Mach Learn (2008) 72: 139–153
DOI 10.1007/s10994-008-5058-6

Robust reductions from ranking to classification

Maria-Florina Balcan · Nikhil Bansal · Alina Beygelzimer · Don Coppersmith · John Langford · Gregory B. Sorkin

Received: 5 September 2007 / Revised: 10 April 2008 / Accepted: 15 April 2008 /
Published online: 26 April 2008
Springer Science+Business Media, LLC 2008

Abstract We reduce ranking, as measured by the Area Under the Receiver Operating Characteristic Curve (AUC), to binary classification. The core theorem shows that a binary classification regret of r on the induced binary problem implies an AUC regret of at most $2r$. This is a large improvement over approaches such as ordering according to regressed scores, which have a regret transform of $r \mapsto nr$ where n is the number of elements.

Keywords Ranking · Classification · Reductions

Editors: Claudio Gentile, Nader H. Bshouty.

M.-F. Balcan
Carnegie Melon University, Pittsburgh, PA, USA
e-mail: ninamf@cs.cmu.edu

N. Bansal · G.B. Sorkin
IBM Thomas J. Watson Research Center, Yorktown Heights, NY, USA

N. Bansal
e-mail: bansal@us.ibm.com

G.B. Sorkin
e-mail: sorkin@us.ibm.com

A. Beygelzimer (✉)
IBM Thomas J. Watson Research Center, Hawthorne, NY, USA
e-mail: beygel@us.ibm.com

D. Coppersmith
IDA Center for Communications Research, Princeton, NJ, USA
e-mail: dcopper@idaccr.org

J. Langford
Yahoo Research, New York, NY, USA
e-mail: jl@yahoo-inc.com

1 Introduction

We consider the problem of ranking a set of instances. In the most basic version, we are given a set of unlabeled instances belonging to two classes, 0 and 1, and the goal is to rank all instances from class 0 before any instance from class 1. A common measure of success for a ranking algorithm is the area under the ROC curve (AUC). The associated loss, $1 - \text{AUC}$, measures how many pairs of neighboring instances would have to be swapped to repair the ranking, normalized by the number of 0s times the number of 1s. The loss is zero precisely when all 0s precede all 1s; one when all 1s precede all 0s. It is greater for mistakes at the beginning and the end of an ordering, which satisfies the intuition that an unwanted item placed at the top of a recommendation list should have a higher associated loss than when placed in the middle.

In binary classification, where the problem is simply to predict whether a label is 0 or 1, the loss is measured by the error rate, i.e., the probability of a misclassification. Here any misclassified instance incurs the same loss independently of how other instances are classified, while the AUC loss depends on the whole (ranked) sequence of instances.

It is natural to ask whether we need fundamentally different algorithms to optimize these two loss functions. This paper shows that the answer is no, in the sense that the problem of optimizing the AUC can be reduced to classification so that good performance on the classification problem implies good performance on the AUC problem. The classification problem is to predict, given a random pair of instances with different labels, whether the first instance should be ordered before the second. We show that there is a robust mechanism for translating any binary classifier learning algorithm into a ranking algorithm.

Relation to score-based ranking A common way to generate a ranking is to learn a *scoring function* $f : X \to \mathbb{R}$, which assigns a real-valued score to each example in the instance space X, and then rank test examples in the order of their scores.

If f is learned by minimizing some loss function that depends on individual examples, this approach is not robust. The fundamental difficulty is exhibited by highly unbalanced test sets. If we have one 1 and many 0s, a pointwise loss on the 1 with a perfect prediction for the 0s can greatly harm the AUC while only slightly affecting the pointwise loss with respect to the induced distribution. For concreteness, let $f(x)$ be 1 if the predicted label is 1, and 0 otherwise. Then if n is the number of elements in the test set, f can induce an AUC loss of 1 with classification loss of $1/n$. Thus such schemes transform pointwise loss l to AUC loss nl. The same observation holds for *regrets* in place of losses: pointwise regret r translates into AUC regret nr. *Regret* here is the difference between the incurred loss and the lowest achievable loss on the problem. The motivation for regret analysis is to separate avoidable loss from noise intrinsic to the problem, so that bounds apply nontrivially even for problems with large intrinsic noise.

To avoid the robustness problem, many score-based ranking algorithms optimize loss functions that depend on pairs of examples, e.g., the probability that a pair of examples with different labels is misranked (see Agarwal et al. 2005; Clémençon et al. 2005; Freund et al. 2003; Rudin et al. 2005).

Another approach, taken in (Cohen et al. 1999; Ailon and Mohri 2007) as well as here, is to learn a *preference function* $c : X \times X \to \{0, 1\}$ on pairs of examples. If $c(x, x') = 1$ then c ranks x higher than x', and $c(x, x') = 0$ indicates the opposite preference. Notice that while a scoring function imposes a total order on the entire instance space, a preference function is not required to be transitive. If, for example, the underlying distribution is such that an optimal preference function is non-transitive, then not imposing a total order on the entire space may result in a better performance on limited subsets on which the algorithm is invoked. Since c is not necessarily consistent with a linear ordering on a test set U, there is a need for a subsequent step which produces an ordering of U based on the learned c. Because c is typically learned using a binary classification algorithm, we call it a classifier.

Relation to the feedback arc set problem Consider a set U with a hidden bipartition into a set of *winners* and a set of *losers*. Imagine that every element (also called player) of U plays all other elements, and the outcome of each play is determined by a classifier c. The tournament induced by c on U does not have to be consistent with any linear ordering. We want to find the best way to rank the elements in U so that all winners are ordered before all losers.

A natural objective, dating back to Slater (1961), is to find an ordering which agrees with the tournament on as many player pairs as possible, i.e., minimizes the number of inconsistent pairs where a higher-ranked player (one ordered closer to the beginning of the list) lost to a lower-ranked player. This is the NP-hard "minimum feedback arc set problem in tournaments". (Although the hardness was conjectured for a long time, it was proved only recently; see (Alon 2006).)

A *mistake* is defined as a winner–loser pair where the loser beats (i.e., is preferred to) the winner. Section 4 proves that a solution to the feedback arc set problem satisfies a basic guarantee: If the classifier c makes at most k mistakes on U, then the algorithm minimizing the number of inconsistent pairs produces an ordering, or equivalently a transitive tournament, with at most $2k$ mistakes on U. Section 5 shows that this bound is tight.

