Open access • Journal Article • DOI:10.1162/NECO.2007.19.3.792

Support Vector Ordinal Regression — Source link

Wei Chu, S. Sathiya Keerthi

Institutions: Columbia University, Yahoo!

Published on: 01 Mar 2007 - Neural Computation (MIT Press)

Topics: Ordinal regression, Ordinal data, Ordinal optimization, Sequential minimal optimization and Support vector machine

Support Vector Ordinal Regression

Wei Chu CHUWEI@CCLS.COLUMBIA.EDU

Center for Computational Learning Systems, Columbia University, New York, NY 10115 USA

S. Sathiya Keerthi SELVARAK@YAHOO-INC.COM

Yahoo! Research, Media Studios North, Burbank, CA 91504 USA

Abstract

In this paper, we propose two new support vector approaches for ordinal regression, which optimize multiple thresholds to define parallel discriminant hyperplanes for the ordinal scales. *Both approaches guarantee that the thresholds are properly ordered at the optimal solution.* The size of these optimization problems is linear in the number of training samples. The SMO algorithm is adapted for the resulting optimization problems; it is extremely easy to implement and scales efficiently as a quadratic function of the number of examples. The results of numerical experiments on some benchmark and real-world data sets, including applications of ordinal regression to information retrieval and collaborative filtering, verify the usefulness of these approaches.

1 Introduction

We consider the supervised learning problem of predicting variables of ordinal scale, a setting that bridges metric regression and classification, and referred to as *ranking learning* or *ordinal regression.* Ordinal regression arises frequently in social science and information retrieval where human preferences play a major role. The training samples are labelled by ranks, which exhibits an ordering among the different categories. In contrast to metric regression problems, these ranks are of finite types and the metric distances between the ranks are not defined. These ranks are also different from the labels of multiple classes in classification problems due to the existence of the ordering information.

There are several approaches to tackle ordinal regression problems in the domain of machine learning. The naive idea is to transform the ordinal scales into numeric values, and then solve the problem as a standard regression problem. Kramer et al. (2001) investigated the use of a regression tree learner in this way. A problem with this approach is that there might be no principled way of devising an appropriate mapping function since the true metric distances between the ordinal scales are unknown in most of the tasks. Another idea is to decompose the original ordinal regression problem into a set of binary classification tasks. Frank and Hall (2001) converted an ordinal regression problem into nested binary classification problems that encode the ordering of the original ranks and then organized the results of these binary classifiers in some *ad hoc* way for prediction. It is also possible to formulate the original problem as a large augmented binary classification problem. Har-Peled et al. (2002) proposed a constraint classification approach that provides a unified framework for solving ranking and multi-classification problems. Herbrich et al. (2000) applied the principle of Structural Risk Minimization (Vapnik, 1995) to ordinal regression leading to a new distribution-independent learning algorithm based on a loss function between pairs of ranks. The main difficulty with these two algorithms (Har-Peled et al., 2002; Herbrich et al., 2000) is that the problem size of these formulations is a quadratic function of the training data size. As for sequential learning, Crammer and Singer (2002) proposed a proceptron-based online algorithm for rank prediction, known as the PRank algorithm.

Shashua and Levin (2003) generalized the support vector formulation for ordinal regression by finding $r-1$ thresholds that divide the real line into r consecutive intervals for the r ordered categories. However there is a problem with their approach: the ordinal inequalities on the thresholds, $b_1 \leq b_2 \leq \ldots \leq b_{r-1}$, are not included in their formulation. This omission may result in disordered thresholds at the solution on some unfortunate cases (see section 5.1 for an example). It is important to make any method free of bad situations where it will not work.

In this paper, we propose two new approaches for support vector ordinal regression. The first one takes only the adjacent ranks into account in determining the thresholds, exactly as Shashua and Levin (2003) proposed, but we introduce explicit constraints in the problem formulation that enforce the inequalities on the thresholds. The second approach is entirely new; it considers the training samples from all the ranks to determine each threshold. Interestingly, we show that, in this second approach, the ordinal inequality constraints on the thresholds are automatically satisfied at the optimal solution though there are no explicit constraints on these thresholds. For both approaches the size of the optimization problems is linear in the number of training samples. We show that the popular SMO algorithm (Platt, 1999; Keerthi et al., 2001) for SVMs can be easily adapted for the two approaches. The resulting algorithms scale efficiently; empirical analysis shows that the cost is roughly a quadratic function of the problem size. Using several benchmark datasets we demonstrate that the generalization capabilities of the two approaches are much better than that of the naive approach of doing standard regression on the ordinal labels.

The paper is organized as follows. In section 2 we present the first approach with explicit inequality constraints on the thresholds, derive the optimality conditions for the dual problem, and adapt the SMO algorithm for the solution. In section 4 we present the second approach with implicit constraints. In section 5 we do an empirically study to show the scaling properties of the two algorithms and their generalization performance. We conclude in section 6.

