Magnitude-Preserving Ranking Algorithms

Technical Report Number: TR2007-887

Corinna Cortes
Google Research,
76 Ninth Avenue, New York, NY 10011

Mehryar Mohri
Courant Institute of Mathematical Sciences and Google Research,
251 Mercer Street, New York, NY 10012

Ashish Rastogi
Courant Institute of Mathematical Sciences,
251 Mercer Street, New York, NY 10012

March 15, 2007

Abstract

This paper studies the learning problem of ranking when one wishes not just to accurately predict pairwise ordering but also preserve the magnitude of the preferences or the difference between ratings, a problem motivated by its crucial importance in the design of search engines, movie recommendation, and other similar ranking systems. We describe and analyze several algorithms for this problem and give stability bounds for their generalization error, extending previously known stability results to non-bipartite ranking and magnitude of preference-preserving algorithms. We also report the results of experiments comparing these algorithms on several datasets and contrast these results with those obtained using an AUC-maximization algorithm.

1 Motivation

The learning problem of ranking has gained much attention in recent years, in part motivated by the development of new search engines and movie recommendation systems Freund et al. [1998], Crammer and Singer [2001], Joachims [2002], Shashua and Levin [2003], Rudin et al. [2005], Agarwal and Niyogi [2005], Herbrich et al. [2000]. The ordering of the list of documents returned by a search engine or an information extraction system, or that of the list of movies supplied by a movie recommendation system is a key aspect of their quality.

In most previous studies, the problem of ranking is formulated as that of learning a scoring function with small pairwise misranking error from a labeled sample of pairwise preferences Freund et al. [1998], Crammer and Singer [2001], Joachims [2002], Rudin et al. [2005], Agarwal and Niyogi [2005]. But, this formulation of the problem and thus the scoring function learned ignore the magnitude of the preferences. In many applications, it is not sufficient to determine if one example is preferred to another. One may further request an assessment of how large that preference is. Taking this magnitude of preference into consideration is critical for example in the design of search engines, which originally motivated our study,

but also in other recommendation systems. For a recommendation system, one may choose to truncate the ordered list returned where a large gap in predicted preference is found. For a search engine, this may trigger a search in parallel corpora to display more relevant results.

This motivated our study of the problem of ranking while preserving the magnitude of preferences, which we will refer to in short by *magnitude-preserving ranking*. The problem that we are studying bears some resemblance with that of ordinal regression McCullagh [1980], McCullagh and Nelder [1983], Shashua and Levin [2003], Chu and Keerthi [2005]. It is however distinct from ordinal regression since in ordinal regression the magnitude of the difference in target values is not taken into consideration in the formulation of the problem or the solutions proposed. In the formulation of ordinal regression by Shashua and Levin [2003], Chu and Keerthi [2005] the *magnitude of the difference* in the target values is not taken into consideration in the cost function. A crucial aspect of the algorithms we propose is that they penalize misranking errors more heavily in the case of larger magnitudes of preferences.

We describe and analyze several algorithms for magnitude-preserving ranking and give stability bounds for their generalization error, extending previously known stability results to non-bipartite ranking and magnitude of preference-preserving algorithms. In particular, our bounds extend the framework of Bousquet and Elisseeff [2000, 2002] to the case of cost functions over pairs of examples, and extend the bounds of Agarwal and Niyogi [2005] beyond the bi-partite ranking problem. Our bounds also apply to algorithms optimizing the so-called *hinge rank loss*.

We also report the results of experiments comparing these algorithms on several datasets and contrast these results with those obtained using RankBoost Freund et al. [1998], Rudin et al. [2005], an algorithm maximizing the Area under the ROC Curve (AUC), or minimizing pairwise misranking.

The remainder of the paper is organized as follows. Section 2 presents stability-based generalization bounds for a family of magnitude-preserving algorithms. Section 3 describes and analyzes these algorithms in detail. Section 4 presents the results of our experiments with these algorithms on several datasets.

2 Stability bounds

In Bousquet and Elisseeff [2000, 2002] stability bounds were given for several regression and classification algorithms. This section shows similar stability bounds for ranking and magnitude-preserving ranking algorithms. This also generalizes the results of Agarwal and Niyogi [2005] which were given in the specific case of bi-partite ranking.

Let S be a sample of m labeled examples drawn i.i.d. from a set X according to some distribution D:

$$(x_1, y_1), \ldots, (x_m, y_m) \in X \times \mathbb{R}. \tag{1}$$

For any $i \in [1, m]$, we denote by S^{-i} the sample derived from S by omitting example (x_i, y_i), and by S^i the sample derived from S by replacing example (x_i, y_i) with an other example (x'_i, y'_i) drawn i.i.d. from X according to D. For convenience, we will sometimes denote by $y_x = y_i$ the label of a point $x = x_i \in X$.

The quality of the ranking algorithms we consider is measured with respect to pairs of examples. Thus, a cost functions c takes as arguments two sample points. For a fixed cost function c, the empirical error $\widehat{R}(h, S)$ of a hypothesis $h : X \mapsto \mathbb{R}$ on a sample S is defined by:

$$\widehat{R}(h, S) = \frac{1}{m^2} \sum_{i=1}^{m} \sum_{j=1}^{m} c(h, x_i, x_j). \tag{2}$$

The true error $R(h)$ is defined by

$$R(h) = \mathrm{E}_{x,x' \sim D}[c(h, x, x')]. \tag{3}$$

The following definitions are natural extensions to the case of cost functions over pairs of those given by Bousquet and Elisseeff [2002].

