Evaluation of Recommendations: Rating-Prediction and Ranking

Harald Steck*
Netflix Inc.
Los Gatos, California
hsteck@netflix.com

ABSTRACT

The literature on recommender systems distinguishes typically between two broad categories of measuring recommendation accuracy: *rating prediction*, often quantified in terms of the root mean square error (RMSE), and *ranking*, measured in terms of metrics like precision and recall, among others. In this paper, we examine both approaches in detail, and find that the dominating difference lies *instead* in the training and test data considered: rating prediction is concerned with *only the observed ratings*, while ranking typically accounts for *all items in the collection*, whether the user has rated them or not. Furthermore, we show that predicting observed ratings, while popular in the literature, only solves a (small) part of the rating prediction task for *any* item in the collection, which is a common real-world problem. The reasons are selection bias in the data, combined with data sparsity. We show that the latter rating-prediction task involves the prediction task *'Who rated What'* as a sub-problem, which can be cast as a classification or ranking problem. This suggests that solving the ranking problem is not only valuable by itself, but also for predicting the rating value of *any* item.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*Data Mining*

Keywords

Recommender Systems; Selection Bias; Rating Prediction; Ranking

1. INTRODUCTION

The idea of recommender systems (RS) is to automatically suggest items to each user that s/he may find appealing, e.g., see [2] for an overview. Prior to a user study or deployment, offline testing on historical data provides a time and cost efficient initial assessment of an RS. A meaningful offline test ideally provides a good approximation to the utility function to be optimized by the deployed system (e.g., user satisfaction, increase in sales). Recommendation accuracy is an important part of such an offline metric.

The literature on recommender systems typically distinguishes between two ways of measuring recommendation accuracy: *rating prediction*, often measured in terms of the root mean square error (RMSE); and *ranking*, which is measured in terms of metrics like precision and recall, among others. In this paper, we examine both kinds of accuracy measures in detail and identify different variants.

Concerning rating prediction, the literature has focused mainly on predicting the rating values for those items that a user has *deliberately chosen to rate*. This kind of data can be collected easily, and is hence readily available for offline training and testing of recommender systems. Moreover, the root mean square error (RMSE), the most popular accuracy metric in the recommender literature, can easily be evaluated on the user-item pairs that actually have a rating value in the data.

The objective of common real-world rating prediction tasks, however, is often different from this scenario: typically, the goal is to predict the rating value for *any* item in the collection, independent of the fact if a user rates it or not. The reason for the difference is that the ratings are missing not at random (MNAR) in the data sets typically collected [14, 13, 22, 23, 17]. For instance, on the Netflix web site, a personalized rating value is predicted for *any* video in the collection, as to provide the user with an indication of enjoyment if s/he watches it.

These two variants of *rating prediction* motivated us to examine it in more detail. In the first part of this paper, we identify *three* variants of rating prediction, and show how they differ from each other in terms of answers they provide to the recommendation problem and in terms of the degree of difficulty to solve them. Moreover, we show that many real-world rating prediction tasks require an additional sub-problem to be solved: as to which user deliberately chooses to assign a rating to which item in the collection (also known as *'Who rated What'* [1]). This sub-problem may be cast as a *classification* or *ranking* problem.

In the second part of this paper, we focus on ranking for solving real-world recommendation tasks. As just motivated, ranking may not only be an important sub-problem of many real-world rating-prediction tasks, but ranking is a

*Part of this work was done while at Bell-Labs, Alcatel-Lucent, Murray Hill, New Jersey.

RecSys'13, October 12–16, 2013, Hong Kong, China.
ACM 978-1-4503-2409-0/13/10.
http://dx.doi.org/10.1145/2507157.2507160.

relevant approach on its own–for instance, when the objective is to select a small subset of items from the entire collection (top N recommendations). We examine three variants of ranking: ranking of all items in the collection, ranking of only those items that the user has not rated yet, and ranking of only those items that the user has already rated. Our experiments provide strong empirical evidence that the difference in ranking accuracy due to these variants is considerably larger than the difference due to different ranking metrics. This suggests that the appropriate choice of training and testing *protocol*, e.g., which user-item pairs to consider, may be more important than the ranking *metric* for solving a given real-world ranking problem.

The main contributions of this paper can be summarized as follows:

1. we identify three variants of rating prediction. We show that the variant that is the main focus in the literature solves only a part of many real-world rating-prediction problems. This suggests further research is needed to develop rating prediction approaches for the (various) real-world tasks.

2. concerning ranking, we find that in practice the choice of an appropriate training and testing protocol makes a bigger difference than the choice of ranking metric.

3. when comparing rating prediction with ranking, we find that their *main* difference is *not* caused by the different metrics (e.g., RMSE vs. ranking metrics). Instead, it is due to the user-item pairs that are taken into account: only the subset of user-item pairs where a rating is *observed* in the data (e.g., for RMSE), vs. *all* user-item pairs–whether the user deliberately chose to assign a rating value to an item or not. As main causes, we identify selection bias in the data combined with data sparsity.

This paper is organized into two main parts: first, we examine rating prediction, which we model as a two-stage problem: the user's deliberate choice as to which item to rate, and which rating value to assign (Section 2.1). This leads immediately to three variants of rating prediction (Section 2.2), and we derive their general relationships and differences in Section 2.3. In Section 2.4, we discuss general implications on modeling the two-stage rating process, while Section 2.5 justifies a particular model approach. In the second part of this paper (Section 3), we outline three variants of ranking protocols (Section 3.1), which is analogous to the variants for rating prediction. In Section 3.2, we briefly review various ranking metrics, so that we are ready to compare the effects of ranking protocols and ranking metrics in our experiments in Section 3.3. We finish with our Conclusions.