Notations Throughout this paper we will use x to denote the input vector of the ordinal regression problem and $\phi(x)$ to denote the feature vector in a high dimensional reproducing kernel Hilbert space (RKHS) related to x by transformation. All computations will be done using the reproducing kernel function only, which is defined as

$$\mathcal{K}(x, x') = \langle \phi(x) \cdot \phi(x') \rangle \tag{1}$$

where $\langle \cdot \rangle$ denotes inner product in the RKHS. Without loss of generality, we consider an ordinal regression problem with r ordered categories and denote these categories as consecutive integers $\boldsymbol{Y} = \{1, 2, \ldots, r\}$ to keep the known ordering information. In the j-th category, where $j \in \boldsymbol{Y}$, the number of training samples is denoted as n^j, and the i-th training sample is denoted as x_i^j

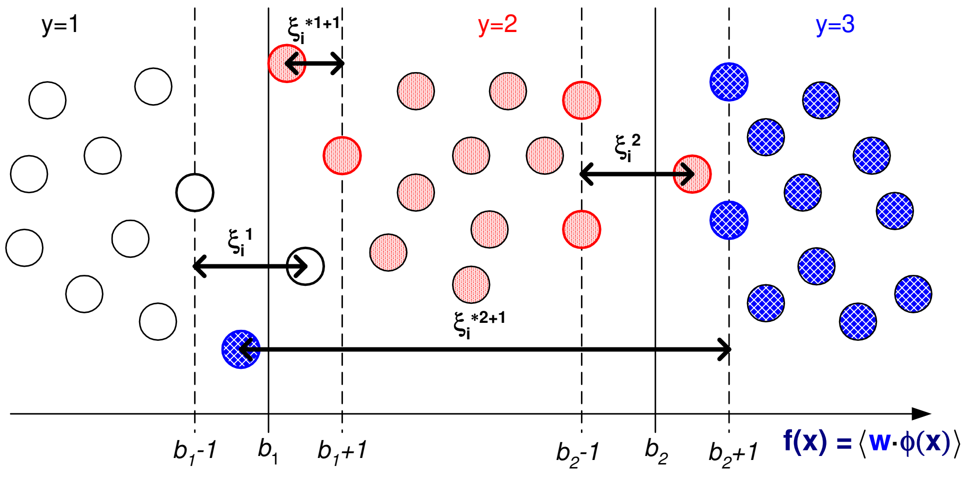


Figure 1: An illustration of the definition of slack variables $\boldsymbol{\xi}$ and $\boldsymbol{\xi}^*$ for the thresholds. The samples from different ranks, represented as circles filled with different patterns, are mapped by $\langle \boldsymbol{w} \cdot \phi(x) \rangle$ onto the axis of function value. Note that a sample from rank $j+1$ could be counted twice for errors if it is sandwiched by $b_{j+1}-1$ and b_j+1 where $b_{j+1}-1 < b_j+1$, and *the samples from rank $j+2$, $j-1$ etc. never give contributions to the threshold b_j.*

where $x_i^j \in \mathbb{R}^d$. The total number of training samples $\sum_{j=1}^r n^j$ is denoted as n. b_j, $j = 1, \ldots, r-1$ denote the $(r-1)$ thresholds.

2 Approach 1: Explicit Constraints on Thresholds

As a powerful computational tool for supervised learning, support vector machines (SVMs) map the input vectors into feature vectors in a high dimensional RKHS (Vapnik, 1995; Schölkopf and Smola, 2002), where a linear machine is constructed by minimizing a regularized functional. For binary classification (a special case of ordinal regression with $r = 2$), SVMs find an optimal direction that maps the feature vectors into function values on the real line, and a single optimized threshold is used to divide the real line into two regions for the two classes respectively. In the setting of ordinal regression, the support vector formulation could attempt to find an optimal mapping direction $\boldsymbol{w}$, and $r-1$ thresholds, which define $r-1$ parallel discriminant hyperplanes for the r ranks accordingly. For each threshold b_j, Shashua and Levin (2003) suggested considering the samples from the two adjacent categories, j and $j+1$, for empirical errors (see Figure 1 for an illustration). More exactly, each sample in the j-th category should have a function value that is less than the lower margin $b_j - 1$, otherwise $\langle \boldsymbol{w} \cdot \phi(x_i^j) \rangle - (b_j - 1)$ is the error (denoted as ξ_i^j); similarly, each sample from the $(j+1)$-th category should have a function value that is greater than the upper margin b_j+1, otherwise $(b_j+1) - \langle \boldsymbol{w} \cdot \phi(x_i^{j+1}) \rangle$ is the error (denoted as ξ_i^{*j+1}).[1] Shashua and Levin (2003) generalized the *primal* problem of SVMs to ordinal regression as follows:

$$\min_{\boldsymbol{w},\boldsymbol{b},\boldsymbol{\xi},\boldsymbol{\xi}^*} \frac{1}{2}\langle \boldsymbol{w} \cdot \boldsymbol{w} \rangle + C \sum_{j=1}^{r-1} \left(\sum_{i=1}^{n^j} \xi_i^j + \sum_{i=1}^{n^{j+1}} \xi_i^{*j+1} \right) \tag{2}$$

subject to

$$\begin{aligned} &\langle \boldsymbol{w} \cdot \phi(x_i^j) \rangle - b_j \leq -1 + \xi_i^j, \\ &\xi_i^j \geq 0, \qquad \text{for } i = 1, \ldots, n^j; \\ &\langle \boldsymbol{w} \cdot \phi(x_i^{j+1}) \rangle - b_j \geq +1 - \xi_i^{*j+1}, \\ &\xi_i^{*j+1} \geq 0, \qquad \text{for } i = 1, \ldots, n^{j+1}; \end{aligned} \tag{3}$$

where j runs over $1, \ldots, r-1$ and $C > 0$.