Definition 1. *A learning algorithm L is said to be uniformly β-stable with respect to the sample S and cost function c if there exists $\beta \geq 0$ such that for all $S \in (X \times \mathbb{R})^m$ and $i \in [1, m]$,*

$$\forall x, x' \in X,\ |c(h_S, x, x') - c(h_{S^{-i}}, x, x')| \leq \beta. \tag{4}$$

Definition 2. *A cost function c is is σ-admissible with respect to a hypothesis set H if there exists $\sigma \geq 0$ such that for all $h, h' \in H$, and for all $x, x' \in X$,*

$$|c(h, x, x') - c(h', x, x')| \leq \sigma(|\Delta h(x')| + |\Delta h(x)|), \tag{5}$$

with $\Delta h = h' - h$.

2.1 Cost functions

We introduce several cost functions related to magnitude-preserving ranking. The first one is the so-called *hinge rank loss* which is a natural extension of the pairwise misranking loss Cortes and Mohri [2004], Rudin et al. [2005]. It penalizes a pairwise misranking by the magnitude of preference predicted or the nth power of that magnitude ($n = 1$ or $n = 2$):

$$c^n_{\mathrm{HR}}(h, x, x') = \begin{cases} 0, \text{ if } (h(x') - h(x))(y_{x'} - y_x) \geq 0 \\ \big|(h(x') - h(x))\big|^n, \text{ otherwise.} \end{cases} \tag{6}$$

c^n_{HR} does not take into consideration the true magnitude of preference $y_{x'} - y_x$ for each pair (x, x') however. The following cost function takes into consideration the true magnitude of preference and penalizes deviations of the predicted magnitude with respect to that. Thus, it matches our objective of magnitude-preserving ranking ($n = 1, 2$):

$$c^n_{\mathrm{MP}}(h, x, x') = \big|(h(x') - h(x)) - (y_{x'} - y_x)\big|^n. \tag{7}$$

A one-sided version of that cost function penalizing only misranked pairs is given by ($n = 1, 2$):

$$c^n_{\mathrm{HMP}}(h, x, x') = \begin{cases} 0, \text{ if } (h(x') - h(x))(y_{x'} - y_x) \geq 0 \\ \big|(h(x') - h(x)) - (y_{x'} - y_x)\big|^n, \text{ otherwise.} \end{cases} \tag{8}$$

Finally, we will consider the following cost function derived from the ϵ-insensitive cost function used in SVM regression (SVR) Vapnik [1998] ($n = 1, 2$):

$$c^n_{\mathrm{SVR}}(h, x, x') = \begin{cases} 0, \text{ if } \big[(h(x') - h(x)) - (y_{x'} - y_x)\big] \leq \epsilon \\ \big|\big|(h(x') - h(x)) - (y_{x'} - y_x)\big| - \epsilon\big|^n, \text{ otherwise.} \end{cases} \tag{9}$$

Note that all of these cost functions are convex functions of $h(x)$ and $h(x')$.

2.2 Magnitude-preserving regularization algorithms

For a cost function c as just defined and a regularization function N, a regularization-based algorithm can be defined as one minimizing the following objective function:

$$F(h, S) = N(h) + C \frac{1}{m^2} \sum_{i=1}^{m} \sum_{j=1}^{m} c(h, x_i, x_j), \tag{10}$$

where $C \geq 0$ is a constant determining the trade-off between the emphasis on the regularization term versus the error term. In much of what follows, we will consider the case where the hypothesis set H is a reproducing

Hilbert space and where N is the squared norm in a that space, $N(h) = \|h\|_K^2$ for a kernel K, though some of our results can straightforwardedly be generalized to the case of an arbitrary convex N. By the reproducing property, for any $h \in H$, $\forall x \in X,\ h(x) = \langle h, K(x,.)\rangle$ and by Cauchy-Schwarz's inequality,

$$\forall x \in X, |h(x)| \leq \|h\|_K \sqrt{K(x,x)}. \tag{11}$$

Assuming that for all $x \in X, K(x,x) \leq \kappa^2$ for some constant $\kappa \geq 0$, the inequality becomes: $\forall x \in X, |h(x)| \leq \kappa \|h\|_K$. With the cost functions previously discussed, the objective function F is then strictly convex and the optimization problem admits a unique solution. In what follows, we will refer to the algorithms minimizing the objective function F with a cost function defined in the previous section as *magnitude-preserving regularization algorithms*.

Lemma 1. *Assume that the hypothesis in H are bounded, that is for all $h \in H$ and $x \in S$, $|h(x) - y_x| \leq M$. Then, the cost functions c^n_{HR}, c^n_{MP}, c^n_{HMP}, and c^n_{SVR} are all σ_n-admissible with $\sigma_1 = 1$, $\sigma_2 = 4M$.*

Proof. We will give the proof in the case of c^n_{MP}, $n = 1, 2$, the other cases can be treated similarly.

