mathbf{p}_u$. The elements of $\mathbf{q}_i = (q_{1i}, \ldots, q_{fi})$ measure the extent to which the item i possesses those factors, positive or negative. The elements of $\mathbf{p}_u = (p_{1u}, \ldots, p_{fu})$ measure the extent of interest u has in items that are high on the corresponding factors, positive or negative. The interaction between user u and item i is captured by the resulting dot product $\mathbf{q}_i \cdot \mathbf{p}_u$. This dot product is then the estimation of the users' overall interest in the items' characteristics Koren and Bell [2011].

$$\mathbf{r}^*_{ui} = \mathbf{q}_i \cdot \mathbf{p}_u = \sum_{j=1}^{f} p_{ju} q_{ji} \tag{2.4}$$

The core problem is how to compute $\mathbf{q}_i$ and $\mathbf{p}_u$. In this situation singular value decomposition (SVD) Golub and Reinsch [1970] comes to use. SVD is a technique that can be used for identifying latent factors in the model.

For a complete $m \times n$ matrix $\mathbf{A}$ of rank r there exists a factorization (Singular Value Decomposition, SVD) as follows:

$$\mathbf{A} = \mathbf{U\Sigma V^T} \tag{2.5}$$

The columns of $\mathbf{U}$ are orthogonal eigenvectors of $\mathbf{AA}^T$. The columns of $\mathbf{V}$ are orthogonal eigenvectors of $\mathbf{A}^T\mathbf{A}$. Eigenvalues $\lambda_1, \ldots, \lambda_r$ of $\mathbf{AA}^T$ are the eigenvalues of $\mathbf{A}^T\mathbf{A}$.

$$\sigma_i = \sqrt{\lambda_i} \tag{2.6}$$

$$\mathbf{\Sigma} = diag\left(\sigma_1, \ldots, \sigma_r\right) \tag{2.7}$$

SVD can then be used to compute optimal low-rank approximations Chua et al. [2003]. The problem of approximation is to find the $\mathbf{A}_k$ of rank k such that:

$$\mathbf{A}_k = \min_{\mathbf{X}:rank(\mathbf{X})=k} \|\mathbf{A} - \mathbf{X}\|_F \tag{2.8}$$

Here $\|\mathbf{A}\|_F$ denotes the Frobenius norm which is calculated as follows:

$$\|\mathbf{A}\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2} \tag{2.9}$$

$\mathbf{A}_k$ and $\mathbf{X}$ are both $m \times n$ matrices and typically, $k \ll r$.

The rank k approximation $\mathbf{A}_k$ is found by setting the smallest $r-k$ singular values in $\mathbf{\Sigma}$ to 0:

$$\mathbf{A}_k = \mathbf{U}\, diag(\sigma_1, \ldots, \sigma_k, 0, \ldots, 0)\, \mathbf{V}_T \tag{2.10}$$

If $\mathbf{U} = (u_1\, u_2\, \ldots\, u_r)$ and $\mathbf{V} = (v_1\, v_2\, \ldots\, v_r)$, then:

$$\mathbf{A}_k = \sum_{i=1}^{k} \sigma_i u_i v_i^T \tag{2.11}$$

Here σ_i, u_i and v_i are the columns of $\mathbf{\Sigma}$, $\mathbf{U}$ and $\mathbf{V}$ respectively.

The prediction models based on MF using SVD replace the missing rating values with the corresponding entries in the matrix $\mathbf{A}_k$.

However, in the recommender systems the matrix of known ratings is always very sparse, i.e. incomplete. In such case the standard SVD is undefined. Furthermore, it is very important to use the few known ratings with care, because there is a high risk of overfitting. In other words, if you approximate too much the existing ratings, then you do not have good performance on ratings that are not present in the training set.

There are different approaches to address the problem of missing data in SVD. One of them, which was used early in the research Kim and Yum [2005]; Sarwar et al. [2000b], is imputation which fills in a missing rating value with a plausible one and makes the rating matrix dense. However, this can produce several problems. One of them is the price paid for the increased size of the data. The other is changing the data largely by incorrect imputation. This is very likely, because of the fact that the amount of data imputed is usually very large (the matrices are sparse).

The more recent research works suggest models that use only the directly observed data and in this way avoid problems that are present when using SVD in MF models and therefore are proven to be very effective for MF Koren [2008]; Koren et al. [2009]; Paterek [2007]; Takács et al. [2007].

In order to learn the factor vectors ($\mathbf{p}_u$ and $\mathbf{q}_i$), the regularized squared error on the set of known ratings is minimized:

$$\min_{\mathbf{p}^*,\mathbf{q}^*} \sum_{(u,i)\in\mathcal{D}_{train}} \left(r_{ui} - \mathbf{q}_i^T \mathbf{p}_u\right)^2 + \lambda_{MF}\left(\|\mathbf{q}_i\|^2 + \|\mathbf{p}_u\|^2\right) \tag{2.12}$$

$\mathcal{D}_{train}$ is the training set, i.e. the set of the user-item pairs (u, i) for which the rating r_{ui} is known and on which the model is built. The constant λ_{MF} is a regularization parameter, which is usually determined by cross validation.

There exist several approaches to minimizing Equation 2.12. Probably the best known ones are *stochastic gradient descent* and *alternating least squares* Koren et al. [2009]. For the sake of brevity and because we use the stochastic gradient descent in our experiments, only this one is described here.

The idea of stochastic gradient descent optimization is looping through ratings in the training set, predicting r_{ui} and computing the associated prediction error e_{ui} given in Equation 2.13.

$$e_{ui} \quad = \quad r_{ui} - \mathbf{q}_i^T \mathbf{p}_u \tag{2.13}$$

The parameters are then modified by a magnitude proportional to γ_{MF} (a learning rate) in the opposite direction of the gradient, giving:

- $\mathbf{q}_i \leftarrow \mathbf{q}_i + \gamma_{MF} \cdot (e_{ui} \cdot \mathbf{p}_u - \lambda_{MF} \cdot \mathbf{q}_i)$
- $\mathbf{p}_u \leftarrow \mathbf{p}_u + \gamma_{MF} \cdot (e_{ui} \cdot \mathbf{q}_i - \lambda_{MF} \cdot \mathbf{p}_u)$

There are additional aspects of matrix factorization models that make them even more superior to the classic neighbor-based methods. First of all, they are very flexible when dealing with different data aspects and other application-specific requirements Koren et al. [2009]. While the simple stochastic gradient descent model described above predicts different rating values based on the captured interactions between users and items ($\mathbf{q}_i^T \mathbf{p}_u$), much of the variation in rating values comes from the effects associated with either users or items, known as *biases* (also called *intercepts*), independent of any interactions. The bias is a typical characteristic pertaining to a user or item. A natural example of a bias in recommender systems data is the trend for some users to give higher (or lower) ratings than the average. Similarly, some items typically are rated higher (or lower) than others. Matrix factorization models easily incorporate biases minimizing the modification of the Equation 2.12 given in Equation 2.14 Koren and Bell [2011].

$$\min_{\mathbf{p}^*,\mathbf{q}^*,\mathbf{b}^*} \sum_{(u,i)\in\mathcal{D}_{train}} \left(r_{ui} - \mu - b_u - b_i - \mathbf{q}_i^T \mathbf{p}_u\right)^2 + \lambda_{MF}\left(\|\mathbf{q}_i\|^2 + \|\mathbf{p}_u\|^2 + b_u^2 + b_i^2\right) \tag{2.14}$$

Here μ is the overall average rating, b_u and b_i indicate the observed deviations from the average, of user u and item i, respectively.

The parameters are modified in the opposite direction of the gradient, giving:

- $b_u \leftarrow b_u + \gamma_{MF} \left(e_{ui} - \lambda_{MF} \cdot b_u\right)$
- $b_i \leftarrow b_i + \gamma_{MF} \left(e_{ui} - \lambda_{MF} \cdot b_i\right)$
- $\mathbf{q}_i \leftarrow \mathbf{q}_i + \gamma_{MF} \cdot \left(e_{ui} \cdot \mathbf{q}_u - \lambda_{MF} \cdot \mathbf{q}_i\right)$
- $\mathbf{p}_u \leftarrow \mathbf{p}_u + \gamma_{MF} \cdot \left(e_{ui} \cdot \mathbf{q}_i - \lambda_{MF} \cdot \mathbf{p}_u\right)$

Moreover, matrix factorization models allow for the possibility to account for temporal changes Koren [2009]; Koren and Bell [2011]. The reason is that the tastes of the users and/or the popularity of the items is never static. This makes it necessary to take into account the temporal dynamics into the model. And the matrix factorization model allows that easily.

2.2 Ranking Using Pairwise Preferences

A second line of research that this thesis is based on analyzes the techniques of ranking using pairwise preferences Fürnkranz and Hüllermeier [2003]; Hüllermeier et al. [2008]. A more general problem of ranking has a long tradition which dates back to around 13th century. Interest in it grew very rapidly with the birth of the Internet which introduced a new problem of ranking Web pages with the interesting solutions such as PageRank Brin and Page [1998]; Page et al. [1999] or HITS Kleinberg [1999].

Another application of ranking problem that interested many researchers was ranking the sports teams Colley [2002]; Keener [1993]; Massey [1997]. Langville and Meyer [2012] (sometimes we will refer to it as 'the book') collects the multitude of existing methods and applications concerning the ranking of sports teams as well as presents several new ideas related to the topic. The approaches described in Langville and Meyer [2012] use direct comparisons of the items in a form of match scores to produce the rankings of the teams.

The recommender systems literature does not provide us with much information on direct comparisons (or pairwise preferences) between items to predict the item ratings. Generally RSs make recommendations of the items based on their ranking that is derived from the existing ratings that the users gave to a subset of the full set of items. However, in some cases the pairwise preferences could be easier and more natural to acquire. We conjecture that in many situations users are more inclined to give pairwise preferences than ratings. This may result from pairwise preferences being easier to perceive than the numerical ratings which are limited to a particular scale, e.g. 5-star scale.

Langville and Meyer [2012] presents quite a number of methods for ranking the items in a set. The items in the examples of the book are most often sports teams. The ranking techniques presented use teams' match score differences to predict the rating differences and in this way produce a ranking of the teams. (Because a term *rating* is a different concept in the ranking literature and in RSs literature, we will use a term *evaluation* to denote the optimal rating value that is to be predicted by the ranking technique from Langville and Meyer [2012].)

Nevertheless, the techniques presented transcend just sports ratings and rankings. All of the methods are equally well applicable for any set of items. The recommender systems case is briefly addressed in the book as well. Actually, the whole chapter (Chapter 10) is dedicated to User Preference Ratings. However, the ideas for ranking in order to make recommendations based on the user preferences suggested in the book are not personalized, i.e. they produce the global ranking of the items (the same for all users).

In order to illustrate the team ranking problem, we will present an example from Langville and Meyer [2012]. Then we will explain how the match score differences are used to predict item evaluations. Finally, we will show how the score difference ideas can be applied to user preferences in order to make a non-personalized ranking (a global recommendation list).

Table 2.1 shows match results, win-loss records and point differentials for 5 Atlantic Coast Conference teams from the 2005 NCAA football season. For example, the table entry 7-52 in the Miami column and Duke row means that in the match 'Duke against Miami' Duke scored 7 and Miami scored 52. 0-4 in the Record column and Duke row means that Duke had 0 wins and 4 losses during the season. -124 in the Point Differential column and Duke row is the difference of the sum of scores that Duke scored against the other teams and the sum of scores that it missed. This is the running example used throughout the book to illustrate the different ranking methods.

	Duke	Miami	UNC	UVA	VT	Record	Point Differential
Duke		**7-52**	21-24	7-38	0-45	**0-4**	**-124**
Miami	52-7		34-16	25-17	27-7	4-0	91
UNC	24-21	16-34		7-5	3-30	2-2	-40
UVA	38-7	17-25	5-7		14-52	1-3	-17
VT	45-0	7-27	30-3	52-14		3-1	90

Table 2.1: Game score data for a small 5-team example

A good scientific rating system tries to accurately reflect relative differences in the overall strength of each team after a reasonable amount of competitions has occurred Langville and Meyer [2012]. A slightly simpler goal is to produce a reasonable ranking that agrees with some expert consensus and reflects well the long-term ability for winning. Many systems actually produce the rankings that reflect the win percentages quite well. However, there is a huge difference between predicting the win percentages and the point spreads (actual score differences).

The simplest ranking system based on the win-loss records produces a ranking by sorting the teams according to the difference between the number of the wins and losses they acquired in a decreasing order. The result of a ranking of 5 teams from this running example produced by such a system is shown in Table 2.2.

On the other hand, a ranking system based on the point differentials produces a ranking by sorting the teams according to the difference between the sum of points the team scored against

Team	wins - losses	Rank
Miami	4	1
VT	2	2
UNC	0	3
UVA	-2	4
Duke	-4	5

Table 2.2: Ranking teams by win-loss differences

all other teams and the sum of points it missed in a decreasing order. The result of a ranking of 5 teams from this running example produced by such a system is shown in Table 2.3.

Team	Point Differential	Rank
Miami	91	1
VT	90	2
UVA	-17	3
UNC	-40	4
Duke	-124	5

Table 2.3: Ranking teams by point differentials

Ranking by either the win-losses or the point differentials is a very naïve approach. Now we will present a more advanced ranking technique based on the point spread prediction that was introduced in Langville and Meyer [2012]. In order to illustrate the issues of building a system that predicts the point spreads, suppose all of the spreads of all of the future matches are known. The aim is to build an optimal item evaluation system (that can be used for ranking the items) using this information. Optimal here means the one in which the differences of the evaluations $v_i - v_j$ are as close as possible to the point spreads (the score differences) $S_{ij} - S_{ji}$when team i plays team j. Suppose also that there exists the perfect evaluation vector

$$\mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix}$$

such that each evaluation difference $v_i - v_j$ is equal to each scoring difference $S_{ij} - S_{ji}$, where S_{ij} and S_{ji} are the respective numbers of points scored by teams i and j when they meet.

$$\mathbf{K} = \begin{pmatrix} 0 & S_{12} - S_{21} & \cdots & S_{1n} - S_{n1} \\ S_{21} - S_{12} & 0 & \cdots & S_{2n} - S_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ S_{n1} - S_{1n} & S_{n2} - S_{2n} & \cdots & 0 \end{pmatrix} \tag{2.15}$$

$$\mathbf{V} = \begin{pmatrix} 0 & v_1 - v_2 & \cdots & v_1 - v_n \\ v_2 - v_1 & 0 & \cdots & v_2 - v_n \\ \vdots & \vdots & \ddots & \vdots \\ v_n - v_1 & v_n - v_2 & \cdots & 0 \end{pmatrix} \tag{2.16}$$

If $\mathbf{K}$ in Equation 2.15 and $\mathbf{V}$ in Equation 2.16 are the score difference and evaluation difference matrices respectively, then

$$\mathbf{K} = \mathbf{V} = \mathbf{v}\mathbf{e}^T - \mathbf{e}\mathbf{v}^T \tag{2.17}$$

Here $\mathbf{e}$ is a column vector of all ones.

As was mentioned, the Equation 2.17 reflects the ideal case. In that case the score difference matrix $\mathbf{K}$ is a rank-two skew symmetric matrix. (A square matrix $\mathbf{A}$ is skew-symmetric if $\mathbf{A}^T = -\mathbf{A}$ Saikia [2009]). $\mathbf{K}$ is skew symmetric because $k_{ij} = -k_{ji} \Rightarrow \mathbf{K}^T = -\mathbf{K}$, and the rank is two because $rank(\mathbf{v}\,\mathbf{e}^T - \mathbf{e}\,\mathbf{v}^T) = 2$ if $\mathbf{v}$ is not a multiple of $\mathbf{e}$ (i.e., if $\mathbf{V} \neq \mathbf{0}$).

In fact the ideal case is very unlikely, therefore the best solution is to construct such evaluation vector $\mathbf{v}$ that minimizes the distance between the score difference matrix $\mathbf{K}$ and the evaluation difference matrix $\mathbf{V} = \mathbf{v}\,\mathbf{e}^T - \mathbf{e}\,\mathbf{v}^T$, i.e., to find a vector $\mathbf{x}$ such that

$$f(\mathbf{x}) \quad = \quad \|\mathbf{K} - \mathbf{V}(\mathbf{x})\|^2 = \|\mathbf{K} - \left(\mathbf{x}\mathbf{e}^T - \mathbf{e}\mathbf{x}^T\right)\|^2 \tag{2.18}$$

is minimal for some matrix norm. The simple and widely used norm is Frobenius norm given in Equation 2.9. When the matrix $\mathbf{A}$ is square, it's Frobenius norm can be written as:

$$\|\mathbf{A}\|_F \quad = \quad \sqrt{\sum_{i,j} a_{ij}^2} = \sqrt{trace\left(\mathbf{A}^T\mathbf{A}\right)} \tag{2.19}$$

The trace of a square matrix $\mathbf{A}$ is defined as the sum of the diagonal entries of $\mathbf{A}$ Saikia [2009].