Instead of solving feedback arc set, another natural way to break cycles is to rank instances according to their number of wins in the tournament produced by c. The way ties are broken is inessential; for definiteness, let us say they are broken against us. Coppersmith et al. (2006) proved that this algorithm provides a 5-approximation for the feedback arc set problem. An approximation, however, does not generally imply that the ratio of the number of mistakes made by the approximation to the number of mistakes made by c is finite. For example, c may make no mistakes (i.e., make correct predictions on all winner–loser pairs) while inducing a non-transitive tournament on the winners or the losers, so an approximation that does not know the labeling can incur a non-zero number of mistakes. We prove, however, that the algorithm that orders the elements by their number of wins, has the same regret and loss transforms as an optimal solution to the NP-hard feedback arc set problem. Again, Sect. 5 shows that solving feedback arc set does no better.

Results The core theorem (Theorem 1) says that a pairwise classifier with regret r implies AUC regret at most $2r$, for arbitrary distributions over instances. For example, if the binary error rate is 20% due to inherent noise and 5% due to the errors made by the classifier, then AUC regret is at most 10%, i.e., only the 5% are at most doubled. The same statement holds for losses in place of regrets. As shown in (Ailon and Mohri 2007) (see Sect. 5), this is best possible with any deterministic algorithm. Section 6 proves that Theorem 1 holds for a natural generalization of AUC regret to multiple labels.

In a subsequent paper, Ailon and Mohri (2007) describe a randomized quick-sort reduction, which guarantees that AUC loss is bounded by binary loss, in expectation over the randomness of the algorithm. They also define a generalization of AUC loss to multiple labels, and show that the expected generalized AUC loss is bounded by twice the binary loss. This generalization encodes the generalization we analyze in Sect. 6 as a special case. Using the decomposition in Theorem 1, the argument in (Ailon and Mohri 2007) could potentially be extended to bound AUC regret, rather than loss. The quick-sort algorithm is more efficient, requiring only $O(n \log n)$ instead of $\Theta(n^2)$ classifier evaluations at test time, which makes it practical in larger settings. On the other hand, the guarantees in (Ailon and Mohri 2007) are in expectation over the algorithm's randomness.

Cohen et al. (1999) operate in a similar two-stage setting: They first learn a preference function that takes a pair of instances and returns a score predicting how certain it is that the first instance should be ranked before the second. The learned function is then evaluated on all pairs of instances in the test set and the instances are ordered using the degree-based algorithm used here. One of the results they show is that the agreement of an optimal feedback arc set ordering with the learned predictions is at most twice the agreement obtained by their algorithm. To translate this result into the language of losses, let MFA be the AUC loss of the minimum feedback arc set ordering and APPROX be the AUC loss of the approximation. Then the result says that $1 - \text{APPROX} \geq \frac{1}{2}(1 - \text{MFA})$ or $\text{APPROX} \leq \frac{1}{2} + \text{MFA}/2$. The result is difficult to compare with the results given here, as the settings are different. A rough comparison requires specializations and yields a bound that is weaker than ours: As we show in Sect. 4, $\text{MFA} \leq 2\,\text{BIN}$, where BIN is the loss of the pairwise predictor, so the result of (Cohen et al. 1999) roughly says that $\text{APPROX} \leq \frac{1}{2} + \text{BIN}$ (which is essentially vacuous because a random ordering has expected AUC loss of 1/2), while we show that $\text{APPROX} \leq 2\,\text{BIN}$, modulo the slight difference in the binary problem.

Curiously, the relationship of ranking to classification is functionally tighter than has been proven for regression to binary classification ($r \mapsto \sqrt{r}$) (Langford and Zadrozny 2005).

2 Preliminaries

Classification A *binary classification problem* is defined by a distribution P over $X \times \{0, 1\}$, where X is some observable feature space and $\{0, 1\}$ is the binary label space. (For simplicity of presentation, assume that the spaces are finite.) The goal is to find a classifier $c : X \to \{0, 1\}$ minimizing the *classification loss* on P given by

$$e(c, P) = \mathbf{Pr}_{(x,y)\sim P}[c(x) \neq y].$$

The *classification regret* of c on P is defined as

$$r(c, P) = e(c, P) - \min_{c^*} e(c^*, P),$$

where the minimum is taken over all classifiers $c^* : X \to \{0, 1\}$.

Ranking Where $U \subseteq X$ is an unlabeled set, and $x, x' \in U$, a *preference function* $\pi(x, x', U) = 1$ if π "prefers" x to x', and 0 otherwise. For simplicity, let $\pi(x, x', U) = 1 - \pi(x', x, U)$ for $x \neq x'$. If π is consistent with some linear ordering of U, we call π itself an *ordering* of U, and denote $\pi_U(x, x') = \pi(x, x', U)$. For a labeled set $S = \{(x_1, y_1), \dots, (x_n, y_n)\}$, let $U_S = \{x_1, \dots, x_n\}$ denote the unlabeled set corresponding to S.

The *AUC loss* of an ordering π_{U_S} on a set $S = \{(x_1, y_1), \dots, (x_n, y_n)\}$ is defined as

$$l(\pi_{U_S}, S) = \frac{\sum_{i \neq j} \mathbf{1}(y_i > y_j)\pi_{U_S}(x_i, x_j)}{\sum_{i<j} \mathbf{1}(y_i \neq y_j)}.$$

Indices i and j in the summations range from 1 to n, and $\mathbf{1}(\cdot)$ is the indicator function which is 1 if its argument is true, and 0 otherwise. By convention, 0s should be ordered ahead of 1s, so any pair where a 1 is ordered before a 0 contributes to the loss.

A pair of examples $(x_1, y_1), (x_2, y_2)$ is called *mixed* if $y_1 \neq y_2$.

An *AUC problem* is defined by a distribution D over $(X \times \{0, 1\})^*$. The goal is to find an ordering π minimizing the expected AUC loss on D, given by

$$l(\pi, D) = \mathbf{E}_{S\sim D} l(\pi_{U_S}, S).$$

As an example, consider the Internet search problem, where there is some underlying distribution of queries, each yielding a set of search results. This process generates a distribution over subsets; whether or not the subsets have the same size is inessential for the analysis. Note that D is allowed to encode arbitrary dependencies between examples.

The *AUC regret* of π on D is given by $r_{\mathrm{AUC}}(\pi, D) = l(\pi, D) - \min_{\pi^*} l(\pi^*, D)$, where the minimum is taken over all preference functions π^* transitive on any subset in the support of D.

Tournaments A *tournament* is a complete graph with no self-loops, in which each edge is directed one way or the other, so that for every pair of vertices $i \neq j$, either $i \to j$ is an edge or $j \to i$ is an edge, but not both. The edge $i \to j$ says that i *beats* j ("i is preferred to j"). Since we adopt the convention that 0s should be ordered ahead of 1s, ideally 0s should beat 1s. We write $\deg(i)$ for the *outdegree* of vertex i, so $\deg(i) = \sum_j \mathbf{1}(i \to j)$, where the indicator function $\mathbf{1}(i \to j)$ is 1 if $i \to j$ is an edge and 0 otherwise. Thus we generally expect 0s to have large outdegree and 1s small outdegree; however, we allow and analyze arbitrary tournaments.