By definition of c^1_{MP}, for all $x, x' \in X$,

$$\begin{array}{l} |c^1_{\mathrm{MP}}(h', x, x') - c^1_{\mathrm{MP}}(h, x, x')| = \\ |(h'(x') - h'(x)) - (y_{x'} - y_x)| - |(h(x') - h(x)) - \\ (y_{x'} - y_x)|. \end{array} \tag{12}$$

Using the identity $|X' - Y| - |X - Y| \leq |X' - X|$, valid for all $X, X', Y \in \mathbb{R}$, this shows that

$$\begin{array}{rl} |c^1_{\mathrm{MP}}(h', x, x') - c^1_{\mathrm{MP}}(h, x, x')| \leq & |\Delta h(x') - \Delta h(x)| \\ \leq & |\Delta h(x')| + |\Delta h(x)|, \end{array} \tag{13}$$

which shows the σ-admissibility of c^1_{MP} with $\sigma = 1$. For c^2_{MP}, for all $x, x' \in X$,

$$\begin{array}{l} |c^2_{\mathrm{MP}}(h', x, x') - c^2_{\mathrm{MP}}(h, x, x')| = \\ ||(h'(x') - h'(x)) - (y_{x'} - y_x)|^2 \\ -|(h(x') - h(x)) - (y_{x'} - y_x)|^2| \\ \leq |\Delta h(x') - \Delta h(x)|(|h'(x') - y_{x'}| + \\ |h(x') - y_{x'}| + |h'(x) - y_x| + |h(x) - y_x|) \\ \leq 4M(|\Delta h(x')| + |\Delta h(x)|), \end{array} \tag{14}$$

which shows the σ-admissibility of c^2_{MP} with $\sigma = 4M$. □

Proposition 1. *Assume that the hypotheses in H are bounded, that is for all $h \in H$ and $x \in S$, $|h(x) - y_x| \leq M$. Then, a magnitude-preserving regularization algorithm as defined above is β-stable with $\beta = \frac{4C\sigma_n^2\kappa^2}{m}$.*

Proof. Fix the cost function to be c, one of the σ_n-admissible cost function previously discussed. Let h_S denote the function minimizing $F(h, S)$ and $h_{S^{-k}}$ the one minimizing $F(h, S^{-k})$. We denote by $\Delta h_S = h_{S^{-k}} - h_S$.

Since the cost function c is convex with respect to $h(x)$ and $h(x')$, $\widehat{R}(h, S)$ is also convex with respect to h and for $t \in [0, 1]$,

$$\begin{array}{l} \widehat{R}(h_S + t\Delta h_S, S^{-k}) - \widehat{R}(h_S, S^{-k}) \leq \\ t\left[\widehat{R}(h_{S^{-k}}, S^{-k}) - \widehat{R}(h_S, S^{-k})\right]. \end{array} \tag{15}$$

Similarly,

$$\begin{array}{l} \widehat{R}(h_{S^{-k}} - t\Delta h_S, S^{-k}) - \widehat{R}(h_{S^{-k}}, S^{-k}) \leq \\ t\left[\widehat{R}(h_S, S^{-k}) - \widehat{R}(h_{S^{-k}}, S^{-k})\right]. \end{array} \tag{16}$$

Summing these inequalities yields

$$\begin{array}{l}\widehat{R}(h_S + t\Delta h_S, S^{-k}) - \widehat{R}(h_S, S^{-k}) + \\ \widehat{R}(h_{S^{-k}} - t\Delta h_S, S^{-k}) - \widehat{R}(h_{S^{-k}}, S^{-k}) \leq 0.\end{array} \tag{17}$$

By definition of h_S and $h_{S^{-k}}$ as functions minimizing the objective functions, for all $t \in [0,1]$,

$$\begin{array}{l}F(h_S, S) - F(h_S + t\Delta h_S, S) \leq 0 \\ F(h_{S^{-k}}, S^{-k}) - F(h_{S^{-k}} - t\Delta h_S, S^{-k}) \leq 0.\end{array} \tag{18}$$

Multiplying Inequality 17 by C and summing it with the two Inequalities 18 lead to

$$\begin{array}{l}A + \|h_S\|_K^2 - \|h_S + t\Delta h_S\|_K^2 + \|h_{S^{-k}}\|_K^2 \\ -\|h_{S^{-k}} - t\Delta h_S\|_K^2 \leq 0.\end{array} \tag{19}$$

with $A = C\left(\widehat{R}(h_S, S) - \widehat{R}(h_S, S^{-k}) + \right.$
$\left.\widehat{R}(h_S + t\Delta h_S, S^{-k}) - \widehat{R}(h_S + t\Delta h_S, S)\right)$. Since

$$\begin{array}{l}A = \frac{C}{m^2}\left[\sum_{i\neq k} c(h_S, x_i, x_k) - c(h_S + t\Delta h_S, x_i, x_k) + \right. \\ \left.\sum_{i\neq k} c(h_S, x_k, x_i) - c(h_S + t\Delta h_S, x_k, x_i)\right],\end{array} \tag{20}$$

by the σ_n-admissibility of c,

$$\begin{array}{rl}|A| &\leq \frac{2Ct\sigma_n}{m^2}\sum_{i\neq k}(|\Delta h_S(x_k)| + |\Delta h_S(x_i)|) \\ &\leq \frac{4Ct\sigma_n\kappa}{m}\|\Delta h_S\|_K.\end{array}$$

Using the fact that $\|h\|_K^2 = \langle h, h\rangle$ for any h, it is not hard to show that

$$\begin{array}{l}\|h_S\|_K^2 - \|h_S + t\Delta h_S\|_K^2 + \|h_{S^{-k}}\|_K^2 - \\ \|h_{S^{-k}} - t\Delta h_S\|_K^2 = 2t(1-t)\|\Delta h_S\|_K^2.\end{array}$$

In view of this and the inequality for $|A|$, Inequality 19 implies $2t(1-t)\|\Delta h_S\|_K^2 \leq \frac{4Ct\sigma_n\kappa}{m}\|\Delta h_S\|_K$, that is after dividing by t and taking $t \to 0$,

$$\|\Delta h_S\|_K \leq \frac{2C\sigma_n\kappa}{m}. \tag{21}$$

By the σ_n-admissibility of c, for all $x, x' \in X$,

$$\begin{array}{l}|c(h_S, x, x') - c(h_{S^{-k}}, x, x')| \\ \leq \sigma_n(|\Delta h_S(x')| + |\Delta h_S(x)|) \\ \leq 2\sigma_n\kappa\|\Delta h_S\|_K \\ \leq \frac{4C\sigma_n^2\kappa^2}{m}.\end{array}$$

This shows the β-stability of the algorithm with $\beta = \frac{4C\sigma_n^2\kappa^2}{m}$. □

To shorten the notation, in the absence of ambiguity, we will write in the following $\widehat{R}(h_S)$ instead of $\widehat{R}(h_S, S)$.