Using the trace function properties Meyer [2000](Equation 2.20 and Equation 2.21) and differentiating function f with respect to x_i's, it can be shown that the optimal item evaluation vector $\mathbf{v}$ can be found by computing the centroid (or mean) of the columns in $\mathbf{K}$ Langville and Meyer [2012].

$$trace\left(a\mathbf{A} + \mathbf{B}\right) = a\,trace\left(\mathbf{A}\right) + trace\left(\mathbf{B}\right) \tag{2.20}$$

$$trace\,(\mathbf{AB}) = trace\,(\mathbf{BA}) \tag{2.21}$$

From Equations 2.20 and 2.21 it follows that:

$$\begin{aligned} f(\mathbf{x}) &= trace\ \left[\mathbf{K} - \left(\mathbf{x}\mathbf{e}^T - \mathbf{e}\mathbf{x}^T\right)\right]^T \left[\mathbf{K} - \left(\mathbf{x}\mathbf{e}^T - \mathbf{e}\mathbf{x}^T\right)\right] \\ &= trace\,\mathbf{K}^T\mathbf{K} - trace\left[\mathbf{K}^T\left(\mathbf{x}\mathbf{e}^T - \mathbf{e}\mathbf{x}^T\right) + \left(\mathbf{x}\mathbf{e}^T - \mathbf{e}\mathbf{x}^T\right)^T\mathbf{K}\right] \\ &+ trace\left[\left(\mathbf{x}\mathbf{e}^T - \mathbf{e}\mathbf{x}^T\right)^T\left(\mathbf{x}\mathbf{e}^T - \mathbf{e}\mathbf{x}^T\right)\right] \end{aligned} \tag{2.22}$$

Because, as was shown before, $\mathbf{K}$ is skew symmetric, it follows that:

$$f(\mathbf{x}) \quad = \quad trace\,\mathbf{K}^T\mathbf{K} - 4\mathbf{x}^T\mathbf{K}\mathbf{e} + 2n\left(\mathbf{x}^T\mathbf{x}\right) - 2\left(\mathbf{e}^T\mathbf{x}\right)^2 \tag{2.23}$$

Therefore, by differentiating f with respect to x_i's and setting the results to 0 $\frac{\partial f}{\partial x_i} = 0$, using the fact that $\frac{\partial x}{\partial x_i} = \mathbf{e}_i$ and the skew symmetry property of $\mathbf{K}$, it can be shown that

$$\mathbf{v} \quad = \quad \frac{\mathbf{K}\mathbf{e}}{n} \tag{2.24}$$

is the optimal evaluation vector that minimizes the difference between score differences and evaluation differences Langville and Meyer [2012]. Then the teams can be ranked according to these evaluations in a decreasing order.

Using the data from our running example in Table 2.1 (the one also used in the book), the $\mathbf{K}$ matrix can be produced:

$$\mathbf{K} = \begin{pmatrix} 0 & -45 & -3 & -31 & -45 \\ 45 & 0 & 18 & 8 & 20 \\ 3 & -18 & 0 & 2 & -27 \\ 31 & -8 & -2 & 0 & 38 \\ 45 & -20 & 27 & 38 & 0 \end{pmatrix}$$

Then the optimal evaluation vector can be computed:

$$\mathbf{v} = \frac{\mathbf{Ke}}{5} = \begin{array}{l} Duke \\ Miami \\ UTC \\ UVA \\ VT \end{array} \begin{pmatrix} -24.8 \\ 18.2 \\ -8 \\ -3.4 \\ 18 \end{pmatrix}$$

The ranking system based on minimization of the difference between score differences and evaluation differences produces a ranking by sorting teams according to the optimal evaluations derived by minimization. The result of a ranking of 5 teams from our running example produced by such a system is shown in Table 2.4.

Team	Optimal evaluation	Rank
Miami	18.2	1
VT	18	2
UVA	-3.4	3
UTC	-8	4
Duke	-24.8	5

Table 2.4: Ranking teams by minimizing the difference between score differences and evaluation differences

As mentioned previously, the evaluation spread techniques are not only applicable to the sports ratings and rankings. Here we will delve a bit deeper into the application of evaluation spread techniques to recommender systems and present the non-personalized ranking technique for recommending items shown in Langville and Meyer [2012].

Suppose we have users' preference data in the form of numerical ratings given by the users to a subset of items. Suppose also, that they are given in a 5-star scale (as in Amazon.com or movielens.org).

The goal that CF RSs raise in such case is to produce a recommendation list of the items to the target user based on patterns of ratings of like-minded users. Langville and Meyer [2012] raise a different goal. They assume that there is a ranking that is different to the one produced by using immediately the ratings and search for a unique ranking given the opinion of all the users. That is, they want to aggregate the preferences of all the users and produce a global unique ranking of the items.

With this goal in mind they build a product evaluation system from user recommendations by reinterpreting the skew-symmetric score-differential matrix **K** in Equation 2.15. Namely, in a team ranking scenario S_{ij} represented the sum of points that team i scored against team j. In an item "recommender" (actually just global ranking) scenario that they introduce, S_{ij} is the average of ratings that item i received from all the users who have rated both items i and j.

If we denote the number of users that have rated both products i and j by $m_{ij} = |U_{ij}|$, then the entries of the *score matrix* in Equation 2.26 (i.e. the average rating that item i receives from users who rated both items i and j) are calculated as follows:

$$S_{ij} = \begin{cases} \frac{1}{m_{ij}} \sum_{u \in U_{ij}} r_{ui} & m_{ij} \neq 0 \\ 0 & m_{ij} = 0 \end{cases} \tag{2.25}$$

r_{ui} is the rating that the user u gave to the item i.

$$\mathbf{S} = \begin{pmatrix} 0 & S_{12} & \cdots & S_{1n} \\ S_{21} & 0 & \cdots & S_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ S_{n1} & S_{n2} & \cdots & 0 \end{pmatrix} \tag{2.26}$$

And the score differences for item i and item j become:

$$k_{ij} \quad = \quad S_{ij} - S_{ji} = \frac{1}{m_{ij}} \sum_{u \in U_{ij}} (r_{ui} - r_{uj}) \tag{2.27}$$

The *score-differential matrix* $\mathbf{K}$ is calculated in the same way as in Equation 2.15.

Using the reasoning that the centroid of the evaluation spreads in the sports teams ranking scenario provides the optimal evaluation derived from point spreads, Langville and Meyer [2012] also derives the conclusion that for a given set of item ratings, the best set of evaluations that can be derived by averaging ratings is given by the centroid. The formula for calculating the optimal evaluations of the items is the same as was derived in a case of team ranking in Equation 2.24.

Here $\mathbf{K}$ is the skew-symmetric matrix of average rating differences given in Equation 2.15 and $\mathbf{e}$ is a column vector of all ones.

Up to now in the item "recommender" scenario the rating data were modified to produce the spread ratings. However, in the RS domain sometimes instead of the rating data, pairwise preferences may be used, either because the numerical ratings are not available, or simply because the binary preferences may reflect the user preferences more accurately Gleich and Lim [2011]; Salimans et al. [2012].

The technique for ranking using pairwise preferences presented in Langville and Meyer [2012] as well is very similar. Instead of ratings, the preference function δ defined in Equation 2.28 is introduced. Here $i \succ_u j$ means that the user u prefers the item i over the item j.

$$\delta_{ij}^{u} = \begin{cases} 1 & i \succ_u j \\ -1 & i \prec_u i \end{cases} \tag{2.28}$$

S_{ij}'s are then defined in Equation 2.29 and $\mathbf{K}$ is the skew-symmetric matrix of pairwise preference score differences defined in Equation 2.30.

$$S_{ij} = \begin{cases} \frac{1}{m_{ij}} \sum_{u \in U_{ij}} \delta_{ij}^{u} & m_{ij} \neq 0 \\ 0 & m_{ij} = 0 \end{cases} \tag{2.29}$$

$$k_{ij} \quad = \quad [K_{ij}] = [S_{ij} - S_{ji}] \tag{2.30}$$

The best set of ratings that can be derived from pairwise preferences are calculated by the same formula as in Equation 2.24.

Let's illustrate the ranking system presented above using a small example. Imagine we have 4 items and 10 users that are comparing some of those items. Suppose a set of pairwise preferences that the users gave to some of the items is summarized by a pairwise preference matrix:

$$\delta = \begin{array}{c} \\ u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ u_7 \\ u_8 \\ u_9 \\ u_{10} \end{array} \begin{array}{c} \begin{array}{cccc} i_1 & i_2 & i_3 & i_4 \end{array} \\ \begin{pmatrix} 1 & & 0 & \\ 1 & & & 0 \\ 0 & & 1 & \\ 0 & 1 & & \\ & 0 & & 1 \\ 0 & 1 & & \\ & 1 & & 0 \\ 1 & & 0 & \\ & 0 & & 1 \\ 0 & & & 1 \end{pmatrix} \end{array}$$

This is a simple example in which all users compare one pair of items each. It means that, for example user u_3 compares items i_1 and i_3 and prefers i_3 over i_1. The resulting pairwise preference score matrix is then as follows:

$$\mathbf{S} = \begin{pmatrix} 0 & 0 & 2/3 & 1/2 \\ 1 & 0 & 0 & 0 \\ 1/3 & 0 & 0 & 0 \\ 1/2 & 1 & 0 & 0 \end{pmatrix} \tag{2.31}$$

The skew-symmetric pairwise preference difference matrix $\mathbf{K}$ is:

$$\mathbf{K} = \begin{pmatrix} 0 & -1 & 1/3 & 0 \\ 1 & 0 & 0 & -1 \\ -1/3 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \tag{2.32}$$

Therefore, the pairwise preference item evaluations derived from Equation 2.24 is:

$$\mathbf{v} \quad = \quad \frac{\mathbf{Ke}}{n} = \frac{\mathbf{Ke}}{4} = \begin{pmatrix} -2/12 \\ 0 \\ -1/12 \\ 3/12 \end{pmatrix} \tag{2.33}$$

The ranking system based on minimization of the difference between pairwise preference score differences and evaluation differences produces a ranking by sorting the items according to the optimal evaluations derived by minimization. The result of a ranking of 4 items from the running example produced by such a system is shown in Table 2.5.

Item	Optimal evaluation	Rank
i_4	3/12	1
i_2	0	2
i_3	-1/12	3
i_1	-2/12	4

Table 2.5: Ranking items by minimizing the difference between pairwise preference score differences and evaluation differences

However, as was already argued before, the ranking produced by such a ranking system has no particular value in recommender systems. It solves a different problem - predicting global evaluations of the items based on their comparisons in a form of rating average differences (or match score differences in sports teams domain) and ranking items based on the predicted evaluations. It does not provide any personalization. That is, if a recommender system based on such a ranking would recommend the items with the highest evaluations, all the users would get the same recommendations no matter what preference history and patterns they have. Because of this reason we aim to modify the item ranking techniques by making them personalized and in this way adapting them to the RSs. In the next chapter such modifications are presented.

2.3 Preference Acquisition

The precision of the CF ranking (and thus the quality of the online experiment that we conduct) depends on the characteristics of the ranking algorithm in the first place. Besides that, the distribution, the quantity and the quality of the data (ratings or binary preferences) known by the system plays a big role in the system's performance Elahi et al. [2013]. Thus, it is important to let the system acquire new and valuable data. Therefore, the right choice and ordering of the items (or item pairs) that will be shown for the user when he is asked to give his preferences is invaluable.

In case of standard collaborative filtering where the users are asked to rate items, the problem is how to choose and order the items that they would have high probability to be rated by the user and whose ratings will better improve the system accuracy. Rating elicitation strategies are analyzed quite extensively Elahi et al. [2011]; Rashid et al. [2008, 2002]. Six most analyzed techniques for explicitly determining the items to ask a user to rate are:

- *entropy*, where items with the largest rating entropy are preferred. Entropy is a diversity measure for categorical (nominal) data.
- *random request*;
- *popularity*, which is measured as the number of ratings for an item, and hence the most frequently rated items are selected;
- $log(popularity) \cdot entropy$, where items that are both popular and have diverse ratings are selected ($log(pop) \cdot entropy$);
- *item-item personalized*, where random items are proposed until the user rates some items. Similarity between items is then computed in order to select other items that the user is likely to have seen. The list of similar movies is updated each time the user rates more items.

It was reported that $log(pop) \cdot entropy$ (Equation 2.34) performs best at the signup scenario.

$$\begin{aligned}(log(pop) \cdot entropy)(i) &= log(m_i) \cdot \left(- \sum_{k=r_{min}}^{r_{max}} \frac{\#_k^i}{m_i} \cdot log(\frac{\#_k^i}{m_i}) \right) \\ &= log(m_i) \cdot \left(-\frac{1}{m_i} \cdot \sum_{k=r_{min}}^{r_{max}} (\#_k^i \cdot log(\#_k^i)) + \frac{log(m_i)}{m_i} \cdot \sum_{k=r_{min}}^{r_{max}} \#_k^i \right) \\ &= log(m_i) \cdot \left(log(m_i) - \frac{1}{m_i} \cdot \sum_{k=r_{min}}^{r_{max}} \#_k^i \cdot log(\#_k^i) \right),\end{aligned} \tag{2.34}$$

where $m_i = |U_i|$, $U_i = \{u \in U | \exists r_{u,i}\}$, r_{max} is the maximum rating and r_{min} is the minimum rating in the whole scale and

$$\#_k^i = |\{r_{ui} : u \in U_i, r_{ui} = k\}|.$$

If the recommender is based on pairwise preferences, a question arises how to choose the item pairs that have a high probability that a user can express pairwise preferences on these pairs and whose pairwise preference would better improve the system accuracy.

However, there is not much research on pairwise preference elicitation strategies in RS. Brun et al. [2010] only mentions that "choosing the appropriate questions to ask - each new question depending on the preceding answers - is a challenging problem". Wang et al. [2012] adapt the vector space model to ranking-based collaborative filtering. That is, they treat each user as a document and his pairwise preferences as terms. Then they use a degree-specialty weighting scheme similar to the TF-IDF scheme used in information retrieval to weight the terms and the cosine similarity to select the neighbors for the target user to make recommendations. They do not discuss it, but the scheme presented could be used as an active learning strategy.

Salimans et al. [2012] propose a Bayesian factor model for learning user preference rankings for the purpose of making product recommendations and show how the model can be used as an active learning strategy where only a small number of most informative items is selected. They aim to select the most informative items to present to the user for feedback by maximizing the expected entropy reduction in the posterior distribution of the parameters contained in the Bayesian factor model.

Chapter 3

Recommendation Techniques

In this chapter we explain the recommendation techniques that we have developed. In Section 3.1 we present a Personalized Differential Matrix with Ratings (PDMR) technique that uses only rating data. In Section 3.2 we describe a Personalized Differential Matrix with Pairwise Preferences (PDMPP) technique that uses only pairwise preference data. In Section 3.3 we present a Personalized Differential Matrix with Ratings and Pairwise Preferences (PDMRPP) technique that uses a combination of ratings and pairwise preferences.

3.1 Personalized Differential Matrix with Ratings (PDMR)

The approach described in Section 2.2 produces a global ranking, i.e. there is no personalization in it. Each user would get the same list of items (in the user preference rating example from Langville and Meyer [2012] the items are products, like in Amazon.com; in our thesis experiments the items are movies), no matter what preferences he has. Therefore, we modify the technique described in Section 2.2 in order to take into account the personalization and temporal slicing.

Personalization is the core of the research on recommender systems. It allows to suggest individual recommendations to users depending on their preferences. Temporal slicing is the concept widely used in databases. It allows expressing queries on time-varying (relational or XML) data. In the context of our thesis we refer to temporal slicing as to the process of dividing the data depending on some datastamp. In our algorithms we use even a more general type of temporal slicing - slicing by session.

In order to reason about the choice to use slicing by session, let's define what a session is. A session is sometimes defined as a limited time of communication between a client and a server. In the RSs context it can be the period of time that the user is logged in the system. Based on the hypothesis that the users are not stable in their rating behavior we conjecture that using temporal slices to define rating (or pairwise preference) acquisition sessions may improve the recommender system. In this way the preferences that are given in the same session are weighted stronger. That is, the preferences that are given in the same session have a greater impact in the model.

While the login session divides the user preferences in a most natural way, sessions can also be artificially created by defining a duration of time that spans the session, i.e. if two ratings are given in a time interval that is shorter than an hour, then they are assigned to the same session. This is in fact how we use temporal slicing in our offline experiments, because the MovieLens dataset has no information about the sessions, just the timestamps of the ratings.

In order to incorporate session data (the more general case of temporal slicing) into the ranking techniques analyzed before, the procedure of ranking presented in Section 2.2 is modified as follows. Instead of computing the *score matrix* **S** and then producing the *score-differential matrix* **K**, which requires calculating the sets U_{ij} for each pair of items i and j, we process the ratings of each user u given in the same session s and compute the rating differences $c_{ij}^{us} = r_{ui}^{s} - r_{uj}^{s}$ for the item pairs that are present in the user's u profile and that were rated in the same session s.

The non-personalized k_{ij}'s are computed as the average of rating differences across all the users and all the sessions:

$$k_{ij} \quad = \quad AVG_{us}\left(c_{ij}^{us}\right) = \frac{1}{m_{ij}} \cdot \sum_{u \in U_{ij}} \frac{1}{|S_u|} \cdot \sum_{s \in S_u} c_{ij}^{us} \tag{3.1}$$

AVG denotes an average, S_u - the set of the sessions in which user u gave some ratings.

