Estimating Probabilities in Recommendation Systems

Mingxuan Sun
Georgia Institute of Technology
cynthia@cc.gatech.edu

Guy Lebanon
Georgia Institute of Technology
lebanon@cc.gatech.edu

Paul Kidwell
Lawrence Livermore National Lab
kidwell1@llnl.gov

Abstract

Modeling ranked data is an essential component in a number of important applications including recommendation systems and web-search. In many cases, judges omit preference among unobserved items and between unobserved and observed items. This case of analyzing incomplete rankings is very important from a practical perspective and yet has not been fully studied due to considerable computational difficulties. We show how to avoid such computational difficulties and efficiently construct a non-parametric model for rankings with missing items. We demonstrate our approach and show how it applies in the context of collaborative filtering.

1 Introduction

Modeling ranked data is an essential component in a number of important applications including recommendation systems and web-search. In its simplest form, ranked data is a set of permutations $\pi : \{1,\ldots,n\} \to \{1,\ldots,n\}$ mapping abstract items to their rankings. In other words, $\pi(j)$ is the rank given to item j and $\pi^{-1}(j)$ is the j-most preferred item. The available sets of permutations $\pi_1,\ldots,\pi_m$ represent preferences issued by judges who are sampled i.i.d. from a population of judges. A judge in this case may be a human being, a computer program, or an abstract hypothesis. The modeling goal is thus to analyze i.i.d. permutations $\pi_1,\ldots,\pi_m \overset{\text{iid}}{\sim} p(\pi)$ and construct an estimator $\hat{p}_m(\pi\,;\pi_1,\ldots,\pi_m)$ for the unknown p.

Although many approaches for estimating p given $\pi_1,\ldots,\pi_m$ are possible, experimental evidence points to the fact that in cases of high n (recommendation systems for example), the distribution p does not follow a simple parametric form such as the Mallows, Bradley-Terry, or Thurstone models [12]. Instead, the distribution p tends to be diffuse and multimodal with different probability mass regions corresponding to different types of judges, e.g. in movie preferences probability modes may correspond to genre as fans of drama, action, comedy, etc. may have similar preferences. For example, Figure 1 demonstrates how parametric assumptions break down with increasing n in the case of voting and recommendation systems (see also [8]).

Several techniques for constructing a non-parametric $\hat{p}$ have been proposed. The simplest is perhaps the relative frequency estimator $\hat{p}_m(\pi\,;\pi_1,\ldots,\pi_m) = m^{-1}\sum_{i=1}^m I(\pi=\pi_i)$. A more accurate non-parametric estimator is based on kernel smoothing [10]

$$\hat{p}_m(\pi\,;\pi_1,\ldots,\pi_m) = m^{-1}\sum_{i=1}^m K_h(T(\pi,\pi_i)), \qquad (1)$$

where $T(\pi,\sigma)$ is a suitable distance between permutations such as Kendall's tau [2] and $K_h(r) = h^{-1}K(r/h)$ is a normalized unimodal density. The bandwidth parameter h represents the amount of smoothing [19] with small h corresponding to narrow kernels and large h corresponding to wide kernels.

In most cases involving large n, expecting that the available data are complete permutations is unrealistic. Typically, when a large number of items are available, different items are observed by different judges who omit preferences among the unobserved items and among observed and unobserved items. Extending the modeling framework (1) to such incomplete ranking data (with or without ties) is currently an open problem. The main difficulty is computational: both parametric and non-parametric models require summation over sets that increase factorially in the number of items n (see Sections 2-3).

Our main contributions in this paper are: (i) developing a computationally efficient scheme for extending (1) to incomplete rankings based on generating functions and combinatorics, (ii) applying it to three estimation problems in recommendation systems: probability estimation, rank prediction, and rule discovery.

Appearing in Proceedings of the 14^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2011, Fort Lauderdale, FL, USA. Volume 15 of JMLR: W&CP 15.

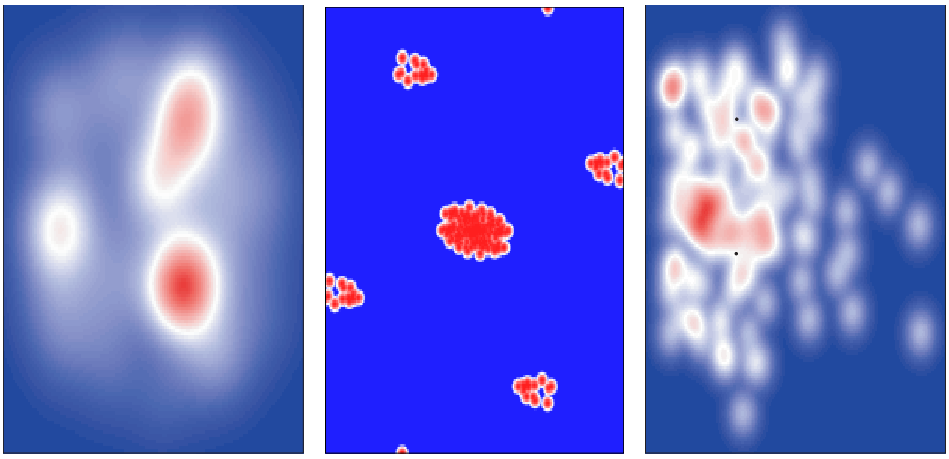


Figure 1: Heat map visualization of the density of ranked data using multidimensional scaling with expected Kendall's Tau distance. The datasets are APA voting (left, $n = 5$), Jester (middle, $n = 100$), and EachMovie (right, $n = 1628$) datasets. None of these cases show a simple parametric form, and the complexity of the density increases with the number of items n. This motivates non-parametric estimators for modeling preferences over a large number of items [8].

2 Incomplete and Tied Rankings

Since permutations play a central role in this paper, we describe the following notations and conventions, which are taken from [2] where more detail may be found. We denote a permutation by listing the items from most to least preferred separated by $\prec$ – $\pi^{-1}(1) \prec \pi^{-1}(2) \prec \cdots \prec \pi^{-1}(n)$, e.g. $\pi(1) = 2, \pi(2) = 3, \pi(3) = 1$ is $3 \prec 1 \prec 2$.

