Rank Aggregation for Presentation-Centric Preference Estimation

Evgeny Televitckiy and **Carmel Domshlak**
Technion - Israel Institute of Technology
Haifa, Israel

Abstract

We consider the connection between the recommendations' selection process and the form in which these recommendations are then presented to the users. On the one hand, a user of an e-commerce site typically provides her preferences as a (very rough) weak ordering over a small subset of all the available items. On the other hand, the recommendations selected for a user are presented to her as this or another projection of a strict ordering over the whole space of items. Aiming at bridging the gap between these two properties of the systems, we suggest and evaluate techniques for rank aggregation in this setting.

Introduction

A recommender system reasons about historical behavior of a community of users to suggest to a user items that the system estimates to be of the user's interest. The motivation for our work came from an observation that, while a significant amount of work has been devoted to the question of estimating user preferences (that is, to the collaborative filtering (CF) and information filtering (IF) algorithms), to our knowledge, this question has been typically considered somewhat in isolation, independently of the concrete way this estimated information is then put in use by the system. For instance, the recommendations selected for a user are presented to her in this or another *concrete form*, and probably the most typical such form is a *strict* ordering (i.e., a *linear list*) over a subset of all items available in the system. If so, then one may wonder whether the way the recommendations are *presented* to the user should not influence the very process of estimating the preferences of that user.

Having this perspective in mind, in this paper we consider the problem of generating a linear list of recommendations for a user. On the technical side, we cast this problem into an optimization problem of aggregating a collection of weak user orderings over subsets of items into a single strict ordering over the entire set of items. We consider and empirically evaluate different algorithmic approaches to this problem. Targeting the most problematic for recommender systems situations, the problem is discussed in the settings of user cold-start and item cold-start (Schein *et al.* 2002; Park *et al.* 2006).

Preliminaries

Given a universe of some items $\mathcal{I} = \{o_1, \ldots, o_N\}$, we say that $\succeq$ is an *ordering with respect to* $\mathcal{I}$ if $\succeq$ is an ordering relation over a subset $\mathcal{I}' \subseteq \mathcal{I}$. First, if the ordering $\succeq$ addresses all the items in $\mathcal{I}$, then $\succeq$ is said to be *complete*, otherwise $\succeq$ is said to be *partial*. In what follows, partial and complete orderings are denoted by $\succeq$ and $\trianglerighteq$. Second, if for each pair of items $o, o' \in \mathcal{I}'$ we have $o \succeq o' \Rightarrow o' \not\succeq o$, then $\succeq$ is said to be *strict*, otherwise $\succeq$ is said to be *weak*. By $\succ$ and $\triangleright$ we denote strict partial and strict complete orderings.

Let $U = \{u_1, \ldots, u_M\}$ be a set of users rating the items $\mathcal{I}$ in terms of some discrete rating scale $L = \{0, \ldots, M\} \subset \mathbb{N}$. Each user u_i provides the system with a *rating function* $\sigma_i : \mathcal{I}_i \mapsto L$ from a subset $\mathcal{I}_i \subseteq \mathcal{I}$. This rating function is assumed to communicate to the system information about the user's preference ordering $\succeq_i$ over $\mathcal{I}_i$, that is,

$$\begin{aligned} \sigma_i(o) > \sigma_i(o') &\Rightarrow o \ \succ_i \ o' \\ o \ \not\succ_i \ o' &\Rightarrow \sigma_i(o) \not\succ \sigma_i(o') \end{aligned} \tag{1}$$

Note that, in practical information systems, the rating scale L is usually very rough, and each user typically rates only a small subset of the entire set of items $\mathcal{I}$. Hence, in general, the individual orderings $\succeq_1, \ldots, \succeq_M$ should be expected to be both weak and partial.

In this work, we focus on the troubled user cold-start and item cold-start situations. In user cold-start, the system strives to provide recommendations to an effectively anonymous user. In item cold-start, the system strives recommending effectively anonymous (opinions-wise) items. If the user interface of the system requires the recommendations to be presented as a linear list of items, in the absence of any additional information,

- the system has to base its recommendations on the rating functions $\sigma_1, \ldots, \sigma_M$ of the previous users, and
- strive to generating a strict, complete ordering $\triangleright^\dagger$ over $\mathcal{I}$ that maximizes the agreement with these rating functions.

