Large-Margin Thresholded Ensembles for Ordinal Regression: Theory and Practice

Hsuan-Tien Lin and Ling Li

Learning Systems Group, California Institute of Technology, USA
htlin@caltech.edu, ling@caltech.edu

Abstract. We propose a thresholded ensemble model for ordinal regression problems. The model consists of a weighted ensemble of confidence functions and an ordered vector of thresholds. We derive novel large-margin bounds of common error functions, such as the classification error and the absolute error. In addition to some existing algorithms, we also study two novel boosting approaches for constructing thresholded ensembles. Both our approaches not only are simpler than existing algorithms, but also have a stronger connection to the large-margin bounds. In addition, they have comparable performance to SVM-based algorithms, but enjoy the benefit of faster training. Experimental results on benchmark datasets demonstrate the usefulness of our boosting approaches.

1 Introduction

Ordinal regression resides between multiclass classification and metric regression in the area of supervised learning. They have many applications in social science and information retrieval to match human preferences. In an ordinal regression problem, examples are labeled with a set of $K \geq 2$ discrete ranks, which, unlike general class labels, also carry ordering preferences. However, ordinal regression is not exactly the same as common metric regression, because the label set is of finite size and metric distance between ranks is undefined.

Several approaches for ordinal regression were proposed in recent years from a machine learning perspective. For example, Herbrich et al. [1] designed an algorithm with support vector machines (SVM). Other SVM formulations were first studied by Shashua and Levin [2], and some improved ones were later proposed by Chu and Keerthi [3]. Crammer and Singer [4] generalized the perceptron learning rule for ordinal regression in an online setting. These approaches are all extended from well-known binary classification algorithms [5]. In addition, they share a common property in predicting: the discrete rank comes from thresholding a continuous potential value, which represents an ordering preference. Ideally, examples with higher ranks should have higher potential values.

In the special case of $K = 2$, ordinal regression is similar to binary classification [6]. If we interpret the similarity from the other side, the confidence function for a binary classifier can be naturally used as an ordering preference. For example, Freund et al. [7] proposed a boosting algorithm, RankBoost, that constructs an ensemble of those confidence functions to form a better ordering preference.

J. Balcazár, P.M. Long, and F. Stephan (Eds.): ALT 2006, LNAI 4264, pp. 319–333, 2006.

However, RankBoost was not specifically designed for ordinal regression. Hence, some efforts are needed when applying RankBoost for ordinal regression.

In this work, we combine the ideas of thresholding and ensemble learning to propose a thresholded ensemble model for ordinal regression. In our model, potential values are computed from an ensemble of confidence functions, and then thresholded to rank labels. It is well-known that ensemble is useful and powerful in approximating complex functions for classification and metric regression [8]. Our model shall inherit the same advantages for ordinal regression. Furthermore, we define margins for the thresholded ensemble model, and derive novel large-margin bounds of its out-of-sample error. The results indicate that large-margin thresholded ensembles could generalize well.

Algorithms for constructing thresholded ensembles are also studied. We not only combine RankBoost with a thresholding algorithm, but also propose two simpler boosting formulations, named ordinal regression boosting (ORBoost). ORBoost formulations have stronger connections with the large-margin bounds that we derive, and are direct generalizations to the famous AdaBoost algorithm [9]. Experimental results demonstrate that ORBoost formulations share some good properties with AdaBoost. They usually outperform RankBoost, and have comparable performance to SVM-based algorithms.

This paper is organized as follows. Section 2 introduces ordinal regression, as well as the thresholded ensemble model. Large-margin bounds for thresholded ensembles are derived in Sect. 3. Then, an extended RankBoost algorithm and two ORBoost formulations, which construct thresholded ensembles, are discussed in Sect. 4. We show the experimental results in Sect. 5, and conclude in Sect. 6.

2 Thresholded Ensemble Model for Ordinal Regression

In an ordinal regression problem, we are given a set of training examples $\mathcal{S} = \{(x_n, y_n)\}_{n=1}^{N}$, where each input vector $x_n \in \mathbb{R}^D$ is associated with an ordinal label (i.e., rank) y_n. We assume that y_n belongs to a set $\{1, 2, \ldots, K\}$. The goal is to find an ordinal regression rule $G(x)$ that predicts the rank y of an unseen input vector x. For a theoretic setting, we shall assume that all input-rank pairs are drawn i.i.d. from some unknown distribution $\mathcal{D}$.

The setting above looks similar to that of a multiclass classification problem. Hence, a general classification error,[1]

$$E_C(G, \mathcal{D}) = \mathcal{E}_{(x,y)\sim\mathcal{D}} [\![G(x) \neq y]\!],$$

can be used to measure the performance of G. However, the classification error does not consider the ordering preference of the ranks. One naive interpretation of the ordering preference is as follows: for an example (x, y) with $y = 4$, if $G_1(x) = 3$ and $G_2(x) = 1$, G_1 is preferred over G_2 on that example. A common practice to encode such preference is to use the absolute error:

$$E_A(G, \mathcal{D}) = \mathcal{E}_{(x,y)\sim\mathcal{D}} \left|G(x) - y\right| .$$

[1] $[\![\cdot]\!] = 1$ when the inner condition is true, and 0 otherwise.

Next, we propose the thresholded ensemble model for ordinal regression. As the name suggests, the model has two components: a vector of thresholds, and an ensemble of confidence functions.