Ranking with ties occur when judges do not provide enough information to construct a total order. In particular, we define tied rankings as a partition of $\{1, \ldots, n\}$ to $k < n$ disjoint subsets $A_1, \ldots, A_k \subset \{1, \ldots, n\}$ such that all items in A_i are preferred to all items in A_{i+1} but no information is provided concerning the relative preference of the items among the sets A_i. We denote such rankings by separating the items in A_i and A_{i+1} with a $\prec$ notation e.g., $A_1 = \{3\}, A_2 = \{2\}, A_3 = \{1, 4\}$ (items 1 and 4 are tied for last place) is denoted as $3 \prec 2 \prec 1, 4$.

Ranking with missing items occur when judges omit certain items from their preference information altogether. For example assuming a set of items $\{1, \ldots, 4\}$, a judge may report a preference $3 \prec 2 \prec 4$, omitting altogether item 1 which the judge did not observe or experience. This is common in situations involving a large number of items n. In this case judges typically provide preference only for the $l \ll n$ items that they observed or experienced. For example, in movie recommendation we may have $n \sim 10^3$ and $l \sim 10^1$.

From a statistical perspective, the observation of incomplete rankings (with or without ties) represents an unknown permutation within the set of permutations that do not contradict the observed preference e.g.,

$$3 \prec 2 \prec 4 = \{1 \prec 3 \prec 2 \prec 4\} \cup \{3 \prec 1 \prec 2 \prec 4\} \cup \{3 \prec 2 \prec 1 \prec 4\} \cup \{3 \prec 2 \prec 4 \prec 1\}. \quad (2)$$

Barring any additional information on which permutations in S are more likely, we assume a uniform distribution resulting in

$$\hat{p}(\pi) = m^{-1} \sum_{i=1}^{m} |S_i|^{-1} \sum_{\sigma \in S_i} K_h(T(\pi, \sigma)),$$

$$\hat{p}(R) = m^{-1} \sum_{i=1}^{m} |S_i|^{-1} \sum_{\sigma \in S_i} \sum_{\pi \in R} K_h(T(\pi, \sigma)), \quad (3)$$

where $S_1, \ldots, S_m$ are the observed incomplete preferences. The main difficulty with the estimator above is the intractable computation of $\sum_{\sigma \in S} K_h(T(\pi, \sigma))$ or $\sum_{\pi \in R} \sum_{\sigma \in S} K_h(T(\pi, \sigma))$ for large n.

3 Efficient Triangular Smoothing with Missing Items

Previous work [10] developed computationally efficient schemes to compute (3) for the Mallows kernel

$$K_h(T(\pi, \sigma)) = \exp\left(-\frac{T(\pi, \sigma)}{h}\right) \prod_{j=1}^{n} \frac{1 - e^{-1/h}}{1 - e^{-j/h}} \quad (4)$$

$$T(\pi, \sigma) = \sum_{i=1}^{n-1} \sum_{l>i} I(\pi\sigma^{-1}(i) - \pi\sigma^{-1}(l)) \quad (5)$$

when the observed preferences are tied rankings. Unfortunately these simplifications do not carry over to the case of incomplete rankings that are often found in practice. In this paper we address this problem by deriving computationally efficient estimators for incomplete rankings. We do so for the non-parametric estimator (3) but instead of the Mallows kernel (4) we use a triangular kernel that is the direct analog of the Euclidean triangular kernel

$$K_h(T(\pi, \sigma)) = \frac{(1 - h^{-1}T(\pi, \sigma))I(h - T(\pi, \sigma))}{C(h)}. \quad (6)$$

Above, $I(x) = 1$ for $x \geq 0$ and 0 otherwise, T is Kendall's Tau defined in (5) and h represent both the support (the kernel is 0 for all larger distances) and the inverse slope of the triangle.

It is generally agreed in the non-parametric kernel smoothing literature that the precise shape of a unimodal kernel is not important (with the exception of histogram kernels). This is motivated by statistical theory which shows that continuous unimodal symmetric kernels achieve the same asymptotic rate. They do differ in the decay constant but only by a bit and the triangular kernel is favorable in that sense: the triangular kernel achieves 98.6% efficiency (100% is the optimal kernel) while Gaussian achieves 95.1% (worse!), tri-weight is 98.7% and bi-weight achieves 99.4% [19].

Combinatorial Generating Function

Generating functions, a tool from enumerative combinatorics, allow the kernel estimators to be readily calculated by concisely expressing the distribution of distances between permutations. Kendall's tau $T(\pi, \sigma)$ is the total number of discordant pairs or inversions between π, σ [17] and thus its computation becomes a combinatorial counting problem. We associate the following generating function with the symmetric group of permutations over n items

$$G_n(z) = \prod_{j=1}^{n-1} \sum_{k=0}^{j} z^k. \tag{7}$$

As shown for example in [17] the coefficient of z^k in $G_n(z)$ corresponds to the number of permutations σ for which $T(\sigma, \pi') = k$ (in this case π' is an arbitrary fixed permutation whose precise choice does has no influence). For example, the distribution of Kendall's tau $T(\cdot, \pi')$ over all permutations of 3 items is described by $G_3(z) = (1 + z)(1 + z + z^2) = 1z^0 + 2z^1 + 2z^2 + 1z^3$ i.e., there is one permutation σ with $T(\sigma, \pi') = 0$, two permutations σ with $T(\sigma, \pi') = 1$, two with $T(\sigma, \pi') = 2$ and one with $T(\sigma, \pi') = 3$. We use the standard notation of extracting the k coefficient of $G_n(z)$ as $[z^k]G(z)$ i.e., $[z^2]G_3(z) = 2$. Another important generating function is

$$H_n(z) = \frac{G_n(z)}{1 - z} = (1 + z + z^2 + z^3 + \cdots)G_n(z),$$

where $[z^k]H_n(z)$ represents the number of permutations σ for which $T(\sigma, \pi') \leq k$.