This agenda boils down to solving an optimization problem

$$\triangleright^\dagger = \operatorname*{argmin}_{\triangleright \ over \ \mathcal{I}} \mathscr{D}(\triangleright, \{\sigma_1, \ldots, \sigma_M\}) \tag{2}$$

for some justifiable measure of distance $\mathscr{D}$ between a single strict, complete ordering over a set, and a collection of rating functions from subsets of this set.

Rank Aggregation for User Cold-Start

For now, let us consider the user cold-start setting only, and assume the system contains no completely cold items, that is, $\mathcal{I} = \bigcup_{i=1}^{M} \mathcal{I}_i$. Considering Eq. 2, the first question is how should we measure distance between a strict, complete ordering $\rhd$ over $\mathcal{I}$, and a collection of rating functions $\{\sigma_1, \ldots, \sigma_M\}$ from subsets of $\mathcal{I}$ to L? Recalling the ordinal semantics of the user ratings (Eq. 1), this question can be reduced to measuring distance between $\rhd$, and the weak, partial orderings $\succeq_1, \ldots, \succeq_M$ induced by the respective ratings $\{\sigma_1, \ldots, \sigma_M\}$. As various measures of distance between orderings have been studied in statistics (Diaconis 1988), this already simplifies our task. For instance, the popular Kendall distance between a pair of strict, complete orderings $\rhd, \rhd'$ over $\mathcal{I}$ is given by $\mathscr{K}(\rhd, \rhd') = |\{(o_i, o_j) \mid i < j, o_i \rhd o_j, o_j \rhd' o_i\}|$, and this distance measure extends in a natural way to a $[0, 1]$-normalized distance between one and many strict, complete orderings as

$$\mathscr{K}(\rhd, \{\rhd_1, \ldots, \rhd_M\}) = \frac{1}{M\binom{|\mathcal{I}|}{2}} \sum_{i=1}^{M} \mathscr{K}(\rhd, \rhd_i). \quad (3)$$

This, however, turns out not to resolve our question entirely. On the good side, the Kendall distance is meaningful for both incomplete and weak orderings $\{\succeq_1, \ldots, \succeq_M\}$ as well. However, the issue of normalization in presence of orderings' incompleteness is somewhat more tricky. Specifically, suppose that user u_i has ranked more items than user u_j. In this case, for a given strict, complete ordering $\rhd$, we may have $\mathscr{K}(\rhd, \succeq_i) > \mathscr{K}(\rhd, \succeq_j)$ just because u_i has ranked more items, and this is definitely something we would like to avoid. For that, we consider a specific lifting of Eq. 3 to aggregation of weak, incomplete orderings, notably

$$\mathscr{K}_*(\rhd, \{\succeq_1, \ldots, \succeq_M\}) \quad = \quad \frac{1}{M} \sum_{i=1}^{M} \frac{\mathscr{K}(\rhd, \succeq_i)}{\Delta_{\mathscr{K}}(\mathcal{I}, \succeq_i)}, \quad (4)$$

where $\Delta_{\mathscr{K}}(\mathcal{I}, \succeq_i) = \max_{\rhd' \; over \; \mathcal{I}} \mathscr{K}(\rhd', \succeq_i)$. While further motivation for this particular form of lifting appear in a wider version of the paper, note that $\mathscr{K}_*$ is sensitive to the amount of ranking information provided by each user, taking into account our uncertainty about the preferences of the users over the items they never ranked.

Computing the extended Kendall distance have been thoroughly studied in recent works on rank aggregation for web meta-search (Dwork *et al.* 2001) and data integration (Fagin *et al.* 2004). In general, computing an optimal ordering $\rhd^\dagger$ with respect to the extended Kendall distance as in Eq. 3 is known to be NP-hard (Dwork *et al.* 2001), and this implies hardness of optimizing Eq. 2 with respect to $\mathscr{K}_*$. On the positive side, however, numerous effective approximation techniques for the former task have been suggested (Dwork *et al.* 2001; Fagin *et al.* 2004). In addition, in the ratings-based recommender systems of our interest, it is possible that simple adaptations of standard CF techniques will prove to be effective for this task as well.

A simple such CF-based procedure corresponds to ranking items based on their average rating across the users. Assuming $\mathcal{I} = \bigcup_{i=1}^{M} \mathcal{I}_i$, let $\sigma_{\mathrm{avg}} : \mathcal{I} \mapsto \mathbb{R}$ be defined as $\sigma_{\mathrm{avg}}(o) = \frac{1}{|U_o|} \sum_{u_i \in U_o} \sigma_i(o)$, where $U_o = \{u_i \mid u_i \in U, o \in \mathcal{I}_i\}$. If $\trianglerighteq_{\mathrm{avg}}$ is the weak, complete ordering induced by the "averaging" function σ_{avg}, the procedure (referred here as AVG-cf) picks a linearization of $\trianglerighteq_{\mathrm{avg}}$ as the approximation $\rhd^*$ of $\rhd^\dagger$.