In PDMR a personalized **K** matrix is calculated for the target user (the user for which the recommendations are made). The values of such **K** matrix weight stronger the rating differences of the more similar users.

w_{uv} denotes the user-user similarity. As was described in Section 2.1, it can be computed using metrics such as (adjusted-) cosine similarity and correlation-based similarity (Pearson, Spearman Correlation). Because there exist experiments reporting better performance of both Pearson correlation and cosine similarity when comparing them, in our experiments we arbitrarily choose Pearson correlation calculated as in Equation 3.2

$$w_{uv} \quad = \quad \frac{\sum_{j \in I_{uv}} (r_{uj} - r_u)(r_{vj} - r_v)}{\sqrt{\sum_{j \in I_{uv}} (r_{uj} - r_u)^2 \sum_{j \in I_{uv}} (r_{vj} - r_v)^2}} \tag{3.2}$$

The problem of using just the Pearson correlation to measure similarity is that the rating data are typically sparse, i.e. user-to-user similarity weights are often computed on few ratings given to common items: I_{uv}. Therefore, we take into account this problem by using the *significance of similarity* metric defined in Equation 3.3, which shows how dependable the measure of similarity is - and not only its value - when making a rating prediction Herlocker et al. [1999].

$$w'_{uv} = \frac{min\left(|I_{uv}|, \gamma\right)}{\gamma} \cdot w_{uv} \tag{3.3}$$

γ is the parameter that is data-dependent and must be cross-validated. Using MovieLens dataset consisting of 100,000 ratings, Herlocker et al. [2002, 1999] found that using $\gamma \geq 25$ could significantly improve the accuracy of the predicted ratings, and that a value of 50 for γ gave the best results in general neighborhood-based collaborative filtering.

For the target user v, the values of the personal $\mathbf{K}$ matrix are calculated as follows:

$$k^v_{ij} = WAVG_{us}\left(c^{us}_{ij}\right) = \frac{1}{\sum_{u \in U_{ij}, s \in S} w'_{u,v}} \sum_{u \in U_{ij}, s \in S} w'_{uv} \cdot c^{vs}_{ij} \tag{3.4}$$

$WAVG$ denotes a weighted average.

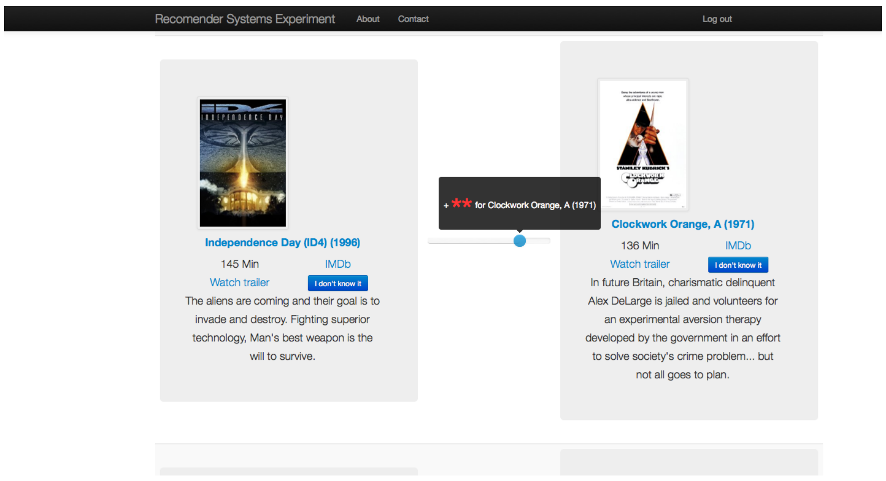
Note that the sum in the denominator of Equation 3.4 ranges over s as well as over u, because if e.g. the pair of items was rated by the same user u in two different sessions s_1 and s_2, then both $c^{us_1}_{ij}$ and $c^{us_2}_{ij}$ will be added to the sum in the numerator. Therefore, in order to normalize the weight w_{uv} must be added twice.

3.2 Personalized Differential Matrix with Pairwise Preferences (PDMPP)

The goal of our thesis is to analyze if incorporating pairwise preferences into collaborative filtering recommendation methods improve the recommendation accuracy. In order to do that we aim to develop the techniques that use pairwise preferences and make personalized recommendations to the users based on them. Going towards this objective, in Section 3.1 we presented the technique that transforms ratings to pairwise preferences and produce the personalized rankings.

In this section the technique derived in Section 3.1 is adapted to the case of direct pairwise preferences instead of numerical ratings. In order to comply with the range of the values of the rating differences $c^{us}_{ij} \in \{-4, -3, -2, -1, 0, 1, 2, 3, 4\}$ (recall that the ratings are given in a 5-star rating scale), we redefine δ (the preference function, which was previously defined in Equation 2.28 as a binary preference function).

This is the approach to record the pairwise preferences that we used in our online experiments asking the users to provide pairwise preferences with the help of the sliders to reflect the required scale. That is, the users were given pairs of movies and asked how much more they like one movie than the other, meaning how many more stars they would give (if they had to) for one movie than for the other. For example, the preference shown in Figure 3.1 means that


Recomender Systems Experiment
About
Contact
Log out
Independence Day (ID4) (1996)
145 Min
IMDb
Watch trailer
I don't know it
+ ** for Clockwork Orange, A (1971)
Clockwork Orange, A (1971)
136 Min
IMDb
Watch trailer
I don't know it

The aliens are coming and their goal is to invade and destroy. Fighting superior technology, Man's best weapon is the will to survive.

In future Britain, charismatic delinquent Alex DeLarge is jailed and volunteers for an experimental aversion therapy developed by the government in an effort to solve society's crime problem... but not all goes to plan.

Figure 3.1: Users Compare Movies

the user would give 2 stars more to the movie "A Clockwork Orange" than to "Independence Day".

While we say that this way of recording pairwise preferences is used in our experiments, the exact model is slightly different as it has to incorporate data of different types - both ratings and pairwise preferences. Here, we suppose we have the set of users that have provided only pairwise preferences and build a model using only this data.

The user preference function δ is defined in Equation 3.5.

$$\delta_{ij}^{us} = \begin{cases} \in \{4,3,2,1\} & i \succ_{us} j \\ 0 & \text{no preference} \\ \in \{-1,-2,-3,-4\} & i \prec_{us} i \end{cases} \tag{3.5}$$

Here $i \succ_{us} j$ means that the user u prefers the item i over the item j in the session s. The magnitude of the preference function value is dependent on the strength of the user preference, i.e. how much one item is preferred to another.

In order to build the Personalized Differential Matrix with Pairwise Preferences model, we need the similarity measure between user profiles consisting of pairwise preferences. We need the similarity measure to be as much compatible to the Pearson correlation among user's ratings as possible because we aim to build a model that combines the preference data of both types - ratings and pairwise preferences - in one model which will be presented in the next section.

The previous RS literature on pairwise preferences did not introduce any similarity measure between user profiles described with pairwise preferences because these preferences were obtained from ratings. Thus, because the original ratings are still available, general variants of similarity measures based on ratings before transforming them to pairwise preferences are used

Brun et al. [2010]. However, here it cannot be computed in this way (or as in Equation 3.2) because the user profile in this case consists of pairwise preferences rather than numerical ratings.

Others represent the users as vectors of either negative or positive (depending on the binary preference given to the item pair) degree-specialty weights similar to term frequency - inverse document frequency weights in information retrieval and calculate the standard cosine similarity between these vectors Wang et al. [2012].

For the user-user similarity computation in PDMPP we use neither of the methods mentioned above, but the Pearson correlation among users' pairwise preferences. Recall that the pairwise preferences are also numerical as we adjusted the scale to be compliant to the 5-star rating differences.

The weight w_{uv} that we use in our experiments measuring user-user similarity in terms of their pairwise preferences is given in Equation 3.6.

$$w_{uv} = \frac{\sum_{i \in I_{uv}, j \in I_{uv}, i<j} (\delta^u_{ij} - \delta^u)(\delta^v_{ij} - \delta^v)}{\sqrt{\sum_{i \in I_{uv}, j \in I_{uv}, i<j} (\delta^u_{ij} - \delta^u)^2 \sum_{i \in I_{uv}, j \in I_{uv}, i<j} (\delta^v_{ij} - \delta^v)^2}} \tag{3.6}$$

Here δ^u is the user's u average of all pairwise preferences.

Let's illustrate the computation of the Pearson correlation among users' u and v pairwise preferences by a small example.

Let

$$\begin{array}{c} \\ i_1 \\ i_2 \\ i_3 \\ i_4 \end{array} \begin{array}{c} \begin{array}{cccc} i_1 & i_2 & i_3 & i_4 \end{array} \\ \begin{pmatrix} & 3 & 1 & \\ -3 & & -1 & \\ -1 & 1 & & 2 \\ & & -2 & \end{pmatrix} \end{array} \quad \text{and} \quad \begin{array}{c} \\ i_1 \\ i_2 \\ i_3 \\ i_4 \end{array} \begin{array}{c} \begin{array}{cccc} i_1 & i_2 & i_3 & 4 \end{array} \\ \begin{pmatrix} & 2 & 0 & \\ -2 & & -1 & -1 \\ 0 & 1 & & \\ & 1 & & \end{pmatrix} \end{array}$$

be the matrices representing the pairwise preferences of users u and v respectively.

Then the Pearson correlation between these two users' pairwise preferences is calculated as follows:

$$\begin{aligned} w_{uv} &= \frac{\left(3 - \frac{5}{4}\right)(2-0) + \left(1 - \frac{5}{4}\right)(0-0) + \left(-1 - \frac{5}{4}\right)(-1-0)}{\sqrt{\left(\left(3 - \frac{5}{4}\right)^2 + \left(1 - \frac{5}{4}\right)^2 \left(-1 - \frac{5}{4}\right)^2\right)\left((2-0)^2 + (0-0)^2 + (-1-0)^2\right)}} \\ &= \frac{\frac{14}{4} + 0 + \frac{9}{4}}{\sqrt{\frac{131}{16} \cdot 5}} \approx 0.82 \end{aligned}$$

Several problems associated with such similarity computation could arise. First of all, pairwise preference matrices are usually very sparse - much sparser than the rating data. Therefore, the probability of finding the common item pairs that are given preferences by two users is even smaller than in the ratings case. For example, if there are 100 items in the dataset, and the two users give 10 ratings each, then the probability that an item is rated by both users is $0.1 \cdot 0.1 = 0.01$. On the other hand, 100 items make $\frac{100(100-1)}{2} = 4950$ item pairs. Thus, if two users give 10 pairwise preferences each, the probability of the pairwise preference given by both users is $\frac{10}{4950} \cdot \frac{10}{4950} \approx 4 \cdot 10^{-6}$. In other words, to have the same probability to be able to compare two users, users have to give 495 pairwise preferences each (which is very unlikely) as opposed to 10 ratings each in case of the rating based technique. This problem might result in the inability to compute similarities between many user pairs.

We address this problem, in both offline and online experiments by filtering out the users that have rated too little items in the training set. However, it does not fully solve the problem - there is still not quite sufficient number of users that can be compared with a target one. Another approach is to compute ratings from the pairwise preferences and then use the standard method for similarity computation. It has it's own disadvantages yet and the comparison of the two approaches is another topic for research.

Similarly as in PDMR presented in the previous section, here we also use significance of similarity to increase the weight of similarity in case of many common item pairs and to decrease it otherwise. The modification of the original *significance of similarity* used for rating data is defined in Equation 3.7.

$$w'_{uv} = \frac{min\left(|\mathcal{P}_{uv}|, \gamma\right)}{\gamma} \cdot w_{uv} \tag{3.7}$$

$\mathcal{P}_{uv}$ denotes the set of item pairs that have been compared by both users u and v. γ is the parameter that must be cross-validated.

The score differences are calculated by a formula similar to Equation 3.4 replacing c^{us}_{ij} with δ^{us}_{ij} in it as in Equation 3.8. For the target user v, the score differences are computed as follows:

$$k^{v}_{ij} = WAVG_{us}\left(\delta^{us}_{ij}\right) = \frac{1}{\sum_{u \in U_{ij}, s \in S} w'_{u,v}} \sum_{u \in U_{ij}, s \in S} w'_{uv} \cdot \delta^{us}_{ij} \tag{3.8}$$

3.3 Personalized Differential Matrix with Ratings and Pairwise Preferences (PDMRPP)

The main motivation for developing the algorithms in Sections 3.1 and 3.2 was the prospect of combining them in order to be able to use data of both types - ratings and pairwise preferences. Such a technique would enable us to test the main hypothesis of our thesis - that incorporating pairwise preferences in a CF model, i.e., using a mixture of ratings and pairwise preferences, can improve the recommendation acuracy.

Suppose, we have data that consists of a set of users that have provided only ratings and a set of users that have provided only pairwise preferences. This is the case that we have in our online experiments using the MovieLens rating data combined with the pairwise preferences collected during the experiment. The techniques that are presented in this section can easily be modified to incorporate the mixed data, i.e., also the users that have provided both ratings and pairwise preferences. That is, the preference type can differ not only between users (users giving ratings and users giving pairwise preferences), but also within users (users giving both ratings and pairwise preferences). Here we present the former case (i.e., preference types differ only between users) because as was mentioned before it is the case of the online experiment conducted by us.

In order to produce the combined model, we use the user preference function defined in Equation 3.5. The ratings that are also available are then converted to pairwise preferences by calculating the rating differences.

Let's illustrate the idea with a simple example where just one session exists. Suppose, we have a rating matrix $\mathbf{R}$ of 10 users and 4 items:

$$\mathbf{R} = \begin{array}{c} \\ u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ u_7 \\ u_8 \\ u_9 \\ u_{10} \end{array} \begin{array}{c} \begin{array}{cccc} i_1 & i_2 & i_3 & i_4 \end{array} \\ \begin{pmatrix} 4 & 2 & 2 & \\ 3 & 1 & & 2 \\ 1 & & 2 & \\ 2 & 4 & 2 & \\ & 3 & 2 & 5 \\ 2 & 3 & & \\ & 4 & 1 & 3 \\ 3 & 1 & 1 & \\ & 3 & 2 & 5 \\ 2 & & 2 & 4 \end{pmatrix} \end{array}$$

Suppose also that the target user v gives such pairwise preferences:

$$\begin{array}{c} \\ i_1 \\ i_2 \\ i_3 \\ i_4 \end{array} \begin{array}{c} \begin{array}{cccc} i_1 & i_2 & i_3 & i_4 \end{array} \\ \begin{pmatrix} & 2 & 0 & \\ -2 & & -1 & -1 \\ 0 & 1 & & \\ & 1 & & \end{pmatrix} \end{array}$$

It means that we have four known user preference function values: $\delta_{12}^v = 2$, $\delta_{13}^v = 0$, $\delta_{23}^v = -1$

and $\delta^v_{24} = -1$. (Because half of the entries are opposite of the other half, they are not listed.)

Then for each user $u_1, \dots, u_{10}$, their ratings are transformed to pairwise preferences. For example, for the users u_1 and u_2 the pairwise preference matrices would be as follows:

$$\begin{array}{c|cccc} & i_1 & i_2 & i_3 & i_4 \\ \hline i_1 & & 2 & 2 & \\ i_2 & -2 & & 0 & \\ i_3 & -2 & 0 & & \\ i_4 & & & & \end{array} \quad \text{and} \quad \begin{array}{c|cccc} & i_1 & i_2 & i_3 & 4 \\ \hline i_1 & & 2 & & 1 \\ i_2 & -2 & & & -1 \\ i_3 & & & & \\ i_4 & -1 & 1 & & \end{array}$$

Thus, we have these inferred user preference function values: $\delta^{u_1}_{12} = 2$, $\delta^{u_1}_{13} = 2$, $\delta^{u_1}_{23} = 0$ and $\delta^{u_2}_{12} = 2$, $\delta^{u_2}_{14} = 1$, $\delta^{u_2}_{24} = -1$.

Then the similarity between the target user v and users u_1 and u_2, computed as in Equation 3.6, would be $w_{vu_1} \approx 0.73$ and $w_{vu_2} \approx 0.91$. For the other users that have provided ratings, the data is transformed and the similarities are computed in the same way.

The weights are again improved by significance of similarity defined in Equation 3.7.

The k^v_{ij}'s are computed as in Equation 3.8 using the inferred user preference function values in place of real ones.

Chapter 4

Research Hypotheses

In this chapter we explain the research hypotheses that were briefly presented in Section 1.2, in more detail.

As mentioned before, the main research hypothesis is whether incorporating pairwise preferences into recommender systems that use standard CF methods improves recommendation prediction accuracy in terms of the normalized Discounted Cumulative Gain ($nDCG$) and precision - the standard IR metrics for measuring recommendation prediction accuracy. We try to prove it by developing a novel technique for producing recommendations that uses pairwise preferences in combination with the ratings (PDMRPP technique) and comparing it to the matrix factorization model that uses only ratings as user preferences which is a state-of-the-art model in CF. The PDMRPP technique is making use of the ideas presented by Langville and Meyer [2012] in a totally different domain and adjusting them for the case of recommender systems. It was described in Section 3.3.