Proposition 1. *The normalization term $C(h)$ is given by $C(h) = [z^h]H_n(z) - h^{-1}[z^{h-1}]\frac{G'_n(z)}{1-z}$.*

Proof. The proof factors the non-normalized triangular kernel $CK_h(\pi, \sigma)$ to $I(h - T(\pi, \sigma))$ and $h^{-1}T(\pi, \sigma)I(h - T(\pi, \sigma))$ and making the following observations. First we note that summing the first factor over all permutations may be counted by $[z^h]H_n(z)$. The second observation is that $[z^{k-1}]G'_n(z)$ is the number of permutations σ for which $T(\sigma, \pi') = k$, multiplied by k. Since we want to sum over that quantity for all permutations whose distance is less than h we extract the $h - 1$ coefficient of the generating function $G'_n(z) \sum_{k \geq 0} z^k = G'_n(z)/(1 - z)$. We thus have

$$\begin{aligned} C = &\sum_{\sigma: T(\pi', \sigma) \leq h} 1 - h^{-1} \sum_{\sigma: T(\pi', \sigma) \leq h} T(\pi', \sigma) \\ = &[z^h]H_n(z) - h^{-1}[z^{h-1}]\frac{G'_n(z)}{1 - z}. \end{aligned}$$

□

Proposition 2. *Computing $C(h)$ has complexity $O(n^4)$.*

Proof. We describe a dynamic programming algorithm to compute the coefficients of G_n by recursively computing the coefficients of G_k from the coefficients of $G_{k-1}, k = 1, \ldots, n$. The generating function $G_k(z)$ has $k(k+1)/2$ non-zero coefficients and computing each of them (using the coefficients of G_{k-1}) takes $O(k)$. We thus have $O(k^3)$ to compute G_k from G_{k-1} which implies $O(n^4)$ to compute $G_k, n = 1, \ldots, n$. We conclude the proof by noting that once the coefficients of G_n are computed the coefficients of $H_n(z)$ and $G_n(z)/(1 - z)$ are computable in $O(n^2)$ as these are simply cumulative weighted sums of the coefficients of G_n. □

Note that computing $C(h)$ for one or many h values may be done offline prior to the arrival of the rankings and the need to compute the estimated probabilities.

Denoting by k the number of items ranked in either S or R or both, the computation of $\hat{p}(\pi)$ in (3) requires $O(k^2)$ online and $O(n^4)$ offline complexity if either non-zero smoothing is performed over the entire data i.e., $\max_{\pi \in R} \max_{i=1}^{n} \max_{\sigma \in S_i} T(\sigma, \pi) < h$ or alternatively, we use the modified triangular kernel $K^*_h(\pi, \sigma) \propto (1 - h^{-1})T(\pi, \sigma)$ which is allowed to take negative values for the most distant permutations (normalization still applies though).

Proposition 3. *For two sets of permutations S, R corresponding to tied-incomplete rankings, the expected distance $\overline{d} = \frac{1}{|S||R|} \sum_{\pi \in S} \sum_{\sigma \in R} T(\pi, \sigma)$ is*

$$\overline{d} = \frac{n(n-1)}{4} - \frac{1}{2} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} (1 - 2p_{ij}(S))(1 - 2p_{ij}(R))$$

$$p_{ij}(U) = \begin{cases} 1/2 & i \text{ and } j \text{ are tied in } U \\ I(\tau_U(j) - \tau_U(i)) & i \text{ and } j \text{ are not tied in } U \\ 1 - \frac{\tau_U(i) + \frac{\phi_U(i)-1}{2}}{k+1} & \text{only } i \text{ is ranked in } U \\ \frac{\tau_U(j) + \frac{\phi_U(j)-1}{2}}{k+1} & \text{only } j \text{ is ranked in } U \\ 1/2 & \text{otherwise} \end{cases}$$

with $\tau_U(i) = \min_{\pi \in U} \pi(i)$, and $\phi_U(i)$ being the number of items that are tied to i in U.

Proof. We note that $\overline{d}$ is an expectation with respect to the uniform measure. We thus start by computing the probability $p_{ij}(U)$ that i is preferred to j for $U = S$ and $U = R$ under the uniform measure. Five scenarios exist for each of $p_{ij}(U)$ corresponding to whether each of i and j are ranked by S and R. Starting with the case that i is not ranked j is ranked we note that i is equally likely to be preferred to any item preferred to and including j and that there are $k+1$ possible rankings for i. Given the uniform distribution over compatible rankings item j is equally likely to appear in positions $\tau_U(j), \ldots, \tau_U(j) + \phi_U(j) - 1$. Thus

$$\begin{aligned} p_{ij} &= \frac{1}{\phi_U(j)} \frac{\tau_U(j)}{k+1} + \cdots + \frac{1}{\phi_U(j)} \frac{\tau_U(j) + \phi_U(j) - 1}{k+1} \\ &= \frac{\tau_U(j) + \frac{\phi_U(j)-1}{2}}{k+1}. \end{aligned}$$

Similarly, if j is unknown and i is known, then $p_{ij} + p_{ji} = 1$. If both i and j are unknown either ordering must be equally likely given the uniform distribution making $p_{ij} = 1/2$. Finally, if both i and j are known $p_{ij} = 1, 1/2, 0$ depending on their preference. Given p_{ij}, linearity of expectation, and the independence between rankings, the change in the expected number of inversions relative to the uniform expectation $n(n-1)/4$ can be found by considering each pair

$$\begin{aligned} \mathrm{E}d(i,j) &= \frac{1}{2} P\left(i \text{ and } j \text{ disagree}\right) - \frac{1}{2} P\left(i \text{ and } j \text{ agree}\right) \\ &= \frac{1}{2}(p_{ij}(\sigma)(1 - p_{ij}(\pi)) + (1 - p_{ij}(\sigma))p_{ij}(\pi)) \\ &\quad -p_{ij}(\sigma)p_{ij}(\pi) - (1 - p_{ij}(\sigma))(1 - p_{ij}(\pi))) \\ &= \frac{-1}{2}\left(1 - 2p_{ij}(\sigma)\right)\left(1 - 2p_{ij}(\pi)\right) \end{aligned}$$