Theorem 1. *Let c be any of the cost functions defined in Section 2.1. Let L be a uniformly β-stable algorithm with respect to the sample S and cost function c and let h_S be the hypothesis returned by L. Assume that the hypotheses in H are bounded, that is for all $h \in H$, sample S, and $x \in S$, $|h(x) - y_x| \leq M$. Then, for any $\epsilon > 0$,*

$$\Pr_{S\sim D}\left[|R(h_S) - \widehat{R}(h_S)| > \epsilon + 2\beta\right] \leq 2e^{-\frac{m\epsilon^2}{2(\beta m + (2M)^n)^2}}.$$

Proof. We apply McDiarmid's inequality McDiarmid [1998] to $\Phi(S) = R(h_S) - \widehat{R}(h_S, S)$. We will first give a bound on $\mathrm{E}[\Phi(S)]$ and then show that $\Phi(S)$ satisfies the conditions of McDiarmid's inequality.

We will denote by $S^{i,j}$ the sample derived from S by replacing x_i with x'_i and x_j with x'_j, with x'_i and x'_j sampled i.i.d. according to D.

Since the sample points in S are drawn in an i.i.d. fashion, for all $i, j \in [1, m]$,

$$\begin{aligned}\mathrm{E}_S[\widehat{R}(h_S, S)] &= \tfrac{1}{m^2}\textstyle\sum_{i=1}^m \sum_{j=1}^m \mathrm{E}[c(h_S, x_i, x_j)] \\ &= \mathrm{E}_{S\sim D}[c(h_S, x_i, x_j)] \\ &= \mathrm{E}_{S^{i,j}\sim D}[c(h_{S^{i,j}}, x'_i, x'_j)] \\ &= \mathrm{E}_{S, x'_i, x'_j \sim D}[c(h_{S^{i,j}}, x'_i, x'_j)].\end{aligned} \tag{22}$$

Note that by definition of $R(h_S)$, $\mathrm{E}_S[R(h_S)] = \mathrm{E}_{S,x'_i,x'_j\sim D}[c(h_S, x'_i, x'_j)]$. Thus, $\mathrm{E}_S[\Phi(S)] = \mathrm{E}_{S,x,x'}[c(h_S, x'_i, x'_j) - c(h_{S^{i,j}}, x'_i, x'_j)]$, and by β-stability (Proposition 1)

$$\begin{aligned}&|\,\mathrm{E}_S[\Phi(S)]| \le \mathrm{E}_{S,x,x'}[|c(h_S, x'_i, x'_j) - c(h_{S^i}, x'_i, x'_j)|] + \\ &\mathrm{E}_{S,x,x'}[|c(h_{S^i}, x'_i, x'_j) - c(h_{S^{i,j}}, x'_i, x'_j)|] \le 2\beta.\end{aligned}$$

Now,

$$\begin{aligned}|R(h_S) - R(h_{S^k})| &= |\,\mathrm{E}_S[c(h_S, x, x') - c(h_{S^k}, x, x')]| \\ &\le \mathrm{E}_S[|c(h_S, x, x') - c(h_{S^k}, x, x')|] \\ &\le \beta.\end{aligned}$$

For any $x, x' \in X$, $|c(h_S, x_k, x_j) - c(h_{S^k}, x_i, x'_k)| < \mathrm{E}_S[|c(h_{S^k}, x, x') - c(h_{S^k}, x, x')|] \le (2M)^n$, where $n = 1$ or $n = 2$. Thus, we have

$$\begin{aligned}&|\widehat{R}(h_S) - \widehat{R}(h_S^k)| \le \\ &\tfrac{1}{m^2}\textstyle\sum_{i\neq k}\sum_{j\neq k}|c(h_S, x_i, x_j) - c(h_{S^k}, x_i, x_j)| + \\ &\tfrac{1}{m^2}\textstyle\sum_{j=1}^m |c(h_S, x_k, x_j) - c(h_{S^k}, x'_k, x_j)| + \\ &\tfrac{1}{m^2}\textstyle\sum_{i=1}^m |c(h_S, x_k, x_j) - c(h_{S^k}, x_i, x'_k)| \\ &\le \tfrac{1}{m^2}(m^2\beta) + \tfrac{m}{m^2}2(2M)^n = \beta + 2(2M)^n/m.\end{aligned}$$

Thus,

$$|\Phi(S) - \Phi(S^k)| \le 2(\beta + (2M)^n/m), \tag{23}$$

and $\Phi(S)$ satisfies the hypotheses of McDiarmid's inequality. □

The following Corollary gives stability bounds for the generalization error of magnitude-preserving regularization algorithms.