3 Ordering by the number of wins

In this section, we describe the reduction and prove the main result.

The reduction consists of two components. The training part, AUC-TRAIN (Algorithm 1), takes a set S of labeled examples of type $X \times \{0, 1\}$ and transforms all mixed pairs in S into binary examples for the oracle learning algorithm. The binary classification problem induced by the reduction is to predict, given a random mixed pair of examples in a subset $U \subseteq X$ together with U itself, whether the first example should be ordered before the second in U.

The reason why the classifier has to depend on U is that a regret reduction cannot throw away any information; otherwise the adversary could encode conflicting pairwise preferences based on the value of the subset. (An alternative is to introduce an assumption that the optimal ordering of any pair of unlabeled instances is independent of the drawn subset, whenever the instances are drawn together (Ailon and Mohri 2007).)

For any process D generating datasets S, we can define the induced distribution by first drawing S from D, then drawing a random mixed pair (x_1, y_1), (x_2, y_2) from S, and generating $(\langle x_1, x_2, U_S\rangle, \mathbf{1}(y_1 < y_2))$. We denote this induced distribution by AUC-TRAIN(D), admittedly overloading the notation.

The test part, DEGREE (Algorithm 2), uses the pairwise classifier c learned in Algorithm 1 to run a tournament on a test set U, and then ranks the elements of U in decreasing order of their number of wins in the tournament, breaking ties arbitrarily. Recall that we expect 0s to beat 1s, and thus have larger outdegree. (In the algorithm and below, we omit angle brackets from calls to c.)

For the analysis, it is best to think of the classifier c as an adversary playing against the ranking algorithm DEGREE. The goal of c is to pay little in classification regret while making DEGREE$(\cdot, c)$ pay a lot in AUC regret.

Algorithm 1 AUC-TRAIN (labeled set S, binary learning algorithm A)

```
1. Define the (multi)set $U_S = \{x : (x, y) \in S\}$.
2. Let $S' = \{(\langle x_1, x_2, U_S\rangle, \mathbf{1}(y_1 < y_2)) : (x_1, y_1), (x_2, y_2) \in S$ and $y_1 \neq y_2\}$.
3. return $c = A(S')$.
```

Algorithm 2 DEGREE (unlabeled set U, classifier c)

```
1. For $x \in U$, let $\deg(x) = |\{x' : c(x, x', U) = 1, x' \in U\}|$.
2. Sort $U$ in descending order of $\deg(x)$, breaking ties arbitrarily.
```

Theorem 1 below reduces the regret problem to the following combinatorial problem. Given a set U with each element labeled either 0 or 1, the adversary c starts with a tournament of its choice where every 0 beats every 1. Then c can choose to invert the outcome of any game between a 0 and a 1, and she is charged for each such "mistake". Again, c can choose any (not necessarily transitive) subtournaments on the 0s and on the 1s for free. The resulting tournament is shown to the algorithm.

Without seeing the labels, the algorithm needs to approximate c's tournament with a transitive tournament (or equivalently, a linear order). The goal of the algorithm is to minimize the number of mistakes it makes (i.e., pairs where a 1 precedes a 0 in the order). If c were itself consistent with a linear order, the algorithm could simply output that, at a cost in mistakes identical to the adversary's. In general it is not, and we would expect the cost of the more-constrained linear order to be higher. Our goal is to show that the reduction is robust in the sense that c cannot cause DEGREE to make many mistakes without making many mistakes itself. More precisely, DEGREE never makes more than twice as many mistakes as c. This combinatorial result (Theorem 2) is invoked $(n-1)$ times in the proof of the main theorem below.

Theorem 1 *For all distributions D and all pairwise classifiers c,*

$$r_{\mathrm{AUC}}(\textsc{Degree}(\cdot, c), D) \le 2r(c, \textsc{Auc-Train}(D)). \tag{1}$$

Note the quantification in the above theorem: it applies to *all* settings where Algorithms 1 and 2 are used; in particular, D may encode arbitrary dependences between examples.

Proof Given an unlabeled test set $U \in X^n$, the joint distribution D induces a conditional distribution $D(Y_1, \ldots, Y_n|U)$ over the set of label sequences $\{0,1\}^n$. In the remainder of the paper, let Q denote this conditional distribution. We prove the theorem by fixing U, taking the expectation over the draw of U at the end.

Let U be identified with $\{1, \ldots, n\}$. The *pairwise loss* of ordering i before j in U is defined as

$$l_Q(i,j) = \mathbf{E}_{y^n \sim Q} \frac{\mathbf{1}(y_i > y_j)}{\sum_{u<v} \mathbf{1}(y_u \ne y_v)}.$$

If $l_Q(i,j) < l_Q(j,i)$, the *regret* $r_Q(i,j)$ of ordering i before j is 0; otherwise, $r_Q(i,j) = l_Q(i,j) - l_Q(j,i)$ and $r_Q(i,j)$ is called *proper*. Thus if $r_Q(i,j)$ is proper, then $r_Q(j,i) = 0$.

Lemma 1 presented below this proof, establishes a basic property of pairwise regrets. It says that for any i, j, and k, if $r_Q(i,j)$ and $r_Q(j,k)$ are proper, $r_Q(i,k) = r_Q(i,j) + r_Q(j,k)$.

We can assume without loss of generality that the ordering minimizing the AUC loss (thus having zero AUC regret) on U is $\langle 1, 2, \ldots, n \rangle$. It is easy to see that all regret-zero pairwise predictions must be consistent with the ordering, i.e., $r_Q(i,j) = 0$ for all $1 \le i < j \le n$. Indeed, all $r_Q(i, i+1)$ must be 0; otherwise swapping i and $i+1$ decreases the overall regret, contradicting the assumption that $\langle 1, 2, \ldots, n \rangle$ is regret minimizing. Thus all $r_Q(i+1, i)$ are proper and repeated application of Lemma 1 implies that, for any $j > i$, $r_Q(j,i)$ is proper, which in turn implies that $r_Q(i,j) = 0$.