Summing over all pairs completes the proof. □

Corollary 1. Denoting the number of items ranked by either S or R or both as k, and assuming either $h > \max_{\pi \in R} \max_{i=1}^{n} \max_{\sigma \in S_i} T(\sigma, \pi)$ or that the modified triangular kernel $K_h^*(\pi, \sigma) \propto (1 - h^{-1})T(\pi, \sigma)$ is used, the complexity of computing $\hat{p}(R)$ in (3) is $O(k^2)$ online and $O(n^4)$ offline.

Proof. Computing (3) reduces to $O(n^4)$ offline computation of the normalization term and $O(k^2)$ online computation in Proposition 3. □

In many cases including all of the experiments in this paper it is not necessary to compute $C(h)$ using the generating function approach (however, see Table 1 for comptuational cost of computing $C(h)$). One reason is that in many cases we care mostly about the ordering of probabilities of competing events rather than precise values and thus we can compute the probabilities up to a constant multiple. Another reason is that we are estimating probabilities of a set of L disjoint events and the normalization term may be computed by summing over the non-normalized probability estimates (this is feasible when L is not too large which happens in most practical cases involving recommendation systems). In these cases the complexity is $O(k^2)$ which is bounded from above by $O(n^2)$.

item n	1000	1500	2000	2500	3000
time (hr)	0.02	0.09	0.27	0.75	1.72

Table 1: Computational cost of $C(h)$ in (6) with respect to the number of items n with bandwidth $h = 0.5 * T_{max}(\pi, \sigma)$.

4 Three Applications

We examine the estimation framework on three movie recommendation datasets: Movielens1M (6040 users over 3952 movies), EachMovie and Netflix (10k users over 1k most rated movies).

Task 1: Estimating Probabilities. We consider here the task of estimating $\hat{p}(R)$ where R is a set of permutations corresponding to a tied incomplete ranking. Such estimates may be used to compute conditional estimates $\hat{P}(R|S)$ which are used to predict which augmentations R of S are highly probable. For example, given an observed preference $3 \prec 2 \prec 5$ we may want to compute $\hat{p}(8 \prec 3 \prec 2 \prec 5 | 3 \prec 2 \prec 5) = \hat{p}(8 \prec 3 \prec 2 \prec 5)/\hat{p}(3 \prec 2 \prec 5)$ to see whether item 8 should be recommended to the user. For simplicity we focus in this section on probabilities of simple events such as $i \prec j$ or $i \prec j \prec k$. The next section deals with more complex events.

In Figure 2 we plot the values of $\hat{p}(i \prec j)$ for i corresponding to three movies: Shrek (family), Catch Me If You Can (drama) and Napoleon Dynamite (comedy) (j varies over the remaining movies). Comparing the three stem plots we observe that Shrek is preferred to almost all other movies, Napoleon Dynamite is less preferred than most other movies, and Catch Me If You Can is preferred to some other movies but less preferred than others. Also interesting is the linear increase of the stem plots for Catch Me If You Can and Napoleon Dynamite and the non-linear increase of the stem plot for Shrek. This is likely a result of the fact that for very popular movies there are only a few comparable movies with the rest being very likely to be less preferred movies ($\hat{p}(i \prec j)$ close to 1). Examining the genre (colors in right panels of Figure 2) we see that family and science fiction are generally preferred to others movies while comedy and romance generally receive lower preferences. The drama, action genres are somewhere in the middle. This is in agreement with the three highest movies in terms of $r(i) = \sum_j \hat{p}(i \prec j)$: Lord of the Rings: The Return of the King, Finding Nemo, and Lord of the Rings:

The Two Towers and the three lowest movies: Maid in Manhattan, Anger Management, and The Royal Tenenbaums.

In a second experiment (see Figure 3) we compare the predictive behavior of the kernel smoothing estimator with that of a parametric model (Mallows model) and the empirical measure (frequency of event occurring in the m samples). We evaluate the performance of a probability estimator via its test set loglikelihood. Since the Mallows model is intractable for large n we use in this experiment $n = 3, 4, 5$.

We observe that the kernel estimator consistently achieves higher test set loglikelihood than the Mallows model and the empirical measure. The former is due to the breakdown of parametric assumptions as indicated by Figure 1 (note that this happens even for n as low as 3). The latter is due to the superior asymptotics of the kernel estimator over the empirical measure.

Task 2: Rank Prediction. Our task here is to predict the ranking of a new unseen item which is presented to the user. We follow the standard procedure in collaborative filtering: the set of users is partitioned to two sets, a training set and a testing set. For each of the test set users we further split the observed items into two sets: one set used for estimating preferences (together with the preferences of the training set users) and the second set to evaluate the performance of the prediction [14].

Given a loss function $L(i, j)$ which measures the loss of predicting rank i when true rank is j (rank here refers to the number of sets of equivalent items that are more or less preferred than the current item) we evaluate a prediction rule by the expected loss. We focus on three loss functions: $L_0(i, j) = 0$ if $i = j$ and 1 otherwise, $L_1(i, j) = |i-j|$ which reduces to the standard CF evaluation technique described in [14], and an asymmetric loss function (rows are estimated number of stars (0-5) and columns are actual number of stars (0-5)

$$L_e = \begin{pmatrix} 0 & 0 & 0 & 3 & 4 & 5 \\ 0 & 0 & 0 & 2 & 3 & 4 \\ 0 & 0 & 0 & 1 & 2 & 3 \\ 9 & 6 & 3 & 0 & 0 & 0 \\ 12 & 9 & 6 & 0 & 0 & 0 \\ 15 & 12 & 9 & 0 & 0 & 0 \end{pmatrix}. \tag{8}$$

In contrast to the L_0 and L_1 loss, L_e captures the fact that recommending bad movies as good movies is worse than recommending good movies as bad.