Thresholded models are widely used for ordinal regression [3,4]. The thresholds can be thought as estimated scales that reflect the discrete nature of ordinal regression. The ordinal regression rule, denoted as $G_{H,\theta}$, is illustrated in Fig. 1. Here $H(x)$ computes the potential value of x, and θ is a $(K-1)$ dimensional ordered vector that contains the thresholds ($\theta_1 \le \theta_2 \le \cdots \le \theta_{K-1}$). We shall denote $G_{H,\theta}$ as G_θ when H is clear from the context. Then, if we let $\theta_0 = -\infty$ and $\theta_K = \infty$, the ordinal regression rule is

$$G_\theta(x) = \min\{k\colon H(x) \le \theta_k\} = \max\{k\colon H(x) > \theta_{k-1}\} = 1 + \sum_{k=1}^{K-1} [\![H(x) > \theta_k]\!].$$

In the thresholded ensemble model, we take an ensemble of confidence functions to compute the potentials. That is,

$$H(x) = H_T(x) = \sum_{t=1}^{T} \alpha_t h_t(x), \quad \alpha_t \in \mathbb{R}.$$

We shall assume that the confidence function h_t comes from a hypothesis set $\mathcal{H}$, and has an output range $[-1, 1]$. A special case of the confidence function, which only outputs -1 or 1, would be called a binary classifier. Each confidence function reflects a possibly imperfect ordering preference. The ensemble linearly combines the ordering preferences with α. Note that we allow α_t to be any real value, which means that it is possible to reverse the ordering preference of h_t in the ensemble when necessary.

Ensemble models in general have been successfully used for classification and metric regression [8]. They not only introduce more stable predictions through the linear combination, but also provide sufficient power for approximating complex functions. These properties shall be inherited by the thresholded ensemble model for ordinal regression.

3 Large-Margin Bounds for Thresholded Ensembles

Margin is an important concept in structural risk minimization [10]. Many large-margin error bounds were proposed based on the intuition that large margins lead to good generalization. They are typically of the form

$$E_1(G, \mathcal{D}) \le E_2(G, \mathcal{S}_u, \Delta) + \text{complexity term}.$$

Here $E_1(G, \mathcal{D})$ is the generalization error of interest, such as $E_A(G, \mathcal{D})$. $\mathcal{S}_u$ denotes the uniform distribution on the set $\mathcal{S}$, and $E_2(G, \mathcal{S}_u, \Delta)$ represents some training error with margin Δ, which will be further explained in this section.

For ordinal regression, Herbrich et al. [1] derived a large-margin bound for a thresholded ordinal regression rule G. Unfortunately the bound is quite restricted

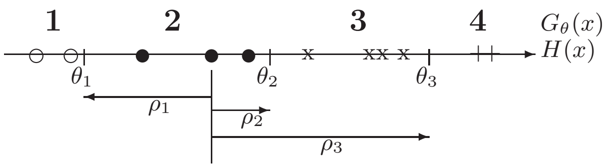


Fig. 1. The thresholded model and the margins of a correctly-predicted example

since it requires that $E_2(G, \mathcal{S}_u, \Delta) = 0$. In addition, the bound uses a definition of margin that has $O(N^2)$ terms, which makes it more complicated to design algorithms that relate to the bound. Another bound was derived by Shashua and Levin [2]. The bound is based on a margin definition of only $O(KN)$ terms, and is applicable to the thresholded ensemble model. However, the bound is loose when T, the size of the ensemble, is large, because its complexity term grows with T.

In this section, we derive novel large-margin bounds of different error functions for the thresholded ensemble model. The bounds are extended from the results of Schapire et al. [11]. Our bounds are based on a margin definition of $O(KN)$ terms. Similar to the results of Schapire et al., our bounds do not require $E_2(G, \mathcal{S}_u, \Delta) = 0$, and their complexity terms do not grow with T.

3.1 Margins

The margins with respect to a thresholded model are illustrated in Fig. 1. Intuitively, we expect the potential value $H(x)$ to be in the correct interval $(\theta_{y-1}, \theta_y]$, and we want $H(x)$ to be far from the boundaries (thresholds):

Definition 1. *Consider a given thresholded ensemble $G_\theta(x)$.*

1. *The margin of an example (x, y) with respect to θ_k is defined as*

$$\rho_k(x, y) = \begin{cases} H(x) - \theta_k, & \text{if } y > k; \\ \theta_k - H(x), & \text{if } y \le k. \end{cases}$$

2. *The normalized margin $\bar{\rho}_k(x, y)$ is defined as*

$$\bar{\rho}_k(x, y) = \rho_k(x, y) \Bigg/ \left(\sum_{t=1}^{T} |\alpha_t| + \sum_{k=1}^{K-1} |\theta_k| \right).$$

Definition 1 is similar to the definition by Shashua and Levin [2], which is analogous to the definition of margins in binary classification. A negative $\rho_k(x, y)$ would indicate an incorrect prediction.

For each example (x, y), we can obtain $(K - 1)$ margins from Definition 1. However, two of them are of the most importance. The first one is $\rho_{y-1}(x, y)$, which is the margin to the left (lower) boundary of the correct interval. The other is $\rho_y(x, y)$, which is the margin to the right (upper) boundary. We will give them special names: the left-margin $\rho_L(x,y)$, and the right-margin $\rho_R(x,y)$. Note that by definition, $\rho_L(x,1) = \rho_R(x,K) = \infty$.

Δ-classification error: Next, we take a closer look at the error functions for thresholded ensemble models. If we make a minor assumption that the degenerate cases $\bar{\rho}_R(x,y) = 0$ are of an infinitesimal probability,

$$\begin{aligned} E_C(G_\theta, \mathcal{D}) &= \mathcal{E}_{(x,y)\sim\mathcal{D}} [\![G_\theta(x) \neq y]\!] \\ &= \mathcal{E}_{(x,y)\sim\mathcal{D}} [\![\bar{\rho}_L(x,y) \leq 0 \text{ or } \bar{\rho}_R(x,y) \leq 0]\!]. \end{aligned}$$

The definition could be generalized by expecting both margins to be larger than Δ. That is, define the Δ-classification error as

$$E_C(G_\theta, \mathcal{D}, \Delta) = \mathcal{E}_{(x,y)\sim\mathcal{D}} [\![\bar{\rho}_L(x,y) \leq \Delta \text{ or } \bar{\rho}_R(x,y) \leq \Delta]\!].$$

Then, $E_C(G_\theta, \mathcal{D})$ is just a special case with $\Delta = 0$.