2. RATING PREDICTION

The root mean square error (RMSE) is by far the most popular accuracy measure in the recommender literature. It is commonly associated with the objective of *rating prediction*, i.e., predicting the rating value that a user would assign to an item which s/he has not rated yet. In this section, we examine this objective in detail, and identify three variants with important differences.

2.1 Modeling the Decision to Rate

There has been recent work on the fact that ratings are *missing not at random (MNAR)* in the data typically collected in recommender system applications [14, 13, 22, 23, 17]. The main reason for this selection bias in the observed data is that the users are free to deliberately choose which items to rate.

This motivates us to model the rating-prediction task as shown in Figure 1 (left): In this graphical model, U denotes the random variable concerning users, and takes values u, where $u \in \{1, ..., n\}$ denotes user IDs, and n is the number of users. Similarly, I is the random variable concerning items, taking values $i \in \{1, ..., m\}$, where m is the number of items in the collection. The decision if a user u deliberately chooses to rate item i is represented by the random variable C, which takes values $c \in \{c_+, c_-\}$, where c_+ denotes that the user chooses to rate the item, while c_- means that the user does *not* deliberately choose to rate the item. Finally, R is the random variable regarding rating values, taking values r. For instance, $r \in \{1, ..., 5\}$ in case of ratings on a scale from 1 to 5 stars like in the Netflix Prize competition [3] or MovieLens data [5].

Note that C depends on the user and the item, such that this model is able to capture any reason that depends on u and i for choosing to give a rating (e.g., the item's popularity [23, 17]). This is more general than the approaches presented in [13], where this decision does not depend on the user-item pair directly, but only on the rating value itself. The dependence on the rating value as a decision-criteria whether to provide a rating is included as a special case in the model in Figure 1: using theory on graphical models, the graph in Figure 1 is *Markov-equivalent* to the graph where the orientation of the edge between C and R is reversed. Markov-equivalence means that both graphical models represent the same probability distribution (even though the graphs look different). The reason why the edge can be reversed is that both nodes C and R in the graph have the same parents, namely U and I. This is a key difference to the graphs used in [13], where C is not connected to the user or item.

Given that all random variables U, I, C, and R are discrete, we assume the most general probability distribution, the multinomial distribution. Note that this may be too general a model assumption as to learn a "good" model in practice, but in the first part of this paper, we are concerned with general insights that we can obtain without resorting to a more specific model.

2.2 Rating-Prediction Variants

Rating prediction is concerned with predicting the *rating value* r that a user u assigns to an item i. It can be cast as predicting the probability of each rating value r that user u assigns to each item i, denoted by $p(r_{u,i}) = p(r|u,i)$. Based on the graphical model in Figure 1, this can be decomposed as follows:

$$\underbrace{p(r|u,i)}_{(1)} = \underbrace{p(r|c_+,u,i)}_{(2)}\, p(c_+|u,i) + \underbrace{p(r|c_-,u,i)}_{(3)}\, p(c_-|u,i) \tag{1}$$

This equation shows that there are *three* variants of rating prediction, i.e., three probabilities concerning rating value r. Additionally, Eq. 1 also shows that these terms are linked via *two* terms regarding the probability that the user de-

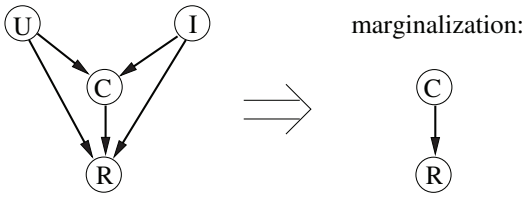

Figure 1: When a user (node U) provides a rating value (node R) for an item (node I), the user typically has the freedom to deliberately choose *which* item(s) to rate. The user's implicit decision as to whether rate an item or not, is represented by node C in the graphical model. The graph on the right hand side is obtained by marginalizing out U and I.

liberately chooses to rate an item (*'Who rated What'* [1]), which is discussed at the end of this sub-section. We first discuss the three variants of rating prediction:

1. $p(r|u,i)$: This is the probability that user u assigns rating value r to item i in the collection. It is important that there is no restriction on which item i it is; it can be *any* item in the collection. The ability to make an accurate rating prediction for any item i is extremely useful in practical applications, as it allows one to find the items with the highest (predicted) enjoyment value from the *entire collection*.

2. $p(r|c_+,u,i)$: This is the probability that user u assigns rating value r to item i under the condition that the user deliberately chooses to rate this item. This condition may not hold for any item i in the collection. In fact, due to data sparsity, it may apply to *only a (small) subset of items*, as shown in the next section. For instance, it is commonly observed in practice that users tend to assign ratings mainly to items they like or know [14, 13, 22, 23, 17]. The reason is that the observed data is collected by allowing users to choose which items to rate. As a consequence, this conditional probability is learned when optimizing RMSE or MAE on the *observed* data.