A priori, the major attractiveness of the AVG-cf procedure (and of its possible variations) is in its simplicity. On the other hand, a shortcoming of AVG-cf is that it assumes *common-sensuality* of the rating scale L across the users. Yet another issue with the AVG-cf procedure is that it does not account for the number of ratings available per item, and it is not clear how such an information can be taken here into account in a semantically reasonable, non-parametric way.

Attempting to escape these limitations of the AVG-cf procedure brings us to consider qualitative rank aggregation techniques, and, in particular, the Markov chain orderings that have been studied in (Dwork *et al.* 2001). Let the states of a stochastic system correspond to the items $\mathcal{I}$, and let the transition distributions of two Marcov chains over $\mathcal{I}$ (called MC3 and MC4) be defined as follows.

- With probability $\epsilon > 0$, pick an item o' uniformly from $\mathcal{I}$, and move to o'. With probability $1 - \epsilon$, if the current state is item o, then the next state is chosen as follows:

MC3 First, uniformly pick a user $u_i \in U_o$, then uniformly pick an item $o' \in \mathcal{I}_i$. If $o' \succ_i o$ then move to o', else stay in o.

MC4 First, uniformly pick an item $o' \in \mathcal{I}$. If $o' \succ_i o$ for a majority of the users that ranked both o and o' (i.e., $U_o \cap U_{o'}$), then move to o', else stay in o.

Having applied the power-iteration method on the transition matrices induced by one of these Markov chains, the obtained (approximation to the) fixed-point distribution x of the chain induces a weak, complete ordering $\trianglerighteq_{\mathrm{mc}}$ over $\mathcal{I}$. The overall procedure then picks a linearization of $\trianglerighteq_{\mathrm{mc}}$ as the approximation $\rhd^*$ of $\rhd^\dagger$.

For our experiments we have used the EachMovie dataset containing user ratings for movies over a period of several years. Due to the large number of experiments we run in this study, we choose to use a randomly-sampled subset of 490 movies, and then dropped all users who ranked less than two movies from this set (as these users do not provide us with any ordinal preference information). The user ratings in this data set are integers $L = \{0, \ldots, 5\}$, with higher rating meaning more positive evaluation. For the evaluation discussed here we have used a train set of 51889 users, and a randomly selected test set of 672 users. The evaluation included AVG-cf, and both Markov chain aggregations.

Surprisingly, while the orderings generated by the two approaches differed substantially from each other, the average performance of all the methods (with respect to $\mathscr{K}_*$) was effectively identical—precision (= $1 - \mathscr{K}_*$) 0.642 for AVG-cf vs. 0.629 for MC3. Given that AVG-cf is not designed to optimize anything directly related to Kendall distance, it was somewhat surprising to find its performance comparable to this of the Markov chain ordering techniques. We believe that this outcome was mainly due to the narrow grading scale L that possibly was indeed common-sensual. In the next

section, however, we show that AVG-cf-style and qualitative rank aggregations do not always perform the same.

Rank Aggregation for Item Cold-Start

While considering rank aggregation for user cold-start situations, we assumed that all the items in the data set have been ranked by at least one previous user. This assumption, however, is well-known to be unrealistic. Thus, next we consider the item cold-start problem within our setting of generating a strictly-ordered list of recommendations. As nothing is known about a new to the system item o, it is custom to assume that user preferences over o will be close to the preferences over items that look similar to o.

In what follows, let the items $\mathcal{I}$ be described in terms of some attributes $X = \{x_1, \ldots, x_n\}$; each item thus can be seen as a vector in $\mathcal{X} = \times Dom(x_i)$. For each item $o \in \mathcal{I}$, let $\mathsf{knn}(o) \subseteq \bigcup_{i=1}^{k} \mathcal{I}_i$ be the set of (up to) k o's nearest neighbors (for some chosen value of k and a distance measure over $\mathcal{X}$) among the previously ranked items in $\mathcal{I}$. A straightforward extension of the AVG-cf procedure to the item cold-start setting is given by modifying Eq. **??** to

$$\sigma_{\text{avg}}(o) = \begin{cases} \frac{\sum_{u_i \in U_o} \sigma_i(o)}{|U_o|}, & o \in \bigcup_{i=1}^{k} \mathcal{I}_i \\ \frac{\sum_{o' \in \mathsf{knn}(o)} \sigma_{\text{avg}}(o')}{|\mathsf{knn}(o)|}, & \text{otherwise} \end{cases}, \quad (5)$$

all other things being equal. Next we introduce three model-based procedures that extend the qualitative rank aggregation techniques from the previous section.