Applied repeatedly, Lemma 1 says that for any pair $i < j$, the regret $r_Q(j,i)$ can be decomposed as $r_Q(j,i) = \sum_{k=i}^{j-1} r_Q(k+1,k)$. Since U is fixed throughout the proof, we omit the subscript U from the produced ordering π_U and the optimal ordering π_U^*. Thus the AUC regret of $\pi = \pi_U$ on Q can be decomposed as a sum of pairwise regrets (where $\langle U, y^n\rangle$ denotes the unlabeled sample U labeled with y^n):

$$\begin{aligned}
r_{\mathrm{AUC}}(\pi, Q) &= l(\pi, Q) - \min_{\pi^*} l(\pi^*, Q)\\
&= \mathbf{E}_{y^n\sim Q}[l(\pi, \langle U, y^n\rangle)] - \min_{\pi^*} \mathbf{E}_{y^n\sim Q}[l(\pi^*, \langle U, y^n\rangle)]\\
&= \mathbf{E}_{y^n\sim Q} \frac{\sum_{i,j} \mathbf{1}(y_i > y_j)\pi(i,j)}{\sum_{u<v} \mathbf{1}(y_u \neq y_v)} - \min_{\pi^*} \mathbf{E}_{y^n\sim Q} \frac{\sum_{i,j} \mathbf{1}(y_i > y_j)\pi^*(i,j)}{\sum_{u<v} \mathbf{1}(y_u \neq y_v)}\\
&= \max_{\pi^*} \mathbf{E}_{y^n\sim Q} \frac{\sum_{i,j}[\mathbf{1}(y_i > y_j)\pi(i,j) - \mathbf{1}(y_i > y_j)\pi^*(i,j)]}{\sum_{u<v} \mathbf{1}(y_u \neq y_v)}\\
&= \sum_{i<j:\pi(j,i)=1} r_Q(j,i) = \sum_{k=1}^{n-1} |\{i \le k < j : \pi(j,i) = 1\}| \cdot r_Q(k+1,k).
\end{aligned}$$

The last equality follows by repeated application of Lemma 1. Note that

$$\min_{\pi^*} \mathbf{E}_U[l(\pi_U^*, Q)] = \mathbf{E}_U[\min_{\pi_U^*} l(\pi_U^*, Q)],$$

because for any U, the minimizer π_U^* of $l(\pi_U^*, Q)$ is the minimizer π^* of $\mathbb{E}_U[l(\pi_U^*, Q)]$ (restricted to U in the last argument).

The classification regret can also be written in terms of pairwise regrets:

$$\begin{aligned}
&r(c, \text{AUC-TRAIN}(Q))\\
&\quad = e(c, \text{AUC-TRAIN}(Q)) - \min_{c^*} e(c^*, \text{AUC-TRAIN}(Q))\\
&\quad = \max_{c^*} \mathbf{E}_{y^n\sim Q}\left[\frac{\sum_{i,j}[\mathbf{1}(y_i > y_j)c(i,j,U) - \mathbf{1}(y_i > y_j)c^*(i,j,U)]}{\sum_{u<v} \mathbf{1}(y_u \neq y_v)}\right]\\
&\quad = \sum_{i<j:c(j,i,U)=1} r_Q(j,i) = \sum_{k=1}^{n-1} |\{i \le k < j : c(j,i,U) = 1\}| \cdot r_Q(k+1,k).
\end{aligned}$$

Let g_k and f_k denote the coefficients with which the term $r_Q(k+1,k)$ appears in the above decompositions of $r_{\mathrm{AUC}}(\pi, Q)$ and $r(c, \text{AUC-TRAIN}(Q))$ respectively. To complete the proof it suffices to show that $g_k \le 2f_k$ for each k.

Fix k and consider a bipartition of U into a set $\{1,\dots,k\}$ of "winners" and a set $\{k+1,\dots,n\}$ of "losers". In this terminology, g_k is the number of winner–loser pairs where the loser has at least as many wins as the winner, and f_k is the number of winner–loser pairs where the loser beats the winner (in the tournament induced by c on U). Theorem 2 below shows that $g_k \le 2f_k$, completing this proof. □

We prove the lemma used in the proof above.

Lemma 1 *For any i, j, and k in $\{1,\dots,n\}$, if $r_Q(i,j)$ and $r_Q(j,k)$ are proper,*

$$r_Q(i,k) = r_Q(i,j) + r_Q(j,k).$$

Proof We have

$$r_Q(i,j) + r_Q(j,k) = \mathbf{E}_{y^n \sim Q} \frac{I(y^n)}{\sum_{u<v} \mathbf{1}(y_u \neq y_v)},$$

where

$$\begin{aligned} I(y^n) &= \mathbf{1}(y_i = 1, y_j = 0) - \mathbf{1}(y_i = 0, y_j = 1) \\ &\quad + \mathbf{1}(y_j = 1, y_k = 0) - \mathbf{1}(y_j = 0, y_k = 1) \\ &= \mathbf{1}(y_i = 1, y_j = 0, y_k = 0) + \mathbf{1}(y_i = 1, y_j = 0, y_k = 1) \\ &\quad - \mathbf{1}(y_i = 0, y_j = 1, y_k = 0) - \mathbf{1}(y_i = 0, y_j = 1, y_k = 1) \\ &\quad + \mathbf{1}(y_i = 0, y_j = 1, y_k = 0) + \mathbf{1}(y_i = 1, y_j = 1, y_k = 0) \\ &\quad - \mathbf{1}(y_i = 0, y_j = 0, y_k = 1) - \mathbf{1}(y_i = 1, y_j = 0, y_k = 1) \\ &= \mathbf{1}(y_i = 1, y_k = 0) - \mathbf{1}(y_i = 0, y_k = 1). \end{aligned}$$

Recall that

$$\mathbf{E}_{y^n \sim Q} \frac{\mathbf{1}(y_i = 1, y_k = 0)}{\sum_{u<v} \mathbf{1}(y_u \neq y_v)} = l_Q(i,k), \qquad \mathbf{E}_{y^n \sim Q} \frac{\mathbf{1}(y_i = 0, y_k = 1)}{\sum_{u<v} \mathbf{1}(y_u \neq y_v)} = l_Q(k,i).$$

Thus $r_Q(i,j) + r_Q(j,k) = l_Q(i,k) - l_Q(k,i) = r_Q(i,k)$. (Since $r_Q(i,j) + r_Q(j,k) > 0$, we have $l_Q(i,k) > l_Q(k,i)$ and $r_Q(i,k)$ is proper.) □

Let T be a tournament and let the vertices of T be arbitrarily partitioned into a set W of "winners" and a set L of "losers". Call the triple (T, W, L) a winner–loser partitioned tournament, and denote it by $\mathbf{T}$. We show that for any $\mathbf{T}$, the number of winner–loser pairs where the loser's degree is larger than or equal to the winner's, is at most twice the number of winner–loser pairs where the loser beats the winner. Formally, define two measures:

$$g(\mathbf{T}) = \sum_{\ell \in L} \sum_{w \in W} \mathbf{1}(\deg(\ell) \geq \deg(w)),$$

$$f(\mathbf{T}) = \sum_{\ell \in L} \sum_{w \in W} \mathbf{1}(\ell \to w).$$

Theorem 2 *For any winner–loser partitioned tournament* $\mathbf{T}$, $g(\mathbf{T}) \leq 2f(\mathbf{T})$.