3. $p(r|c_-,u,i)$: This is the probability that user u would assign rating value r to item i under the condition that the user would not deliberately choose to rate this item. In practice, this rating value could be elicited, for instance, by providing the user with some reward for completing this task. These ratings are typically costly to elicit, and burdensome for the user to provide. For this reason, ratings under this condition are typically not available, which obviously makes it difficult to estimate this probability distribution accurately. Due to data sparsity, this distribution applies to the vast majority of user-item pairs, as discussed in more detail in the next section. In the rare case that they were collected, for instance in the Yahoo!Music data [13], it becomes obvious that this conditional distribution of rating values can be very different from the rating distribution under the condition that the user deliberately chose an item to rate. For instance, compare Figures 2(a) and (b) in [13], or Figure 1 in [17].

The decomposition in Eq. 1 also shows that, in addition, the two *conditional* probabilities $p(r|c_+,u,i)$ and $p(r|c_-,u,i)$ have to be estimated as to obtain valid rating predictions for *any* item in the collection. These probabilities capture whether user u deliberately chooses to rate item i or not. This shows their importance for rating prediction of *any* item, and motivates further research beyond the few works conducted so far, like the KDD Cup 2007, titled *'Who rated What'* [1]; there, one of the insights was that it was difficult to make accurate predictions as to which user deliberately chooses to rate which items.

2.3 Relationship of the three Variants

This section presents our main results concerning rating prediction, i.e., a general relationship between the three variants of rating prediction.

2.3.1 *Marginal Probabilities*

A necessary condition for accurate *personalized* rating prediction is that the *average* (and hence unpersonalized) rating predictions have to be accurate on average as well. This is outlined in detail in the following. To this end, let us consider $p(r)$, which is the probability of rating value r averaged over all users u and items i. Based on our graphical model in Figure 1, this can be derived by marginalizing over u and i in Eq. 1:

$$\begin{aligned} p(r) &= \sum_{c,u,i} p(r|c,u,i)\cdot p(c|u,i)\cdot p(u)\cdot p(i) && (2)\\ &= \sum_{c} p(r|c)\cdot p(c) && (3)\\ &= p(r|c_+)\cdot p(c_+) + p(r|c_-)\cdot p(c_-) && (4)\\ &= p(r|c_-) + p(c_+)\left\{p(r|c_+) - p(r|c_-)\right\} && (5)\\ &\approx p(r|c_-) && (6) \end{aligned}$$

The line-by-line comments are as follows: equality (2) in the first line follows from the definition of $p(r)$ in terms of the graphical model in Figure 1. In equality (3), the random variables U and I are marginalized out from the probability distribution, so that we obtain the marginal distribution over C and R, which is also represented in Figure 1 (right). This is valid [11] because the class of graphical models representing probability distributions from the exponential family, like the multinomial distribution used here, is closed when marginalizing out over a variable that is connected with all its edges to the same clique[1] in the graph, which is obviously the case in Figure 1. Equality (4) re-states the previous line, using $c \in \{c_+, c_-\}$, and equality (5) uses $p(c_-) = 1 - p(c_+)$. The interesting approximation in the last line is accurate due the *sparsity of the data*, which is typical for recommender applications: based on the graph, we have

$$p(c_+) = \sum_{u,i} p(c_+|u,i)\cdot p(u)\cdot p(i). \qquad (7)$$

Any accurate prediction for $p(c_+)$ has to be close to the empirical estimate $\hat{p}(c_+)$, which is given by

$$\hat{p}(c_+) = \frac{\#ratings}{|U|\cdot|I|} = \text{data sparsity}$$

[1]A clique is a completely connected sub-graph.

where $\#ratings$ is the number of ratings in the data; $|U| = n$ and $|I| = m$ denote the number of users and items, respectively. Hence, $p(c_+)$ equals the data sparsity. In typical applications the data sparsity is in the single-digit percent range, and often even one or more orders of magnitudes lower. For instance, it was about 1% in the Netflix Prize data [3], which may be considered a relatively dense data set compared to other recommender applications. Hence, the approximation in Eq. 6 is accurate up to a few percent in general. This is orders of magnitudes more accurate than any real-world rating-prediction accuracy, e.g., see RMSE for the Netflix Prize data [3] or MovieLens data [5]. We hence arrive at

Conclusion 1: *Due to data sparsity, the rating prediction variants 1. and 3., as outlined in Section 2.2, have to be closely related. In contrast, variant 2. is not required to be similar to variants 1. or 3.*

This suggests that there is a difference between the main focus of the literature (variant 2.), and many real-world rating-prediction problems (variants 1. and 3.).

2.3.2 Average Ratings

The *average* of the predicted rating values of a recommender system has an important impact on its accuracy in terms of RMSE (and similarly for MAE). This is outlined in the following.

Analogous to Eq. (2) - (6), one obtains for the average rating value:

$$\begin{aligned} \mathrm{E}[R] &= \sum_{r,c,u,i} r \cdot p(r|c,u,i) \cdot p(c|u,i) \cdot p(u) \cdot p(i) \quad (8) \\ &= \mathrm{E}[R|c_-] + p(c_+)\left\{\mathrm{E}[R|c_+] - \mathrm{E}[R|c_-]\right\} \quad (9) \\ &\approx \mathrm{E}[R|c_-] \quad (10) \end{aligned}$$

where $\mathrm{E}[\cdot]$ denotes the (conditional) expectation/average of the random variable R, i.e., the rating values. Not surprisingly, this confirms the previous relationships among the three variants of rating prediction: the average rating value of variants 1. and 3. has to be approximately the same due to data sparsity, while the average rating of variant 2. is not required to be similar.