Our first procedure AGR for a model-based rank aggregation is depicted in Figure 1. Informally, AGR corresponds to extending the qualitative rank aggregation schemes towards their *generalization* over the entire space of possible items $\mathcal{X}$ (and thus, in particular, to the cold items in $\mathcal{I}$). The first step of AGR constitutes exactly the qualitative aggregation of orderings induced by the explicit user ratings $\sigma_1, \ldots, \sigma_M$. In the second step, the procedure suggests a real-valued *ranking function $\hat{\sigma}$ from the entire space $\mathcal{X}$* that aims at generalizing the preference information captured by the aggregative ordering $\succeq$. Ultimately, such a function should both satisfy the ordering constraints posed by $\succeq$, that is $\forall o, o' \in \bigcup_{i=1}^{k} \mathcal{I}_i : \; o \succ o' \Rightarrow \hat{\sigma}(o) > \hat{\sigma}(o')$, and generalize beyond the rated items $\bigcup_{i=1}^{k} \mathcal{I}_i$. In other words, such a ranking function can be seen as a binary classifier over *pairs* of items, with the target of classification being not a class label, but a binary relation over the items (Herbrich, Graepel, & Obermayer 2000; Joachims 2002). In machine learning practice, this ranking function is selected, from a family of ranking functions, to minimize the classification (= ranking) error on the training pairs as above. Importantly, such a minimization of classification error has a clear interpretation in terms of minimizing ranking error in terms of Kendall distance between the input ordering constraints and the ordering induced by $\hat{\sigma}$.

Here we consider the family of linear ranking functions over the item attributes X. Finding such a function that minimizes classification error on our training data is equivalent to finding a weight vector $\vec{\mathbf{w}}_{\mathsf{agr}} \in \mathbb{R}^n$ so that the maximum number of the inequalities $\forall o, o' \in \bigcup_{i=1}^{k} \mathcal{I}_i, o \succ o' : \vec{\mathbf{w}}_{\mathsf{agr}} \cdot \vec{o} > \vec{\mathbf{w}}_{\mathsf{agr}} \cdot \vec{o'}$, where $\succ$ is the strict part of the aggregative ordering $\succeq$, are satisfied. This optimization problem is known to be NP-hard, yet it is possible to approximate its solution by introducing a non-negative slack variable ξ for each linear constraint, and then minimizing the sum of these slack variables. On the other hand, as suggested by theory and practice of classification Support Vector Machines (SVM) (Cortes & Vapnik 1995), adding regularization for margin maximization to the objective is likely to increase the generalization power of the learned function $\vec{\mathbf{w}}_{\mathsf{agr}}$ (that is, $\hat{\sigma}$) well beyond the training pairs of items. This setting results in the optimization problem

$$\begin{aligned} &\text{minimize: } \frac{1}{2}\vec{\mathbf{w}}_{\mathsf{agr}} \cdot \vec{\mathbf{w}}_{\mathsf{agr}} + C \sum \xi_{o,o'} \\ &\text{subject to: } \forall \xi_{o,o'} : \; \xi_{o,o'} > 0 \\ &\forall o, o' \in \bigcup_{i=1}^{k} \mathcal{I}_i, o \succ o' : \; \vec{\mathbf{w}}_{\mathsf{agr}} \cdot \vec{o} > \vec{\mathbf{w}}_{\mathsf{agr}} \cdot \vec{o'} + 1 - \xi_{o,o'} \end{aligned} \quad (6)$$

that is equivalent to that of classification SVM on difference vectors $(\vec{o} - \vec{o'})$, and thus can be efficiently solved using one of the standard tools for classification SVM.

At first view, the AGR procedure is appealing because both steps (a) and (b) aim at minimizing the Kendall distance between their inputs and outputs. However, the performance of the AGR rank aggregation on the EachMovie dataset turns out to be not very impressive. Our analysis showed that this failure should be attributed to the *combination* of steps (a) and (b) as in AGR. The pitfall in this combination is that step (a) does generate a good ordering $\succeq$, but this ordering is not forced to depend in any reasonable way on the item-describing features X. As a result, step (b) then fails to effectively generalize $\succeq$ by a ranking function from $\mathcal{X}$.