Corollary 1. *Let L be a magnitude-preserving regularization algorithm and let c be the corresponding cost function and assume that for all $x \in X, K(x, x) \le \kappa^2$. Assume that the hypothesis set H is bounded, that is for all $h \in H$, sample S, and $x \in S$, $|h(x) - y_x| \le M$. Then, with probability at least $1 - \delta$,*

- *for $n = 1$,*
$$R(h_S) \le \widehat{R}(h_S) + \frac{8\kappa^2 C}{m} + 2(2\kappa^2 C + M)\sqrt{\frac{2}{m}\log\frac{2}{\delta}};$$
- *for $n = 2$,*
$$R(h_S) \le \widehat{R}(h_S) + \frac{128\kappa^2 C M^2}{m} + 4M^2(16\kappa^2 C + 1)\sqrt{\frac{2}{m}\log\frac{2}{\delta}}.$$

Proof. By Proposition 1, these algorithms are β-stable with $\beta = \frac{4C\sigma_n^2\kappa^2}{m}$. □

These bounds are of the form $R(h_S) \le \widehat{R}(h_S) + O(\frac{C}{\sqrt{m}})$. Thus, they are effective for values of $C \gg \frac{1}{\sqrt{m}}$. In the next sections, we will examine some of these magnitude preserving algorithms in more detail.

3 Algorithms

The regularization algorithms based on the cost functions c^n_{MP} and c^n_{SVR} correspond closely to the idea of preserving the magnitude of preferences since these cost functions penalize deviations of a predicted difference of score from the target preferences. We will refer by MPRank to the algorithm minimizing the regularization-based objective function based on c^n_{MP}:

$$F(h,S) = \|h\|^2_K + C\frac{1}{m^2}\sum_{i=1}^{m}\sum_{j=1}^{m} c^n_{\mathrm{MP}}(h,x_i,x_j), \tag{24}$$

and by SVRank to the one based on the cost function c^n_{SVR}

$$F(h,S) = \|h\|^2_K + C\frac{1}{m^2}\sum_{i=1}^{m}\sum_{j=1}^{m} c^n_{\mathrm{SVR}}(h,x_i,x_j). \tag{25}$$

For a fixed n, $n = 1, 2$, the same stability bounds hold for both algorithms as seen in the previous section. However, their running-time complexity is significantly different.

3.1 MPRank

We will examine the algorithm in the case $n = 2$. Let $\Phi : X \mapsto F$ be the mapping from X to the reproducing Hilbert space. The hypothesis set H that we are considering is that of linear functions h, that is $\forall x \in X, h(x) = w \cdot \Phi(x)$. The objective function can be expressed as follows

$$\begin{aligned} F(h,S) &= \|w\|^2 + C\frac{1}{m^2}\sum_{i=1}^{m}\sum_{j=1}^{m}\big[(w\cdot\Phi(x_j)- \\ &\quad w\cdot\Phi(x_i)) - (y_j - y_i)\big]^2 \\ &= \|w\|^2 + \frac{2C}{m}\sum_{i=1}^{m}\|w\cdot\Phi(x_i) - y_i\|^2 - \\ &\quad 2C\|w\cdot\bar{\Phi} - \bar{y}\|^2, \end{aligned}$$

where $\bar{\Phi} = \frac{1}{m}\sum_{i=1}^{m}\Phi(x_i)$ and $\bar{y} = \frac{1}{m}\sum_{i=1}^{m} y_i$. The objective function can thus be written with a single sum over the training examples, which translates in a more efficient computation of the solution.

Let N be the dimension of the feature space F. For $i = 1, \ldots, m$, let $\mathbf{M}_{x_i} \in \mathbb{R}^{N\times 1}$ denote the column matrix representing $\Phi(x_i)$, $\mathbf{M}_{\bar{\Phi}} \in \mathbb{R}^{N\times 1}$ a column matrix representing $\bar{\Phi}$, $\mathbf{W} \in \mathbb{R}^{N\times 1}$ a column matrix representing the vector w, $M_Y \in \mathbb{R}^{m\times 1}$ a column matrix whose ith component is y_i, and $M_{\bar{Y}} \in \mathbb{R}^{m\times 1}$ a column matrix with all its components equal to $\bar{y}$. Let $\mathbf{M}_X, \mathbf{M}_{\bar{X}} \in \mathbb{R}^{N\times m}$ be the matrices defined by:

$$\mathbf{M}_X = [\mathbf{M}_{x_1} \ldots \mathbf{M}_{x_m}] \quad \mathbf{M}_X = [\mathbf{M}_{\bar{\Phi}} \ldots \mathbf{M}_{\bar{\Phi}}]. \tag{26}$$

Then, the expression giving F can be rewritten as

$$F = \|W\|^2 + \frac{2C}{m}\|\mathbf{M}_X^\top\mathbf{W} - \mathbf{M}_Y\|^2 - \frac{2C}{m}\|\mathbf{M}_{\bar{X}}^\top\mathbf{W} - \mathbf{M}_{\bar{Y}}\|^2.$$

The gradient of F is then given by: $\nabla F = 2\mathbf{W} + \frac{4C}{m}\mathbf{M}_X(\mathbf{M}_X^\top\mathbf{W} - \mathbf{M}_Y) - \frac{4C}{m}\mathbf{M}_{\bar{X}}(\mathbf{M}_{\bar{X}}^\top\mathbf{W} - \mathbf{M}_{\bar{Y}})$. Setting $\nabla F = 0$ yields the unique closed form solution of the convex optimization problem:

$$\begin{aligned} \mathbf{W} = C'\big(\mathbf{I} + C'(\mathbf{M}_X - \mathbf{M}_{\bar{X}})(\mathbf{M}_X - \mathbf{M}_{\bar{X}})^\top\big)^{-1} \\ (\mathbf{M}_X - \mathbf{M}_{\bar{X}})(\mathbf{M}_Y - \mathbf{M}_{\bar{Y}}), \end{aligned} \tag{27}$$