A problem with the above formulation is that the natural ordinal inequalities on the thresholds, i.e., $b_1 \leq b_2 \leq \ldots \leq b_{r-1}$ cannot be guaranteed to hold at the solution. To tackle this problem, we explicitly include the following constraints in (3):

$$b_{j-1} \leq b_j, \quad \text{for } j = 2, \ldots, r-1. \tag{4}$$

2.1 Primal and Dual Problems

By introducing two auxiliary variables $b_0 = -\infty$ and $b_r = +\infty$, the modified *primal* problem in (2)–(4) can be equivalently written as follows:

$$\min_{\boldsymbol{w},\boldsymbol{b},\boldsymbol{\xi},\boldsymbol{\xi}^*} \quad \frac{1}{2}\langle \boldsymbol{w} \cdot \boldsymbol{w} \rangle + C \sum_{j=1}^{r} \sum_{i=1}^{n^j} \left(\xi_i^j + \xi_i^{*j} \right) \tag{5}$$

[1]The superscript $*$ in ξ_i^{*j+1} denotes that the error is associated with a sample in the adjacent upper category of the j-th threshold.

subject to

$$\begin{aligned} &\langle \boldsymbol{w} \cdot \phi(x_i^j) \rangle - b_j \leq -1 + \xi_i^j, && \xi_i^j \geq 0, \forall i, j; \\ &\langle \boldsymbol{w} \cdot \phi(x_i^j) \rangle - b_{j-1} \geq +1 - \xi_i^{*j}, && \xi_i^{*j} \geq 0, \forall i, j; \\ &b_{j-1} \leq b_j, \forall j. \end{aligned} \tag{6}$$

The *dual* problem can be derived by standard Lagrangian techniques. Let $\alpha_i^j \geq 0$, $\gamma_i^j \geq 0$, $\alpha_i^{*j} \geq 0$, $\gamma_i^{*j} \geq 0$ and $\mu^j \geq 0$ be the Lagrangian multipliers for the inequalities in (6). The Lagrangian for the *primal* problem is:

$$\begin{aligned} &\mathcal{L}_e = \tfrac{1}{2}\langle \boldsymbol{w} \cdot \boldsymbol{w} \rangle + C \textstyle\sum_{j=1}^{r} \sum_{i=1}^{n^j} \left(\xi_i^j + \xi_i^{*j} \right) \\ &- \textstyle\sum_{j=1}^{r} \sum_{i=1}^{n^j} \alpha_i^j (-1 + \xi_i^j - \langle \boldsymbol{w} \cdot \phi(x_i^j) \rangle + b_j) \\ &- \textstyle\sum_{j=1}^{r} \sum_{i=1}^{n^j} \alpha_i^{*j} (-1 + \xi_i^{*j} + \langle \boldsymbol{w} \cdot \phi(x_i^{*j}) \rangle - b_{j-1}) \\ &- \textstyle\sum_{j=1}^{r} \sum_{i=1}^{n^j} \gamma_i^j \xi_i^j - \sum_{j=1}^{r} \sum_{i=1}^{n^j} \gamma_i^{*j} \xi_i^{*j} - \sum_{j=1}^{r} \mu^j (b_j - b_{j-1}). \end{aligned} \tag{7}$$

The KKT conditions for the *primal* problem require the following to hold:

$$\tfrac{\partial \mathcal{L}_e}{\partial \boldsymbol{w}} = \boldsymbol{w} - \textstyle\sum_{j=1}^{r} \sum_{i=1}^{n^j} \left(\alpha_i^{*j} - \alpha_i^j \right) \phi(x_i^j) = 0; \tag{8}$$

$$\tfrac{\partial \mathcal{L}_e}{\partial \xi_i^j} = C - \alpha_i^j - \gamma_i^j = 0, \ \ \forall i, \ \ \forall j; \tag{9}$$

$$\tfrac{\partial \mathcal{L}_e}{\partial \xi_i^{*j}} = C - \alpha_i^{*j} - \gamma_i^{*j} = 0, \ \ \forall i, \ \ \forall j; \tag{10}$$

$$\tfrac{\partial \mathcal{L}_e}{\partial b_j} = - \textstyle\sum_{i=1}^{n^j} \alpha_i^j - \mu^j + \sum_{i=1}^{n^{j+1}} \alpha_i^{*j+1} + \mu^{j+1} = 0, \forall j.$$

Note that the dummy variables associated with b_0 and b_r, i.e. μ^1, μ^r, α_i^{*1}'s and α_i^r's, are always zero. The conditions (9) and (10) give rise to the constraints $0 \leq \alpha_i^j \leq C$ and $0 \leq \alpha_i^{*j} \leq C$ respectively. Let us now apply Wolfe duality theory to the *primal* problem. By introducing the KKT conditions (8)–(10) into the Lagrangian (7) and applying the kernel trick (1), the *dual* problem becomes a maximization problem involving the Lagrangian multipliers $\boldsymbol{\alpha}$, $\boldsymbol{\alpha}^*$ and $\boldsymbol{\mu}$:

$$\max_{\boldsymbol{\alpha}, \boldsymbol{\alpha}^*, \boldsymbol{\mu}} \sum_{j,i} (\alpha_i^j + \alpha_i^{*j}) - \frac{1}{2} \sum_{j,i} \sum_{j',i'} (\alpha_i^{*j} - \alpha_i^j)(\alpha_{i'}^{*j'} - \alpha_{i'}^{j'}) \mathcal{K}(x_i^j, x_{i'}^{j'}) \tag{11}$$

subject to

$$\begin{aligned}
&0 \le \alpha_i^j \le C, && \forall i,\ \forall j, \\
&0 \le \alpha_i^{*j+1} \le C, && \forall i,\ \forall j, \\
&\textstyle\sum_{i=1}^{n^j} \alpha_i^j + \mu^j = \sum_{i=1}^{n^{j+1}} \alpha_i^{*j+1} + \mu^{j+1}, && \forall j, \\
&\mu^j \ge 0, && \forall j,
\end{aligned} \tag{12}$$

where j runs over $1, \ldots, r-1$. Leaving the dummy variables out of account, the size of the optimization problem is $2n - n^1 - n^r$ ($\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}^*$) plus $r-2$ (for $\boldsymbol{\mu}$).