For example, consider a test user whose observed preference is $3 \prec 4, 5, 6 \prec 10, 11, 12 \prec 23 \prec 40, 50, 60 \prec 100, 101$. We may withhold the preferences of items $4, 11$ for evaluation purposes. The recommendation systems then predict a rank of 1 for item 4 and a rank of 4 for item 11. Since the true ranking of these items are 2 and 3 the L_1 loss is $|1 - 2| = 1$ and $|3 - 4| = 1$.

In our experiment, we use the kernel estimator $\hat{p}$ to predict ranks that minimize the posterior loss and thus adapts to customized loss functions such as L_e. This is an advantage of a probabilistic modeling approach over more ad-hoc rule based recommendation systems. Figure 4 compares the performance of our estimator to several standard baselines in the collaborative filtering literature: two older memory based methods vector similarity (sim1), correlation (sim2) e.g., [1], and a recent state-of-the-art non-negative matrix (NMF) factorization (gnmf) [9]. The kernel smoothing estimate performed similarly to the NMF approach. We note that kernel smoothing is essentially a memory based method whose functional form is similar to the two other memory based methods (correlation and vector similarity) that performed much worse.

Task 3: Rule Discovery. In the third task, we used the estimator $\hat{p}$ to detect noteworthy association rules of the type $i \prec j \Rightarrow k \prec l$ (if i is preferred to j then it is probably the case that k is preferred to l). Such association rules are important for both business analytics (devising marketing and manufacturing strategies) and recommendation system engineering. Specifically, we used $\hat{p}$ to select sets of four items i, j, k, l for which the mutual information $I(i \prec j\,; k \prec l)$ is maximized. We detected the precise shape of the rule (i.e., $i \prec j \Rightarrow k \prec l$ rather than $j \prec i \Rightarrow k \prec l$ by examining the summands in the mutual information expectation).

Figure 7 shows the top 10 rules that were discovered. These rules isolate viewer preferences for genres such as fantasy, romantic comedies, animation, and action (note however that genre information was not used in the rule discovery). To quantitatively evaluate the rule discovery process we judge a rule $i \prec j \Rightarrow k \prec l$ to be good if i, k are of the same genre and j, l are of the same genre. This evaluation appears in Figure 6 where it is contrasted with the same rule discovery process (maximizing mutual information) based on the empirical measure. The superior performance of $\hat{p}$ as compared to the empirical nature is evident in that the kernel estimator $\hat{p}$ consistently outperforms the empirical measure.

In another rule discovery experiment, we used $\hat{p}$ to detect association rules of the form i ranked highest $\Rightarrow$ j ranked second highest by selecting i, j that maximize the score $\frac{p(\pi(i)=1,\pi(j)=2)}{p(\pi(i)=1)p(\pi(j)=2)}$ between pairs of movies in the Netflix data. We similarly detected rules of the form i ranked highest $\Rightarrow$ j ranked lowest by maximizing the scores $\frac{p(\pi(i)=1,\pi(j)=\text{last})}{p(\pi(i)=1)p(\pi(j)=\text{last})}$ between pairs of movies. Figure 8 (top) shows the top 9 rules of 100 most rated movies which clearly capture movie preference of similar genre, e.g. romance, comedies, and action. Figure 8 (bottom) shows the top 9 rules

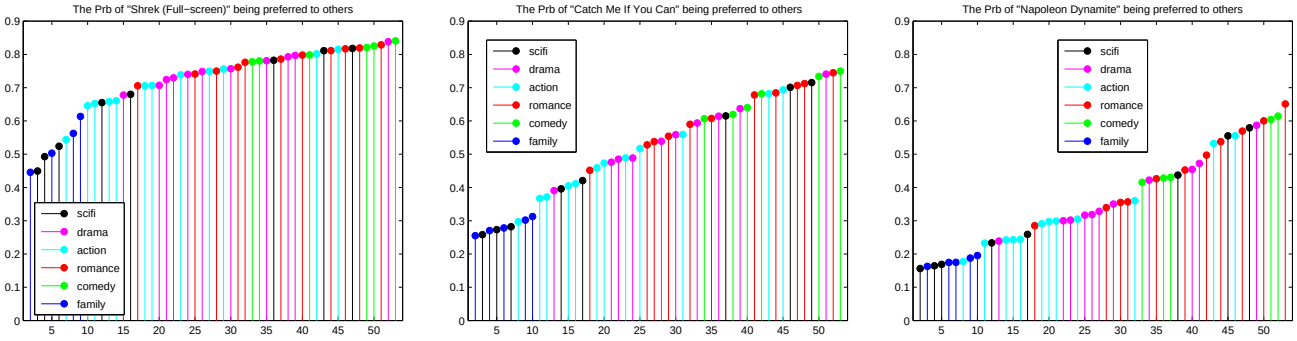


Figure 2: The value $\hat{p}(i \prec j)$ where i is fixed for three movies Shrek (left), Catch Me If You Can (middle) and Napoleon Dynamite (right) and j ranges over the most ranked 53 Netflix movies.