Targeting this pitfall of AGR, next we push the aggregation one step farther in the process. In the procedure GAR (see Figure 1), first, we functionally generalize the *individual* preference information provided by the users via their ratings $\sigma_1, \ldots, \sigma_M$. Such individual ranking functions $\hat{\sigma}_1, \ldots, \hat{\sigma}_M$ can be generated via M independent optimization problems as in Eq. 6, each restricted to the constraints of a certain user $u_i \in U$. At the next step, the GAR procedure aggregates the generated ranking functions $\{\hat{\sigma}_1, \ldots, \hat{\sigma}_M\}$ into a single ranking function $\hat{\sigma}$ that aims at capturing the "average" taste across the users. In case of linear ranking functions (considered here), $\vec{\mathbf{w}}_{\mathsf{gar}} = \frac{1}{M} \sum_{i=1}^{M} \vec{\mathbf{w}}_i$ appears to be a natural and semantically justifiable candidate for such an "averaging" function $\hat{\sigma}$. However, the performance of GAR on EachMovie is still unsatisfactory. While it eliminates the major shortcoming of AGR, it introduces a new shortcoming of its own: Simple averaging of the individual user models does not prove to be very effective. Hence, next we proceed in a different direction.

The lessons learnt with the AGR and GAR procedures suggest stratifying the model-based rank aggregation by pushing the aggregation step even farther down the process. This is exactly what we converge to in our third procedure, GRA, depicted in Figure 1. The first step here is identical to this of

AGR *(aggregate, generalize, and rank)*

(a) Aggregate the weak, partial orderings $\succeq_1, \ldots, \succeq_M$ induced by the user ratings $\sigma_1, \ldots, \sigma_M$ into a single (possibly weak) partial ordering $\succeq$.

(b) From $\succeq$, generate a ranking function $\hat{\sigma} : \mathcal{X} \mapsto \mathbb{R}$ that generalizes the ordering $\succeq$ to the entire item space $\mathcal{X} \supset \mathcal{I}$.

(c) Generate a strict, complete ordering $\rhd^*$ by picking an arbitrary linearization of the ordering $\hat{\succeq}$ induced by $\hat{\sigma}$ according to Eq. 1.

GAR *(generalize, aggregate, and rank)*

(a) For each user u_i, generate a ranking function $\hat{\sigma}_i : \mathcal{X} \mapsto \mathbb{R}$ that generalizes the ordering $\succeq_i$ induced by σ_i to the entire item space $\mathcal{X} \supseteq \mathcal{I}$.

(b) Aggregate $\{\hat{\sigma}_1, \ldots, \hat{\sigma}_M\}$ into a single ranking function $\hat{\sigma}$ aiming at capturing the average preferences of the users U.

(c) Generate a strict, complete ordering $\rhd^*$ by picking an arbitrary linearization of the ordering $\hat{\succeq}$ induced by $\hat{\sigma}$ according to Eq. 1.

GRA *(generalize, rank, and aggregate)*

(a) For each user u_i, generate a ranking function $\hat{\sigma}_i : \mathcal{X} \mapsto \mathbb{R}$ that generalizes the ordering $\succeq_i$ induced by σ_i to the entire item space $\mathcal{X} \supseteq \mathcal{I}$.

(b) For each ratings-less item $o \in \mathcal{I} \setminus \bigcup_{i=1}^{k} \mathcal{I}_i$:

- Select a subset $\hat{U}_o \subseteq U$.
- For each $u_i \in \hat{U}_o$, (schematically) add o into $\mathcal{I}_i$.

(c) Aggregate the individual user orderings $\hat{\succeq}_1, \ldots, \hat{\succeq}_M$ over the (extended) item sets $\mathcal{I}_1, \ldots, \mathcal{I}_M$ into a strict, complete ordering $\rhd^*$ over $\mathcal{I}$.

Figure 1: Model-based qualitative rank aggregations.

the GAR procedure. However, instead of then gluing up the individual ranking functions $\hat{\sigma}_i$ into a single function $\hat{\sigma}$, we use them directly as a source of *pseudo-ranking* of the entire item set $\mathcal{I}$ "on behalf of" each user. Note that we apply the pseudo-ranking not only to the cold items $o \in \mathcal{I} \setminus \bigcup_{i=1}^{k} \mathcal{I}_i$, but also to the items the user has actually ranked. The (optional) selection at step (b) restricts the pseudo-ranking of each cold item o only to a selected subset $\hat{U}_o$ of users that are considered to be relatively "reliable" pseudo-rankers for this particular item o.