In fact, there is strong empirical evidence, for instance, provided by the Yahoo! Music data[2] [13], suggesting that these average rating values can actually be very different: Figures 2 (a) and (b) in [13] show that, on a rating scale from 1 to 5, we have:

- $\mathrm{E}[R|c_-] \approx \mathrm{E}[R] \approx 1.8$: this is the average rating when users where asked to rate songs that were *randomly* selected by Yahoo! (instead of selected by the users). These rating data may hence be considered as (approximately) fulfilling the *missing at random* condition [18, 12], such that its rating distribution provides an unbiased estimate of the (unknown) true distribution.

- $\mathrm{E}[R|c_+] \approx 2.9$: this is the average rating when users were free to deliberately choose which items to rate. Evidently, this value is significantly larger than 1.8, suggesting a strong selection bias.

[2]The actual Yahoo! Music data set was not available to us, so that we had to resort to the histograms published in [13].

This difference in the average rating is extremely large– when compared to typical improvements in terms of RMSE, as outlined in the following. In the Netflix Prize data, the winning approach achieved $RMSE \approx 0.86$, which was so difficult to achieve that it required about three years of research work. In comparison, when simply predicting the average rating for all users and all items, one achieves $RMSE \approx 1.0$. This shows that an improvement in RMSE of 0.14 (on a rating scale of 1 to 5) may look small, but is actually very large. The numbers for MovieLens data are slightly different, but also here, the improvement of a sophisticated approach over a simple approach in terms of RMSE is much less than 1.

This shows that a difference in the average predicted rating of about 1 can easily dominate over the improvement of a more accurate recommender system. This becomes clear from the following thought experiment: Let us assume we train a recommender system on available rating data where the users were free to choose which items to rate. This is the typical scenario for collecting data. Let its RMSE (on a test set) be given by $RMSE_0$; its average predicted rating value will be close to the average rating value in the training data, $\mathrm{E}[R|c_+]$. This recommender system will hence be accurate for variant 2. Now, let us consider the common real-world task of rating-prediction for *any/each* item in the collection (variant 1): in this case, the (unknown) true average rating is $\mathrm{E}[R]$; now, the previously trained recommender system has a bias $b = \mathrm{E}[R|c_+] - \mathrm{E}[R]$. This results in a degraded RMSE: $RMSE_1 = \sqrt{RMSE_0^2 + b^2}$. Using the numerical values from above, this suggests that $RMSE_1 \gg 1$. Interestingly, this is worse than RMSE≈ 1, which can be achieved by an (unpersonalized) recommender system that predicts the average rating $\mathrm{E}[R]$. Even though the value $\mathrm{E}[R]$ may be unknown in many practical applications, it may also be possible to determine its value with some additional efforts, for instance, by running a truly randomized experiment with a small subset of the users. This leads us to the following conclusion:

Conclusion 2: *A recommender system with a low RMSE concerning the rating-prediction variant 2, is not guaranteed to achieve a low RMSE regarding rating-prediction tasks 1 or 3.*

Among these three variants, variant 1 refers to a common real-world rating-prediction task, like e.g., on the Netflix web site, which provides a personalized rating prediction for any video on its website. While many excellent solutions have been developed for variant 2 in the literature, this conclusion suggests that additional research is needed as to develop accurate prediction models for variant 1.

2.4 Implications for Modeling

Conclusions 1 and 2 above show that a key challenge in building real-world recommender systems is that rating distributions $p(R|U,I)$ (variant 1) or $p(R|c_-,U,I)$ (variant 3) are relevant for the user experience in many real-world applications; in contrast, the data that are readily available in large quantities follow the distribution $p(R|c_+,U,I)$ (i.e., variant 2). Given the practical importance of variants 1 and 3, it is crucial to build recommender systems that account for the items (and their ratings) that a user has *not* rated. Developing solutions for these new objectives of rating-prediction is an interesting area for future work.

In the following, we outline a Bayesian approach that uses an informative prior distribution that incorporates the rat-

ing distributions of the items that were *not* rated by a user. There are different ways of defining this prior distribution. First, one may run an experiment and elicit ratings for random items from users, like in the Yahoo! Music data. This provides a good estimate of the ratings concerning items that a user would not have rated otherwise. But it is also a costly experiment, and puts a burden onto users. Especially, if the items to rate are movies, it would be very time-consuming for users to watch random movies as they might not enjoy them.

Second, one may use a prior distribution with a small number of free parameters, which can be then tuned to achieve the desired result, e.g., by cross-validation. Such an approach was outlined in [22] for rating data, and a probabilistic version with prior distributions in [25]. The latter paper also shows that several rating values assigned to the same item i by a user u, i.e., a *distribution* of rating values $p(R|u,i)$, is equivalent to using the *average* rating $\bar{r}_{u,i} = \mathrm{E}[R|u,i] = \sum_r r \cdot p(r|u,i)$ when *optimizing* the least squares objective function.[3] This allows one to parameterize the prior distribution in terms of its mean. In [22], a single mean rating value is used for all users and items. It is an interesting result that, when this mean rating value is optimized as to achieve the best ranking[4] performance on the Netflix test set, a mean rating value of about 2 is found. This appears like a reasonable value for approximating the (unknown) mean rating value for the items that a user did not rate, as it agrees well with the results found for the Yahoo! Music data. Hence, this approach may provide a way for approximating relevant parameters of the (otherwise unknown) rating distribution concerning the items that a user did *not* rate. The approach in [22] is summarized in the following section; it will also be used in our experiments later in this paper.