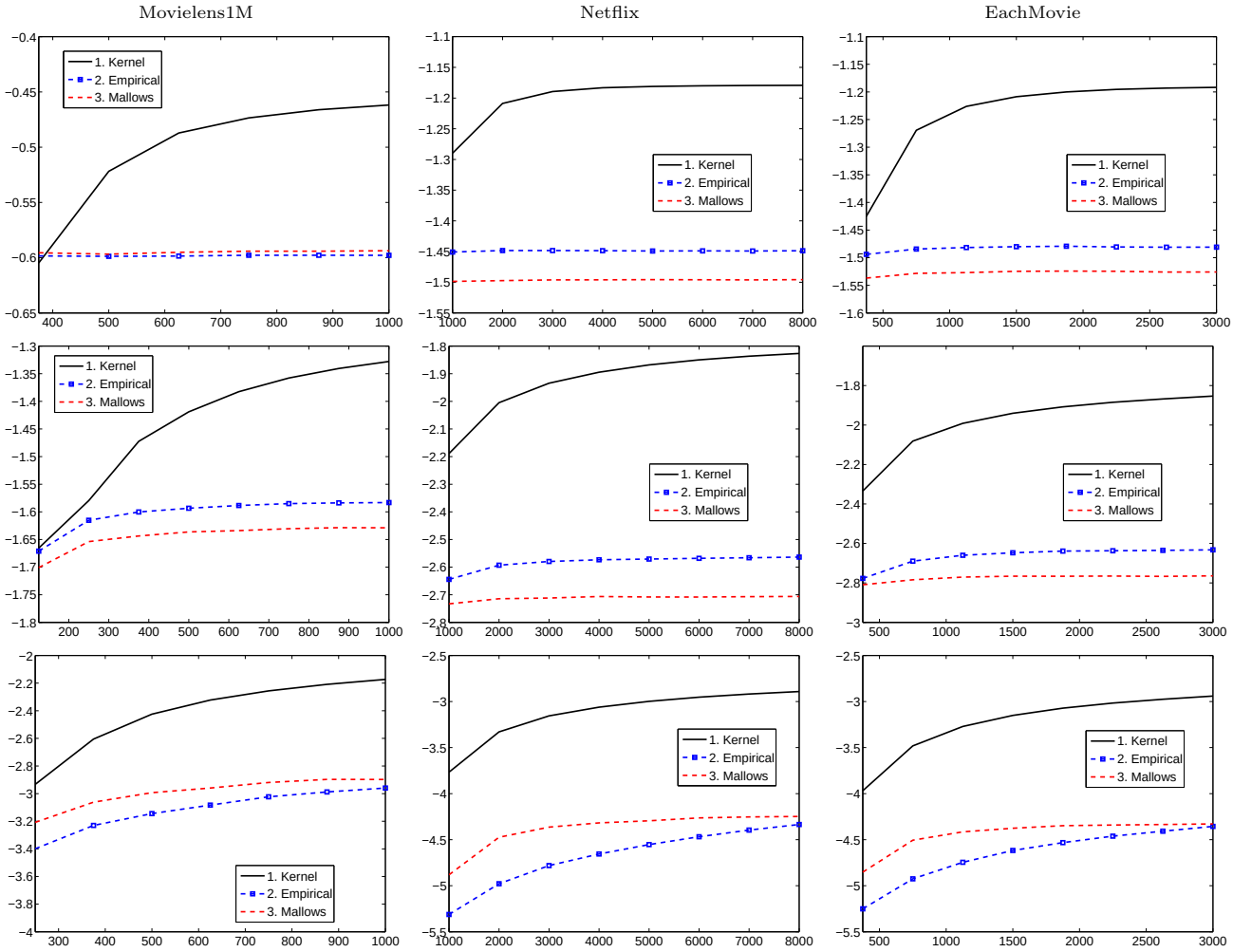


Figure 3: The test-set log-likelihood for kernel smoothing, Mallows model, and the empirical measure with respect to training size m for a small number of items $n = 3, 4, 5$ (top, middle, bottom rows) on three datasets. Both Mallows model (intractable for large n which is why $n \leq 5$ in the experiment) and the empirical measure perform worse than the kernel estimator $\hat{p}$.

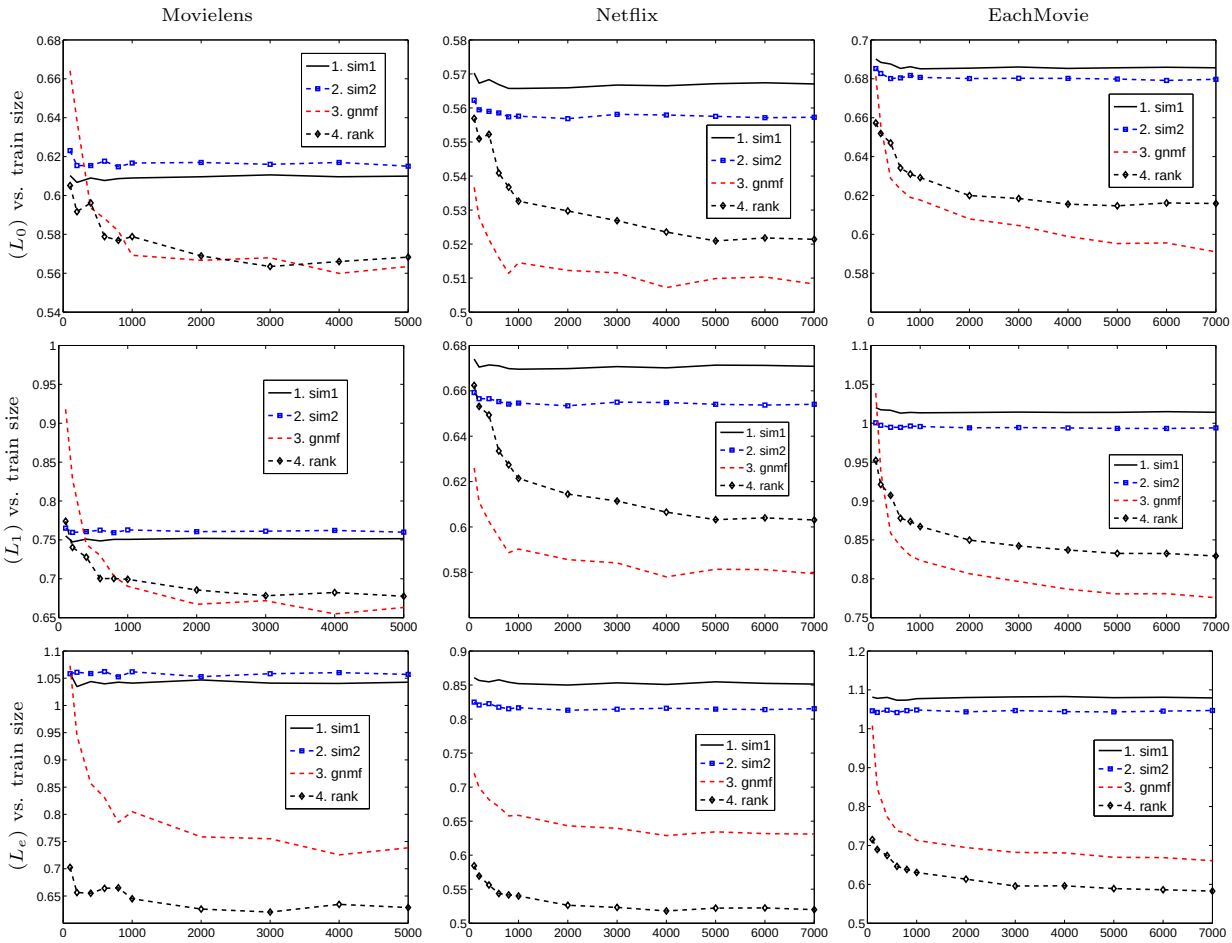
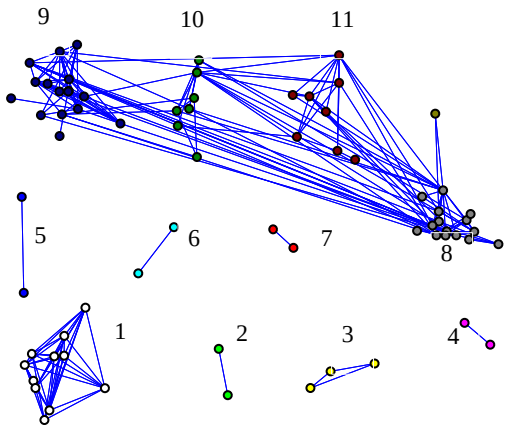