where $C' = \frac{2C}{m}$. Here, we are using the identity $\mathbf{M}_X\mathbf{M}_X^\top - \mathbf{M}_{\bar{X}}\mathbf{M}_{\bar{X}}^\top = (\mathbf{M}_X - \mathbf{M}_{\bar{X}})(\mathbf{M}_X - \mathbf{M}_{\bar{X}})^\top$, which is not hard to verify. This provides the solution of the primal problem. Using the fact the matrices $(\mathbf{I} + C'(\mathbf{M}_X - \mathbf{M}_{\bar{X}})(\mathbf{M}_X - \mathbf{M}_{\bar{X}})^\top)^{-1}$ and $\mathbf{M}_X - \mathbf{M}_{\bar{X}}$ commute leads to:

$$\begin{aligned}\mathbf{W} = C'(\mathbf{M}_X - \mathbf{M}_{\bar{X}})\big(\mathbf{I} + C'(\mathbf{M}_X - \mathbf{M}_{\bar{X}})\\ (\mathbf{M}_X - \mathbf{M}_{\bar{X}})^\top\big)^{-1}(\mathbf{M}_Y - \mathbf{M}_{\bar{Y}}).\end{aligned} \tag{28}$$

This helps derive the solution of the dual problem. For any $x' \in X$,

$$h(x') = C'\mathbf{K}'(\mathbf{I} + \bar{\mathbf{K}})^{-1}(\mathbf{M}_Y - \mathbf{M}_{\bar{Y}}), \tag{29}$$

where $\mathbf{K}' \in \mathbb{R}^{1\times m}$ is the row matrix whose ith component is $K(x', x_i) - \frac{1}{m}\sum_{j=1}^m K(x', x_j)$ and $\bar{\mathbf{K}}$ is the kernel matrix defined by

$$\begin{aligned}\tfrac{1}{C'}(\bar{\mathbf{K}})_{ij} = & K(x_i, x_j) - \frac{1}{m}\sum_{k=1}^m (K(x_i, x_k) + K(x_j, x_k))\\ & + \frac{1}{m^2}\sum_{k=1}^m\sum_{l=1}^m K(x_k, x_l),\end{aligned}$$

for all $i, j \in [1, m]$. The solution of the optimization problem for MPRank is close to that of a kernel ridge regression problem, but the presence of additional terms makes it distinct, a fact that can also be confirmed experimentally. However, remarkably, it has the same computational complexity, due to the fact that the optimization problem can be written in terms of a single sum, as already pointed out above. The main computational cost of the algorithm is that of the matrix inversion, which can be computed in time $O(N^3)$ in the primal, and $O(m^3)$ in the dual case, or $O(N^{2+\alpha})$ and $O(m^{2+\alpha})$, with $\alpha \approx .376$, using faster matrix inversion methods such as that of Coppersmith and Winograd.

3.2 SVRank

We will examine the algorithm in the case $n = 1$. As with MPRank, the hypothesis set H that we are considering here is that of linear functions h, that is $\forall x \in X, h(x) = w \cdot \Phi(x)$. The constraint optimization problem associated with SVRank can thus be rewritten as

$$\begin{aligned}&\text{minimize} \quad F(h, S) = \|w\|^2 + C\frac{1}{m^2}\sum_{i=1}^m\sum_{j=1}^m(\xi_{ij} + \xi_{ij}^*)\\ &\text{subject to} \begin{cases} w \cdot (\Phi(x_j) - \Phi(x_i)) - (y_j - y_i) \leq \epsilon + \xi_{ij}\\ (y_j - y_i) - w \cdot (\Phi(x_j) - \Phi(x_i)) \leq \epsilon + \xi_{ij}^*\\ \xi_{ij}, \xi_{ij}^* \geq 0,\end{cases}\end{aligned}$$

for all $i, j \in [1, m]$. Note that the number of constraints are quadratic with respect to the number of examples. Thus, in general, this results in a problem that is more costly to solve than that of MPRank.

Introducing Lagrange multipliers $\alpha_{ij}, \alpha_{ij}^* \geq 0$, corresponding to the first two sets of constraints and $\beta_{ij}, \beta_{ij}^* \geq 0$ for the remaining constraints leads to the following Lagrange function

$$\begin{aligned}&L = \|w\|^2 + C\tfrac{1}{m^2}\sum_{i=1}^m\sum_{j=1}^m(\xi_{ij} + \xi_{ij}^*) +\\ &\sum_{i=1}^m\sum_{j=1}^m \alpha_{ij}(w \cdot (\Phi(x_j) - \Phi(x_i)) - (y_j - y_i) - \epsilon + \xi_{ij}) +\\ &\sum_{i=1}^m\sum_{j=1}^m \alpha_{ij}^*(-w \cdot (\Phi(x_j) - \Phi(x_i)) + (y_j - y_i) - \epsilon + \xi_{ij}^*)\\ &+ \sum_{i=1}^m\sum_{j=1}^m(\beta_{ij}\xi_{ij} + \beta_{ij}^*\xi_{ij}^*).\end{aligned}$$

Table 1: Performance results for MPRank, SVRank, and RankBoost.