The *dual* problem (11)–(12) is a convex quadratic programming problem. Once the $\boldsymbol{\alpha}$, $\boldsymbol{\alpha}^*$ and $\boldsymbol{\mu}$ are obtained by solving this problem, $\boldsymbol{w}$ is obtained from (8). The determination of the b_j's will be addressed in the next section. The discriminant function value for a new input vector x is

$$f(x) = \langle \boldsymbol{w} \cdot \phi(x) \rangle = \sum_{j,i} (\alpha_i^{*j} - \alpha_i^j) \mathcal{K}(x_i^j, x). \tag{13}$$

The predictive ordinal decision function is given by $\arg\min_i \{i : f(x) < b_i\}$.

2.2 Optimality Conditions for the Dual

To derive proper stopping conditions for algorithms that solve the *dual* problem and also determine the thresholds b_j's, it is important to write down the optimality conditions for the *dual*. Though the resulting conditions that are derived below look a bit clumsy because of the notations, the ideas behind them are very much similar to those for the binary SVM classifier case, as the optimization problem is actually composed of $r-1$ binary classifiers with a shared mapping direction $\boldsymbol{w}$ and the additional constraint (4). The Lagrangian for the *dual* can be written down as follows:

$$\begin{aligned}
&\mathcal{L}_d = \tfrac{1}{2} \textstyle\sum_{j,i} \sum_{j',i'} (\alpha_i^{*j} - \alpha_i^j)(\alpha_{i'}^{*j'} - \alpha_{i'}^{j'}) \mathcal{K}(x_i^j, x_{i'}^{j'}) \\
&+ \textstyle\sum_{j=1}^{r-1} \beta_j (\sum_{i=1}^{n^j} \alpha_i^j - \sum_{i=1}^{n^{j+1}} \alpha_i^{*j+1} + \mu^j - \mu^{j+1}) \\
&- \textstyle\sum_{j,i} (\eta_i^j \alpha_i^j + \eta_i^{*j} \alpha_i^{*j}) - \sum_{j,i} (\pi_i^j (C - \alpha_i^j) \\
&+ \pi_i^{*j} (C - \alpha_i^{*j})) - \textstyle\sum_{j=2}^{r-1} \lambda^j \mu^j - \sum_{j,i} (\alpha_i^j + \alpha_i^{*j})
\end{aligned}$$

where the Lagrangian multipliers η_i^j, η_i^{*j}, π_i^j, π_i^{*j} and λ^j are non-negative, while β_j can take any value.

The KKT conditions associated with $\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}^*$ can be given as follows:

$$\begin{aligned}
&\frac{\partial \mathcal{L}_d}{\partial \alpha_i^j} = -f(x_i^j) - 1 - \eta_i^j + \pi_i^j + \beta_j = 0, \pi_i^j \geq 0,\\
&\eta_i^j \geq 0, \pi_i^j(C - \alpha_i^j) = 0, \eta_i^j \alpha_i^j = 0, \text{ for } i = 1, \ldots, n^j;\\
&\frac{\partial \mathcal{L}_d}{\partial \alpha_i^{*j+1}} = f(x_i^{j+1}) - 1 - \eta_i^{*j+1} + \pi_i^{*j+1} - \beta_j = 0,\\
&\pi_i^{*j+1} \geq 0, \eta_i^{*j+1} \geq 0, \pi_i^{*j+1}(C - \alpha_i^{*j+1}) = 0,\\
&\eta_i^{*j+1}\alpha_i^{*j+1} = 0, \text{ for } i = 1, \ldots, n^{j+1};
\end{aligned} \tag{14}$$

where $f(x)$ is defined as in (13) and j runs over $1, \ldots, r-1$, while the KKT conditions associated with the μ^j are

$$\beta_j - \beta_{j-1} - \lambda^j = 0, \ \lambda^j \mu^j = 0, \ \lambda^j \geq 0, \tag{15}$$

where $j = 2, \ldots, r-1$. The conditions in (14) can be re-grouped into the following six cases:

$$\begin{array}{lll}
\text{case 1}: & \alpha_i^j = 0 & f(x_i^j) + 1 \leq \beta_j \\
\text{case 2}: & 0 < \alpha_i^j < C & f(x_i^j) + 1 = \beta_j \\
\text{case 3}: & \alpha_i^j = C & f(x_i^j) + 1 \geq \beta_j \\
\text{case 4}: & \alpha_i^{*j+1} = 0 & f(x_i^{j+1}) - 1 \geq \beta_j \\
\text{case 5}: & 0 < \alpha_i^{*j+1} < C & f(x_i^{j+1}) - 1 = \beta_j \\
\text{case 6}: & \alpha_i^{*j+1} = C & f(x_i^{j+1}) - 1 \leq \beta_j
\end{array}$$

We can classify any variable into one of the following six sets:

$$\begin{aligned}
&I_{0a}^j \stackrel{\text{def}}{=} \{i \in \{1, \ldots, n^j\} : 0 < \alpha_i^j < C\}\\
&I_{0b}^j \stackrel{\text{def}}{=} \{i \in \{1, \ldots, n^{j+1}\} : 0 < \alpha_i^{*j+1} < C\}\\
&I_1^j \stackrel{\text{def}}{=} \{i \in \{1, \ldots, n^{j+1}\} : \alpha_i^{*j+1} = 0\}\\
&I_2^j \stackrel{\text{def}}{=} \{i \in \{1, \ldots, n^j\} : \alpha_i^j = 0\}\\
&I_3^j \stackrel{\text{def}}{=} \{i \in \{1, \ldots, n^j\} : \alpha_i^j = C\}\\
&I_4^j \stackrel{\text{def}}{=} \{i \in \{1, \ldots, n^{j+1}\} : \alpha_i^{*j+1} = C\}
\end{aligned}$$