In order to test the hypothesis we conduct an online experiment in a movie domain. We use MovieLens dataset as the basis of both recommender systems models (PDMRPP and MF). Besides that, additional user preference data of the users taking part in an experiment is gathered by us. In the experiment some users are asked to provide preferences in a form of ratings while the others - in a form of pairwise preferences. The users are then treated differently depending on a form of preferences they have provided. The ones who have provided ratings are given recommendation list produced by the matrix factorization model that uses MovieLens rating data and our users' rating data. The ones who have provided pairwise preferences are given recommendation list produced by the PDMRPP technique that uses MovieLens rating data and our users' pairwise preference data. All the users are asked to provide feedback about the recommendations they have received by indicating what are the good recommendations. This enables us to compute $nDCG$ and precision in order to compare the techniques.

Besides the recommendation prediction accuracy, we conjectured that the pairwise preference usage in RS improves the users' perceived recommendation quality. Perceived recommendation quality is measured by a questionnaire that contains the following statements:

1. I liked the items recommended by the system;
2. The recommended items fitted my preference;
3. The recommended items were well-chosen;
4. The recommended items were relevant.
5. The system recommended too many bad items;
6. I didn't like any of the recommended items;
7. The items I selected were "the best among the worst".

We also expect the perceived recommendation quality to increase in terms of a pooled score that combines the users' responses to both the negative and the positive statements mentioned above. In order to test these assumptions, after the users get the recommendation lists we ask them to evaluate the strength of the statements listed above in a 5-point Likert scale.

There are several side hypotheses which we are interested in as well. One of them is that the users are more inclined to give pairwise preferences than ratings because they get more satisfaction from comparing items than from giving ratings to them. We test this side assumption in the online experiment by asking the users to provide feedback about the preference elicitation process (not taking into account the recommendation prediction quality) in a form of statements that are listed below:

1. I have fun using the system;
2. Using the system is a pleasant experience;
3. The system makes me more aware of my choice options.
4. I feel bored when I am using the system.

We use a 5-point Likert scale for users' responses and expect the higher/lower scores for positive/negative aspects in case when the users are comparing items than in case when they rate them.

The strongest assumption (i.e. the most likely) out of the statements about the preference elicitation process mentioned above is that the system in which the users compare the movies is making them much more aware of their choice options than the system in which the users rate the movies. We believe that if such an improvement exists, it can serve as an incentive for the users to provide more preferences and thus enable the system to produce more accurate recommendations. Also, the overall user satisfaction in the preference elicitation process is expected to be higher among the users who provide pairwise preferences in terms of a pooled score that combines the users' responses to the statements mentioned above.

Motivated by the reasoning that the conversational way of eliciting users' preferences has certain advantages, we raise another assumption. We hypothesize that adding a step in a RS which asks the users to provide some more pairwise preferences results in a recommendation prediction accuracy improvement in terms of $nDCG$ and precision. We test this assumption by asking the users to mark the movies that they know in the recommendation step of our online experiment, i.e. when they get the recommendation list. Then we ask them to compare those movies that they have marked as known by providing pairwise preferences between them. We then produce another recommendation list where the users mark the good recommendations again and test whether the recommendation accuracy in terms of $nDCG$ and precision is higher than the one of the initial recommendation list.

As was mentioned before, most of the research about pairwise preferences in recommender systems that we are aware of apply the transformation step that creates pairwise preferences from the existing ratings, which causes a loss of information. We believe that such a loss of information actually exists, but we do not expect it to be extreme. We make two hypotheses with respect to this issue. Firstly, the movie ranking accuracy of the Non-Personalized Differential Matrix with Ratings NPDMR (Langville and Meyer [2012] variant) technique is expected to be lower than the ranking accuracy based on the movie rating averages (RA). Secondly, the ranking accuracy of the Personalized Differential Matrix with Ratings PDMR (developed by us, but using the transformation of ratings to pairwise preferences) is assumed to be lower than the movie ranking accuracy of the matrix factorization model. In both cases the hypothesis is based on Kendall's τ - the ranking correlation measure - comparisons.

To test these hypotheses, we conduct offline experiments. We use MovieLens data split into test and training sets. First, for each user we produce the real ranking of the items from the test set that he has rated. Then we build the NPDMR and RA models using the users' ratings transformed to pairwise preferences and produce two rankings: one for each model (NPDMR and RA). Kendall's τs between NPDMR and real ranking as well as between RA and real ranking are computed and compared. We expect that NPDMR will have a lower Kendall's τ than RA because of the explained information loss.

Similarly, we build the PDMR and MF models using the users' ratings transformed to pairwise preferences and produce two rankings: one for each model (PDMR and MF). Kendall's τs between PDMR and real ranking as well as between MF and real ranking are computed and compared. We expect that PDMR will have a lower Kendall's τ than MF again because of the explained information loss.

Another hypothesis that we raise originates from the fact that the users are not stable in their rating behavior. Thus, we hypothesize that using temporal slices to define user preference acquisition sessions may improve rating prediction accuracy of the recommender system. We test this assumption in an offline experiment (using the same MovieLens dataset as previously mentioned) by defining sessions as a duration of 24 hours. If the two ratings were given closer in time than 24 hours then we assume they were given in the same session. We incorporate such sessions in our developed PDMR model and expect the item ranking quality increase compared to the MF algorithm in terms of Kendall's τ.

Chapter 5

Preference Acquisition

The online experiment that we conduct in order to test the research hypotheses described in the previous chapter includes two different recommender systems. The detailed description of the experimental strategy will be given in the next chapter. In this chapter we explain one technical issue that is very important for the quality of the experiment.

One of the two recommender systems that we implement is using a MF technique that is based solely on rating data and the other is using PDMRPP technique that is based on both rating data and pairwise preference data. Therefore, we have to acquire preferences in two different forms: ratings and pairwise preferences.

Because the precision of the CF ranking outcome (and therefore the quality of the online experiment that we conduct) depends not only on the characteristics of the ranking algorithm but also on the distribution, the quantity and the quality of the data (ratings or binary preferences) known by the system, it is important to let the system acquire new and valuable data. Therefore, the right choice and ordering of the movies (or movie pairs) that will be shown for the user when he is asked to give his preferences is invaluable.

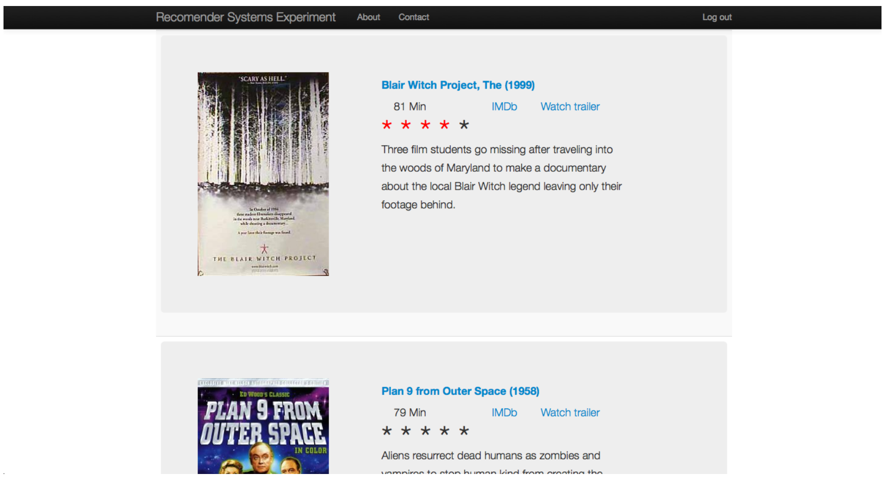
To understand the problem better, imagine the users are asked by the system to provide ratings for the movies as in Figure 5.1 or pairwise preferences as in Figure 5.2. But how the system knows which movies (or movie pairs) to show first? Let's split this problem in two cases: rating request generation for movies, and pairwise preference request generation for movie pairs.

In Sections 5.1 and 5.2 we will present the strategies for rating requests generation for movies and pairwise preference requests generation for movie pairs, respectively.

5.1 Rating Requests' Generation For Movies

The criteria for choosing the movies that the user is solicited to rate (Figure 5.1) should clearly include:

1. The presented movies should have high probability that can be rated by the user, i.e. the



Blair Witch Project, The (1999)

81 Min

★★★★★

Three film students go missing after traveling into the woods of Maryland to make a documentary about the local Blair Witch legend leaving only their footage behind.

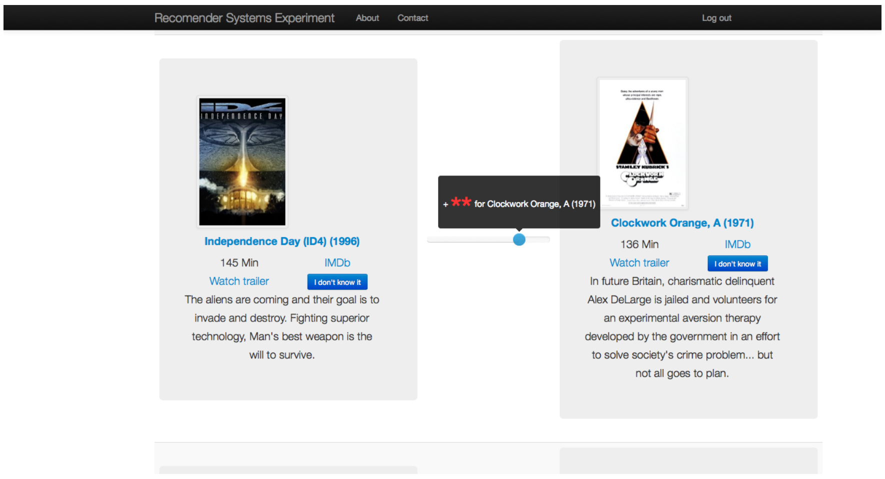
Plan 9 from Outer Space (1958)

79 Min

★★★★★

Aliens resurrect dead humans as zombies and

Figure 5.1: Users Rate Movies in System R



Independence Day (ID4) (1996)

145 Min

The aliens are coming and their goal is to invade and destroy. Fighting superior technology, Man's best weapon is the will to survive.

+ ** for Clockwork Orange, A (1971)

Clockwork Orange, A (1971)

136 Min

In future Britain, charismatic delinquent Alex DeLarge is jailed and volunteers for an experimental aversion therapy developed by the government in an effort to solve society's crime problem... but not all goes to plan.

Figure 5.2: Users Compare Movies in System P

probability that the user has seen the movie should be high;

2. The presented movies should be those whose ratings will better improve the system accuracy. In some sense they must be the movies that, after the new rating is acquired, are improving the predicted ordering of the movies.

Six techniques for explicitly determining the items to ask a user to rate analyzed by Rashid et al. [2002] and presented previously in Section 2.3 include *entropy*, *random request*, *popularity*, $log(popularity) \cdot entropy$ and *item-item personalized*.

It was reported that $log(pop) \cdot entropy$ (previously given in Equation 2.34) performs best at the signup scenario. However, we argue that $log(pop) \cdot entropy$ measure has a disadvantage in case of the numerical ratings. Generally, entropy is a diversity measure for categorical (nominal) data. Therefore, the entropy increases equally in presence of two very different and two just slightly different items, i.e. the entropy for the set $S = \{1, 5\}$ would be the same as for the set $T = \{2, 3\}$ $(= 1)$.

Therefore, we propose a modification of this measure, which solves the nominal data problem. Instead of using entropy we propose to use variance. Variance is a diversity measure for ordinal data. Thus, it is more suitable for the ratings than entropy.

The chosen measure - $log(pop) \cdot variance$ - is given in Equation 5.1. It addresses the two criteria listed above by scoring higher the most popular movies ($log(pop)$ part) with the most diverse ratings (*variance* part). The higher the value of this score for the item i, the higher the item is ranked.

$$(log(pop) \cdot variance)(i) = log(m_i) \cdot \left(\frac{1}{m_i} \cdot \sum_{u \in U_i} (r_{ui} - r_i)^2 \right) \tag{5.1}$$

5.2 Pairwise Preference Requests' Generation For Movie Pairs

The criteria for choosing the movie pairs that the user is asked to compare (Figure 5.2) should clearly include:

1. The presented movie pairs should have high probability that a user can express pairwise preferences on these pairs, i.e. the probability that the user has seen both movies should be high;
2. The presented movie pairs should be those for which the knowledge of their pairwise preference will better improve the system accuracy. In some sense they must be the pairs that, after the new pairwise preference is acquired, the system, retrained on this new set, will improve the predicted ordering of the movies.

The importance of the second criteria can be illustrated by an example. Suppose that the movie pairs presented to the user to compare are ordered solely by their popularity (and so the second criterion is not addressed). Then think of some movie that is much more popular than the others. Then at the beginning the user will be asked to compare all the other movies with this one. Clearly, it is not the best scenario as the user either most of the time will prefer it over the others or vice versa. Also, think about the movies belonging to a common series of which all of the movies are very popular. It is not hard to find examples: "Star Wars", "Lord of the Rings", "Hanibal Lecter Trilogy". Then the user will be asked to compare the different parts of this movie even if they are very similarly rated by the users, i.e. if some user likes one part, most likely he also likes the other parts. Thus, the pairwise preferences given by the user will not provide much added value to the model.

To address both the first and the second criteria listed above, we score a pair of movies according to Equation 5.2. Then the movie pairs are presented to the users for acquiring their preference by decreasing value of their scores. This score accounts for items' popularity by incorporating the multiplication of the logarithms of the item popularities. Also, by incorporating a measure of decorrelation between the ratings of two movies ($1 - correlation$), it chooses the least correlated movie pairs.

$$log(m_i) \cdot log(m_j) \cdot \left(1 - \frac{\sum_{u \in U_{i,j}} (r_{ui} - r_i) \cdot (r_{uj} - r_j)}{\sqrt{\sum_{u \in U_i} (r_{ui} - r_i)^2 \cdot \sum_{u \in U_j} (r_{uj} - r_j)^2}}\right), \tag{5.2}$$

Both of the measures in Equation 5.1 and Equation 5.2 were used in our online experiments in order to improve the recommendation quality by enhancing the preference elicitation process.

Chapter 6

Experimental Strategy

The hypotheses of the thesis that we test in both offline and online experiments were described in Chapter 4. In brief, we have such expectations:

- In our online experiment, we expect to measure an improvement of recommendation accuracy in terms of $nDCG$ and precision when pairwise preferences are being incorporated into the standard CF model (PDMRPP) as opposed to the state-of-the-art CF model (MF);
- In the online experiment, we expect the users to be more satisfied with the preference elicitation process when the system asks them to compare movies rather than rate them;
- In the online experiment, we expect the system that uses both ratings and pairwise preferences as user preference data (PDMRPP technique) to perform better in terms of the users' perceived recommendation quality as opposed to the system that uses only ratings as user preference data (MF technique);
- In the online experiment, we expect to measure an improvement of the recommendation accuracy in terms of $nDCG$ and precision as additional pairwise preferences are gathered in the PDMRPP; however, we expect a lower improvement after additional ratings are gathered in case of the system using only rating data (MF);
- In the offline experiment, we expect to observe a minor advantage of the rating average technique over the NPDMR technique and a minor advantage of the MF over the PDMR in terms of rating accuracy measured by Kendall's τ. The reason of this hypothesis is the loss of information induced by the transformation of ratings to pairwise preferences in Differential Matrix with Ratings techniques;
- In the offline experiments, we expect the improvement of rating accuracy in terms of Kendall's τ when the session data is incorporated in the PDMR model as compared to the case when it is not taken into account.

Table 6.1 lists the recommendation techniques and their properties used in the experiments.

Full name	Acronym	Experiment	Data
Matrix Factorization	MF	online & offline	ratings
Rating Average	RA	offline	ratings
Personalized Differential Matrix with Ratings	PDMR	offline	ratings
Personalized Differential Matrix with Pairwise Preferences	PDMPP	-	pairwise preferences
Personalized Differential Matrix with Ratings and Pairwise Preferences	PDMRPP	online	ratings & pairwise preferences

Table 6.1: Summary of Recommendation Techniques and Their Properties

In this chapter we explain the strategy of the experiments conducted in order to test the hypotheses summarized above. In Section 6.1 and 6.2 we describe the experimental strategy of offline and online experiments, respectively.

6.1 Offline Experiments

The main objective is to see if the proposed method produces a ranked list that is close to the target and to check how close it is compared with the ranking produced by matrix factorization. Let's first describe the data that we use in order to perform such a comparison.

We use MovieLens[1] 100k data set. MovieLens data sets were collected by the GroupLens Research Project[2] at the University of Minnesota. The dataset used for our experiment:

- Consists of 100,000 ratings (1-5) from 943 users on 1682 movies.
- Each user has rated at least 20 movies.
- Simple demographic info for the users (age, gender, occupation, zip) is given.

The data was collected through the MovieLens web site[3] during a seven-month period from September 19th, 1997 through April 22nd, 1998. This data has been cleaned up by the providers by removing the users who had less than 20 ratings or did not have complete demographic information.