Data set	Mean Squared Difference			Mean 1-Norm Difference		
	MPRank	SVRank	RBoost	MPRank	SVRank	RBoost
MovieLens 20-40	2.01 ± 0.02	2.43 ± 0.13	12.88 ± 2.15	1.04 ± 0.05	1.17 ± 0.03	2.59 ± 0.04
MovieLens 40-60	2.02 ± 0.06	2.36 ± 0.16	20.06 ± 2.76	1.04 ± 0.02	1.15 ± 0.07	2.99 ± 0.12
MovieLens 60-80	2.07 ± 0.05	2.66 ± 0.09	21.35 ± 2.71	1.06 ± 0.01	1.24 ± 0.02	3.82 ± 0.23
Jester 20-40	51.34 ± 2.90	55.00 ± 5.14	77.08 ± 17.1	5.08 ± 0.15	5.40 ± 0.20	5.97 ± 0.16
Jester 40-60	46.77 ± 2.03	57.75 ± 5.14	80.00 ± 18.2	4.98 ± 0.13	5.27 ± 0.20	6.18 ± 0.11
Jester 60-80	49.33 ± 3.11	56.06 ± 4.26	88.61 ± 18.6	4.88 ± 0.14	5.25 ± 0.19	6.46 ± 0.20
Books	4.00 ± 3.12	3.64 ± 3.04	7.58 ± 9.95	1.38 ± 0.60	1.32 ± 0.56	1.72 ± 1.05

Taking the gradients, setting them to zero, and applying the Karush-Kuhn-Tucker conditions leads to the following dual maximization problem

$$\begin{aligned} \text{maximize } & \frac{1}{2}\sum_{i,j=1}^{m}\sum_{k,l=1}^{m}(\alpha_{ij}^{*}-\alpha_{ij})(\alpha_{kl}^{*}-\alpha_{kl})K_{ij,kl}- \\ & \epsilon\sum_{i,j=1}^{m}(\alpha_{ij}^{*}+\alpha_{ij})+\sum_{i,j=1}^{m}(\alpha_{ij}^{*}-\alpha_{ij})(y_j-y_i) \\ \text{subject to } & 0\leq\alpha_{ij},\alpha_{ij}^{*}\leq C,\forall i,j\in[1,m], \end{aligned}$$

where $K_{ij,kl}=K(x_i,x_k)+K(x_j,x_l)-K(x_i,x_l)-K(x_j,x_k)$. This quadratic optimization problem can be solved in a way similar to SVM regression (SVR) Vapnik [1998] by defining a kernel K' over pairs with $K'((x_i,x_j),(x_k,x_l))=K_{ij,kl}$, for all $i,j,k,l\in[1,m]$, and associating the target value y_i-y_j to the pair (x_i,x_j).

The computational complexity of the quadratic programming with respect to pairs makes this algorithm less attractive for relatively large samples.

4 Experiments

In this section, we report the results of experiments with two of our magnitude-preserving algorithms, MPRank and SVRank.

The algorithms were tested on three publicly available data sets commonly used for collaborative filtering: MovieLens, Book-Crossings, and Jester Joke. All datasets are available from the following URL: `http://www.grouplens.org/taxonomy/term/14`.

4.1 MovieLens Dataset

The MovieLens dataset consists of approximately 1M ratings by 6,040 users for 3,900 movies. Ratings are integers in the range of 1 to 5. For each user, a different predictive model is designed. The ratings of that

user on the 3,900 movies (not all movies will be rated) form the target values y_i. The other users' ratings of the ith movie form the ith input vector x_i.

We followed the experimental set-up of Freund et al. [1998] and grouped the reviewers according to the number of movies they have reviewed. The groupings were $20-40$ movies, $40-60$ movies, and $60-80$ movies.

Test reviewers were selected among users who had reviewed between 50 and 300 movies. For a given test reviewer, 300 reference reviewers were chosen at random from one of the three groups and their rating were used to form the input vectors. Training was carried out on half of the test reviewer's movie ratings and testing was performed on the other half. The experiment was done for 300 different test reviewers and the average performance recorded. The whole process was then repeated ten times with a different set of 300 reviewers selected at random. We report mean values and standard deviation for these ten repeated experiments for each of the three groups.

Missing review values in the input features were populated with the median review score of the given reference reviewer.

4.2 Jester Joke Dataset

The Jester Joke Recommender System dataset contains 4.1M continuous ratings in the range -10.00 to +10.00 of 100 jokes from 73,496 users. The experiments were set-up in the same way as for the MovieLens dataset.

4.3 Book-Crossing Dataset

The book-crossing dataset contains 278,858 users and 1,149,780 ratings for 271,379 books. The low density of ratings makes predictions very noisy in this task. Thus, we required users to have reviewed at least 200 books, and then only kept books with at least 10 reviews. This left us with a dataset of 89 books and 131 reviewers. For this dataset, each of the 131 reviewers was in turn selected as a test reviewer, and the other 130 reviewers served as input features. The results reported are mean values and standard deviations over these 131 leave-one-out experiments.

4.4 Performance Measures and Results

The performance measures we report correspond to the problem we are solving. The cost function of MPRank is designed to minimize the squared difference between all pairs of target values, hence we report the mean squared difference (MSD) over all pairs in the test set of size m' of a hypothesis h:

$$\frac{1}{m'^2}\sum_{i=1}^{m'}\sum_{j=1}^{m'}\left((h(x_j)-h(x_i))-(y_j-y_i)\right)^2. \tag{30}$$

The cost function of SVRank minimizes the absolute value of the difference between all pairs of examples, hence we report the average of the 1-norm difference, M1D:

$$\frac{1}{m'^2}\sum_{i=1}^{m'}\sum_{j=1}^{m'}\left|(h(x_j)-h(x_i))-(y_j-y_i)\right|. \tag{31}$$

The results for MPRank and SVRank are obtained using Gaussian kernels. The width of the kernel and the other cost function parameters were first optimized on a held-out sample. The performance on their respective cost functions was optimized and the parameters fixed at these values.