Δ-boundary error: The "or" operation of $E_C(G_\theta, \mathcal{D}, \Delta)$ is not easy to handle in the proof of the coming bounds. An alternative choice is the Δ-boundary error:

$$E_B(G_\theta, \mathcal{D}, \Delta) = \mathcal{E}_{(x,y)\sim\mathcal{D}} \begin{cases} [\![\bar{\rho}_R(x,y) \leq \Delta]\!], & \text{if } y = 1; \\ [\![\bar{\rho}_L(x,y) \leq \Delta]\!], & \text{if } y = K; \\ \frac{1}{2} \cdot \left([\![\bar{\rho}_L(x,y) \leq \Delta]\!] + [\![\bar{\rho}_R(x,y) \leq \Delta]\!] \right), & \text{otherwise.} \end{cases}$$

The Δ-boundary error and the Δ-classification error are equivalent up to a constant. That is, for any $(G_\theta, \mathcal{D}, \Delta)$,

$$\tfrac{1}{2} E_C(G_\theta, \mathcal{D}, \Delta) \leq E_B(G_\theta, \mathcal{D}, \Delta) \leq E_C(G_\theta, \mathcal{D}, \Delta). \tag{1}$$

Δ-absolute error: We can analogously define the Δ-absolute error as

$$E_A(G_\theta, \mathcal{D}, \Delta) = \mathcal{E}_{(x,y)\sim\mathcal{D}} \sum_{k=1}^{K-1} [\![\bar{\rho}_k(x,y) \leq \Delta]\!].$$

Then, if we assume that the degenerate cases $\rho_k(x,y) = 0$ happen with an infinitesimal probability, $E_A(G_\theta, \mathcal{D})$ is just a special case with $\Delta = 0$.

3.2 Large-Margin Bounds

An important observation for deriving our bounds is that E_B and E_A can be written with respect to an additional sampling of k. For example,

$$E_A(G_\theta, \mathcal{D}, \Delta) = (K-1)\mathcal{E}_{(x,y)\sim\mathcal{D}, k\sim\{1,\ldots,K-1\}_u} [\![\bar{\rho}_k(x,y) \leq \Delta]\!].$$

Equivalently, we can define a distribution $\hat{\mathcal{D}}$ by $\mathcal{D}$ and $\{1,\ldots,K-1\}_u$ to generate the tuple (x,y,k). Then $E_A(G_\theta, \mathcal{D})$ is simply the portion of nonpositive $\bar{\rho}_k(x,y)$ under $\hat{\mathcal{D}}$. Consider an extended training set $\hat{\mathcal{S}} = \{(x_n, y_n, k)\}$

with $N(K-1)$ elements. Each element is a possible outcome from $\hat{\mathcal{D}}$. Note, however, that these elements are not all independent. For example, $(x_n, y_n, 1)$ and $(x_n, y_n, 2)$ are dependent. Thus, we cannot directly use the whole $\hat{\mathcal{S}}$ as a set of i.i.d. outcomes from $\hat{\mathcal{D}}$.

Fortunately, some subsets of $\hat{\mathcal{S}}$ contain independent outcomes from $\hat{\mathcal{D}}$. One way to extract such subsets is to choose one k_n from $\{1, \ldots, K-1\}_u$ for each example (x_n, y_n) independently. The subset would be named $\mathcal{T} = \{(x_n, y_n, k_n)\}_{n=1}^N$. Then, we can obtain a large-margin bound of the absolute error:

Theorem 1. *Consider a set $\mathcal{H}$, which contains only binary classifiers, is negation-complete,*[2] *and has VC-dimension d. Let $\delta > 0$, and $N > d + K - 1 = \hat{d}$. Then with probability at least $1-\delta$ over the random choice of the training set $\mathcal{S}$, every thresholded ensemble $G_\theta(x)$, where the associated H is constructed with $h \in \mathcal{H}$, satisfies the following bound for all $\Delta > 0$:*

$$E_A(G_\theta, \mathcal{D}) \leq E_A(G_\theta, \mathcal{S}_u, \Delta) + O\left(\frac{K}{\sqrt{N}}\left(\frac{\hat{d}\log^2(N/\hat{d})}{\Delta^2} + \log\frac{1}{\delta}\right)^{1/2}\right).$$

Proof. The key is to reduce the ordinal regression problem to a binary classification problem, which consists of training examples derived from $(x_n, y_n, k_n) \in \mathcal{T}$:

$$(X_n, Y_n) = \begin{cases} \big((x_n, \mathbf{1}_{k_n}), +1\big), & \text{if } y_n > k_n; \\ \big((x_n, \mathbf{1}_{k_n}), -1\big), & \text{if } y_n \leq k_n, \end{cases} \tag{2}$$

where $\mathbf{1}_m$ is a vector of length $(K-1)$ with a single 1 at the m-th dimension and 0 elsewhere. The test examples are constructed similarly with $(x, y, k) \sim \hat{\mathcal{D}}$. Then, large-margin bounds for the ordinal regression problem can be inferred from those for the binary classification problem, as shown in Appendix A. □

Similarly, if we look at the boundary error,

$$E_B(G_\theta, \mathcal{D}, \Delta) = \mathcal{E}_{(x,y)\sim\mathcal{D}, k\sim\mathcal{B}_y} [\![\bar{\rho}_k(x, y) \leq \Delta]\!],$$

for some distribution $\mathcal{B}_y$ on $\{L, R\}$. Then, a similar proof leads to

Theorem 2. *For the same conditions as of Theorem 1,*

$$E_B(G_\theta, \mathcal{D}) \leq E_B(G_\theta, \mathcal{S}_u, \Delta) + O\left(\frac{1}{\sqrt{N}}\left(\frac{\hat{d}\log^2(N/\hat{d})}{\Delta^2} + \log\frac{1}{\delta}\right)^{1/2}\right).$$

Then, a large-margin bound of the classification error can immediately be derived by applying (1).