Let's now clarify what the main problem is. The data described above is split into test and training sets. For each user from the test set, the real ranking of the items from the test set that he has rated is produced. Let's call it a target ranking. Then two rankings are produced for each user as well: one ranking is produced by PDMR algorithm developed by us and the

[1]http://www.movielens.org/
[2]http://www.grouplens.org/
[3]http://movielens.umn.edu/

other by MF. As a result we have three ranked lists of the same movies, i.e., the movies from the test set that a user has rated, for each user. The problem is to compare how close to the target ranking each of two produced ranking lists (using PDMR and MF) is, i.e. how strong the correlation between the PDMR and target ranking and between MF and target ranking is.

A well known measure of ranking correlation coefficient is Kendall's τ Kendall [1938], Kendall [1970]. There are several variations of this measure. The simplest one is Kendall's τ_a. Let's briefly discuss how it is calculated. Imagine, that user u has rated the items $i_1, i_2, \ldots, i_{10}$ from the test set in such a way that the item i_1 is ranked highest, i_2 - second highest and similarly until i_{10} which he has rated lowest. Therefore, ranking

$$i_1, i_2, i_3, i_4, i_5, i_6, i_7, i_8, i_9, i_{10} \tag{6.1}$$

is the target ranking for user u. Now suppose, that algorithm X using the ratings from the training set, produced such a ranking of the items $i_1, \ldots, i_{10}$:

$$i_1, i_4, i_7, i_2, i_3, i_6, i_5, i_8, i_9, i_{10} \tag{6.2}$$

Let's now define that a pair of items A and B is concordant in two ranked lists if the items are ordered in the same way (A before B in both lists or B before A in both lists) in those lists. If on the other hand A is before B in one list but B is before A in the other list, then the pair is discordant in those two lists. For example, the pair of movies i_1 and i_2 is concordant between the target ranking and X ranking, but the pair of movies i_2 and i_4 is discordant between the target ranking and X ranking.

Kendall's τ_a is then defined as follows:

$$\tau_a = \frac{l_c - l_d}{l_0} \tag{6.3}$$

where l_0 is the number of all possible pairs of items, l_c is the number of concordant pairs of items and l_d is the number of discordant pairs of items.

In our example, the number of concordant pairs of items between target ranking and ranking produced by algorithm X - l_c^X - is 38, while the number of discordant pairs l_d^X is 7. Therefore, Kendall's τ_a between target ranking and the ranking produced by X is:

$$\tau_a^X = \frac{l_c^X - l_d^X}{l_0} = \frac{38 - 7}{\frac{10 \cdot (10-1)}{2}} = \frac{31}{45} \tag{6.4}$$

The range of possible values of Kendall's τ_a is $[-1, 1]$. In case the two rankings are the same, the coefficient has value 1. In case one ranking is the reverse of the other, the coefficient has value 1. In case the two rankings are independent, Kendall's τ_a is expected to be close to 0.

However, in our offline experiment the target ranking is produced for each user by sorting the user's ratings which are given in a 5-star rating scale. It is obvious, that in case a user has rated more than 5 movies in the test set, there will necessarily be ties in the target ranking for that user. A tie in a ranking occurs when two or more items are ranked the same. For example, several movies are rated 5 stars by some user in the test set and thus they are all ranked first in the target ranking for that user. A group of ties is a set consisting of more than one item ranked the same. For example, in our experiment the movies that are rated 5 stars by some user in the test set and therefore ranked first in the target ranking of that user, form one group of ranking ties. If there is only one movie that is rated 5 stars by a user in the test set and therefore ranked first in the target ranking for that user, it does not form a group of ranking ties.

Kendall's τ_a does not take into account the ties in rankings. However, Kendall's τ_b is the modification of it which makes adjustments for ties Agresti [2010]. Because the rating scale is small (5-star) in the MovieLens data and thus the ties in the target rankings are unavoidable, in our experiments we use Kendall's τ_b given in Equation 6.5:

$$\tau_b = \frac{l_c - l_d}{\sqrt{(l_0 - l_1) \cdot (l_0 - l_2)}} \tag{6.5}$$

where l_c is the number of concordant pairs, l_d is the number of discordant pairs, l_0 is the number of all possible pairs (Equation 6.6), l_1 is the number of ties in the first ranking list (Equation 6.7) and l_2 is the number of ties in the second ranking list (Equation 6.8).

$$l_0 = \frac{n \cdot (n-1)}{2} \tag{6.6}$$

$$l_1 = \sum_{i \in T_1} \frac{t_i \cdot (t_i - 1)}{2} \tag{6.7}$$

$$l_2 = \sum_{j \in T_2} \frac{t_j \cdot (t_j - 1)}{2} \tag{6.8}$$

where n is the number of elements ranked, T_1 is the set of groups of ties in the first ranking list, T_2 is the set of groups of ties in the second ranking list, t_i is the number of tied values in the i^{th} group of ties of the first ranking list and t_j is the number of tied values in the j^{th} group of ties of the second ranking list.

Let's recall now the main problem - it is to compare how close to the target ranking each of two produced ranking lists (using PDMR and MF). We already have a coefficient of correlation between two ranked lists - Kendall's τ_b. What we need next is a statistic for testing the difference of the correlation between target ranking and PDMR ranking and the correlation between the target ranking and MF ranking.

There exists a statistic z_B derived from Kendall's τ_b for testing the presence of correlation between two ranked lists Agresti [2010]. z_B is given in Equation 6.9. It has the same distribution as τ_b which is approximately equal to a standard normal distribution when the ranked lists are statistically independent. We do not present the derivation of the test statistic z_B and the explanation of the values v, v_0, v_t, v_u, v_1 and v_2 here. More details can be found in Agresti [2010].

$$z_B = \frac{l_c - l_d}{\sqrt{v}} \tag{6.9}$$

$$v = \frac{v_0 - v_t - v_u}{18} + v_1 + v_2$$

$$v_0 = n(n-1)(2n+5)$$

$$v_t = \sum_i t_i(t_i - 1)(2t_i + 5)$$

$$v_u = \sum_j u_j(u_j - 1)(2u_j + 5)$$

$$v_1 = \frac{\sum_i t_i(t_i-1) \sum_j u_j(u_j-1)}{2n(n-1)}$$

$$v_2 = \frac{\sum_i t_i(t_i-1)(t_i-2) \sum_j u_j(u_j-1)(u_j-2)}{9n(n-1)(n-2)}$$

If $z_B \sim \tau_b$ and the distribution of τ_b is approximately equal to standard normal, then in order to test the difference of two τ_bs, i.e., the difference of the correlations between two pairs of ranked lists, we derive the test:

$$\frac{z_B^1 - z_B^2}{\sqrt{2}} \tag{6.10}$$

which is also approximately standard normal.

This is the test that we use in our offline experiments. Their results are presented in Chapter 7.

6.2 Online Experiments

In this section we describe the online experiment that we conduct in order to test the hypotheses described in Chapter 4 and briefly reminded at the beginning of this chapter.

In order to test these hypotheses, we implemented two recommender systems: one that is based on the Personalized Differential Matrix with Ratings and Pairwise Preferences (PDM-RPP) model that is described in Section 3.3 and one that is based on the Matrix Factorization model that is described in Section 2.1.

The experiment conducted consists of two stages. During the first stage of the experiment the data about user likes and dislikes are gathered in a form of either ratings or pairwise preferences depending on the system to which they are assigned. Besides that, users are asked questions about the preference elicitation process. During the second stage of the experiment, the users are given the lists of recommendations and asked to provide feedback about them. This allows us to compare the different aspects of the two techniques that we hypothesize about: personalized PDMRPP algorithm and MF algorithm.

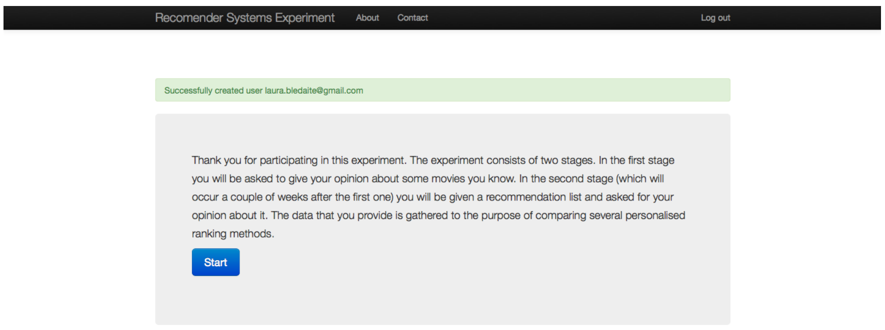
Before the start of the experiment the system is bootstrapped with the MovieLens data - the ratings of 100 movies selected by the criterion described in Chapter 5. It is worth noting that the data gathered during the first stage is just a minority compared to the existing ratings taken from MovieLens dataset. Because of that the PDMRPP model uses a mixture of preference data (ratings from MovieLens dataset and pairwise preferences gathered from new users during our experiment), where the fraction of the pairwise preferences is much smaller than the fraction of ratings. However, even if the Matrix Factorization model uses only rating data, it is also a mixture of the ratings from MovieLens dataset and the ratings gathered from the new users during our experiment. The same as for the PDMRPP model, the fraction of ratings gathered during our experiment is much smaller than the fraction of rating from MovieLens dataset.

First Stage

At the very beginning the users are invited to take part in the experiment by email. The first time a user comes to the website, he has to register entering the email address only. The email address is necessary because the users have to be noticed when the second stage begins. The system does not require password because our experiment consists of two stages. Therefore, users would have to remember their passwords. And generally, they tend to forget not important passwords, especially the ones that they are not using extensively. As a result, the more users forget their password, the more experiment participants we lose.

The experiment follows a between group strategy. One group (we will refer to it as group R because the users provide ratings) consists of the users using the system R based on the MF technique, another group (we will refer to it as group P because the users provide preferences) consists of the users using the system P based on PDMRPP technique. When the user registers he is assigned to one of these two systems: R or P. Users using the system R provide ratings, while users using the system P provide pairwise preferences.

In order to make the comparison of the algorithms as compatible as possible, the number of the users in both groups must be approximately equal. To ensure this, every other user is assigned to a different system. For example, odd users according to the registration time are assigned to system R, even users - to system P. After the user successfully registers to the system, he is redirected to a page with a short explanation of the experiment he is taking part in, the data that is gathered and the reasons why the data is necessary. The screenshot of this step is given in Figure 6.1.



Successfully created user laura.bledaite@gmail.com

Thank you for participating in this experiment. The experiment consists of two stages. In the first stage you will be asked to give your opinion about some movies you know. In the second stage (which will occur a couple of weeks after the first one) you will be given a recommendation list and asked for your opinion about it. The data that you provide is gathered to the purpose of comparing several personalised ranking methods.

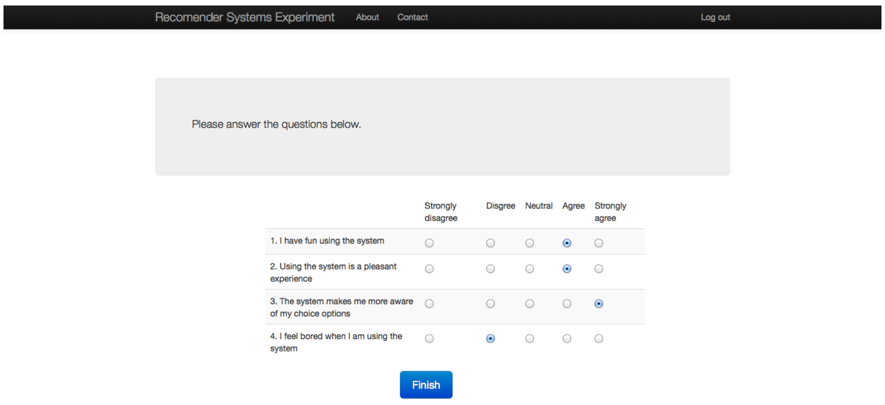
Figure 6.1: Explanation of the Experiment

The precision of the ranking outcome depends not only on the characteristics of the ranking algorithm but also on the distribution, the quantity and the quality of the user preference data that the system collects. Thus, it is important to let the system acquire new and valuable data. Therefore, in order to get the valuable preference data we select 100 movies from the MovieLens dataset that score highest according to the criterion described in Chapter 5.

On the next page, in case the user is attached to group R, the list of 10 movies per page (out of 100 selected movies) is shown. The user is asked to rate as many movies as he would like to. The fact that the user is not asked a specific number of ratings to provide, will also enable us to compare the two approaches by means of the pure amount of data (5 star ratings versus pairwise preferences) that the user deliberately provides. Every listed movie is presented by a title, a production year, a photo, a short description and a trailer. Next to this, the 5 star rating scale is given, where the user has to mark his/her rating. The screenshot of the page where users are asked to rate the movies was shown in Figure 5.1 when we presented the preference acquisition technique in Chapter 5.

In case the user is attached to group P, a list of movie pairs is shown on the next page. The user is asked to provide as many pairwise preferences of movies as he would like to in order to enable us to compare the pure amount of data (ratings versus pairwise preferences) that the user deliberately provides. The paginated list contains all the possible movie pairs (the selection and the ordering is described in Chapter 5). We show 10 movie pairs per page. The pagination is used because of the large number of pairs. If a user finds a movie that he does not know he can press the button"I dont know it" in order to filter out all comparisons containing this movie. The screenshot of the page where users are asked to compare the movies was shown in Figure 5.2 when we presented the preference acquisition technique in Chapter 5.

When the user finishes entering the ratings or the pairwise preferences, he presses the button "Finish". He is then redirected to the page where several statements about the preference elicitation process (that were presented in Chapter 4 where we described the research hypotheses) are given in Figure 6.2. He has to state his opinion on them. A 5-level Likert scale is used to scale the responses.



Please answer the questions below.

	Strongly disagree	Disgree	Neutral	Agree	Strongly agree
1. I have fun using the system	○	○	○	●	○
2. Using the system is a pleasant experience	○	○	○	●	○
3. The system makes me more aware of my choice options	○	○	○	○	●
4. I feel bored when I am using the system	○	●	○	○	○

Figure 6.2: Statements at the End of the First Stage

After that the thanks are expressed and the user is made aware that when the second stage starts, he will be informed by email again.

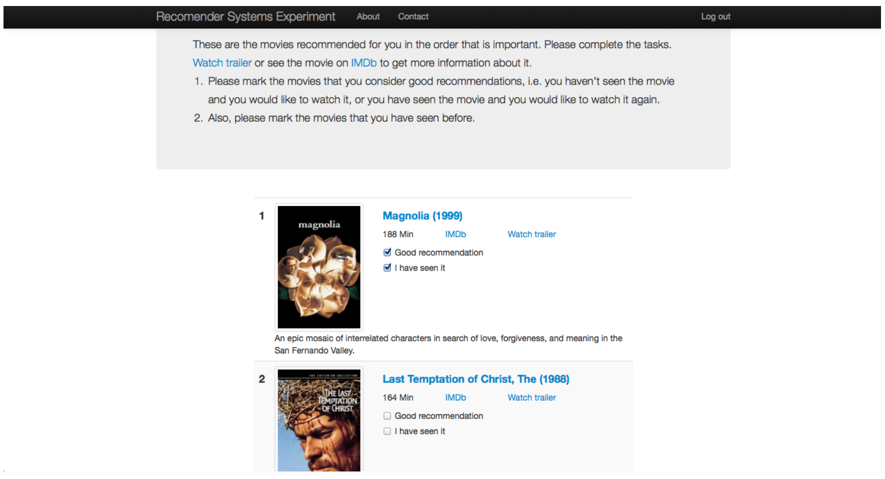
Second Stage

The users are invited to the second stage by email. The interaction with the system differs again based on the group. When the user accesses the system during the second stage, first he is redirected to the page with a reminder about the tasks that he has to fulfill. The "Start" button leads him to the beginning of the second stage of the experiment.

The movies from the list of 100 movies' selection are split into two equal parts. We split them because in our experiment we intend to compare the recommendation lists produced by the model based only on the preferences given in the first stage and the model based also on the additional preferences that are gathered after the initial recommendation list is produced. We conjecture that the latter produces a recommendation list of higher accuracy in terms of $nDCG$ and precision.

The splits are produced as follows. The movies that were not rated by the user during the first stage are ordered by the $log(pop) \cdot variance$ criterion. Odd movies from this ordering are attached to split 1, even ones to split 2. We do not include the rated movies in the splits because in the experiment we include only 100 movies. Therefore, if we include the rated movies among the possible recommendations, it is a high chance that many high rated movies (and no unrated ones) will occur in the recommendation lists. Thus, in this case it will be more difficult to reflect the recommendation accuracy of the RSs being compared.

The initial recommendation list is produced only from movies in split 1 which contains less than 50 movies. The user is then presented with a page containing a list of 5 movie recommendations (Figure 6.3). In case the user is from group R (in the first stage he provided ratings), the list is generated using MF algorithm, otherwise, using PDMRPP algorithm.



These are the movies recommended for you in the order that is important. Please complete the tasks.
Watch trailer or see the movie on IMDb to get more information about it.