The results are reported in Table 1. They demonstrate that the magnitude-preserving algorithms are both successful at minimizing their respective objective. MPRank obtains the best MSD values and the two algorithms obtain comparable M1D values. However, overall, in view of these results and the superior

Table 2: Comparison of MPRank and RankBoost for pairwise misrankings.

Data set	Pairwise Misrankings	
	MPRank	RBoost
MovieLens 40-60	0.471 ± 0.005	0.476 0 ± 0.007
MovieLens 60-80	0.442 ± 0.005	0.463 ± 0.011
Jester 20-40	0.414 ± 0.005	0.479 ± 0.008
Jester 40-60	0.418 ± 0.007	0.432 ± 0.005
Jester 60-80	0.400 ± 0.017	0.417 ± 0.008

computational efficiency of MPRank already pointed out in the previous section, we consider MPRank as the best performing algorithm for such tasks.

To further examine the ranking properties of MPRank we conducted a number of experiments where we compared the pairwise misranking performance of the algorithm to that of RankBoost, an algorithm designed to minimize the number of pairwise misrankings Rudin et al. [2005]. We used the same features for RankBoost as for MPRank that is we used as weak rankers threshold functions over other reviewers' ratings. As for the other algorithms, the parameter of RankBoost, that is the number of boosting rounds required to minimize pairwise misranking was determined on a held-out sample and then fixed at this value.

Table 2 shows a comparison between these two algorithms. It reports the fraction of pairwise misrankings for both algorithms using the same experimental set-up as previously described:

$$\frac{\sum_{i,j=1}^{m'} 1_{y_i > y_j \wedge h(x_i) \leq h(x_j)}}{\sum_{i,j=1}^{m'} 1_{y_i > y_j}}. \tag{32}$$

The results show that the pairwise misranking error of MPRank is comparable to that of RankBoost. This further increases the benefits of MPRank as a ranking algorithm.

We also tested the performance of RankBoost with respect to MSD and M1D (see Table 1). RankBoost is not designed to optimize these performance measure and does not lead to competitive results with respect to MPRank and SVRank on any of the datasets examined.

5 Conclusion

We presented several algorithms for magnitude-preserving ranking problems and provided stability bounds for their generalization error. We also reported the results of several experiments on public datasets comparing these algorithms. We view accurate magnitude-preserving ranking as an important problem for improving the quality of modern recommendation and rating systems. An alternative for incorporating the magnitude of preferences in cost functions is to use weighted AUC, where the weights reflect the magnitude of preferences and extend existing algorithms. This however, does not exactly coincide with the objective of preserving the magnitude of preferences.

References

Shivani Agarwal and Partha Niyogi. Stability and generalization of bipartite ranking algorithms. In *COLT*, 2005.

Olivier Bousquet and André Elisseeff. Stability and generalization. *J. Mach. Learn. Res.*, 2:499–526, 2002. ISSN 1533-7928.

Olivier Bousquet and Andre Elisseeff. Algorithmic stability and generalization performance. In *NIPS*, pages 196–202, 2000. URL `citeseer.ist.psu.edu/article/bousquet00algorithmic.html`.

Wei Chu and S. Sathiya Keerthi. New approaches to support vector ordinal regression. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pages 145–152, New York, NY, USA, 2005. ACM Press. ISBN 1-59593-180-5. doi: http://doi.acm.org/10.1145/1102351.1102370.

Corinna Cortes and Mehryar Mohri. AUC Optimization vs. Error Rate Minimization. In *Advances in Neural Information Processing Systems (NIPS 2003)*, volume 16, Vancouver, Canada, 2004. MIT Press. URL `http://www.cs.nyu.edu/\~{}mohri/postscript/auc.pdf`.

K. Crammer and Y. Singer. Pranking with ranking. In *Proceedings of the conference on Neural Information Processing Systems (NIPS)*, 2001. URL `citeseer.ist.psu.edu/crammer01pranking.html`.

Yoav Freund, Raj Iyer, Robert E. Schapire, and Yoram Singer. An efficient boosting algorithm for combining preferences. In Jude W. Shavlik, editor, *Proceedings of ICML-98, 15th International Conference on Machine Learning*, pages 170–178, Madison, US, 1998. Morgan Kaufmann Publishers, San Francisco, US. URL `citeseer.ist.psu.edu/article/freund98efficient.html`.

Ralf Herbrich, Thore Graepel, and Klaus Obermayer. Large margin rank boundaries for ordinal regression. In Smola, Bartlett, Schoelkopf, and Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 115–132. MIT Press, Cambridge, MA, 2000.

T. Joachims. Evaluating retrieval performance using clickthrough data, 2002. URL `citeseer.ist.psu.edu/article/joachims02evaluating.html`.

P. McCullagh. Regression models for ordinal data. *Journal of the Royal Statistical Society B*, 42(2), 1980.

P. McCullagh and J. A. Nelder. *Generalized Linear Models*. Chapman & Hall, London, 1983.

Colin McDiarmid. Concentration. In *Probabilistic Methods for Algorithmic Discrete Mathematics*, pages 195–248, 1998.

Cynthia Rudin, Corinna Cortes, Mehryar Mohri, and Robert E. Schapire. Margin-Based Ranking Meets Boosting in the Middle. In *Proceedings of COLT 2005*, volume 3359 of *Lecture Notes in Computer Science*, pages 63–78. Springer, Heidelberg, Germany, June 2005.

A. Shashua and A. Levin. Ranking with large margin principle: Two approaches. In *Proceedings of the conference on Neural Information Processing Systems (NIPS)*, 2003. URL `citeseer.ist.psu.edu/article/shashua03ranking.html`.

Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, New York, 1998.