Figure 4: The prediction loss with respect to training size on three datasets. The kernel smoothing estimator performs similar to the state of the art gnmf (matrix factorization) and better than other similarity based approaches.



	Movie Titles
1	American Beauty, Lost in Translation, Pulp Fiction, Kill Bill I,II, Memento, The Royal Tenenbaums, Napoleon Dynamite,..
2	Spider-Man, Spider-Man II
3	Lord of the Rings I,II,III
4	The Bourne Identity, The Bourne Supremacy
5	Shrek, Shrek II
6	Meet the parents, American Pie
7	Indiana Jones and the Last Crusade, Raiders of the Lost Ark
8	The Patriot, Pearl Harbor, Men of Honor, John Q, The General's Daughter, National Treasure, Troy, The Italian Job,..
9	Miss Congeniality, Sweet Home Alabama,Two Weeks Notice, 50 First Dates, The Wedding Planner,Maid in Manhattan,Titanic,..
10	Men in Black I,II, Bruce Almighty, Anger Management, Mr. Deeds, Tomb Raider, The Fast and the Furious
11	Independence Day, Con Air, Twister, Armageddon, The Rock, Lethal Weapon 4, The Fugitive, Air Force One

Figure 5: A graph corresponding to the 100 most rated Netflix movies where edges represent high affinity as determined by the rule discovery process (see text for more details).

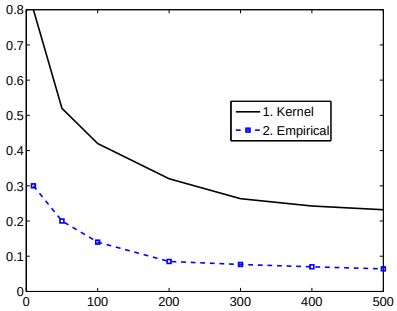


Figure 6: A quantitative evaluation of rule discovery. The x axis represents the number of rules discovered and the y axis represents the frequency of good rules in the discovered rules. A rule $i \prec j \Rightarrow k \prec l$ is considered good if i, k are of the same genre and j, l are of the same genre.

Shrek $\prec$ LOTR 1	$\Rightarrow$	Shrek 2$\prec$ LOTR 3
Shrek $\prec$ LOTR 1	$\Rightarrow$	Shrek 2$\prec$ LOTR 2
Shrek 2 $\prec$ LOTR 1	$\Rightarrow$	Shrek$\prec$ LOTR 3
Kill Bill 2 $\prec$ National Treasure	$\Rightarrow$	Kill Bill 1 $\prec$ I. Robot
Shrek 2 $\prec$ LOTR 1	$\Rightarrow$	Shrek 2$\prec$ LOTR 2
LOTR 1 $\prec$ Monsters, Inc.	$\Rightarrow$	LOTR 2$\prec$ Shrek
National Treasure $\prec$ Kill Bill 2	$\Rightarrow$	Pearl Harbor $\prec$ Kill Bill 1
LOTR 1 $\prec$ Monsters, Inc.	$\Rightarrow$	LOTR 3$\prec$ Shrek
Lose a Guy in 10 days $\prec$ Kill Bill 2	$\Rightarrow$	50 First Dates$\prec$ Kill Bill 1

Figure 7: Top 10 rules discovered by $\hat{p}$ on Netflix.

which represents like and dislike of different movie types, e.g. like of romance leads to dislike of action.

In a third experiment, we used $\hat{p}$ to construct an undirected graph whose vertices are items (Netflix movies) and two nodes i,j are connected by an edge if the average score of the rule i ranked highest $\Rightarrow$ j ranked second highest and the rule j ranked highest $\Rightarrow$ i ranked second highest is higher than a certain threshold. Figure 5 shows the graph for the 100 most rated movies in Netflix (only movies with vertex degree greater than 0 are shown). The clusters in the graph corresponding to vertex color and numbering were obtained using a graph partitioning algorithm and the graph is embedded in a 2-D plane using standard graph visualization technique. Within each of the identified clusters movies are clearly similar with respect to genre, while an even finer separation can be observed when looking at specific clusters. For example, clusters 6 and 9 both contain comedy movies, where as cluster 6 tends toward slapstick humor and cluster 9 contains romantic comedies.