Since the number of edges from L to W is equal to the total number of edges out of L minus the number of edges from L to L, we can rewrite

$$f(\mathbf{T}) = \sum_{\ell \in L} \sum_{w \in W} \mathbf{1}(\ell \to w) = \sum_{\ell \in L} \deg(\ell) - \binom{|L|}{2}.$$

Both $f(\mathbf{T})$ and $g(\mathbf{T})$ depend only on the degrees of the vertices of T, so rather than working with a (labeled) tournament, a relatively complex object, we can work with a (labeled) degree sequence.

Landau's theorem (Landau 1953) says that there exists a tournament with outdegree sequence $d_1 \le d_2 \le \cdots \le d_n$ if and only if, for all $1 \le i \le n$, $\sum_{j=1}^{i} d_j \ge \sum_{j=1}^{i}(j-1)$, with equality for $i = n$.

Recall that a sequence $\langle a_1, \dots, a_n \rangle$ is *majorized* by $\langle b_1, \dots, b_n \rangle$ if the two sums are equal and if, when each sequence is sorted in non-increasing order, the prefix sums of the b sequence are at least as large as (dominate) those of the a sequence. (For a comprehensive treatment of majorization, see (Marshall and Olkin 1979).) Landau's condition is precisely that $\langle d_1, \dots, d_n \rangle$ is majorized by $\langle 0, \dots, n-1 \rangle$. (With the sequences sorted in increasing order, Landau's condition is that prefix sums of the degree sequence dominate those of the progression, which is the same as saying that the suffix sums of the degree sequence are dominated by the suffix sums of the progression.) This allows us to take advantage of well-known properties of majorization, notably that if A' is obtained by averaging together any elements of A, then A majorizes A'.

This allows us to restate Theorem 2 in terms of a sequence and majorization, rather than a tournament, but first we relax the constraints. First, where the original statement requires elements of the degree sequence to be non-negative integers, we allow them to be non-negative reals. Second, the original statement requires that we attach a winner/loser label to each element of the degree sequence. Instead, we aggregate equal elements of the degree sequence, and for a degree d_i of (integral) multiplicity m_i, assign arbitrary non-negative (but not necessarily integral) portions to winners and losers: $w_i + \ell_i = m_i$.

Let $\mathbf{D} = (D, W, L)$ be such a generalized "winner–loser labeled compressed sequence". Note that the majorization condition applies only to the values $\{d_i, m_i\}$, not the labeling. The definitions of f and g above are easily extended to this broader domain: $g(\mathbf{D}) = \sum_i \sum_{j \le i} l_i w_j$, $f(\mathbf{D}) = \sum_i l_i d_i - \binom{\sum_i l_i}{2}$, where we define $\binom{x}{2} = x(x-1)/2$ for all x (not just integers). If we prove $g \le 2f$ over this larger domain, the inequality holds in particular for plain winner–loser labeled degree sequences (the case where all weights happen to be integral). That is, Theorem 3, below, implies Theorem 2.

Theorem 3 *For any winner–loser labeled compressed sequence* $\mathbf{D} = (D, W, L)$ *where D is majorized by* $\langle 0, \dots, n-1 \rangle$, $g(\mathbf{D}) \le 2f(\mathbf{D})$.

Proof We begin with an outline of the proof. Define a compressed sequence $\mathbf{D}$ as being *canonical* if it consists of at most three degrees: a smallest degree d_1 having only losers ($w_1 = 0$), a middle degree d_2 potentially with both winners and losers ($w_2, \ell_2 \ge 0$), and a largest degree d_3 having only winners ($\ell_3 = 0$). We first establish that any canonical sequence has $g(\mathbf{D}) - 2f(\mathbf{D}) \le 0$. We then show how to transform *any* degree sequence to a canonical one with a larger (or equal) value of $g - 2f$, which completes the argument.

We first show that a canonical sequence $\mathbf{D}$ has $g - 2f \le 0$. For the canonical configuration, $g = w_2 \ell_2$ and $f = \ell_1 d_1 + \ell_2 d_2 - \binom{\ell_1 + \ell_2}{2}$, and hence our goal is to show that

$$\ell_1 d_1 + \ell_2 d_2 \ge (\ell_1 + \ell_2)(\ell_1 + \ell_2 - 1)/2 + w_2 \ell_2 / 2. \tag{2}$$

By Landau's condition applied to ℓ_1 and $\ell_1 + w_2 + \ell_2$, we have the following two relations:

$$\ell_1 d_1 \ge \binom{\ell_1}{2} \tag{3}$$

and

$$\ell_1 d_1 + (\ell_2 + w_2) d_2 \ge \binom{\ell_1 + w_2 + \ell_2}{2}. \tag{4}$$

Multiplying (3) by $w_2/(\ell_2+w_2)$ and (4) by $\ell_2/(\ell_2+w_2)$ and adding them, we obtain that

$$\ell_1 d_1 + \ell_2 d_2 \geq \frac{1}{\ell_2+w_2}\left(w_2\binom{\ell_1}{2} + \ell_2\binom{\ell_1+\ell_2+w_2}{2}\right). \tag{5}$$

A simple calculation shows that the right side of inequality (5) is exactly equal to the right hand side of (2). This proves that $g \leq 2f$ for a canonical sequence.

If a sequence is not canonical then there are two consecutive degrees d_i and d_j $(j=i+1)$ such that one of the cases 1a, 1b, or 2 (described below) holds. In each case we apply a transformation producing from the degree sequence $\mathbf{D}$ a new degree sequence $\mathbf{D}'$, where:

- the total weight of winners in $\mathbf{D}'$ is equal to that of $\mathbf{D}$; similarly for losers, and thus for the total weight; furthermore, the total weight on each degree remains integral;
- $\mathbf{D}'$ maintains the majorization needed for Landau's theorem;
- the value of $g-2f$ is at least as large for $\mathbf{D}'$ as for $\mathbf{D}$; and
- either the number of nonzero values w_i and ℓ_i or the number of distinct degrees d_i is strictly smaller for $\mathbf{D}'$ than for $\mathbf{D}$, and the other is no larger for $\mathbf{D}'$ than for $\mathbf{D}$.

We first sketch the cases and then detail the transformations.

Case 1a d_i has only winners ($l_i=0$).
Apply Transformation 1a, combining the two degrees into one.
Case 1b d_j has only losers ($w_j=0$).
Apply Transformation 1b, combining the two degrees into one.
Case 2 All of w_i, l_i, w_j and l_j are nonzero.
Apply Transformation 2, leaving the degrees the same but transforming the weights so that one of them is equal to 0 and one of the preceding cases applies, or the weights obey an equality allowing application of Transformation 3, which combines the two degrees into one.