2.5 Model and Training

In this section, we briefly review the low-rank matrix-factorization (MF) model named *AllRank* in [22], which was introduced as a point-wise ranking approach that accounts for all (unrated) items for each user. However, one can also view it from the perspective of rating prediction: the observed ratings in the data (i.e., $p(r|c_+,u,i)$), are complemented by imputed ratings with low values (as to approximate the unknown $p(r|c_-,u,i)$). As a result, an approximation to $p(r|u,i)$ is achieved, which applies to *any* item in the collection. For comparison, also the standard MF approach of minimizing RMSE on the *observed* ratings is discussed, and denoted by MF-RMSE.

The matrix of predicted ratings $\hat{R} \in \mathbb{R}^{m \times n}$, where m denotes the number of items, and n the number of users, is modeled as

$$\hat{R} = r_0 + PQ^\top, \tag{11}$$

with matrices $P \in \mathbb{R}^{m \times j_0}$ and $Q \in \mathbb{R}^{n \times j_0}$, where the rank $j_0 \ll m, n$; and $r_0 \in \mathbb{R}$ is a (global) offset.

[3]Note that this holds only for optimization/maximum-a-posteriori estimates, but does not apply for estimating the full posterior distribution, e.g., by means of Markov Chain Monte Carlo.

[4]Note that, in contrast to RMSE, (some) ranking measures can be applied to the entire collection of items, and hence account for both items with and without ratings assigned by the user.

For computationally efficient training, the square error (with the usual L2-norm regularization) is optimized:

$$\sum_{\text{all } u} \sum_{\text{all } i} W_{i,u} \cdot \left\{ \left(R^{\text{o\&i}}_{i,u} - \hat{R}_{i,u} \right)^2 + \lambda \left(\sum_{j=1}^{j_0} P^2_{i,j} + Q^2_{u,j} \right) \right\}, \tag{12}$$

where $\hat{R}_{i,u}$ denotes the ratings predicted by the model in Eq. 11; and $R^{\text{o\&i}}_{i,u}$ equals the actual rating value in the training data if observed for item i and user u; otherwise the value $R^{\text{o\&i}}_{i,u} = r_0 \in \mathbb{R}$ is imputed. The key is in the training weights,

$$W_{i,u} = \begin{cases} 1 & \text{if } R^{\text{obs}}_{i,u} \text{ observed} \\ w_0 & \text{otherwise} \end{cases} . \tag{13}$$

In *AllRank* [22], the weight assigned to the imputed ratings is *positive*, i.e., $w_0 > 0$ [22], and the imputed rating value is *lower* than the observed average rating in the data. This captures the fact that the (unknown) probability $p(r|c_-,u,i)$ is larger for lower rating values, compared to the observed probabilities $p(r|c_+,u,i)$. In contrast, *MF-RMSE* is obtained for $w_0 = 0$. This seemingly small difference has the important effect that AllRank is trained on a combination of both distributions, $p(r|c_-,u,i)$ and $p(r|c_+,u,i)$, geared towards approximating $p(r|u,i)$. In contrast, MF-RMSE is optimized towards $p(r|c_+,u,i)$. Due to this difference, in [22] AllRank was found to achieve a considerably larger top-N hit-rate or recall when ranking *all* items in the collection, compared to various state-of-the-art approaches optimized on the observed ratings only, see results in [22] and compare to [10, 4].

Alternating least squares can be used for computationally efficient training of AllRank [22] and MF-RMSE. We found the following values for the tuning parameters in Eq. 12 for the Netflix data (see also [22]) to yield the best results: $r_0 = 2$, $w_0 = 0.005$ and $\lambda = 0.04$ for AllRank; and $\lambda = 0.07$ for MF-RMSE (and $w_0 = 0$); we use rank $j_o = 50$ for both approaches.

While there are several state-of-the-art approaches, e.g., [6, 9, 10, 16, 19], that achieve a lower RMSE on *observed* ratings than MF-RMSE does, note that their test performances on *all* (unrated) items is quite similar to each other, e.g., see [10, 4]. This is interesting, as some of these approaches, like [16, 19, 10, 20], actually account in some sense for the MNAR nature of the data–but in the context of *observed* ratings (minimizing RMSE). Note that AllRank does not only apply to explicit feedback data, but can also be used for implicit feedback data, similar to [8, 15].

3. RANKING

In this section, we examine *ranking* as a means for assessing recommendation accuracy. Ranking is a useful approach when the recommendation task is, for each user, to pick a *small* number, say N, of items from among *all available* items in the collection. We divide the ranking task into two parts in this section: ranking protocols and ranking metrics, as outlined and compared to each other in the following.

3.1 Ranking Protocols

With ranking protocols, we refer to the fact as to which items are ranked. One may distinguish between two slightly

different variants: (a) all items are eligible for recommendation, including the items that have been rated by the user in the past (this assumes that a user may consume an item possibly several times, e.g., listen to a song on the on-line radio); (b) only all those items are eligible for recommendation that have not been rated by the user in the past (this assumes that a user consumes each item at most once, e.g., purchase of a movie DVD). This immediately suggests the corresponding ranking protocols:

- *all items:* for each user, all items are considered, whether rated or not by the user (in the training or test set).
- *rated test-items:* for each user, only those items are considered that have been rated by the user in the test set. Note that the ratings for those items follow the same distribution $p(r|c_+, u, i)$ as in the training data, as in both cases the user has deliberately chosen to rate these items.
- *all unrated items:* for each user, only those items are considered that have not yet been rated by the user in the training set. Note that this contains items with and without ratings in the test data.