1. Please mark the movies that you consider good recommendations, i.e. you haven't seen the movie and you would like to watch it, or you have seen the movie and you would like to watch it again.
2. Also, please mark the movies that you have seen before.

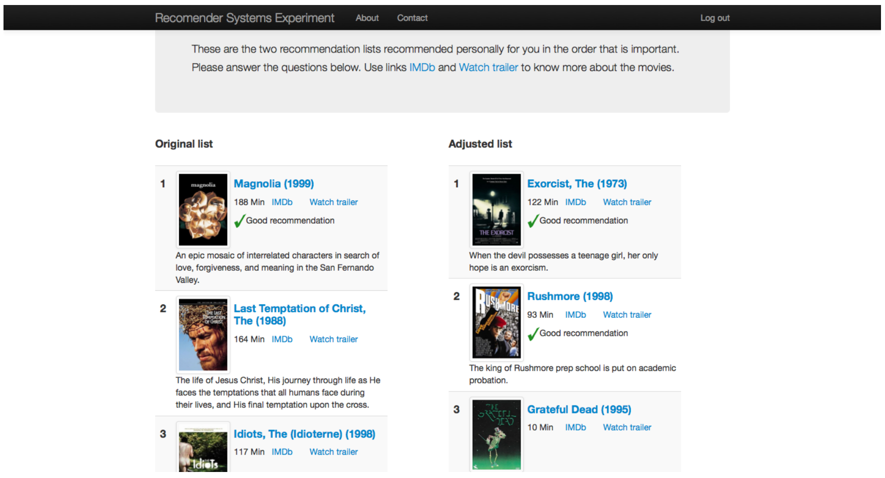
1 Magnolia (1999)
188 Min IMDb Watch trailer
☑ Good recommendation
☑ I have seen it
An epic mosaic of interrelated characters in search of love, forgiveness, and meaning in the San Fernando Valley.

2 Last Temptation of Christ, The (1988)
164 Min IMDb Watch trailer
☐ Good recommendation
☐ I have seen it

Figure 6.3: List of Recommendations

The movies are presented with the same information as in the first stage: a title, a production year, a photo and a trailer. The trailers of the recommended movies are included in order to help user better answer the questions related to this list and evaluate recommendations. First of all, the user is asked to mark all the movies that are good recommendations for him (with an explanation what a good recommendation means). We need to know which recommendations were good, in order to calculate $nDCG$ and precision and compare the different techniques (recall research hypotheses in Chapter 5). Secondly, the user is asked to mark the movies in the recommendation list that he already knows. The set of known movies among those that were recommended initially are needed because we intend to compare the initial recommendation list and the recommendation list improved by updating the model with the additional preferences.

When the user presses "Continue", he is redirected to the page where he is asked additional feedback about the movies that he has marked as seen before. In case he is interacting with the system R, he rates those movies, otherwise, he compares them. Afterwards, he is redirected to a page with a new list of recommendations which is made by MF and PDMRPP for users interacting with system R and P respectively. The user marks the good and the seen recommendations again.



These are the two recommendation lists recommended personally for you in the order that is important.
Please answer the questions below. Use links IMDb and Watch trailer to know more about the movies.

Original list

1 Magnolia (1999)
188 Min IMDb Watch trailer
✓ Good recommendation
An epic mosaic of interrelated characters in search of love, forgiveness, and meaning in the San Fernando Valley.

2 Last Temptation of Christ, The (1988)
164 Min IMDb Watch trailer
The life of Jesus Christ, His journey through life as He faces the temptations that all humans face during their lives, and His final temptation upon the cross.

3 Idiots, The (Idioterne) (1998)
117 Min IMDb Watch trailer

Adjusted list

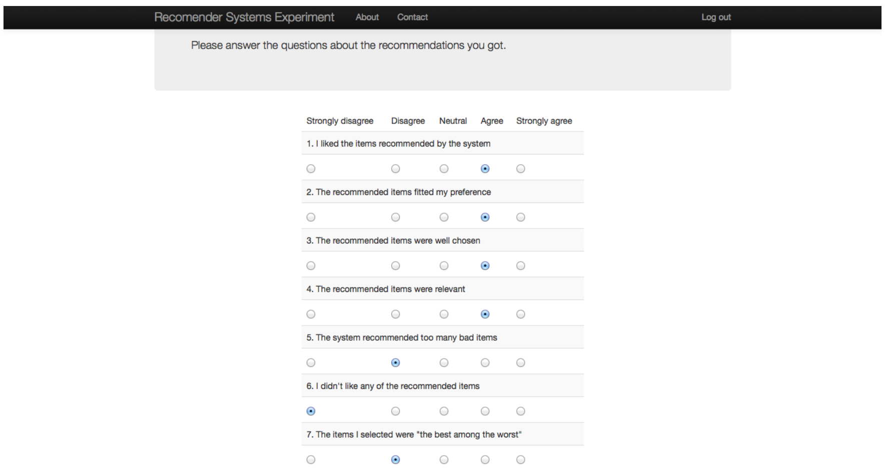
1 Exorcist, The (1973)
122 Min IMDb Watch trailer
✓ Good recommendation
When the devil possesses a teenage girl, her only hope is an exorcism.

2 Rushmore (1998)
93 Min IMDb Watch trailer
✓ Good recommendation
The king of Rushmore prep school is put on academic probation.

3 Grateful Dead (1995)
10 Min IMDb Watch trailer

Figure 6.4: Comparison of Recommendation Lists

Finally, the user is shown both the original and the improved recommendation lists (together with the information about which movies were marked as good and which were marked as seen in both of the lists) and is asked to compare them (Figure 6.4). The system shows both recommendation lists in one page because we think that this allows the user to better compare them. The slider is used for comparison. When he is done, he presses the "Continue" button and is redirected to a page with a list of questions that evaluate the perceived recommendation quality (Figure 6.5). The set of questions, that has been already presented in Chapter 4 where we described the research hypotheses, has already been used in similar experiments Knijnenburg et al. [2012].



Please answer the questions about the recommendations you got.

Statement	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
1. I liked the items recommended by the system	○	○	○	✓	○
2. The recommended items fitted my preference	○	○	○	✓	○
3. The recommended items were well chosen	○	○	○	✓	○
4. The recommended items were relevant	○	○	○	✓	○
5. The system recommended too many bad items	○	✓	○	○	○
6. I didn't like any of the recommended items	✓	○	○	○	○
7. The items I selected were "the best among the worst"	○	✓	○	○	○

Figure 6.5: Statements at the End of the Second Stage

Chapter 7

Offline Experiment Results

As was mentioned before, most of the research about pairwise preferences in recommender systems that we are aware of apply the transformation step that creates pairwise preferences from the existing ratings, which is a loss of information. We believe that such a loss of information actually exists, but we do not expect it to be extreme. We raise two hypotheses in relation to this loss of information. Firstly, the item ranking quality of the NPDMR technique (Langville and Meyer [2012] variant described in Section 2.2) is expected to be close to the ranking quality based on the item rating averages. Secondly, the ranking quality of the PDMR technique (developed by us and described in Section 3.1, but using the transformation of ratings to pairwise preferences) is assumed to be close to the item ranking quality of the matrix factorization model. In both cases the hypothesis will be tested using Kendalls τ_b - the ranking correlation measure that takes into account ranking ties - comparisons.

Overall, the offline tests that were performed include two main comparisons:

1. Non-Personalized Differential Matrix with Ratings (NPDMR) versus Rating Average (RA).
2. Personalized Differential Matrix with Ratings (PDMR) versus MF.

Before presenting the results of the offline experiments, let's recall some notation already used in the thesis and also introduce some new. We defined a training set $\mathcal{D}_{train}$ to be the set of the user-item pairs (u, i) for which the rating r_{ui} is known and on which we build some RS model. Let's define a test set $\mathcal{D}_{test}$ to be the set of the user-item pairs (u, i) for which the rating r_{ui} is known, but on which we test the outcome of some RS model.

7.1 Non-Personalized Differential Matrix with Ratings (NPDMR) versus Rating Average (RA)

We split the MovieLens data into test and training sets with exactly 10 ratings per user in the test set. For each user we produce the target ranking of the items in the test set $\mathcal{D}_{test}$. That is, we order the movies according to the available ratings in the test set and call this ordering the target ranking. Then we build the NPDMR model described in Section 2.2 using the users' ratings from the training set transformed to pairwise preferences and for each user produce a predicted ranking of the movies in the test set based on this model. We also build a ranking based on item rating averages (we call it RA), i.e. order the movies in the test set according to their overall rating average. Kendall's τ_bs between the ranking based on the NPDMR model and the target ranking as well as between the ranking based on the rating average and the target ranking are computed. We calculate the averages of Kendall's τ_bs between the target ranking and the rankings based on each of the models (NPDMR and RA) over all the users u for which there exist ratings in the test set, i.e., over the users $u : (u,i) \in \mathcal{D}_{test}$. The calculation of Kendall τ_b average is given in Equation 7.1. We test whether the correlation between the target ranking and the ranking based on the NPDMR model differs from the correlation between the target ranking and the ranking based on the RA using the z_B test explained in Section 6.1. We expect that the NPDMR will be farther from the target ranking than RA in terms of Kendall's τ_b because of the explained information loss.

$$\overline{\tau_b}^{\text{model}} = \frac{1}{m} \cdot \sum_{u:(u,i)\in\mathcal{D}_{test}} \tau_b^{u,model} \tag{7.1}$$

Here $\tau_b^{u,model}$ is the Kendall's τ_b between the target ranking and the ranking produced by some RS model calculated for a user u.

The results of this experiment can be seen in Table 7.1. The first column of the table shows the average of the Kendall's τ_bs over all the users $u : (u,i) \in \mathcal{D}_{test}$ based on the NPDMR algorithm. The second column shows the average of the Kendall's τ_bs over all the users $u : (u,i) \in \mathcal{D}_{test}$ based on the ranking produced by ordering the movies by overall movie rating average.

$\overline{\tau_b}^{\text{NPDMR}}$	$\overline{\tau_b}^{\text{RA}}$	z_B test value	p-value
0.2602	0.2816	1.5427	0.1229

Table 7.1: Test for the Difference of Ranking Correlation with the Target Ranking for NPDMR and RA

From Table 7.1 we can see that the ranking based on overall movie rating average (RA) slightly outperforms the ranking based on NPDMR model, i.e. the correlation between the former and the target ranking is higher than the correlation between the latter and the target ranking in terms of Kendall's τ_b. However, the difference is not statistically significant.

We conjectured that NPDMR is farther from the target ranking than RA in terms of Kendall's τ_b because of the explained information loss. However, we were not able to prove that which means that the loss of information mentioned before is not extreme. In spite of that, the slight difference of the ranking quality suggests that the real advantages of the models using pairwise preferences could only be noticed in a situation where the pairwise preferences are collected directly and not the ratings transformed to the pairwise preferences.

7.2 Personalized Differential Matrix with Ratings (PDMR) versus Matrix Factorization (MF)

We use the same split of MovieLens data into test and training sets with exactly 10 ratings per user in the test set. For each user we produce the target ranking of the items in the test set $\mathcal{D}_{test}$ that he has rated in the same way as explained previously. Then we build the PDMR model described in Section 3.1 using the users' ratings from the training set transformed to pairwise preferences and for each user we produce a ranking of the movies that the user has rated and that are in the test set, i.e., of the movies $i : (u, i) \in \mathcal{D}_{test}$, based on this model. Also, for each user u we build a ranking of the movies in the test set that he has rated, i.e., of the movies $i : (u, i) \in \mathcal{D}_{test}$, based on MF model described in Section 2.1. Kendall's τ_bs between the ranking based on the PDMR and the target ranking as well as between the ranking based on the MF model and the target ranking are computed. We calculate averages of Kendall's τ_bs between the target ranking and the ranking based on each of the models (PDMR and MF) over all the users u for which there exist ratings in the test set $\mathcal{D}_{test}$. We test whether the correlation between the target ranking and the ranking based on the PDMR model differs from the correlation between the target ranking and the ranking based on the MF model using the z_B test explained in Section 6.1. We expect that PDMR will have a lower correlation with the target ranking compared to MF.

For Matrix Factorization model we use the parameters proposed originally by Funk [2006]: learning rate $\gamma_{MF} = 0.001$, regularization parameter $\lambda_{MF} = 0.02$. We perform the same test described above for different variations of the PDMR algorithm. We generate four variants by using or not the significance parameter and session data (see Section 3.1).

In this experiment two movies are considered to be rated in the same session if they are rated within 24 hours.

To get the optimal value of γ - the parameter of the significance of similarity - for the PDMR, five-fold cross-validation was performed. The original five 80%/20% splits $(u_1, u_2, u_3, u_4, u_5)$ of the data into training and test data from the MovieLens dataset were used. The dataset contains 100000 ratings. We use the same splits that were used by Herlocker et al. [1999]. They were generated as follows:

1. split u_1 has ratings 1 through 20000 in the test set and ratings 20001 through 100000 in the training set;
2. split u_2 has ratings 20001 through 40000 in the test set and ratings 1 through 20000 plus ratings 40001 through 100000 in the training set;
3. split u_3 has ratings 40001 through 60000 in the test set and ratings 1 through 40000 plus ratings 60001 through 100000 in the training set;
4. split u_4 has ratings 60001 through 80000 in the test set and ratings 1 through 60000 plus ratings 80001 through 100000 in the training set;
5. split u_4 has ratings 80001 through 100000 in the test set and ratings 1 through 80000 in the training set.

Clearly, the test sets of the splits $u1, \ldots, u5$ are disjoint. However, this way of generating the sets does not guarantee that for each user there exist ratings in the test set as it is in the data split used in the previous experiment. The experiment is repeated with each training and test set and the results are averaged. Figure 7.1 shows the results of the cross-validation. We found $\hat{\gamma} = 21$ to be the optimal γ value based on the averaging the Kendall's τ_bs across all the users in the test set of each split and across all the five splits. Therefore, we use it in our further experiments.

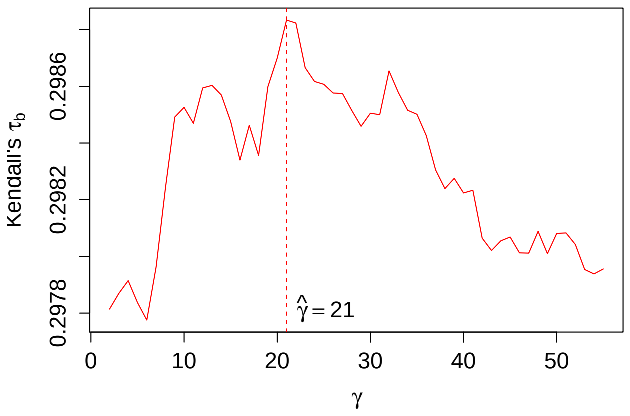


Figure 7.1: γ Cross-Validation

The results of this experiment can be seen in Tables 7.2 and 7.3. Tables 7.2 lists the averages (over the users $u : (u, i) \in \mathcal{D}_{test}$) of Kendall's τ_b between the rankings based on each of the variations of the PDMR model described previously and the target ranking as well as the average (over the users $u : (u, i) \in \mathcal{D}_{test}$) of the Kendall's τ_b between the target ranking and the ranking based on the MF model.

PDMR without significance, without session	PDMR without significance, with session	PDMR with significance, without session	PDMR with significance, with session	MF
0.2858	0.2780	0.2856	0.2782	0.2991

Table 7.2: Kendall's τ_b averages for PDMR and MF

Test	p-value	z_B
PDMR without session, without significance vs MF	0.9672	0.3334
PDMR without session, with significance vs MF	1.5369	0.1243
PDMR with session, without significance vs MF	0.9842	0.3250
PDMR with session, with significance vs MF	1.5309	0.1258

Table 7.3: Test for the Difference of Ranking Correlation with the Target Ranking for PDMR and MF

Table 7.3 lists the p-values of the tests comparing the correlation between the rankings produced by each of the variations and the target ranking as well as the corresponding z_B test values explained in Section 6.1. As can be seen from Tables 7.2 and 7.3, the ranking accuracy based on all the techniques is equivalent. That is, the correlation between the MF and the target ranking is not significantly higher than the correlation between any of the PDMR variations and the target ranking in terms of Kendall's τ_b.

We conjectured that PDMR is farther from the target ranking than MF in terms of Kendall's τ_b because of the explained information loss. However, we were not able to prove that which means that the loss of information mentioned before is not extreme (the same result as in the non-personalized case). In spite of that, the slight difference of the ranking quality suggests that the real advantages of the models using pairwise preferences could only be noticed in a situation where the pairwise preferences are collected directly and not the ratings transformed to the pairwise preferences.

In order to conduct the experiments that are not biased by this loss of information, we implemented two online recommender system. One of the systems uses the PDMRPP model which was described in Section 6.2 where the real pairwise preferences are collected. The other uses the MF model described in Section 2.1. The results of the experiment using these models are presented in Chapter 8.

Chapter 8

Online Experiment Results

In this chapter we present the results of the online experiments that we have conducted with the aim of testing the hypotheses presented in Chapter 4. In Section 8.1 we describe the outcome of the evaluation of the pairwise preference elicitation method. In Section 8.2 we describe and analyze the results of PDMRPP and MF recommendation accuracy tests. In Section 8.3 the improvement of recommendation accuracy in both algorithms (MF and PDMRPP) is analyzed. In Section 8.4 the perceived recommendation quality in two scenarios - when the system is based on PDMRPP and MF model - is compared and analyzed.