Either there is some pair i,j to which one of the cases applies, or the sequence is canonical. We argue this by showing that if there is no pair i,j for which Cases 1a or 1b apply, then either the sequence is canonical, or there is a pair to which Case 2 applies. First, note that for any $i\neq n$, $l_i>0$ (else Case 1a applies to $i,i+1$) and for any $i\neq 1$, $w_i>0$ (else Case 1b applies to $i-1,i$). In particular, for any $1<i<n$, both $l_i, w_i>0$. If $n\geq 4$ this implies immediately that Case 2 applies to the pair 2, 3. If $n=1$, $\mathbf{D}$ is automatically canonical. If $n=2$ and $l_2=0$ or $w_1=0$ then $\mathbf{D}$ is canonical, while if both $l_2, w_1>0$ we may apply Case 2 (since, as we first argued, $l_1, w_2>0$). Finally, if $n=3$, we know $l_1,l_2,w_2,w_3>0$. If $w_1=l_3=0$ then $\mathbf{D}$ is canonical, and otherwise Case 2 applies.

Transformation 1a: In Case 1a, where d_i has only winners, change $\mathbf{D}$ to a new sequence $\mathbf{D}'$ by replacing the pair $(d_i,w_i,0)$, (d_j,w_j,l_j) by their "average": the single degree (d',w',l'), where

$$w'=w_i+w_j, \qquad l'=l_j, \qquad d'=\frac{w_i d_i+(w_j+l_j)d_j}{w_i+w_j+l_j}.$$

The stated conditions on a transformation are easily checked. The total weight of winners is clearly preserved, as is the total weight of losers and the total degree (out-edges). Summing weights preserves integrability. The number of distinct degrees is reduced by one, and the number of nonzero weights may be decreased by one or may remain unchanged. The Landau majorization condition holds because $\mathbf{D}'$, as an averaging of $\mathbf{D}$, is majorized by it, and majorization is transitive. The only non-trivial condition is the non-decrease in $g - 2f$. The number of loser–winner pairs where the loser outranks the winner remains the same, so $g(\mathbf{D}) = g(\mathbf{D}')$. Also, f depends only on the total weight of losers (which is unchanged) and on the average degree of losers. This average degree would be unchanged if w_i were 0; since $w_i \geq 0$, the average degree may decrease. Thus $f(\mathbf{D}) \geq f(\mathbf{D}')$, and $(g - 2f)(\mathbf{D}) \leq (g - 2f)(\mathbf{D}')$, as desired.

Transformation 1b: Symmetrically to Transformation 1a, obtain $\mathbf{D}'$ by replacing the pair of labeled weighted degrees (d_i, w_i, l_i) and $(d_j, 0, l_j)$ with a single one (d', w', l'), where $w' = w_i$, $l' = l_i + l_j$, and $d' = [(l_i + w_i)d_i + l_j d_j]/(l_i + w_i + l_j)$.

Transformation 2: Where w_i, l_i, w_j and l_j are all nonzero, we begin with one case, which leads to one other. In the usual case, we transform $\mathbf{D}$ to $\mathbf{D}'$ by replacing the pair (d_i, w_i, l_i), (d_j, w_j, l_j) with $(d_i, w_i + x, l_i - x)$, $(d_j, w_j - x, l_j + x)$, for some value of x (positive or negative) to be determined. This affects only the labeling, not the weighted degree sequence itself, and is therefore legitimate as long as the four quantities $w_i + x$, $l_i - x$, $w_j - x$ and $l_j + x$ are all non-negative.

Defining $\Delta = (g - 2f)(\mathbf{D}') - (g - 2f)(\mathbf{D})$, we wish to choose x to make $\Delta > 0$.

$$\begin{aligned}\Delta &= \Big\{\big[(l_j + x)(w_i + x + w_j - x) + (l_i - x)(w_i + x)\big] - \big[l_j(w_i + w_j) + l_i w_i\big]\Big\}\\ &\quad - 2\Big\{\big[(l_i - x)d_i + (l_j + x)d_j\big] - \big[l_i d_i + l_j d_j\big]\Big\}\\ &= x(w_j + l_i - 2(d_j - d_i) - x) = x(a - x),\end{aligned}$$

where $a = w_j + l_i - 2(d_j - d_i)$. This is a simple quadratic expression with negative coefficient on x^2, so its value increases monotonically as x is varied from 0 to $a/2$, where the maximum is obtained. (Note that a may be negative.) If $a = 0$ then we do not use this transformation but Transformation 3, below. Otherwise, vary x from 0 to $a/2$ stopping when x reaches $a/2$ or when any of $w_i + x$, $l_i - x$, $w_j - x$ and $l_j + x$ becomes 0. Call this value $x^\star$, and use it to define the transformation.

If any of $w_i + x$, $l_i - x$, $w_j - x$ and $l_j + x$ is 0 then the number of nonzero weights is decreased (while the number of distinct degrees is unchanged). Otherwise, $x^\star = a/2$. In that case, the new $\mathbf{D}'$ has $a = 0$ (the optimal "weight shift" has already been performed). With $a = 0$ we apply Transformation 3, which reduces the number of nonzero weights.

Transformation 3: Similar to Cases 1a and 1b, transform $\mathbf{D}$ to $\mathbf{D}'$ by replacing the pair (d_i, w_i, l_i), (d_j, w_j, l_j) with a single degree (d', w', l') that is their weighted average,

$$w' = w_i + w_j, \qquad l' = l_i + l_j, \qquad d' = \frac{(w_i + l_i)d_i + (w_j + l_j)d_j}{w_i + l_i + w_j + l_j}.$$

This gives

$$\begin{aligned}\Delta &= (g - 2f)(\mathbf{D}') - (g - 2f)(\mathbf{D}) = (l_i w_j) - 2(l_i d' + l_j d' - l_i d_i - l_j d_j)\\ &= l_i w_j + \frac{2(d_j - d_i)(w_i l_j - w_j l_i)}{w_i + l_i + w_j + l_j}.\end{aligned}$$

We apply this transformation only in the case where Transformation 2 fails to give any improvement because its "a" expression is equal to 0, i.e., $d_j - d_i = (w_j + l_i)/2$. Making the corresponding substitution gives

$$\begin{aligned}\Delta &= l_i w_j + \frac{(w_j + l_i)(w_i l_j - w_j l_i)}{w_i + l_i + w_j + l_j}\\ &= \frac{(l_i w_j)(l_j + w_i) + (l_j w_i)(l_i + w_j)}{w_i + l_i + w_j + l_j} > 0.\end{aligned}$$

This reduces the number of distinct degrees by one, without increasing the number of nonzero weights.