Before we are ready to present experimental results, we briefly review various ranking metrics.

3.2 Ranking Metrics

There is a multitude of performance measures discussed in the RS literature, e.g., see [7, 21] for an overview. Here, we only have space to briefly review the ranking metrics used in our experiments in Table 1.

Information retrieval and ranking measures may be divided into ones that require *binary* feedback/rating values, and ones that can account for multiple values (e.g., ratings 1,...,5).

The binary measures can be applied naturally to binary feedback data (e.g., purchases, clicks), while explicit feedback data (e.g., rating values 1,...,5) have to be discretized into two values: *relevant* and not relevant to a user. For instance, in our experiments with Netflix data, we consider 5-star ratings as relevant to a user (i.e., the user definitely likes these movies),[5] while we consider all items with other or no rating values as not relevant.

Precision is defined as $\text{P@}N = m^{+,N}/N$, where $m^{+,N}$ is the number of relevant items among the top N items; N is chosen by the researcher. Obviously, if only a small fraction of all relevant items is observed, then P@N underestimates the 'true' precision on the (unknown) complete data. For this reason, the absolute value of P@N on observed data has little meaning, as the fraction of missing relevant items is typically unknown, and the 'true' precision on the complete data cannot be determined. However, P@N on available data may still be useful for relative comparisons, as to determine the best among competing models. Relevant items being missing are a main reason for the low precision values in Table 1.

While P@N captures only the number of relevant items in the top-N items, **Mean Average Precision** also accounts for the ranking within the top-N items. It is defined as $\text{MAP@}N = \frac{1}{m^+}\sum_i^N \frac{i}{\text{Rank}^{+,N}_{\text{sort},i}}$, where $\text{Rank}^{+,N}_{\text{sort},i}$ is the i^{th} element of the sorted ranks (in ascending order, i.e., the highest ranked item has the smallest rank value) of the relevant items in the top N (all relevant and not relevant items are ranked together). The total number of relevant items is denoted by m^+. Like precision, when computed from the observed relevant ratings only, it underestimates the 'true' MAP@N value on the unknown complete data. This follows immediately from its definition, and is also reflected by the small values in Table 1. We define MAP = MAP@m, where m is the number of *all* items. It is well known that MAP may also be viewed as the area under the precision recall curve.

Recall is defined as $\text{R@}k = m^{+,N}/m^+$, where $m^{+,N}$ is the number of relevant items among the top N items, and m^+ is the number of all relevant items. While many researchers may prefer precision over recall, recall has interesting properties in the context of missing ratings: if we assume that *relevant* ratings are missing at random (while allowing all other rating values to be missing not at random), then R@N can be estimated without bias from observed MNAR data [22]. This is very desirable, as its absolute value then has a meaning in the sense that it provides information on the expected performance of the recommender system regarding the unknown complete data (i.e., all items), which exactly is what is experienced by the user.

The **Area under the Recall curve (ATOP)** is related to recall via $\text{ATOP} = \frac{1}{m}\sum_{N=1}^m \text{R@}N$, as defined in [22]. Note that ATOP can also be expressed in terms of the average rank of the relevant items, and it is also a leading order approximation of the area under the ROC curve (AUC) if $m^+ \ll m$ [22].

An example of a *multiple-value ranking measure* is **normalized Discounted Cumulative Gain (nDCG)**. When some rating values are missing in the test data, however, it cannot be readily evaluated. One approach is to ignore missing ratings, and to use only observed ratings [24], like in the case of RMSE. The user-experience depends, however, on the ranking of *all* items, as outlined above. To this end, we amend nDCG as to account for missing ratings as well. We take the simple approach of imputing (the same) value Y_0 for each missing rating. Then nDCG is defined w.r.t. *all* items, and can be evaluated in the usual way, as outlined in the following. The permutation/ordering π of rating values Y_i on the entire list of items i in the test data results in the ordered list or ratings $Y_{\pi,i}$. In particular, we are interested in the permutation that sorts according to the predicted scores of the various items: $Y_{\text{sort},i}$ denotes the list or ratings in descending order of the predicted scores of the items (highest ranked item is first in list). Moreover, the best and worst possible permutations are also of interest. As the maximum value of DCG may vary across different data sets, the normalized version is typically defined as

$$\text{nDCG@}N = \frac{\text{DCG}_{\text{sort}}\text{@}N}{\max_\pi \text{DCG}_\pi\text{@}N},$$

where

$$\text{DCG}_\pi\text{@}N = \sum_{i=1}^N \frac{2^{Y_{\pi,i}} - 1}{\log(i+1)}.$$

Given that most ratings are missing and we impute Y_0, the absolute value of DCG is notably affected by the im-

[5]When we considered both 4 and 5 star ratings as relevant, we obtained qualitatively identical results/conclusions.