Data Collected

We recall that the online experiment consists of two stages. The users are invited to take part in it by email. In the first stage they are asked to provide preferences about the movies. In the second stage they get recommendations for unseen movies and are asked to provide feedback about them. The detailed explanation of the experiment is given in Section 6.2.

The users were collected for the experiment mostly by using social media channels and e-mail address lists. Many of them are aged around 25 to 35. High percentage of them are either undergraduate, graduate, PhD students, recent graduates or university staff. We think that the sample is quite representative of a real population of recommender systems' users in general. Even though, there is a slight bias towards the users with a technical background among the users of our experiment, we do not think it should distort the experiment results.

There were around 97 users registered to the experiment. However, for each of the specific tests that we perform, we use a different number of users. This is because not all the users finished the whole experiment and for each of the tests we need the information gathered in a different stage of the experiment.

8.1 Users' Satisfaction for the Preference Elicitation Process: Ratings versus Pairwise Preferences

As mentioned in the description of the experiment (Section 6.2), after the users provide their preferences about the movies in a form of ratings or pairwise preferences, they answer the set of questions related to the preference elicitation process. The questions are evaluated using a five-level Likert scale ranging from "Totally disagree" to "Totally agree".

We hypothesize that the preference elicitation process of pairwise preferences makes users more satisfied compared to providing ratings. We are interested in this because we think it can lead to the better recommendation accuracy of the system because more preference data could be collected, which we also test and the results of which will be presented later in this chapter in Section 8.2.

In this section we compare users' satisfaction in the preference elicitation process when they give ratings versus when they give pairwise preferences.

89 users reached the state of the experiment where they answer some questions about the preference elicitation process. The questions are asked after the users provide preferences about the movies at the end of the first stage of the experiment. Thus, it does not involve the recommendation accuracy. 44 users were assigned to system R (i.e., they had to rate the movies) and 45 were assigned to system P (i.e., they had to provide pairwise preferences between the movies).

The statements that are used to evaluate the preference elicitation process are listed below:

1. I have fun using the system;
2. Using the system is a pleasant experience;
3. The system makes me more aware of my choice options.
4. I feel bored when I am using the system.

In order to analyze the users' satisfaction in the preference elicitation process, we follow two ways:

1. Compare the systems in terms of a pooled score that reflects the overall users' satisfaction in the preference elicitation process;
2. Compare the systems in terms of user responses to separate questions that reflect different aspects of preference elicitation process.

Users' Satisfaction for the Preference Elicitation Process in Terms of a Pooled Score

In order to evaluate the overall users' satisfaction in preference elicitation process we use a pooled score that combines the responses to the questions listed above. We borrow the idea of a pooled score from Brooke [1996]. He introduces a pooled score to measure the system usability.

The System Usability Scale (SUS) is a set of 10-question Likert scale representing the pooled evaluation of system usability. According to Brooke [1996], SUS yields a single number representing a composite measure of the overall usability of the system being studied. In order to calculate the SUS Score, one has to sum the score contributions from each item. Each item's score contribution will range from 0 to 4. For sentences where larger agreement means a higher perceived satisfaction the score contribution is the scale position minus 1. For sentences where larger agreement means a lower perceived satisfaction the contribution is 5 minus the scale position. The resulting score then has to be scaled to $[0, 100]$.

In order to make a pooled score that measures the users' satisfaction in preference elicitation process, we borrow the pooling idea that was used for SUS Score. We customize it for our case of four questions. Because the first three statements represent the positive aspect, we adjust them as the odd items in SUS, i.e. subtract 1 from the response. Similarly, because the fourth statement represents the negative aspect, we adjust it as the even items in SUS, i.e. $5-$ response. We also scale the pooled score to $[0, 100]$.

Table 8.1 shows the pooled scores for systems R and P and the two-sided t-test p-value.

Pooled Score for System R	Pooled Score for System P	t-test p-value
62.78409	66.66667	0.318

Table 8.1: Pooled Scores for Preference Elicitation Process and t-test p-value

The system that asks users to provide pairwise preferences is receiving a larger preference elicitation satisfaction score but this difference is not statistically significant, i.e., p-value of the two-sided t-test is equal 0.318.

Users' Satisfaction for the Preference Elicitation Process in Terms of Separate Statement Responses

Warachan [2012] researches the appropriate ways to analyze two independent groups of Likert scale data in the Ph.D. of Mathematics and Statistics dissertation. He states that for the Likert scale data, if the sample size is small or midsize under any distribution shapes, the Mann-Whitney test will be the preferred procedure to be used because it will be robust and has high statistical power. The t-test is suitable to be used with large sample size ($n > 100$) under the uniform, moderate skewed or symmetric distribution. The Kolmogorov-Smirnov test is not recommended for the ordinal 5-point and 7-point Likert scale data with lot of ties because the lack of robustness property Warachan [2012].

We perform the Mann-Whitney test (as it is suggested as the most suitable) and also Kruskal-Wallis test (for the sake of comparison) to test whether the population distributions of

Question \Test	Mann-Whitney	Kruskal-Wallis
I have fun using the system	0.8644	0.8609
Using the system is a pleasant experience	0.6221	0.619
The system makes me more aware of my choice options	0.0001231 ***	0.000121 ***
I feel bored when I am using the system	0.8145	0.8111

Table 8.2: Separate Statement Scores' t-test p-values for Preference Elicitation Process

the responses to the questions are identical in systems R and P without assuming them to follow the normal distribution. Table 8.2 reports the p-values values for both of the tests mentioned.

For statement 'The system makes me more aware of my choice options' the average response among the users interacting with system P is higher than the average response among the users interacting with system R. As can be seen from Table 8.2, for statement 'The system makes me more aware of my choice options' the difference between user responses is significant, i.e., the interface with pairwise preferences performs significantly better than the interface with ratings in making people more aware of their choice options. This is a natural and expected conclusion as it encourages one to compare the items and thus to discern their own tastes. Non of the other questions show significant differences in preference elicitation process between the two user interfaces.

8.2 Recommendation Prediction Accuracy: Matrix Factorization (MF) versus Personalized Differential Matrix with Ratings and Pairwise Preferences (PDM-RPP)

In the second stage of the experiment, when the users get their recommendation lists, they are asked to mark the good recommendations. It enables us to compare the accuracy of the recommendation processes using MF versus PDMRPP. We conjecture that the system which exploits pairwise preferences should better rank the recommended items.

The normalized Discounted Cumulative Gain ($nDCG$) Manning et al. [2008] is a popular measure of the ranking accuracy of the recommendations produced by a system widely used in information retrieval (IR). The score accumulates the gain from the top of the recommendation list to the bottom with the gain of each recommended item discounted at lower ranks. Thus, it is suitable for testing our hypothesis.

Besides $nDCG$, we also use precision - a standard IR metric that measures the recommendation accuracy by calculating the proportion of the relevant items recommended among all the items recommended.

For this experiment we have 69 (34 assigned to system R and 35 assigned to system P) users that have finished both stages of the experiment, i.e. they entered their preferences (first stage) and they evaluated the recommendations (second stage). We could rely on less users for this analysis since some of the users registered have not finished the whole experiment, i.e., did not reach the state where they had to evaluate the recommendations. We use the user selection of good recommendations in order to compute *nDCG* and precision.

Table 8.3 shows *nDCG* and precision of the two systems and the statistical test for significance of the differences.

	MF	PDMRPP	p-value
nDCG	0.8054144	0.7873218	0.7433
precision	0.4814815	0.4689655	0.8196

Table 8.3: System R (MF) versus System P (PDMRPP) in Terms of *nDCG* and Precision Using All Users That Finished Both Stages

As can be seen from the average *nDCG* and precision values in Table 8.3, there is no significant difference between the recommendation accuracy of system R (using MF algorithm) system P (using PDMRPP) when the experiment uses the data of all the users that have finished both stages.

However, as was mentioned before, in the online experiment we use a mixture of the preference data. The system R uses a combination of MovieLens ratings and the ratings provided by the users attached to group R during the experiment. The proportion of non-MovieLens ratings is very big in the whole set of ratings used by the system R. Namely, $36,539$ Movielens and 1415 non-Movielens ratings, i.e. only 3.9% of all the ratings are non-Movielens. The system P uses a combination of MovieLens ratings and the pairwise preferences provided by the users attached to group P. Similarly, the proportion of the pairwise preferences is very small in the whole set of user preferences (MovieLens ratings plus our users' pairwise preferences). Namely, $36,539$ Movielens ratings and $2,262$ pairwise preferences. Therefore, the loss of information introduced by the transformation of the ratings into pairwise preferences in the PDMRPP technique still exists.

Moreover, it is very hard to separate the problem of selecting the appropriate movies that a user is asked to compare from the advantage of the higher satisfaction in providing pairwise preferences than ratings. As was shown in the previous section, the system where the users provide pairwise preferences makes users more satisfied than the one in which users provide ratings. Therefore, one could expect that more pairwise preferences collected could improve the system's recommendation accuracy. Yet, as can be seen from the results, this was not the case. And the reason is very likely the problematic movie pair selection.

Another reason why there was no significant improvement in this experiment is that the data collected may have been noisy in a way that some of the users could have skimmed through the tasks fast without really thinking about the feedback that they are giving. And this is a very important aspect to consider.

One way to remove some part of noisiness in the data collected is to filter out the users that have skipped some of the tasks in the experiment. This way we hope to filter out the users who are either not interested or not enough knowledgeable of the domain. We perform the same experiment with only the users that have provided additional preferences after they got the initial recommendation list expecting that this subset of users better represents the real population. We have 30 users that have provided additional ratings or preferences (16 using system R and 14 using system P).

	nDCG			precision		
	MF	PDMRPP	p-value	MF	PDMRPP	p-value
before	0.5374682	0.7927419	0.02443 *	0.4500000	0.5076923	0.2765
after	0.6620382	0.8401720	0.04609 *	0.4923077	0.6428571	0.04389 *

Table 8.4: System R (MF) versus System P (PDMRPP) in Terms of *nDCG* and Precision Using Only Users That Provided Additional Preferences

Table 8.4 shows *nDCG* and precision of the two systems before and after the additional preferences and the statistical tests for significance of the differences between two systems. The results confirm our assumption. Before the additional preferences, i.e. when comparing the initial recommendation lists, the PDMRPP performs significantly better than the MF in terms of *nDCG*. After the additional preferences, i.e. when comparing the improved recommendation lists, the PDMRPP performs significantly better than the MF in terms of both *nDCG* and precision. Thus, we prove our most important research hypothesis that the pairwise preference incorporation in the standard CF algorithms improves the recommendation accuracy.

8.3 Improvement of the Recommendation Prediction Accuracy after Additional Preferences of the Target User

In the second stage of the experiment, first the recommendation list that the users get is produced using one half of the movies. Then after the users provide additional preferences, the recommendation list is produced using another half of the movies. The movies from the list of 100 movies selection (explained in Section 5) are split into two equal parts. We split them because we intend to compare the recommendation list which is produced by the model using only the preferences given in the first stage with the list computed by the system when the additional preferences are gathered.

The reason why we split the movies to make recommendation lists is that it is very likely that only the seen movies from the initial recommendation list will be replaced by the next best ranked movies. The significant changes in an initial recommendation list are unlikely because (1) it is very short (5 movies) and (2) the number of additional preferences is small (the users can only provide preferences about the movies from the initial list that they have seen, i.e. ≤ 5 ratings or ≤ 10 pairwise preferences).

In this experiment we conjecture that the recommendation list using also the additional preferences collected after showing the first recommendation list for the user is more accurate than the recommendation list using only the preferences collected in the first stage in terms of $nDCG$ and precision.

When the users get the first recommended list of movies, besides marking the good recommendations they also mark the movies that they have seen. Next, they provide additional ratings or pairwise preferences for them. Then the users get a second recommendation list, which is produced using the other half of the movies and mark the good recommendations again. This approach enables us to test the improvement of the accuracy of the recommendation processes after additional preferences using both algorithms. We measure $nDCG$ again for each of the systems for the first versus second recommendation list to asses if any of these differences are statistically significant.

For this experiment we have to filter out all the users that have not checked any movies as seen in case of system R and that have checked 1 or less movies as seen in case of system P, i.e. they could not provide any additional preferences that could improve the recommendation quality in the next step. We have 30 users that have provided additional ratings or preferences (16 using system R and 14 using system P).

On average, 2.0625 additional ratings and 1.9286 additional pairwise preferences were provided by the users using system R and system P, respectively.

Table 8.5 shows $nDCG$ and precision of the two systems before and after the additional preferences and the statistical test for significance of the differences before and after the additional preferences for each system.

	nDCG			precision		
	avg before	avg after	p-value	avg before	avg after	p-value
MF	0.5374682	0.6620382	0.1803	0.4500000	0.4923077	0.3464
PDMRPP	0.7927419	0.8401720	0.2948	0.5076923	0.6428571	0.0348 *

Table 8.5: Improvement of Recommendation Accuracy ($H_1 : T_{before} < T_{after}$) After Additional Data $T \in \{nDCG; precision\}$

Note, that the $nDCG$ and precision averages of both systems before the additional preferences in Table 8.5 are different from the corresponding averages given in the previous experiment in Table 8.3. The reason is that in this experiment we only use the users that have provided additional preferences. In this experiment we calculate the averages of the $nDCG$ and precision measures only among these users in both cases: before and after the additional preferences. On the contrary, in the previous experiment, we calculate the averages of the measures among all the users that have reached the state where they marked the good recommendations in the first recommendation list. Therefore, the number of the users used for the experiments differs and thus differs the average values of the measures used for comparison. Clearly, the numbers are the same as in Table 8.4 where we perform the PDMRPP versus MF experiment with a subset of users that have provided additional preferences.

As can be seen from Table 8.5, there exists statistically significant improvement in the precision of recommendation system P (i.e. when PDMRPP algorithm is used) under 0.05 level of significance. We hypothesized that the recommendation accuracy improves in terms of both *nDCG* and precision. As can be seen, there exists the improvement of both measures: *nDCG* and precision, but the improvement of *nDCG* is not statistically significant.

However, the significant improvement of precision suggests that the non-significant improvement of *nDCG* might have occurred only because of a too small sample. Recall, that we had to choose only those users that have provided additional preferences. That is, in case of system P they had to have seen at least 2 movies out of the 5 recommended movies in order to be able to provide at least one pairwise preference. Clearly, quite a low percentage of users were able to provide additional pairwise preferences and thus the sample of users used for this experiment was very small, which could have been the reason of a too high p-value to prove the improvement.

8.4 Perceived Recommendation Quality: Matrix Factorization versus Personalized Differential Matrix with Ratings and Pairwise Preferences (PDMRPP)

After the second stage of the experiment is finished, the users are asked a set of statements that evaluate their perceived recommendation quality. The statements are listed below:

1. I liked the items recommended by the system;
2. The recommended items fitted my preference;
3. The recommended items were well-chosen;
4. The recommended items were relevant.
5. The system recommended too many bad items;
6. I didn't like any of the recommended items;
7. The items I selected were "the best among the worst".

Similarly as in Section 8.1 where the preference elicitation process was analyzed, in order to compare the users' perceived recommendation quality, we follow two ways:

1. Compare the systems in terms of a pooled score that reflects the overall users' perceived recommendation quality;

2. Compare the systems in terms of user responses to separate statements that reflect different aspects of perceived recommendation quality.

The same as for comparison of recommendation accuracy of MF and PDMRPP algorithms in Section 8.2, we have 69 (34 from group A and 35 from group B) users that have finished both stages of the experiment. Thus, 69 users have filled in the questionnaire at the end of the first stage of the experiment. We can use this data to compare their perceived recommendation quality.

Perceived Recommendation Quality in Terms of a Pooled Score

In order to evaluate the overall perceived recommendation quality we use a pooled score that combines the responses to the questions listed above. We borrow the idea of a pooled score from Brooke [1996]. He introduces a pooled score to measure the system usability.

We build a pooled score for our case of seven questions from the second stage questionnaire (the statements are listed above) similarly as we did in Section 8.1. Because the first four statements represent the positive aspect, we adjust them us the odd items in SUS, i.e. subtract 1 from the response. Similarly, because the last three statements represent the negative aspect, we adjust them as the even items in SUS, i.e. $5-$ response. We also scale the score to $[0, 100]$.

Table 8.6 shows the pooled scores for systems R and P and the two-sided t-test p-value.

Pooled Score for System R	Pooled Score for System P	t-test p-value
61.34454	63.26531	0.6625

Table 8.6: Pooled Scores for Perceived Recommendation Quality and t-test p-value

We conjectured that the system P should have the higher overall perceived recommendation quality than the system R. We formulated this conjecture assuming that the users that use system P are more inclined to provide preferences and therefore more preference data is gathered resulting in the better recommendation quality. In our experiment users provided on average 30 ratings and 49 pairwise preferences. Thus, it is a sensible assumption. As can be seen from Table 8.6, the system that asks users to provide pairwise preferences (system P) is indeed receiving a larger perceived recommendation quality score but, unfortunately, this difference is not statistically significant, i.e., p-value of the two-sided t-test is equal 0.6625.