Let us define $I_0^j \overset{\text{def}}{=} I_{0a}^j \cup I_{0b}^j$, $I_{up}^j \overset{\text{def}}{=} I_0^j \cup I_1^j \cup I_3^j$ and $I_{low}^j \overset{\text{def}}{=} I_0^j \cup I_2^j \cup I_4^j$. We further define $F_{up}^i(\beta_j)$ on the set I_{up}^j as

$$F_{up}^i(\beta_j) = \begin{cases} f(x_i^j) + 1 & \text{if} \quad i \in I_{0a}^j \cup I_3^j \\ f(x_i^{j+1}) - 1 & \text{if} \quad i \in I_{0b}^j \cup I_1^j \end{cases}$$

and $F_{low}^i(\beta_j)$ on the set I_{low}^j as

$$F_{low}^i(\beta_j) = \begin{cases} f(x_i^j) + 1 & \text{if} \quad i \in I_{0a}^j \cup I_2^j \\ f(x_i^{j+1}) - 1 & \text{if} \quad i \in I_{0b}^j \cup I_4^j \end{cases}$$

Then the conditions can be simplified as

$$\beta_j \le F_{up}^i(\beta_j)\, \forall i \in I_{up}^j \text{ and } \beta_j \ge F_{low}^i(\beta_j)\, \forall i \in I_{low}^j,$$

which can be compactly written as:

$$b_{low}^j \le b_{up}^j \quad \text{for } j = 1, \ldots, r-1 \tag{16}$$

where $b_{up}^j = \min\{F_{up}^i(\beta_j) : i \in I_{up}^j\}$ and $b_{low}^j = \max\{F_{low}^i(\beta_j) : i \in I_{low}^j\}$.

The KKT conditions in (15) indicate that the condition, $\beta_{j-1} \le \beta_j$ always holds, and that $\beta_{j-1} = \beta_j$ if $\mu^j > 0$. To merge the conditions (15) and (16), let us define

$$\tilde{B}_{low}^j = \max\{b_{low}^k : k = 1, \ldots, j\} \text{ and } \tilde{B}_{up}^j = \min\{b_{up}^k : k = j, \ldots, r-1\},$$

where $j = 1, \ldots, r-1$. The overall optimality conditions can be simply written as

$$B_{low}^j \le B_{up}^j \ \ \forall j \tag{17}$$

where

$$B_{low}^j = \begin{cases} \tilde{B}_{low}^{j+1} & \text{if } \mu^{j+1} > 0 \\ \tilde{B}_{low}^j & \text{otherwise} \end{cases} \text{ and } B_{up}^j = \begin{cases} \tilde{B}_{up}^{j-1} & \text{if } \mu^j > 0 \\ \tilde{B}_{up}^j & \text{otherwise.} \end{cases} \tag{18}$$

It should be easy to see from the above sequence of steps that, $\{\beta_j\}$ *and* $\{\mu_j\}$ *are optimal for* *the problem in (11)–(12) iff (17) is satisfied.*

We introduce a tolerance parameter $\tau > 0$, usually 0.001, to define approximate optimality conditions. The overall stopping condition becomes

$$\max\{B^j_{low} - B^j_{up} : j = 1, \ldots, r-1\} \leq \tau. \tag{19}$$

From the conditions in (14) and (3), it is easy to see the close relationship between the b_j's in the *primal* problem and the multipliers β_j's. In particular, at the optimal solution, β_j and b_j are identical. Thus b_j can be taken to be any value from the interval, $[B^j_{low}, B^j_{up}]$. We can resolve any non-uniqueness by simply taking $b_j = \frac{1}{2}(B^j_{low} + B^j_{up})$. Note that the KKT conditions in (15), coming from the additive constraints in (4) we introduced in Shashua and Levin's formulation, enforce $B^{j-1}_{low} \leq B^j_{low}$ and $B^{j-1}_{up} \leq B^j_{up}$ at the solution, which guarantee that the thresholds specified in these feasible regions will satisfy the inequality constraints $b_{j-1} \leq b_j$; without the constraints in (4), the thresholds might be disordered at the solution! Here we essentially consider an optimization problem of multiple learning tasks where individual optimality conditions are coupled together by a joint constraint. We believe this is a non-trivial extension and useful in other applications.

3 SMO Algorithm

In this section we adapt the SMO algorithm (Platt, 1999; Keerthi et al., 2001) for the solution of (11)–(12). The key idea of SMO consists of starting with a valid initial point and optimizing only one pair of variables at a time while fixing all the other variables. The suboptimization problem of the two active variables can be solved analytically. Table 1 presents an outline of the SMO implementation for our optimization problem.

In order to determine the pair of *active variables* to optimize, we select the *active threshold* first. The index of the *active threshold* is defined as $J = \arg\max_j\{B^j_{low} - B^j_{up}\}$. Let us assume that B^J_{low} and B^J_{up} are actually defined by $b^{j_o}_{low}$ and $b^{j_u}_{up}$ respectively, and that the two multipliers associated with $b^{j_o}_{low}$ and $b^{j_u}_{up}$ are α_o and α_u. The pair of multipliers (α_o, α_u) is optimized from