Table 1: Test results of two different RS approaches (MF-RMSE and AllRank) regarding various accuracy measures and three different item sets on the Netflix data. Obviously, AllRank considerably outperforms MF-RMSE when tested on all (unrated) items (tables a1 and a2), while the opposite holds true when tested on observed ratings (table b). The difference due to the various accuracy measures is small in comparison.

measure	(a) observed and missing ratings in test data				
		(a1) all items		(a2) all unrated items	
	worst case	MF-RMSE	AllRank	MF-RMSE	AllRank
RMSE	–	–	–	–	–
MAE	–	–	–	–	–
P@20	0.0000 ±0.0000	0.0205 ±0.0001	**0.0352** ±0.0001	0.0218 ±0.0001	**0.0382** ±0.0001
MAP	0.0000 ±0.0000	0.1136 ±0.0004	**0.2146** ±0.0006	0.1298 ±0.0007	**0.2810** ±0.0010
R@20	0.0000 ±0.0000	0.3680 ±0.0019	**0.6319** ±0.0017	0.3927 ±0.0022	**0.6864** ±0.0024
ATOP	0.0000 ±0.0000	0.8962 ±0.0007	**0.9596** ±0.0006	0.9018 ±0.0008	**0.9646** ±0.0007
nDCG @20	0.6405 ±0.0006	0.6856 ±0.0007	**0.7335** ±0.0006	0.6906 ±0.0006	**0.7518** ±0.0006
nDCG	0.9649 ±0.0001	0.9704 ±0.0001	**0.9745** ±0.0001	0.9709 ±0.0001	**0.9763** ±0.0001

measure	(b) only observed ratings in test data		
	worst case	MF-RMSE	AllRank
RMSE	–	**0.9224** ±0.0006	1.0731 ±0.0022
MAE	–	**0.7215** ±0.0006	0.8612 ±0.0016
P@2	0.0000 ±0.0000	**0.5989** ±0.0016	0.5935 ±0.0016
MAP	0.0000 ±0.0000	**0.7098** ±0.0007	0.7013 ±0.0006
R@2	0.0000 ±0.0000	**0.4412** ±0.0014	0.4358 ±0.0006
ATOP	0.0000 ±0.0000	**0.7412** ±0.0006	0.7305 ±0.0005
nDCG @2	0.1939 ±0.0003	0.8049 ±0.0009	0.8041 ±0.0009
nDCG	0.6742 ±0.0003	**0.9208** ±0.0003	0.9195 ±0.0003

puted value. In particular, the minimum value of DCG may be markedly larger than 0, see Table 1. While its absolute value does not provide information on the user experience, nDCG may still be useful for comparing different recommender systems with each other.

When a measure refers only to the top N items of the ranked list of items, this is denoted by '@N'; otherwise it refers to the entire list. While smaller values of RMSE and MAE indicate better accuracy, larger values of the (binary and multi-level) ranking measures are better. While all the above measures apply to each user, their average over all users is reported in our experiments, where each user is weighted by the number of relevant ratings (i.e., more active users have a larger weight). We also considered uniform weights among the users, and found that the obtained results for the accuracy measures were almost unchanged.

3.3 Experiments

This section outlines our results on the Netflix data [3]. Note that we obtained qualitatively identical experimental results/conclusions on the MovieLens data [5]. The Netflix Prize data [3] contains 17,770 movies and almost half a million users. About 100 million ratings are available. Ratings are observed for about 1% of all possible movie-user pairs. The ratings take integer values from 1 (worst) to 5 (best). The provided data are already split into a training and a probe set. We removed the probe set from the provided training set as to split these data into a training set and a disjoint test set.

Table 1 summarizes our comparison of the various offline tests for two different RS approaches: MF-RMSE and AllRank (see Section 2.5 for their definitions). Note that we used the usual split of the Netflix data for training and testing, except for evaluating the ranking measures in table (b): there we only considered users with at least 20 ratings, randomly assigned 10 ratings to the test set (so that we have a fair number of items to rank), and the remaining ones to the training set (on this data, RMSE=0.88 and 0.96 for MF-RMSE and AllRank, respectively). Hence, the top-2 items represent the top 20% of items in table (b), while in table (a) the top-20 items correspond to 2% (for each user, we used a random subset of 1,000 items without a rating, besides the rated items in the test set, as to allow for a direct comparison to the results in [10, 22, 4]). We further split the test set randomly into five sets of about equal size as to obtain an estimate of the standard deviation regarding each measure. We also report the worst values, which are obtained by ranking the items in the test data in the worst possible order, as this value can be quite large (especially when we impute a value for the missing ratings for nDCG in table (a)).

The results in Table 1 concerning each individual measure are discussed in Section 3.2, together with the measures' definitions.

Overall, the most striking observation in Table 1 is the clear difference between testing on (a) all (unrated) items vs. (b) only the observed ratings in the test set: AllRank is found best in (a), and MF-RMSE in (b). In comparison, the difference due to the various accuracy measures is very small. This dominating difference due to the ranking protocol may be explained by the fact that the ratings are missing not at random (MNAR) in the available data [14, 13, 22]. Note that an offline test on all (unrated) items intuitively reflects much better the situation in many real-world recommendation tasks (Table 1 (a1) and (a2)); and it evidently is very different from testing on observed ratings (Table 1(b))– whether using RMSE (like in, e.g., [6, 9, 10, 16, 19]) or a ranking metric (like in, e.g., [24]).