The data used for evaluation of all the above methods was identical to this used for evaluating the basic AVG-cf and MC3/MC4 tools, except for that now the training and testing data divide between the movies. We randomly selected $\alpha = 50$ movies for testing, and used the rest 440 movies for training. The movies have been described by 26 *boolean* attributes capturing the genres of the movie, the period the movie was created, and the geographic origin of the movie. For the KNN parameters of AVG-cf we used simple L_1 as the distance measure between the movies, and optimized the number of neighbors to $k = 50$. In GRA, a user has been used as a pseudo-ranker for the movie o if she ranked one of the k nearest neighbors of o.

The table below shows the methods' performance (in terms of precision $= 1 - \mathscr{K}_*$) on the *union* of the training and testing data. It is easy to see that, AVG-cf and GAR clearly outperform the other two methods, with GAR performing slightly better than AVG-cf.

AGR	GAR	GRA(MC3)	GRA(MC4)	AVG-cf
0.583	0.603	**0.652**	0.651	0.645

From these results, however, it is impossible to extract the marginal precision of the techniques on the cold items. Hence, below we evaluate the effectiveness in relative ordering of the cold items only. For that, let $\mathscr{K}^{(o)}(\succeq, \succeq')$ be the number of ordering disagreements between $\succeq$ and $\succeq'$ but only on item pairs containing the item o. Given that, let $\mathscr{K}_*^{(o)}(\rhd, \{\succeq_1, \ldots, \succeq_M\})$ be defined similarly to $\mathscr{K}_*$ in Eq. 4, but with respect to $\mathscr{K}^{(o)}$ instead of $\mathscr{K}$. Focusing on the cold items only, here we consider two alternative measures of precision with respect to the cold items $o_1, \ldots, o_\alpha$ and the single-item-oriented distance measure $\mathscr{K}_*^{(o)}$:

$$p_1 = \frac{1}{\alpha} \sum_{i=1}^{\alpha} \left(1 - \mathscr{K}_*^{(o_i)}(\rhd, \{\succeq_1, \ldots, \succeq_M\})\right)$$

$$p_k = \sum_{i=1}^{\alpha} \frac{\hat{U}_{o_i}}{\sum_{j=1}^{\alpha} \hat{U}_{o_j}} \left(1 - \mathscr{K}_*^{(o_i)}(\rhd, \{\succeq_1, \ldots, \succeq_M\})\right)$$

We run α separate additions of the cold items to the training set, and averaged the performance of all the methods with respect to p_1 and p_2. The results are presented in the table below.

	p_1	p_2
AGR	0.642	0.640
GAR	0.586	0.623
GRA(MC4)	0.674	**0.687**
GRA(MC3)	**0.681**	**0.687**
AVG-cf	0.672	0.639

First, these results show that both MC3 and MC4 versions of GRA significantly outperformed AVG-cf with respect to the precision measure p_2. Currently we perform a more detailed introspection into the results of both experiments, but, in any event, the performance of GRA already appears very promising.

References

Cortes, C., and Vapnik, V. N. 1995. Support–vector networks. *Machine Learning Journal* 20:273–297.

Diaconis, P. 1988. *Group Representation in Probability and Statistics*. IBM Lecture Series 11.

Dwork, C.; Kumar, R.; Naor, M.; and Sivakumar, D. 2001. Rank aggregation methods for the Web. In *WWW*, 613–622.

Fagin, R.; Kumar, R.; Mahdian, M.; Sivakumar, D.; and Vee, E. 2004. Comparing and aggregating rankings with ties. In *PODS*.

Herbrich, R.; Graepel, T.; and Obermayer, K. 2000. Large margin rank boundaries for ordinal regression. In *Advances in Large Margin Classifiers*. 115–132.

Joachims, T. 2002. Optimizing search engines using clickthrough data. In *KDD*, 154–161.

Park, S.-T.; Pennock, D.; Madani, O.; Good, N.; and DeCoste, D. 2006. Naive filterbots for robust cold-start recommendations. In *KDD*, 699–705.

Schein, A. I.; Popescul, A.; Ungar, L. H.; and Pennock, D. M. 2002. Methods and metrics for cold-start recommendations. In *SIGIR*, 253–260.