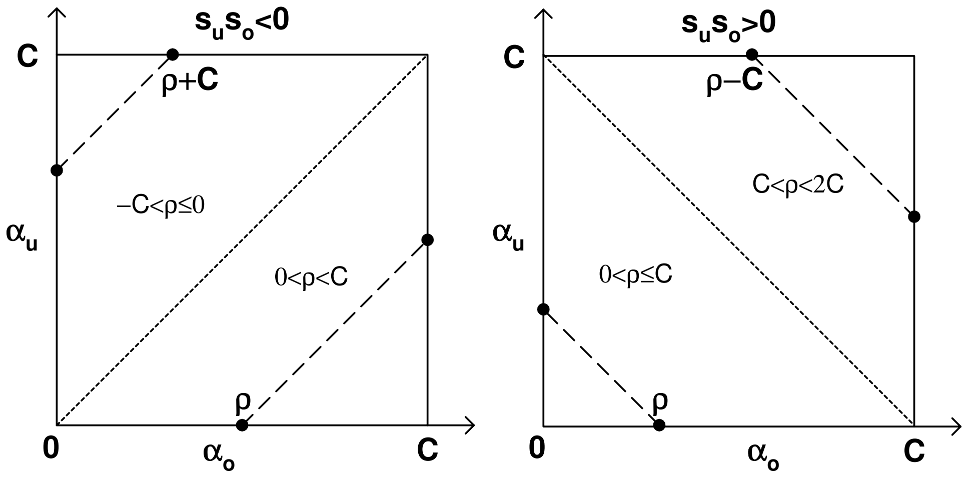


Figure 2: An illustration of the boundary for the active variables. The cases corresponding to $s_u s_o < 0$ and $s_u s_o > 0$ are presented in the left and right part respectively. The dotted line separates the box into two regions for ρ with different values. α_o and α_u need to be updated jointly along the dashed line within the box, due to the constraints in (12).

Table 1: The basic framework of the SMO algorithm for support vector ordinal regression using explicit threshold constraints.

SMO	start at a valid point, $\boldsymbol{\alpha}$, $\boldsymbol{\alpha}^*$ and $\boldsymbol{\mu}$, that satisfy (12), find the current B^j_{up} and B^j_{low} $\forall j$
Loop	do 1. determine the *active threshold* J 2. optimize the pair of active variables and the set $\boldsymbol{\mu}_a$ 3. compute B^j_{up} and B^j_{low} $\forall j$ at the new point while the optimality condition (19) has not been satisfied
Exit	return $\boldsymbol{\alpha}$, $\boldsymbol{\alpha}^*$ and $\boldsymbol{b}$

the current point $(\alpha_o^{old}, \alpha_u^{old})$ to reach the new point, $(\alpha_o^{new}, \alpha_u^{new})$.

It is possible that $j_o \neq j_u$. In this case, named as *cross update*, more than one equality constraint in (12) is involved in the optimization that may update the variable set $\boldsymbol{\mu}_a = \{\mu^{\min\{j_o,j_u\}+1}, \ldots, \mu^{\max\{j_o,j_u\}}\}$, a subset of $\boldsymbol{\mu}$. In the case of $j_o = j_u$, named as *standard update*, only one equality constraint is involved and the variables of $\boldsymbol{\mu}$ are kept intact, i.e. $\boldsymbol{\mu}_a = \emptyset$. These suboptimization problems can be solved analytically, and the detailed formulas for updating are given in the following.

The following equality constraints have to be taken into account in the consequent suboptimization problem:

$$\sum_{i=1}^{n^j} \alpha_i^j + \mu^j = \sum_{i=1}^{n^{j+1}} \alpha_i^{*j+1} + \mu^{j+1}, \quad \min\{j_o, j_u\} \le j \le \max\{j_o, j_u\}. \tag{20}$$

Under these constraints the suboptimization problem for α_o, α_u and $\boldsymbol{\mu}_a$ can be written as

$$\min_{\alpha_o,\alpha_u,\boldsymbol{\mu}_a} \frac{1}{2} \sum_{j,i} \sum_{j',i'} (\alpha_i^{*j} - \alpha_i^j)(\alpha_{i'}^{*j'} - \alpha_{i'}^{j'}) \mathcal{K}(x_i^j, x_{i'}^{j'}) - \alpha_o - \alpha_u \tag{21}$$

subject to the bound constraints $0 \le \alpha_o \le C$, $0 \le \alpha_u \le C$ and $\boldsymbol{\mu}_a \ge 0$. The equality constraints (20) can be summarized for α_o and α_u as a linear constraint $\alpha_o + s_o s_u \alpha_u = \alpha_o^{old} + s_o s_u \alpha_u^{old} = \rho$, where

$$s_o = \begin{cases} -1 & \text{if} \quad i_o \in I_{0a}^{j_o} \cup I_2^{j_o} \\ +1 & \text{if} \quad i_o \in I_{0b}^{j_o} \cup I_4^{j_o} \end{cases}$$
$$s_u = \begin{cases} -1 & \text{if} \quad i_u \in I_{0a}^{j_u} \cup I_3^{j_u} \\ +1 & \text{if} \quad i_u \in I_{0b}^{j_u} \cup I_1^{j_u} \end{cases} \tag{22}$$

It is quite straightforward to verify that, for the above optimization problem which is restricted to α_o, α_u and $\boldsymbol{\mu}_a$, one can use the same derivations that led to (17) to show that the optimality conditions for the restricted problem is nothing but $b_{low}^{j_o} \le b_{up}^{j_u}$. Since we started with this condition being violated, we are guaranteed that the solution of the restricted optimization problem will lead to a strict improvement in the objective function.

To solve the restricted optimization problem, first consider the case of a positive definite kernel matrix. For this case, the unbounded solution can be exactly determined as

$$\begin{aligned} \alpha_o^{new} &= \alpha_o^{old} + s_o \Delta\mu \\ \alpha_u^{new} &= \alpha_u^{old} - s_u \Delta\mu \\ \mu_{new}^j &= \mu_{old}^j - d_\mu \Delta\mu \quad \forall \mu^j \in \boldsymbol{\mu}_a \end{aligned} \tag{23}$$

where

$$d_\mu = \begin{cases} +1 & \text{if} \quad j_o \le j_u \\ -1 & \text{if} \quad j_o > j_u \end{cases} \tag{24}$$

and

$$\Delta\mu = \frac{-f(x_o) + f(x_u) + s_o - s_u}{\mathcal{K}(x_o, x_o) + \mathcal{K}(x_u, x_u) - 2\mathcal{K}(x_o, x_u)}, \tag{25}$$

Now we need to check the box bound constraint on α_o and α_u and the constraint $\mu^j \geq 0$. We have to adjust $\Delta\mu$ towards 0 to draw the out-of-range solution back to the nearest boundary point of the feasible region (see Figure 2, and update variables accordingly as in (23). Note that *the cross update allows α_o and α_u to be associated with the same sample.* For the positive semi-definite kernel matrix case, the denominator of (25) can vanish. Still, leaving out the denominator defines the descent direction which needs to be traversed till the boundary is hit, in order to get the optimal solution.