Corollary 1. *For the same conditions as of Theorem 1,*

$$E_C(G_\theta, \mathcal{D}) \leq 2E_C(G_\theta, \mathcal{S}_u, \Delta) + O\left(\frac{1}{\sqrt{N}}\left(\frac{\hat{d}\log^2(N/\hat{d})}{\Delta^2} + \log\frac{1}{\delta}\right)^{1/2}\right).$$

[2] $h \in \mathcal{H} \Longleftrightarrow (-h) \in \mathcal{H}$, where $(-h)(x) = -\big(h(x)\big)$ for all x.

Similar bounds can be derived with another large-margin theorem [11, Theorem 4] when $\mathcal{H}$ contains confidence functions rather than binary classifiers. These bounds provide motivations for building algorithms with margin-related formulations.

4 Boosting Algorithms for Thresholded Ensembles

The bounds in the previous section are applicable to thresholded ensembles generated from any algorithms. One possible algorithm, for example, is an SVM-based approach [3] with special kernels [12]. In this section, we focus on another branch of approaches: boosting. Boosting approaches can iteratively grow the ensemble $H(x)$, and have been successful in classification and metric regression [8]. Our study includes an extension to the RankBoost algorithm [7] and two novel formulations that we propose.

4.1 RankBoost for Ordinal Regression

RankBoost [7] constructs a weighted ensemble of confidence functions based on the following large-margin concept: for each pair (i, j) such that $y_i > y_j$, the difference between their potential values, $H_t(x_i) - H_t(x_j)$, is desired to be positive and large. Thus, in the t-th iteration, the algorithm chooses (h_t, α_t) to approximately minimize

$$\sum_{y_i > y_j} e^{-H_{t-1}(x_i) - \alpha_t h_t(x_i) + H_{t-1}(x_j) + \alpha_t h_t(x_j)}. \tag{3}$$

Our efforts in extending RankBoost for ordinal regression are discussed as follows:

Computing α_t: Two approaches can be used to determine α_t in RankBoost [7]:

1. Obtain the optimal α_t by numerical search (confidence functions) or analytical solution (binary classifiers).
2. Minimize an upper bound of (3).

If $h_t(x_n)$ is monotonic with respect to y_n, the optimal α_t obtained from approach 1 is ∞, and one single h_t would dominate the ensemble. This situation not only makes the ensemble less stable, but also limits its power. For example, if $(y_n, h_t(x_n))$ pairs for four examples are $(1, -1)$, $(2, 0)$, $(3, 1)$, and $(4, 1)$, ranks 3 and 4 on the last two examples cannot be distinguished by h_t. We have frequently observed such a degenerate situation, called *partial matching*, in real-world experiments, even when h_t is as simple as a decision stump. Thus, we shall use approach 2 for our experiments. Note, however, that when partial matching happens, the magnitude of α_t from approach 2 can still be relatively large, and may cause numerical difficulties.

Obtaining θ: After RankBoost computes a potential function $H(x)$, a reasonable way to obtain the thresholds based on training examples is

$$\theta = \operatorname{argmin}_{\vartheta} E_A(G_{\vartheta}, \mathcal{S}_u). \tag{4}$$

The combination of RankBoost and the absolute error criterion (4) would be called RankBoost-AE. The optimal range of ϑ_k can be efficiently determined by dynamic programming. For simplicity and stability, we assign θ_k to be the middle value in the optimal range. The algorithm that aims at E_C instead of E_A can be similarly derived.

4.2 Ordinal Regression Boosting with Left-Right Margins

The idea of ordinal regression boosting comes from the definition of margins in Sect. 3. As indicated by our bounds, we want the margins to be as large as possible. To achieve this goal, our algorithms, similar to AdaBoost, work on minimizing the exponential margin loss.

First, we introduce a simple formulation called ordinal regression boosting with left-right margins (ORBoost-LR), which tries to minimize

$$\sum_{n=1}^{N} \left[e^{-\rho_L(x_n, y_n)} + e^{-\rho_R(x_n, y_n)} \right] . \tag{5}$$

The formulation can be thought as maximizing the soft-min of the left- and right-margins. Similar to RankBoost, the minimization is performed in an iterative manner. In each iteration, a confidence function h_t is chosen, its weight α_t is computed, and the vector θ is updated. If we plug in the margin definition to (5), we can see that the iteration steps should be designed to approximately minimize

$$\sum_{n=1}^{N} \left[\varphi_n e^{\alpha_t h_t(x_n) - \theta_{y_n}} + \varphi_n^{-1} e^{\theta_{y_n - 1} - \alpha_t h_t(x_n)} \right] , \tag{6}$$

where $\varphi_n = e^{H_{t-1}(x_n)}$. Next, we discuss these three steps in detail.

Choosing h_t: Mason et al. [13] explained AdaBoost as a gradient descent technique in function space. We derive ORBoost-LR using the same technique. We first choose a confidence function h_t that is close to the negative gradient:

$$h_t = \operatorname*{argmin}_{h \in \mathcal{H}} \sum_{n=1}^{N} h(x_n) \left(\varphi_n e^{-\theta_{y_n}} - \varphi_n^{-1} e^{\theta_{y_n - 1}} \right) .$$

This step can be performed with the help of another learning algorithm, called the base learner.