While RMSE and MAE can only be applied to observed ratings, other measures may be applied either to all items or to only the observed ratings. For instance, as shown in Table

1, the model with the highest recall on the *observed* ratings may not be the model with the highest recall on *all* items. This suggests that it is not sufficient to test a recommender system on various accuracy measures. Instead it is more important whether it is tested on all items, or on observed ratings only–i.e., the ranking protocol appears to be crucial.

As expected, the accuracy when testing on (a1) all items is lower than on (a2) all unrated items in Table 1, but only slightly due to data sparsity.

Conclusions

In the literature, the assessment of recommendation accuracy is typically divided into (1) *rating prediction* and (2) *ranking*. In this paper, we examined their details, and found that the crucial difference between these two methods lies *instead* in the distribution of the rating values considered: (a) rating-values of items that the user deliberately chooses to rate vs. (b) rating-values of all the items in the catalog (whether the user deliberately chooses to rate them or not). In the vast majority of the literature, (1) goes implicitly with (a), as RMSE naturally can only be measured with respect to observed ratings in the data; and (2) implicitly goes with (b), as there exist ranking metrics that can be applied to all items in the catalog.

Like for ranking, also for rating prediction, many *real-world* problems are concerned with *all* items in the catalog, i.e., *variant (b)*. As we have shown, the popular rating-prediction variant (1)(a) is very different from the common real-world scenario (1)(b). Interestingly, as to solve (1)(b), a classification or ranking problem has to be solved as well, namely the *'Who rated What'* problem. The common real-world rating-prediction problem (1)(b) is hence more involved to solve than the ranking problem. For this reason, practical applications may benefit from first solving the ranking problem, as top N recommendations by themselves are already very useful.

4. REFERENCES

[1] KDD Cup 2007. Task 1: Who rated what, 2007. `http://www.cs.uic.edu/ liub/KDD-cup-2007/proceedings.html`.

[2] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17:734–49, 2005.

[3] J. Bennet and S. Lanning. The Netflix Prize. In *Workshop at SIGKDD-07, ACM Conference on Knowledge Discovery and Data Mining*, 2007.

[4] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-N recommendation tasks. In *ACM Conference on Recommender Systems*, pages 39–46, 2010.

[5] MovieLens data. homepage: `http://www.grouplens.org/node/73`, 2006.

[6] S. Funk. Netflix update: Try this at home, 2006. `http://sifter.org/šimon/journal/20061211.html`.

[7] A. Gunawardana and G. Shani. A survey of accuracy evaluation metrics of recommendation tasks. *Journal of Machine Learning Research*, 10:2935–62, 2009.

[8] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *IEEE International Conference on Data Mining (ICDM)*, pages 263–72, 2008.

[9] R. Keshavan, A. Montanari, and S. Oh. Matrix completion from noisy entries. *Journal of Machine Learning Research*, 11:2057–78, 2010.

[10] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *ACM Conference on Knowledge Discovery and Data Mining*, pages 426–34, 2008.

[11] S. L. Lauritzen. *Graphical Models*. Oxford University Press, 1996.

[12] R. Little and D. B. Rubin. *Statistical Analysis with missing data*. Wiley, 1986.

[13] B. Marlin and R. Zemel. Collaborative prediction and ranking with non-random missing data. In *ACM Conference on Recommender Systems (RecSys)*, pages 5–12, 2009.

[14] B. Marlin, R. Zemel, S. Roweis, and M. Slaney. Collaborative filtering and the missing at random assumption. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 267–75, 2007.

[15] R. Pan, Y. Zhou, B. Cao, N. Liu, R. Lukose, M. Scholz, and Q. Yang. One-class collaborative filtering. In *IEEE International Conference on Data Mining (ICDM)*, pages 502–11, 2008.

[16] A. Paterek. Improving regularized singular value decomposition for collaborative filtering. In *KDDCup*, 2007.

[17] B. Pradel, N. Usunier, and P. Gallinari. Ranking with non-random missing ratings: influence of popularity and positivity on evaluation metrics. In *ACM Conference on Recommender Systems (RecSys)*, pages 147–54, 2012.

[18] D. B. Rubin. Inference and missing data. *Biometrika*, 63:581–92, 1976.

[19] R. Salakhutdinov, A. Mnih, and G. Hinton. Restricted Boltzmann machines for collaborative filtering. In *International Conference on Machine Learning (ICML)*, pages 791–8, 2007.

[20] R. Salakhutdinov and N. Srebro. Collaborative filtering in a non-uniform world: Learning with the weighted trace norm. In *Advances in Neural Information Processing Systems 24 (NIPS)*, 2010.

[21] G. Shani and A. Gunawardana. Evaluating recommendation systems. In *Recommender Systems Handbook*, pages 257–97. Springer, 2011.

[22] H. Steck. Training and testing of recommender systems on data missing not at random. In *ACM Conference on Knowledge Discovery and Data Mining*, pages 713–22, 2010.

[23] H. Steck. Item popularity and recommendation accuracy. In *ACM Conference on Recommender Systems (RecSys)*, pages 125–32, 2011.

[24] M. Weimer, A. Karatzoglou, Q. Le, and A. Smola. Cofi rank - maximum margin matrix factorization for collaborative ranking. In *Advances in Neural Information Processing Systems (NIPS)*, 2008.

[25] Y. Xin and H. Steck. Multi-value probabilistic matrix factorization for IP-TV recommendations. In *ACM Conference on Recommender Systems (RecSys)*, pages 221–8, 2011.