It should be noted that, working with just any violating pair of variables does not automatically ensure the convergence of the SMO algorithm. However, since the *most violating pair* is always chosen, the ideas of Keerthi and Gilbert (2002) can be adapted to prove the convergence of the SMO algorithm.

4 Approach 2: Implicit Constraints on Thresholds

In this section we present a new approach to support vector ordinal regression. Instead of considering only the empirical errors from the samples of adjacent categories to determine a threshold, we allow the samples in all the categories to contribute errors for each threshold. This kind of loss functions was also suggested by Srebro et al. (2005) for collaborative filtering. A very nice property of this approach is that the ordinal inequalities on the thresholds are satisfied automatically at the optimal solution in spite of the fact that such constraints on the thresholds are not explicitly included in the new formulation. We give a proof on this property in the following section.

Figure 3 explains the new definition of slack variables $\boldsymbol{\xi}$ and $\boldsymbol{\xi}^*$. For a threshold b_j, the function values of all the samples from all the lower categories, should be less than the lower margin $b_j - 1$; if that does not hold, then $\xi^j_{ki} = \langle \boldsymbol{w} \cdot \phi(x^k_i) \rangle - (b_j - 1)$ is taken as the error associated with the sample x^k_i for b_j, where $k \leq j$. Similarly, the function values of all the samples from the upper categories should be greater than the upper margin $b_j + 1$; otherwise $\xi_{ki}^{*j} = (b_j + 1) - \langle \boldsymbol{w} \cdot \phi(x_i^k) \rangle$ is the error associated with the sample x_i^k for b_j, where $k > j$. Here, the subscript ki denotes that the slack variable is associated with the i-th input sample in the k-th category; the superscript j denotes that the slack variable is associated with the lower categories of b_j; and the superscript $*j$ denotes that the slack variable is associated with the upper categories of b_j.

4.1 Primal Problem

By taking all the errors associated with all $r - 1$ thresholds into account, the *primal* problem can be defined as follows:

$$\min_{\boldsymbol{w}, \boldsymbol{b}, \boldsymbol{\xi}, \boldsymbol{\xi}^*} \frac{1}{2} \langle \boldsymbol{w} \cdot \boldsymbol{w} \rangle + C \sum_{j=1}^{r-1} \left(\sum_{k=1}^{j} \sum_{i=1}^{n^k} \xi_{ki}^{j} + \sum_{k=j+1}^{r} \sum_{i=1}^{n^k} \xi_{ki}^{*j} \right) \tag{26}$$

subject to

$$\begin{aligned} &\langle \boldsymbol{w} \cdot \phi(x_i^k) \rangle - b_j \leq -1 + \xi_{ki}^{j}, \quad \xi_{ki}^{j} \geq 0, \\ &\text{ for } k = 1, \ldots, j \text{ and } i = 1, \ldots, n^k; \\ &\langle \boldsymbol{w} \cdot \phi(x_i^k) \rangle - b_j \geq +1 - \xi_{ki}^{*j}, \quad \xi_{ki}^{*j} \geq 0, \\ &\text{ for } k = j+1, \ldots, r \text{ and } i = 1, \ldots, n^k; \end{aligned} \tag{27}$$

where j runs over $1, \ldots, r - 1$. Note that there are $r - 1$ inequality constraints for each sample x_i^k (one for each threshold).

To prove the inequalities on the thresholds at the optimal solution, let us consider the situation where $\boldsymbol{w}$ is fixed and only the b_j's are optimized. Note that the ξ_{ki}^{j} and ξ_{ki}^{*j} are automatically determined once the b_j are given. To eliminate these variables, let us define, for $1 \leq k \leq r$,

$$\begin{aligned} I_k^{\text{low}}(b) &\overset{\text{def}}{=} \{ i \in \{1, \ldots, n^k\} : \langle \boldsymbol{w} \cdot \phi(x_i^k) \rangle - b \geq -1 \}, \\ I_k^{\text{up}}(b) &\overset{\text{def}}{=} \{ i \in \{1, \ldots, n^k\} : \langle \boldsymbol{w} \cdot \phi(x_i^k) \rangle - b \leq 1 \}. \end{aligned}$$

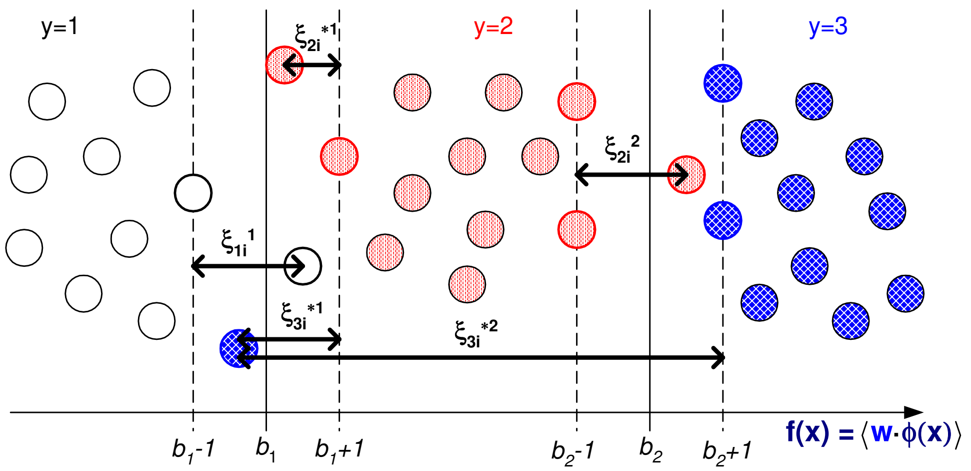


Figure 3: An illustration on the new definition of slack variables $\boldsymbol{\xi}$ and $\boldsymbol{\xi}^*$ that imposes implicit constraints on the thresholds. All the samples are mapped by $\langle \boldsymbol{w} \cdot \phi(x) \rangle$ onto the axis of function values. Note the term ξ_{3i}^{*1} in this graph.