Concluding the argument, we have shown that any non-canonical configuration $\mathbf{D}$ can be replaced by a configuration with a strictly smaller total of distinct degrees and nonzero weights, and at least as large a value of $g - 2f$. Since $\mathbf{D}$ had at most n distinct degrees and $2n$ nonzero weights originally, a canonical configuration $\mathbf{D}^\star$ is reached after at most $3n - 1$ transformations. (All that is important is that the number of transformations is finite: that a canonical configuration is eventually reached.) Then, $(g-2f)(\mathbf{D}) \le (g-2f)(\mathbf{D}^\star) \le 0$. □

A further generalization of Theorem 3 may be found in (Bansal et al. 2006).

4 An upper bound for minimum feedback arc set

This section shows an analog of Theorem 2 for an optimal solution to the feedback arc set problem. (The decomposition argument in Theorem 1 is algorithm-independent and applies here as well.) For a tournament T and an ordering π, a *back edge* is an edge $i \to j$ in T such that j is ordered before i in π. Let $\mathrm{back}(T, \pi)$ denote the number of back edges induced by π in T.

For a winner–loser partitioned tournament $\mathbf{T} = (T, W, L)$ and any minimum feedback arc set ordering π of T, let $g'(\mathbf{T}, \pi)$ be the number of winner–loser pairs where the loser comes before the winner in π, and as before let

$$f(\mathbf{T}) = \sum_{\ell \in L} \sum_{w \in W} \mathbf{1}(\ell \to w)$$

be the number of winner–loser pairs where the loser beats the winner.

Theorem 4 *For any winner–loser partitioned tournament* $\mathbf{T} = (T, W, L)$ *and any minimum feedback arc set ordering* π *of* $\mathbf{T}$, $g'(\mathbf{T}, \pi) \le 2f(\mathbf{T})$.

Proof Let k_w be the smallest possible number of back edges in the subtournament induced by W. Define k_l similarly for the subtournament induced by L. Let k_w^π and k_l^π be the number of back edges in π that go from W to W and from L to L, respectively. Denote the number of remaining (i.e., winner–loser or loser–winner) back edges in π by k_o^π.

Consider another ordering σ where all winners are ordered before all losers, and both the winners and the losers are ordered optimally among themselves, i.e., with k_w and k_l back edges respectively. The number of back edges in σ is $\mathrm{back}(\mathbf{T}, \sigma) = k_w + k_l + f(\mathbf{T})$. But we also have $\mathrm{back}(\mathbf{T}, \sigma) \ge \mathrm{back}(\mathbf{T}, \pi)$ since π minimizes the number of back edges, and thus $k_w + k_l + f(\mathbf{T}) \ge k_w^\pi + k_l^\pi + k_o^\pi$. Since $k_w \le k_w^\pi$ and $k_l \le k_l^\pi$ by definition of k_w and k_l, we have $f(\mathbf{T}) \ge k_o^\pi$.

Consider any winner–loser pair with the loser ordered before the winner. If $w \to l$ is the edge, it is a back edge in π and thus is counted by k_o^π. If $l \to w$ is the edge instead, it is counted by $f(\mathbf{T})$. Thus $g'(\mathbf{T},\pi)$ is at most $k_o^\pi + f(\mathbf{T})$. Since $f(\mathbf{T}) \geq k_o^\pi$, this number is never more than $2f(\mathbf{T})$, which implies $g'(\mathbf{T},\pi) \leq 2f(\mathbf{T})$. □

The proof above immediately shows that for any $\mathbf{T}$ and any ordering π_α of $\mathbf{T}$ with at most $(1+\alpha)\cdot \mathrm{opt}(\mathbf{T})$ back edges, $g'(\mathbf{T},\pi_\alpha) \leq (2+\alpha)f(\mathbf{T}) + \alpha(k_w + k_l)$, where $\mathrm{opt}(\mathbf{T})$ is the minimum number of back edges in $\mathbf{T}$, and k_w and k_l are as in the proof above. Thus, a FAS approximation does not generally guarantee that if f is 0, then so is g' (a guarantee provided by DEGREE and FAS). For example, $\mathbf{T}$ may make correct statements about all winner–loser pairs while inducing a non-transitive tournament on the winners or the losers, so an approximation that does not know the winner–loser labeling can incur a non-zero number of mistakes.

5 Lower bounds

We first show that Theorem 2 is best possible: the DEGREE ranking really can make twice as many mistakes as the adversary. Recall that f denotes the number of winner–loser pairs where the loser beats the winner, and g the number of winner–loser pairs where the loser outranks the winner. The example below generates an infinite family of tournaments with $g = 2f$.

Example 1 With n odd, let every vertex have degree $(n-1)/2$; note that the degree sequence $\langle \frac{n-1}{2}, \ldots, \frac{n-1}{2} \rangle$ does indeed respect Landau's condition, so it is realizable as a tournament. Label $(n-1)/2$ of the vertices as winners and $(n+1)/2$ as losers. With ties broken against us, all winners are ordered after all losers. This gives $f = \frac{n+1}{2} \cdot \frac{n-1}{2} - \binom{(n+1)/2}{2} = (n+1)(n-1)/8$, while $g = \frac{n+1}{2} \cdot \frac{n-1}{2} = (n+1)(n-1)/4 = 2f$. (A similar example gives a lower bound of $2 - O(1/n)$ with ties broken optimally for the algorithm.)

In a subsequent paper, Ailon and Mohri (2007) give a simple algorithm-independent example showing that no deterministic algorithm can achieve a constant factor of less than 2 on the regret ratio. The example puts all the probability mass on a single three element subset. The induced tournament is a directed 3-cycle, and the bipartition is chosen adversarily depending on the ordering output by the algorithm. By symmetry, there are only two different orderings (clockwise and counterclockwise). In both cases, the adversary can make the algorithm pay for both mixed pairs while paying for only one misordering. This example implies that Theorem 4 is also best possible.

6 Generalization to multipartite ranking

The result in Sect. 3 can be extended to the case when examples belong to more than two classes. In the extreme case, all examples can have different labels. The labels typically form an ordered set, arising naturally in applications where labels represent discretized ratings (e.g., survey results can be graded from 'strongly agree' to 'strongly disagree', search results can be graded from 'most relevant' to 'least relevant').

A natural way of extending the definition of ranking loss to account for the order in the values, is to weigh the loss of putting an example with label y_j before an example with label

$y_i < y_j$ by the difference $y_j - y_i$. Formally, the generalized AUC loss of an ordering π on a set $S = (1, y_1), \ldots, (n, y_n)$ is defined as

$$l(\pi, S) = \frac{\sum_{i \neq j} \pi(i, j, \{1, \ldots, n\})(y_i - y_j)_+}{\sum_{i<j} |y_i - y_j|},$$

where we use the operator $(A)_+ = A \cdot \mathbf{1}(A > 0)$.