Computing α_t: Similar to RankBoost, we minimize an upper bound of (6), which is based on a piece-wise linear approximation of e^x for $x \in [-1, 0]$ and $x \in [0, 1]$. The bound can be written as $W_+ e^{\alpha} + W_- e^{-\alpha}$, with

$$W_+ = \sum_{h_t(x_n) > 0} h_t(x_n) \varphi_n e^{-\theta_{y_n}} - \sum_{h_t(x_n) < 0} h_t(x_n) \varphi_n^{-1} e^{\theta_{y_n - 1}} ,$$

$$W_- = \sum_{h_t(x_n) > 0} h_t(x_n) \varphi_n^{-1} e^{\theta_{y_n - 1}} - \sum_{h_t(x_n) < 0} h_t(x_n) \varphi_n e^{-\theta_{y_n}} .$$

Then, the optimal α_t for the bound can be computed by $\frac{1}{2} \log \frac{W_-}{W_+}$.

Note that the upper bound is equal to (6) if $h_t(x_n) \in \{-1, 0, 1\}$. Thus, when h_t is a binary classifier, the optimal α_t can be exactly determined. Another remark here is that α_t is finite under some mild conditions which make both W_+ and W_- positive. Thus, unlike RankBoost, ORBoost-LR rarely sets α_t to ∞.

Updating θ: Note that when the pair (h_t, α_t) is fixed, (6) can be reorganized as $\sum_{k=1}^{K-1} W_{k,+}e^{\theta_k} + W_{k,-}e^{-\theta_k}$. Then, each θ_k can be computed analytically, uniquely, and independently. However, when each θ_k is updated independently, the thresholds may not be ordered. Hence, we propose to add an additional ordering constraint to (6). That is, choosing θ by solving

$$\begin{aligned} \min_{\vartheta} \quad & \sum_{k=1}^{K-1} W_{k,+}e^{\vartheta_k} + W_{k,-}e^{-\vartheta_k} \\ \text{s.t.} \quad & \vartheta_1 \le \vartheta_2 \le \cdots \le \vartheta_{K-1}. \end{aligned} \tag{7}$$

An efficient algorithm for solving (7) can be obtained from by a simple modification of the pool adjacent violators (PAV) algorithm for isotonic regression [14].

Combination of the steps: ORBoost-LR works by combining the three steps above sequentially in each iteration. Note that after h_t is determined, α_t and θ_t can be either jointly optimized, or cyclically updated. However, we found that joint or cyclic optimization does not always introduce better performance, and could sometimes cause ORBoost-LR to overfit. Thus, we only execute each step once in each iteration.

4.3 Ordinal Regression Boosting with All Margins

ORBoost with all margins (ORBoost-All) operates on

$$\sum_{n=1}^{N} \sum_{k=1}^{K-1} e^{-\rho_k(x_n, y_n)} \tag{8}$$

instead of (6). The derivations for the three steps are almost the same as ORBoost-LR. We shall just make some remarks.

Updating θ: When using (8) to update the thresholds, we have proved that each θ_k can be updated uniquely and independently, while still being ordered [5]. Thus, we do not need to implement the PAV algorithm for ORBoost-All.

Relationship between algorithm and theory: A simple relation is that for any Δ, $e^{-A\bar{\rho}_k(x_n,y_n)}$ is an upper bound of $e^{-A\Delta} \cdot [\![\bar{\rho}_k(x_n, y_n) \le \Delta]\!]$. If we take A to be the normalization term of $\bar{\rho}_k$, we can see that

- ORBoost-All works on minimizing an upper bound of $E_A(G_\theta, \mathcal{S}_u, \Delta)$.
- ORBoost-LR works to minimizing an upper bound of $E_B(G_\theta, \mathcal{S}_u, \Delta)$, or $\frac{1}{2}E_C(G_\theta, \mathcal{S}_u, \Delta)$.

ORBoost-All not only minimizes an upper bound, but provably also minimizes the term $E_A(G_\theta, \mathcal{S}_u, \Delta)$ exponentially fast with a sufficiently strong choice of h_t. The proof relies on an extension of the training error theorem of AdaBoost [11, Theorem 5]. Similar proof can be used for ORBoost-LR.

Connection to other algorithms: ORBoost approaches are direct generalizations of AdaBoost using the gradient descent optimization point of view. In the special case of $K = 2$, both ORBoost approaches are almost the same as AdaBoost with an additional term θ_1. Note that the term θ_1 can be thought as the coefficient of a constant classifier. Interestingly, Rudin et al. [6] proved the connection between RankBoost and AdaBoost when including a constant classifier in the ensemble. Thus, when $K = 2$, RankBoost-EA, ORBoost-LR, and ORBoost-All, all share some similarity with AdaBoost.

ORBoost formulations also have connections with SVM-based algorithms. In particular, ORBoost-LR has a counterpart of SVM with explicit constraints (SVM-EXC), and ORBoost-All is related to SVM with implicit constraints (SVM-IMC) [3]. These connections follow closely with the links between AdaBoost and SVM [12, 15].

5 Experiments

In this section, we compare the three boosting formulations for constructing the thresholded ensemble model. We also compare these formulations with SVM-based algorithms.

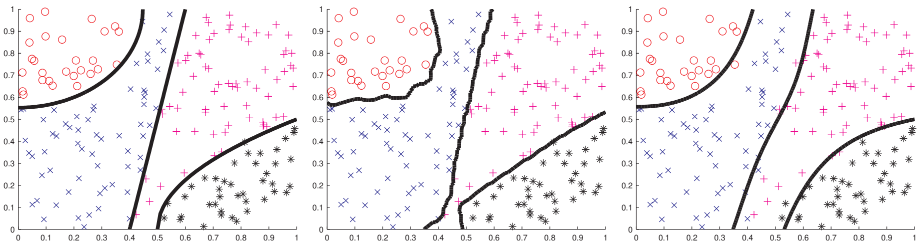
Two sets of confidence functions are used in the experiments. The first one is the set of perceptrons $\{\mathrm{sign}(w^T x + b) : w \in \mathbb{R}^D, b \in \mathbb{R}\}$. The RCD-bias algorithm is known to work well with AdaBoost [16], and is adopted as our base learner.