The reason why the difference is not large may be the difficult choice of the appropriate movie pairs shown for the user to compare in the system P (which may decrease the recommendation quality). Also, because of the small fraction of the pairwise preferences (as compared to the ratings) used in the model PDMRPP, the data loss explained previously still exists and thus makes it hard for the PDMRPP model to be as accurate as the model using ratings directly (MF).

Perceived Recommendation Quality in Terms of Separate Statement Responses

In order to evaluate the perceived recommendation quality in terms of different aspects of the perceived recommendation quality we compare the systems in terms of the responses to the 7 statements listed previously.

Again the Mann-Whitney and Kruskal-Wallis tests are performed to test whether the systems R and P differ in terms of the responses to the statements without assuming them to follow the normal distribution. Table 8.7 reports the averages of the responses to the statements mentioned and the p-values values for both of the tests mentioned. The responses here are converted into numerical scale by setting 'Totally disagree' = 1, 'Disagree' = 2, 'Neutral' = 3, 'Agree' = 4 and 'Totally agree' = 5.

Question	System R avg	System P avg	Mann-Whitney p-value	Kruskal-Wallis p-value
I liked the items recommended by the system	3.588235	3.657143	0.5278	0.5235
The recommended items fitted my preference	3.470588	3.571429	0.5538	0.5493
The recommended items were well-chosen	3.352941	3.485714	0.3164	0.3134
The recommended items were relevant	3.323529	3.457143	0.4291	0.4253
The system recommended too many bad items	2.941176	2.657143	0.2896	0.2868
I didn't like any of the recommended items	1.882353	2.028571	0.5618	0.5575
The items I selected were "the best among the worst"	2.705882	2.8	0.6622	0.6576

Table 8.7: Response Averages of Separate Statement Responses for Perceived Recommendation Quality and t-test p-values

It is clear from the Table 8.7 that the system P outperforms the system R almost in all the statements related to the perceived recommendation quality. That is, the averages of the responses to all 4 positive statements are higher in system P than in system R. Similarly, the average of the responses to one of the three negative statements is lower in system P than in system R. Thus, there is a clear tendency favoring the system P in terms of perceived recommendation quality.

Even if the differences are not statistically significant, it is very likely that in a similar experiment that collects more users, the positive improvement of the perceived recommendation quality would be proven.

Chapter 9

Discussion and Conclusions

In this thesis we have analyzed how the exploitation of pairwise preferences in Collaborative Filtering algorithms can improve Recommender Systems.

The research was motivated by the intuition that pairwise preferences can be easier for a user to provide than numerical ratings. Because of that a user can be more inclined to provide more preferences in the form of items pairwise preferences than ratings. Thus, a system using pairwise preferences may collect more preference data and as a result provide better recommendation accuracy.

In recommender systems this approach to model user preferences has not received much interest. Therefore, in this thesis we have analyzed some techniques that use pairwise preferences in order to produce a global ranking of items. We incorporated personalization into these techniques and in this way adjusted them to serve the RSs needs. We call the derived models Personalized Differential Matrix (PDM).

Using MovieLens dataset consisting of 100,000 ratings we experimented with the derived technique that uses ratings (PDMR) comparing them with the Matrix Factorization model. We showed that the techniques that we based our research on have a ranking accuracy which is close to the MF ranking accuracy. Though, if the technique is used when transforming the rating data into pairwise preferences it suffers from data loss.

In order to test whether the PDM technique has a better recommendation accuracy when direct pairwise preferences are incorporated in the model than the MF technique using only rating data, we conducted an online experiment. In this experiment we implemented two recommender systems: one based on Personalized Differential Matrix with Ratings and Pairwise Preferences model (PDMRPP) and one based on MF model. The users taking part in this two stage experiment had to provide either ratings or pairwise preferences depending on the system they were using. In the second stage they had to provide feedback about the recommendations that were produced for them by either PDMRPP or MF technique.

We proved that the system in which users compare movies performs significantly better in making users more aware of their choice options. We also showed that such system has a slightly higher users' satisfaction in the preference elicitation process in general, though the difference was not statistically significant.

The perceived recommendation quality of the system using PDMRPP model was slightly lower in the system using MF model in terms of a pooled score consisting of a number of statements about the perceived recommendation quality. However, for the difference to be statistically significant, a higher number of users is needed.

We showed that the PDMRPP technique is close to the MF technique in terms of recommendation accuracy measured by $nDCG$ and precision when the whole set of users is used. Moreover and most importantly, we showed that PDMRPP outperforms MF in terms of both $nDCG$ and precision when a subset of more interested and/or knowledgeable users are used. However, we noticed that even if the system using pairwise preferences has an advantage of being more satisfiable and thus gathering more preference data, it also suffers from an obvious difficulty. The biggest problem of such a system is to acquire pairwise preferences wisely. That is, the problem of pairwise preference request generation is more complex than the problem of rating request generation. This is because a quadratic number of movie pairs as opposed to the movies makes the probability that a user is able to provide a pairwise preference much smaller than the probability that he is able to provide a rating.

Therefore, for the system using PDMRPP model to outperform the system using MF model in terms of recommendation accuracy, it's advantage of a more satisfiable preference elicitation process has to outweigh it's disadvantage of a more complex preference request generation. Therefore, it is quite a difficult task for a system that gathers only pairwise preferences from the beginning.

However, the techniques presented can be very useful in improving recommendation accuracy of recommender systems that use active learning strategies. As proven in the thesis, there is a significant improvement in recommendation accuracy after only a couple of additional pairwise preferences are provided in a system using PDMRPP model. Contrary, such an improvement is not as big after a couple of additional ratings are provided in a system using MF model. Therefore, based on the conclusions of our experiments we strongly believe that the RSs would highly benefit by incorporating the user interface features that ask users to compare some specific well selected items from time to time.

We suggest several branches of further research to improve the techniques presented in this thesis. First of all, a deeper analysis of the pairwise preference request generation is required. As mentioned, the problem of item pair selection has a bigger complexity because of a quadratic number of item pairs and a low probability of the pair being compared as opposed to the items to be rated. Therefore, a careful selection of item pairs is very important for a system to perform well.

Secondly, further work is needed in the development of a suitable user interface that makes use of the active learning strategies in RSs based on PDMRPP model with mixed preference data. More online experiments with real users are required to discover better user interface features, because this is a very important part of a RS which should not be separated from the analysis of the models. The reason is that even if a model performs better on some dataset in the offline experiments it does not mean it will also perform better in a real system because the user interface built to gather data for that model might be too complex or not as satisfiable as the one built for another model.

Another possible branch of research is a better use of session data in the PDMRPP techniques. In the thesis we showed how such data can be incorporated in the PDMRPP techniques. However, we only tested session data ideas on historical MovieLens data where we had to perform a data transformation from ratings to pairwise preferences. We believe that a more detailed analysis of session data ideas would add value to the RS using PDMRPP models. The online experiments should be performed to test the advantages of session data incorporation in the PDMRPP techniques using direct pairwise preferences.

In conclusion, recommender systems using pairwise preference data are still quite a new research area which is starting to gain more popularity. We believe that recommender systems could highly benefit from a deeper investigation of this topic. It could lead to more satisfiable, useful and intelligent recommender systems. There is a number of challenging real world problems related to the pairwise preferences in recommender systems and they require more investigation.

References

A. Agresti. *Analysis of Ordinal Categorical Data.* John Wiley & Sons, 2 edition, 2010. 42, 43

S. S. Anand and B. Mobasher. Intelligent techniques for web personalization. In *Lecture Notes in Artificial Intelligence*, pages 1–36. Springer, 2005. 8

R. M. Bell and Y. Koren. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *In Proc. of the 2007 Seventh IEEE International Conference on Data Mining*, pages 43–52, 2007. 6

D. Billsus and M. J. Pazzani. Learning collaborative information filters. In *In Proc. of the 15th International Conference on Machine Learning*, pages 46–54, 1998. 6

D. Billsus, C. A. Brunk, C. Evans, B. Gladish, and M. Pazzani. Adaptive interfaces for ubiquitous web access. *Communications of the ACM*, 45:34–38, 2002. 6

J.S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *in Proc. of the 14th Conference on Uncertainty in Artificial Intelligence*, 1998. 8

S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30:107–117, 1998. 12

J. Brooke. Sus-a quick and dirty usability scale. *Usability Evaluation in Industry*, pages 189–194, 1996. 58, 64

A. Brun, A. Hamad, O. Buffet, and A. Boyer. Towards preference relations in recommender systems. In *in Proc. of the ECML/PKDD Workshop on Preference Learning*, 2010. 2, 22, 27

M. T. Chua, R. E. Funderlicb, and R. J. Plemmons. Structured low rank approximation. *Linear Algebra and its Applications*, 366:157–172, 2003. 9

W. N. Colley. Colley's bias free college football ranking method: The colley matrix explained, 2002. URL `www.Colleyrankings.com`. 12

C. Desrosiers and G. Karypis. A comprehensive survey of neighborhood-based recommendation methods. In F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors, *Recommender Systems Handbook*, pages 107–144. Springer Science and Business Media, 2011. 7

M. Elahi, V. Repšys, and F. Ricci. Rating elicitation strategies for collaborative filtering. In *in Proc. of the International Conference on E-Commerce and Web Technologies*, pages 160–171, 2011. 21

M. Elahi, F. Ricci, and N. Rubens. Active learning strategies for rating elicitation in collaborative filtering: A system-wide perspective. *ACM Transactions on Intelligent Systems and Technology (ACM TIST)*, 5, 2013. 21

S. Funk. Netflix update: Try this at home, 2006. URL `http://sifter.org/simon/journal/20061211.html`. 52

J. Fürnkranz and E. Hüllermeier. Pairwise preference learning and ranking. In *in Proc. of the Fourteenth European Conference on Machine Learning*, pages 145–156, 2003. 12

D. F. Gleich and L. Lim. Rank aggregation via nuclear norm minimization. In *in Proc. of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 60–68, 2011. 18

K. Goldberg, T. Roeder, D. Gupta, and C. Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4:133–151, 2001. 6

G. Golub and C. Reinsch. Singular value decomposition and least squares solutions. *Numerische Mathematik*, 14:403–420, 1970. 9

J. Herlocker, J. A. Konstan, and J. Riedl. An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. *Information Retrieval*, 5:287–310, 2002. 25

J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *in Proc. of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 230–237, 1999. 24, 25, 52

J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. Riedl. Evaluating collaborative filtering recommender systems. In *ACM Transactions on Information Systems,*, volume 22, pages 5–53, 2004. 2

W. Hill, L. Stead, M. Rosenstein, and G. Furnas. Recommending and evaluating choices in a virtual community of use. In *In Proc. of the SIGCHI Conference on Human Factors in Computing Systems*, pages 194–201, 1995. 6

E. Hüllermeier, J. Fürnkranz, Weiwei Cheng, and Klaus Brinker. Label ranking by learning pairwise preferences. *Artificial Intelligence*, 172:1897–1916, 2008. 12

J. P. Keener. The perron-frobenius theorem and the ranking of football teams. *SIAM Review*, 35:80–93, 1993. 12

M.G. Kendall. A new measure of rank correlation. *Biometrika*, 30:81–93, 1938. 41

M.G. Kendall. *Rank Correlation Methods*. Charles Griffin, 4 edition, 1970. 41

D. Kim and B. Yum. Collaborative filtering based on iterative principal component analysis. *Expert Systems with Applications*, 28:823–830, 2005. 10

J. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46: 604–632, 1999. 12

B. P. Knijnenburg, M. C. Willemsen, Z. Gantner, H. Soncu, and C. Newell. Explaining the user experience of recommender systems. *User Modeling and User-Adapted Interaction*, 22: 441–504, 2012. 48

J. A. Konstan, B. N. Miller, and D. Maltz J. L. Herlocker L. R. Gordon J. Riedl. Grouplens: Applying collaborative filtering to usenet news. *Communications of the ACM*, 40:77–87, 1997. 6

Y. Koren. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In *in Proc. 14th ACM SIGKDD Intl Conference Knowledge Discovery and Data Mining*, pages 426–434, 2008. 10

Y. Koren. Collaborative filtering with temporal dynamics. In *in Proc. of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 447–456, 2009. 2, 12

Y. Koren and R. Bell. Advances in collaborative filtering. In F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors, *Recommender Systems Handbook*, pages 145–186. Springer Science and Business Media, 2011. 2, 6, 7, 8, 9, 11, 12

Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *IEEE Computer*, 42:30–37, 2009. 9, 10, 11

A. N. Langville and C. D. Meyer. *Who's #1?: The Science of Rating and Ranking*. Princeton University Press, 2012. 2, 4, 12, 13, 14, 15, 16, 17, 18, 23, 31, 33, 50

G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7:76–80, 2003. 6

P. Lops, M. de Gemmis, and G. Semeraro. Content-based recommender systems: State of the art and trends. In F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors, *Recommender Systems Handbook*, pages 73–105. Springer Science and Business Media, 2011. 6

C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008. 59

K. Massey. Statistical models applied to the rating of sports teams. Bachelors thesis, Bluefield College, 1997. 12

C. D. Meyer. *Matrix Analysis and Applied Linear Algebra*. Society for Industrial and Applied Mathematics, 2000. 15

B. N. Miller, I. Albert, Lam S. K., J. A. Konstan, and J. Riedl. Movielens unplugged: Experiences with an occasionally connected recommender system. In *In Proc. of the 8th International Conference on Intelligent User Interfaces*, pages 263–266, 2003. 6

B. Mobasher, H. Dai, T. Luo, and M. Nakagawa. Discovery and evaluation of aggregate usage profiles for web personalization. *Data Mining and Knowledge Discovery*, 6:61–82, 2002. 6

R. J. Mooney and L. Roy. Content-based book recommending using learning for text categorization. In *in Proc. of the Fifth ACM Conference on Digital Libraries*, 2000. 6

L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, 1999. URL `http://ilpubs.stanford.edu:8090/422/`. 12

A. Paterek. Improving regularized singular value decomposition for collaborative filtering. In *in Proc. of KDD Cup and Workshop*, pages 39–42, 2007. 10

A. M. Rashid, G. Karypis, and J. Riedl. Learning preferences of new users in recommender systems: An information theoretic approach. *SIGKDD Explorations*, 10:90–100, 2008. 21

A.M. Rashid, I. Alberta, D. Cosley, S.K. Lam, S.M. McNee, J.A. Konstan, and J. Riedl. Getting to know you: Learning new user preferences in recommender systems. In *in Proc. of the International Conference on Intelligent User Interfaces*, pages 127–134, 2002. 21, 36

P. Resnick, N. Iacovou, M. Sushak, P. Bergstrom, and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *in Proc. of the 1994 Computer Supported Collaborative Work Conference*, ACM, 1994. 8

F. Ricci and F. Del Missier. Supporting travel decision making through personalized recommendation. In Clare-Marie Karat, Jan Blom, and John Karat, editors, *Designing Personalized User Experiences for eCommerce*, pages 221–251. Kluwer Academic Publisher, 2004. 6

F. Ricci, L. Rokach, and B. Shapira. Introduction to recommender systems handbook. In F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors, *Recommender Systems Handbook*, pages 1–35. Springer Science and Business Media, 2011. 6

P. K. Saikia. *Linear Algebra*. Pearson Education India, 2009. 15

T. Salimans, U. Paquet, and T. Graepel. Collaborative learning of preference rankings. In *in Proc. of the Sixth ACM Conference on Recommender Systems*, pages 261–264, 2012. 18, 22

B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Analysis of recommendation algorithms for e-commerce. In *in Proc. of the 2nd ACM Conference on Electronic Commerce*, ACM, pages 158–167, 2000a. 8

B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Application of dimensionality reduction in recommender system - a case study. In *in Proc. of the WebKDD '00: Web Mining for E-Commerce Workshop*, ACM, 2000b. 8, 10

B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *in Proc. of the 10th International World Wide Web Conference*, 2001. 8

G. Shani and A. Gunawardana. Evaluating recommendation systems. In F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors, *Recommender Systems Handbook*, pages 257–297. Springer Science and Business Media, 2011. 2

U. Shardanand and P. Maes. Social information filtering: Algorithms for automating "word of mouth". In *In Proc. of the SIGCHI Conference on Human factors in Computing Systems*, pages 210–217, 1995. 6, 8

G. Takács, I. Pilászy, and B. Németh. Major components of the gravity recommendation system. *SIGKDD Explorations*, 9:80–84, 2007. 10

L. Terveen, W. Hill, B. Amento, D. McDonald, and J. Creter. Phoaks: A system for sharing recommendations. *Communications of the ACM*, 40:59–62, 1997. 6

S. Wang, J. Sun, B. J. Gao, and J. Ma. Adapting vector space model to ranking-based collaborative filtering. In *21st ACM International Conference on Information and Knowledge Management (CIKM '12)*, 2012. 1, 22, 27

B. Warachan. *Appropriate statistical analysis for two independent groups of likert-type data*. PhD thesis, American University, 2012. 58