5 Related Work

Collaborative filtering or recommendation system has been an active research area in computer science since the 1990s. The earliest efforts made a prediction for the rating of items based on the similarity of the test user and the training users [15, 1, 5]. Specifically, these attempts used similarity measures such as Pearson correlation [15] and Vector cosine similarity [1, 5]

Kill Bill 1	$\Rightarrow$	Kill Bill 2
Maid in Manhattan	$\Rightarrow$	The Wedding Planner
Two Weeks Notice	$\Rightarrow$	Miss Congeniality
The Royal Tenenbaums	$\Rightarrow$	Lost in Translation
The Royal Tenenbaums	$\Rightarrow$	American Beauty
The Fast and the Furious	$\Rightarrow$	Gone in 60 Seconds
Spider-Man	$\Rightarrow$	Spider-Man 2
Anger Management	$\Rightarrow$	Bruce Almighty
Memento	$\Rightarrow$	Pulp Fiction

Maid in Manhattan	$\Rightarrow$	Pulp Fiction
Maid in Manhattan	$\Rightarrow$	Kill Bill: 1
How to Lose a Guy in 10 Days	$\Rightarrow$	Pulp Fiction
The Royal Tenenbaums	$\Rightarrow$	Pearl Harbor
The Wedding Planner	$\Rightarrow$	The Matrix
Peal Harbor	$\Rightarrow$	Memento
Lost in Translation	$\Rightarrow$	Pearl Harbor
The Day After Tomorrow	$\Rightarrow$	American Beauty
The Wedding Planner	$\Rightarrow$	Raiders of the Lost Ark

Figure 8: Top (like A)$\Rightarrow$(like B) (top) and top (like A)$\Rightarrow$(dislike B) (bottom) rules discovered by $\hat{p}$ on Netflix.

to evaluate the similarity level between different users. More recent work includes user and movie clustering [1, 18, 20], item-item similarities [16], Bayesian networks [1], dependence network [4] and probabilistic latent variable models [14, 6, 13].

Parametric alternatives to our non-parametric estimator include MLE for the Mallows model [11, 3], the Bradley Terry model, and the Thurstone model [12]. A different class of estimators are based on Fourier analysis on the symmetric group [2, 7]. These techniques vary in their statistical accuracy and computational efficiency with the latter becoming crucial for large n.

6 Discussion

Estimating distributions from tied and incomplete data is a central task in many applications with perhaps the most obvious one being collaborative filtering. An accurate estimator $\hat{p}$ enables going beyond the traditional item-rank prediction task. It can be used to compute probabilities of interest, find association rules, and a wide range of additional analysis tasks.

We demonstrate the first non-parametric estimator for such data that is computationally tractable i.e., polynomial rather than exponential in n. The computation is made possible using generating function and dynamic programming techniques. We show experimentally that it performs similar to a state-of-the-art matrix factorization method and substantially outperforms other memory based methods (to which it is similar functionally). An advantage of our method is that its probabilistic nature makes it naturally suited for tasks that go beyond rank prediction such as finding association rules, optimizing prediction for customized loss functions, and deriving confidence bounds.

References

[1] J. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proc. of Uncertainty in Artificial Intelligence*, 1998.

[2] P. Diaconis. *Group Representations in Probability and Statistics*. Institute of Mathematical Statistics, 1988.

[3] M. A. Fligner and J. S. Verducci. Multistage ranking models. *Journal of the American Statistical Association*, 83:892–901, 1988.

[4] D. Heckerman, D. Maxwell Chickering, C. Meek, R. Rounthwaite, and C. Kadie. Dependency networks for inference, collaborative filtering, and data visualization. *Journal of Machine Learning Research*, 1, 2000.

[5] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, page 237. ACM, 1999.

[6] T. Hofmann. Latent semantic models for collaborative filtering. *ACM Transactions on Information Systems*, 22(1):115, 2004.

[7] J. Huang, C. Guestrin, and L. Guibas. Fourier theoretic probabilistic inference over permutations. *Journal of Machine Learning Research*, 10(May):997–1070, 2010.

[8] P. Kidwell, G. Lebanon, and W. S. Cleveland. Visualizing incomplete and partially ranked data. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1356–1363, 2008.

[9] N. D. Lawrence and R. Urtasun. Non-linear matrix factorization with gaussian processes. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 601–608. ACM, 2009.

[10] G. Lebanon and Y. Mao. Non-parametric modeling of partially ranked data. *Journal of Machine Learning Research*, 9:2401–2429, 2008.

[11] C. L. Mallows. Non-null ranking models. *Biometrika*, 44:114–130, 1957.

[12] J. I. Marden. *Analyzing and modeling rank data*. CRC Press, 1996.

[13] B. Marlin. Modeling user rating profiles for collaborative filtering. In *Advances in Neural Information Processing Systems*, 2004.

[14] D. M. Pennock, E. Horvitz, S. Lawrence, and C. L. Giles. Collaborative filtering by personality diagnosis: A hybrid memory- and model-based approach. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, 2000.

[15] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the Conference on Computer Supported Cooperative Work*, 1994.

[16] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, 2001.

[17] R. P. Stanley. *Enumerative Combinatorics*, volume 1. Cambridge University Press, 2000.

[18] L. H. Ungar and D. P. Foster. Clustering methods for collaborative filtering. In *AAAI Workshop on Recommendation Systems*, 1998.

[19] M. P. Wand and M. C. Jones. *Kernel Smoothing*. Chapman and Hall/CRC, 1995.

[20] G. R. Xue, C. Lin, Q. Yang, W. S. Xi, H. J. Zeng, Y. Yu, and Z. Chen. Scalable collaborative filtering using cluster-based smoothing. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 114–121. ACM New York, NY, USA, 2005.