The second set is $\{\tanh(w^T x + b) \colon w^T w + b^2 = \gamma^2\}$, which contains normalized sigmoid functions. Note that sigmoid functions smoothen the output of perceptrons, and the smoothness is controlled by the parameter γ. We use a naive base learner for normalized sigmoid functions as follows: RCD-bias is first performed to get a perceptron. Then, the weights and bias of the perceptron are normalized, and the outputs are smoothened. Throughout the experiments we use $\gamma = 4$, which was picked with a few experimental runs on some datasets.

5.1 Artificial Dataset

We first verify that the idea of the thresholded ensemble model works with an artificial 2-D dataset (Fig. 2(a)). Figure 2(b) depicts the separating boundaries of the thresholded ensemble of 200 perceptrons constructed by ORBoost-All. By combining perceptrons, ORBoost-All works reasonably well in approximating the nonlinear boundaries. A similar plot can be obtained with ORBoost-LR. RankBoost-AE cannot perform well on this dataset due to numerical difficulties (see Subsect. 4.1) after only 5 iterations.

If we use a thresholded ensemble of 200 normalized sigmoid functions, it is observed that ORBoost-All, ORBoost-LR, and RankBoost-AE perform similarly.



(a) the target (b) with perceptron (c) with sigmoid

Fig. 2. An artificial 2-D dataset and the learned boundaries with ORBoost-All

The result of ORBoost-All (Fig. 2(c)) shows that the separating boundaries are much smoother because each sigmoid function is smooth. As we shall discuss later, the smoothness can be important for some ordinal regression problems.

5.2 Benchmark Datasets

Next, we perform experiments with eight benchmark datasets[3] that were used by Chu and Keerthi [3]. The datasets are quantized from some metric regression datasets. We use the same $K = 10$, the same "training/test" partition ratio, and also average the results over 20 trials. Thus, we can compare RankBoost and ORBoost fairly with the SVM-based results of Chu and Keerthi [3].

The results on the abalone dataset with T up to 2000 are given in Fig. 3. The training errors are shown in the top plots, while the test errors are shown in the bottom plots. Based on these results, we have several remarks:

RankBoost vs. ORBoost: RankBoost-AE can usually decrease both the training classification and the training absolute errors faster than ORBoost algorithms. However, such property often lead to consistently worse test error than both ORBoost-LR and ORBoost-All. An explanation is that although the RankBoost ensemble orders the training examples well, the current estimate of θ is not used to decide (h_t, α_t). Thus, the two components (H_T, θ) of the thresholded ensemble model are not jointly considered, and the greediness in constructing only H_T results in overfitting. In contrast, ORBoost-LR and ORBoost-All take into consideration the current θ in choosing (h_t, α_t) and the current H_T in updating θ. Hence, a better pair of (H_T, θ) could be obtained.

ORBoost-LR vs. ORBoost-All: Both ORBoost formulations inherit a good property from AdaBoost: not very vulnerable to overfitting. ORBoost-LR is better on test classification errors, while ORBoost-All is better on test absolute errors. This is partially justified by our discussion in Subsect. 4.3 that the two formulations minimize different margin-related upper bounds. A similar observation was made by Chu and Keerthi [3] on SVM-EXC and SVM-IMC algorithms.

[3] pyrimdines, machineCPU, boston, abalone, bank, computer, california, and census.

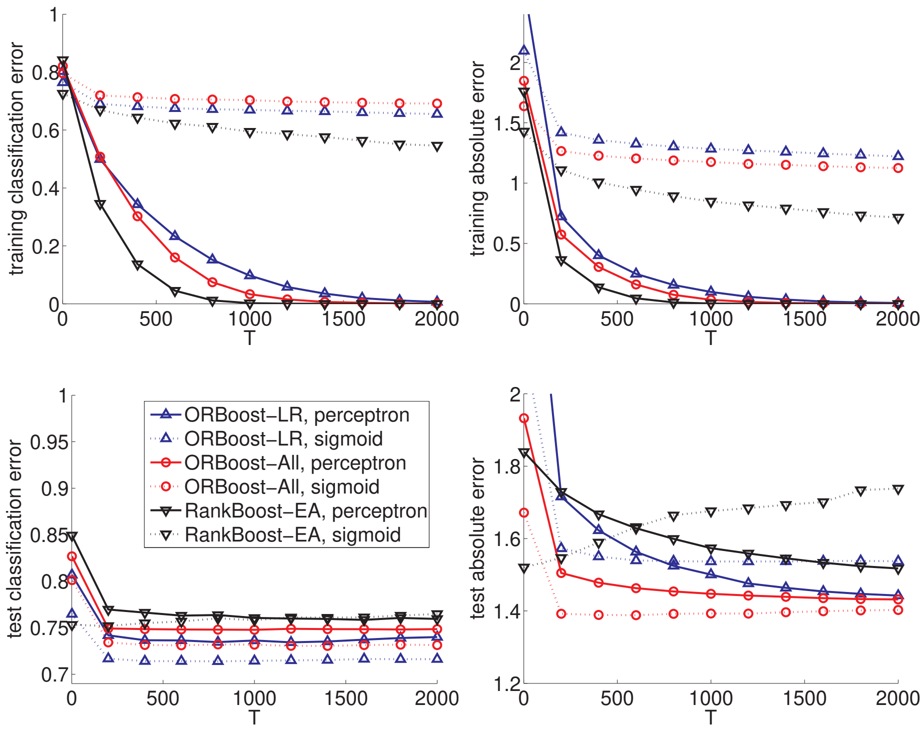


Fig. 3. Errors on the abalone dataset over 20 runs

Note, however, that ORBoost-LR with perceptrons minimizes the training classification error slower than ORBoost-All on this dataset, because the additional ordering constraint of θ in ORBoost-LR slows down the convergence.