As in the bipartite case, let Q denote the distribution of label sequences of the set $\{1, \ldots, n\}$. Then the generalized loss of ordering i before j is

$$l_Q(i, j) = \mathbf{E}_{y^n \sim Q} \frac{(y_i - y_j)_+}{\sum_{u<v} |y_u - y_v|}.$$

Regrets are defined as before. Recall that $r_Q(i, j)$ is *proper* if $l_Q(i, j) - l_Q(j, i) \geq 0$.

The lemma below extends Lemma 1 to multiple labels.

Lemma 2 *For any i, j, and k in $\{1, \ldots, n\}$, if $r_Q(i, j)$ and $r_Q(j, k)$ are proper,*

$$r_Q(i, j) + r_Q(j, k) = r_Q(i, k).$$

Proof Consider any label assignment $y^n = y_1 \ldots y_n$ and let $Z(y^n) = \frac{Q(y^n)}{\sum_{u<v} |y_u - y_v|}$.

A simple case analysis shows that y^n contributes equally to both sides of the lemma statement. Indeed, if $y_i \leq y_j \leq y_k$, the contribution is zero to either side. If $y_i \geq y_j \geq y_k$, it contributes $Z(y^n)((y_i - y_j) + (y_j - y_k))$ to $r_Q(i, j) + r_Q(j, k)$ and $Z(y^n)(y_i - y_k)$ to $r_Q(i, k)$, so the contributions are equal. It remains to consider two cases.

Case 1. $y_i \leq y_j$ and $y_j \geq y_k$. The contribution of y^n to the left side is $Z(y^n)[-(y_j - y_i) + (y_j - y_k)]$. If $y_i \leq y_k$, it adds $Z(y^n)(y_k - y_i)$ to $l_Q(k, i)$; otherwise, it adds $Z(y^n)(y_i - y_k)$ to $l_Q(i, k)$. In both cases, its contribution to the right side, $l_Q(i, k) - l_Q(k, i)$, is $Z(y^n)(y_i - y_k)$, and so the contributions are equal.

Case 2. $y_i \geq y_j$ and $y_j \leq y_k$. Similarly to Case 1, y^n donates $Z(y^n)[(y_i - y_j) - (y_k - y_j)]$ to the left side, and $Z(y^n)(y_i - y_k)$ to the right side, completing the proof. □

The proof of Theorem 1 carries through without modifications, using the generalized definition of losses and regrets and Lemma 2 in place of Lemma 1.

7 Relation to generalization bounds and other work

A number of papers analyze generalization properties of ranking algorithms (see, e.g., Freund et al. 2003; Agarwal et al. 2005; Agarwal and Niyogi 2005; Clémençon et al. 2005; Rudin et al. 2005). These papers analyze ranking directly by estimating the rate of convergence of empirical estimates of the ranking loss to its expectation. The bounds typically involve some complexity parameter of the class of functions searched by the algorithms (which serves as a regularizer), and some additional quantities considered relevant for the analysis. The examples are assumed to be drawn independently from some fixed distribution.

The type of results in this paper is different. We bound the realized AUC performance in terms of the realized classification performance, thus transferring performance from classification to ranking. Since the analysis is relative, it does not have to rely on any assumptions about the way the world produces data. In particular, the bounds apply when there are arbitrary high-order dependencies between examples, which is important in a number of applications where ranking is of interest.

A generalization result for the induced classification problem implies, via the reduction, a generalization result for the AUC problem. In situations where rankings are drawn iid from some base distribution D, the induced independent sample from AUC-TRAIN(D) can be formed by first drawing a ranking from D and then picking a mixed pair at random. Repeating this process many times produces an iid sample set for which standard rate of convergence analysis bounds hypothesis class regret.

Cortes and Mohri (2004) analyzed the relationship between the AUC and the error rate on the same classification problem, treating the two as different loss functions. They derived expressions for the expected value and the standard deviation of the AUC over all classifications with a fixed number of errors, under the assumption that all such classifications are equiprobable (i.e., the classifier is as likely to err on any one example as on any other). There is no connection with the present work.

Acknowledgements We would like to thank COLT reviewers for their comments and suggestions, and Nir Ailon and Mehryar Mohri for helpful discussions.

References

Agarwal, S., & Niyogi, P. (2005). Stability and generalization of bipartite ranking algorithms. In *Proceedings of the eighteenth annual conference on computational learning theory (COLT)* (pp. 32–47).

Agarwal, S., Har-Peled, S., & Roth, D. (2005). A uniform convergence bound for the area under the ROC curve. In *Proceedings of the 10th international workshop on artificial intelligence and statistics*.

Ailon, N., & Mohri, M. (2007). *An efficient reduction of ranking to classification* (Technical Report TR-2007-903). New York University.

Alon, N. (2006). Ranking tournaments. *SIAM Journal on Discrete Mathematics*, *20*, 137–142.

Bansal, N., Coppersmith, D., & Sorkin, G. B. (2006). *A winner–loser labeled tournament has at most twice as many outdegree misrankings as pair misrankings* (IBM Research Report RC24107). November 2006.

Clémençon, S., Lugosi, G., & Vayatis, N. (2005). Ranking and scoring using empirical risk minimization. In *Proceedings of the eighteenth annual conference on computational learning theory (COLT)* (pp. 1–15).

Cohen, W., Schapire, R., & Singer, Y. (1999). Learning to order things. *Journal of Artificial Intelligence Research*, *10*, 243–270.

Coppersmith, D., Fleischer, L., & Rudra, A. (2006). Ordering by weighted number of wins gives a good ranking for weighted tournaments. In *Proceeding of the 17th annual symposium on discrete algorithms (SODA)* (pp. 776–782).

Cortes, C., & Mohri, M. (2004). AUC optimization versus error rate minimization. In *Advances in neural information processing systems (NIPS)*.

Freund, Y., Iyer, R., Schapire, R., & Singer, Y. (2003). An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, *4*, 933–969.

Landau, H. (1953). On dominance relations and the structure of animal societies, III: the condition for a score structure. *Bulletin of Mathematical Biophysics*, *15*, 143–148.

Langford, J., & Zadrozny, B. (2005). Estimating class membership probabilities using classifier learners. In *Proceedings of the 10th international workshop on artificial intelligence and statistics*.

Marshall, A., & Olkin, I. (1979). *Mathematics in science and engineering: Vol. 143. Inequalities: theory of majorization and its applications*. New York.

Rudin, C., Cortes, C., Mohri, M., & Schapire, R. (2005). Margin-based ranking meets boosting in the middle. In *Proceedings of the eighteenth annual conference on computational learning theory (COLT)*.

Slater, P. (1961). Inconsistencies in a schedule of paired comparisons. *Biometrika*, *48*, 303–312.