Perceptron vs. sigmoid: Formulations with sigmoid functions have consistently higher training error, which is due to the naive choice of base learner and the approximation of α_t. However, the best test performance is also achieved with sigmoid functions. One possible reason is that the abalone dataset is quantized from a metric regression dataset, and hence contains some properties such as smoothness of the boundaries. If we only use binary classifiers like perceptrons, as depicted in Fig. 2(b), the boundaries would not be as smooth, and more errors may happen. Thus, for ordinal regression datasets that are quantized from metric regression datasets, smooth confidence functions may be more useful than discrete binary classifiers.

We list the mean and standard errors of all test results with $T = 2000$ in Tables 1 and 2. Consistent with the results on the abalone dataset, RankBoost-AE almost always performs the worst; ORBoost-LR is better on classification errors, and ORBoost-All is slightly better on absolute errors. When compared with SVM-IMC on classification errors and SVM-EXC on absolute errors [3],

Table 1. Test classification error of ordinal regression algorithms

data set	RankBoost-AE perceptron	RankBoost-AE sigmoid	ORBoost-LR perceptron	ORBoost-LR sigmoid	ORBoost-All perceptron	ORBoost-All sigmoid	SVM-EXC [3]
pyr.	0.758±0.015	0.767±0.020	**0.731±0.019**	**0.731±0.018**	0.744±0.019	0.735±0.017	0.752±0.014
mac.	0.717±0.022	0.669±0.011	**0.610±0.009**	0.633±0.011	**0.605±0.010**	0.625±0.014	0.661±0.012
bos.	0.603±0.006	0.578±0.008	0.580±0.006	**0.549±0.007**	0.579±0.006	0.558±0.006	0.569±0.006
aba.	0.759±0.001	0.765±0.002	0.740±0.002	**0.716±0.002**	0.749±0.002	0.731±0.002	0.736±0.002
ban.	0.805±0.001	0.822±0.001	0.767±0.001	0.777±0.002	0.771±0.001	0.776±0.001	**0.744±0.001**
com.	0.598±0.002	0.616±0.001	0.498±0.001	0.491±0.001	0.499±0.001	0.505±0.001	**0.462±0.001**
cal.	0.741±0.001	0.690±0.001	0.628±0.001	**0.605±0.001**	0.626±0.001	0.618±0.001	0.640±0.001
cen.	0.808±0.001	0.780±0.001	0.718±0.001	**0.694±0.001**	0.722±0.001	0.701±0.001	0.699±0.000

(results that are within one standard error of the best are marked in bold)

Table 2. Test absolute error of ordinal regression algorithms

data set	RankBoost-AE perceptron	RankBoost-AE sigmoid	ORBoost-LR perceptron	ORBoost-LR sigmoid	ORBoost-All perceptron	ORBoost-All sigmoid	SVM-IMC [3]
pyr.	1.619±0.078	1.590±0.077	**1.340±0.049**	1.402±0.052	1.360±0.046	1.398±0.052	**1.294±0.046**
mac.	1.573±0.191	1.282±0.034	**0.897±0.019**	0.985±0.018	**0.889±0.019**	0.969±0.025	0.990±0.026
bos.	0.842±0.014	0.829±0.014	0.788±0.013	**0.758±0.015**	0.791±0.013	0.777±0.015	**0.747±0.011**
aba.	1.517±0.005	1.738±0.008	1.442±0.004	1.537±0.007	1.432±0.003	1.403±0.004	**1.361±0.003**
ban.	1.867±0.004	2.183±0.007	1.507±0.002	1.656±0.005	1.490±0.002	1.539±0.002	**1.393±0.002**
com.	0.841±0.003	0.945±0.004	0.631±0.002	0.634±0.003	0.626±0.002	0.634±0.002	**0.596±0.002**
cal.	1.528±0.006	1.251±0.004	1.042±0.004	0.956±0.002	0.977±0.002	**0.942±0.002**	1.008±0.001
cen.	2.008±0.006	1.796±0.005	1.305±0.003	1.262±0.003	1.265±0.002	**1.198±0.002**	1.205±0.002

(results that are within one standard error of the best are marked in bold)

both ORBoost formulations have similar errors as the SVM-based algorithms. Note, however, that ORBoost formulations with perceptrons or sigmoid functions are much faster. On the census dataset, which contains 6000 training examples, it takes about an hour for ORBoost to finish one trial. But SVM-based approaches, which include a time-consuming automatic parameter selection step, need more than four days. With the comparable performance and significantly less computational cost, ORBoost could be a useful tool for large datasets.

6 Conclusion

We proposed a thresholded ensemble model for ordinal regression, and defined margins for the model. Novel large-margin bounds of common error functions were proved. We studied three algorithms for obtaining thresholded ensembles. The first algorithm, RankBoost-AE, combines RankBoost and a thresholding algorithm. In addition, we designed two new boosting approaches, ORBoost-LR and ORBoost-All, which have close connections with the large-margin bounds. ORBoost formulations are direct extensions of AdaBoost, and inherit its advantage of being less venerable to overfitting.

Experimental results demonstrated that ORBoost formulations have superior performance over RankBoost-AE. In addition, they are comparable to SVM-based algorithms in terms of test error, but enjoy the advantage of faster training. These properties make ORBoost formulations favorable over SVM-based algorithms on large datasets.

ORBoost formulations can be equipped with any base learners for confidence functions. In this work, we studied the perceptrons and the normalized sigmoid functions. Future work could be exploring other confidence functions for ORBoost, or extending other boosting approaches to perform ordinal regression.

Acknowledgment

We thank Yaser S. Abu-Mostafa, Amrit Pratap, and the anonymous reviewers for helpful comments. Hsuan-Tien Lin is supported by the Caltech Division of Engineering and Applied Science Fellowship.

References

1. Herbrich, R., Graepel, T., Obermayer, K.: Large margin rank boundaries for ordinal regression. In: Advances in Large Margin Classifiers. MIT Press (2000) 115–132
2. Shashua, A., Levin, A.: Ranking with large margin principle: Two approaches. In: Advances in Neural Information Processing Systems 15, MIT Press (2003) 961–968
3. Chu, W., Keerthi, S.S.: New approaches to support vector ordinal regression. In: Proceedings of ICML 2005, Omnipress (2005) 145–152
4. Crammer, K., Singer, Y.: Online ranking by projecting. Neural Computation **17** (2005) 145–175
5. Li, L., Lin, H.T.: Ordinal regression by extended binary classification. Under review (2007)
6. Rudin, C., Cortes, C., Mohri, M., Schapire, R.E.: Margin-based ranking meets boosting in the middle. In: Learning Theory: COLT 2005, Springer-Verlag (2005) 63–78
7. Freund, Y., Iyer, R., Shapire, R.E., Singer, Y.: An efficient boosting algorithm for combining preferences. Journal of Machine Learning Research **4** (2003) 933–969
8. Meir, R., Rätsch, G.: An introduction to boosting and leveraging. In: Advanced Lectures on Machine Learning. Springer-Verlag (2003) 118–183
9. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: Machine Learning: ICML 1996, Morgan Kaufmann (1996) 148–156
10. Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer-Verlag (1995)
11. Schapire, R.E., Freund, Y., Bartlett, P., Lee, W.S.: Boosting the margin: A new explanation for the effectiveness of voting methods. The Annals of Statistics **26** (1998) 1651–1686
12. Lin, H.T., Li, L.: Infinite ensemble learning with support vector machines. In: Machine Learning: ECML 2005, Springer-Verlag (2005) 242–254
13. Mason, L., Baxter, J., Bartlett, P., Frean, M.: Functional gradient techniques for combining hypotheses. In: Advances in Large Margin Classifiers. MIT Press (2000) 221–246
14. Robertson, T., Wright, F.T., Dykstra, R.L.: Order Restricted Statistical Inference. John Wiley & Sons (1988)

15. Rätsch, G., Mika, S., Schölkopf, B., Müller, K.R.: Constructing boosting algorithms from SVMs: An application to one-class classification. IEEE Transactions on Pattern Analysis and Machine Intelligence **24** (2002) 1184–1199
16. Li, L.: Perceptron learning with random coordinate descent. Technical Report CaltechCSTR:2005.006, California Institute of Technology (2005)

A Proof of Theorem 1

As shown in (2), we first construct a transformed binary problem. Then, the problem is modeled by an ensemble function $F(x)$ defined on a base space

$$\mathcal{F} = \mathcal{H} \cup \{s_k\}_{k=1}^{K-1} .$$

Here $s_k(X) = -\operatorname{sign}(X_{D+k} - 0.5)$ is a decision stump on dimension $(D+k)$. It is not hard to show that the VC-dimension of $\mathcal{F}$ is no more than $\hat{d} = d + K - 1$.

Without loss of generality, we normalize $G_\theta(x)$ such that $\sum_{t=1}^{T} |\alpha_t| + \sum_{k=1}^{K-1} |\theta_k|$ is 1. Then, consider the associated ensemble function

$$F(X) = \sum_{t=1}^{T} \alpha_t h_t(X) + \sum_{k=1}^{K-1} \theta_k s_k(X).$$

An important property for the transform is that for every (X, Y) derived from the tuple (x, y, k), $YF(X) = \bar{\rho}_k(x, y)$.

Because $\mathcal{T}$ contains N i.i.d. outcomes from $\hat{\mathcal{D}}$, the large-margin theorem [11, Theorem 2] states that with probability at least $1 - \delta/2$ over the choice of $\mathcal{T}$,

$$\mathcal{E}_{(x,y,k)\sim\hat{\mathcal{D}}}[YF(X) \le 0] \le \frac{1}{N}\sum_{n=1}^{N} [\![Y_n F(X_n) \le \Delta]\!] + O\left(\frac{1}{\sqrt{N}}\left(\frac{\hat{d}\log^2(N/\hat{d})}{\Delta^2} + \log\frac{1}{\delta}\right)^{1/2}\right) . \quad (9)$$

Since $YF(X) = \bar{\rho}_k(x, y)$, the left-hand-side is $\frac{1}{K-1} E_A(G_\theta, \mathcal{D})$.

Let $b_n = [\![Y_n F(X_n) \le \Delta]\!] = [\![\bar{\rho}_{k_n}(x_n, y_n) \le \Delta]\!]$, which is a Boolean random variable. An extended Chernoff bound shows that when each b_n is chosen independently, with probability at least $1 - \delta/2$ over the choice of b_n,

$$\frac{1}{N}\sum_{n=1}^{N} b_n \le \frac{1}{N}\sum_{n=1}^{N} \mathcal{E}_{k_n \sim \{1,\cdots,K-1\}_u} b_n + O\left(\frac{1}{\sqrt{N}}\left(\log\frac{1}{\delta}\right)^{1/2}\right) . \quad (10)$$

The desired result can be obtained by combining (9) and (10), with a union bound and $\mathcal{E}_{k_n \sim \{1,\cdots,K-1\}_u} b_n = \frac{1}{K-1} E_A(G_\theta, \mathcal{S}_u, \Delta)$. □