It is easy to see that b_j is optimal iff it minimizes the function

$$e_j(b) = \sum_{k=1}^{j} \sum_{i \in I_k^{\text{low}}(b)} \big(\langle \boldsymbol{w} \cdot \phi(x_i^k) \rangle - b + 1\big) + \sum_{k=j+1}^{r} \sum_{i \in I_k^{\text{up}}(b)} \big(-\langle \boldsymbol{w} \cdot \phi(x_i^k) \rangle + b + 1\big) \tag{28}$$

Let $B_j^\star$ denote the set of all minimizers of $e_j(b)$. By convexity, $B_j^\star$ is a closed interval. Given two intervals $B_1 = [c_1, d_1]$ and $B_2 = [c_2, d_2]$, we say $B_1 \leq B_2$ if $c_1 \leq c_2$ and $d_1 \leq d_2$.

Lemma 1. $B_1^\star \leq B_2^\star \leq \cdots \leq B_{r-1}^\star$

Proof. The "right side derivative" of e_j with respect to b is

$$g_j(b) = -\textstyle\sum_{k=1}^{j} |I_k^{\text{low}}(b)| + \sum_{k=j+1}^{r} |I_k^{\text{up}}(b)| \tag{29}$$

Take any one j and consider $B_j^\star = [c_j, d_j]$ and $B_{j+1}^\star = [c_{j+1}, d_{j+1}]$. Suppose $c_j > c_{j+1}$. Define $b_j^\star = c_j$ and $b_{j+1}^\star = c_{j+1}$. Since $b_{j+1}^\star$ is strictly to the left of the interval $B_j^\star$ that minimizes e_j, we have $g_j(b_{j+1}^\star) < 0$. Since $b_{j+1}^\star$ is a minimizer of e_{j+1} we also have $g_{j+1}(b_{j+1}^\star) \geq 0$. Thus we have $g_{j+1}(b_{j+1}^\star) - g_j(b_{j+1}^\star) > 0$; also, by (29) we get

$$0 < g_{j+1}(b_{j+1}^\star) - g_j(b_{j+1}^\star) = -|I_{j+1}^{\text{low}}(b_{j+1}^\star)| - |I_{j+1}^{\text{up}}(b_{j+1}^\star)|$$

which is impossible. In a similar way, $d_j > d_{j+1}$ is also not possible. This proves the lemma.

If the optimal b_j are all unique,[2] then Lemma 1 implies that the b_j satisfy the natural ordinal ordering. Even when one or more b_j's are non-unique, Lemma 1 says that there exist choices for the b_j that obey the natural ordering. The fact that the order preservation comes about automatically is interesting and non-trivial, which differs from the PRank algorithm (Crammer and Singer, 2002) where the order preservation on the thresholds is easily brought in via their update rule.

It is also worth noting that Lemma 1 holds even for an extended problem formulation that allows the use of different costs (different C values) for different misclassifications (class k misclassified as class j can have a C_k^j). In applications such as collaborative filtering such a problem formulation can be very appropriate; for example, an A rated movie that is misrated as D may need to be penalized much more than if a B rated movie is misrated as D. Shashua and Levin's formulation and its extension given in section 2 of this paper do not precisely support such a differential cost structure. This is another good reason in support of the implicit problem formulation of the current section.

4.2 Dual Problem

Let $\alpha_{ki}^j \geq 0$, $\gamma_{ki}^j \geq 0$, $\alpha_{ki}^{*j} \geq 0$ and $\gamma_{ki}^{*j} \geq 0$ be the Lagrangian multipliers for the inequalities in (27). Using ideas parallel to those in section 2.1 we can show that the *dual* of (26)–(27) is the following maximization problem that involves only the multipliers $\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}^*$:

$$\max_{\boldsymbol{\alpha},\boldsymbol{\alpha}^*} -\frac{1}{2}\sum_{k,i}\sum_{k',i'}\left(\sum_{j=1}^{k-1}\alpha_{ki}^{*j}-\sum_{j=k}^{r-1}\alpha_{ki}^{j}\right)\left(\sum_{j=1}^{k'-1}\alpha_{k'i'}^{*j}-\sum_{j=k'}^{r-1}\alpha_{k'i'}^{j}\right)\mathcal{K}(x_i^k,x_{i'}^{k'})+\sum_{k,i}\left(\sum_{j=1}^{k-1}\alpha_{ki}^{*j}+\sum_{j=k}^{r-1}\alpha_{ki}^{j}\right) \tag{30}$$

subject to

$$\begin{aligned}
&\sum_{k=1}^{j}\sum_{i=1}^{n^k}\alpha_{ki}^j = \sum_{k=j+1}^{r}\sum_{i=1}^{n^k}\alpha_{ki}^{*j} && \forall j\\
&0 \leq \alpha_{ki}^j \leq C && \forall j \text{ and } k \leq j\\
&0 \leq \alpha_{ki}^{*j} \leq C && \forall j \text{ and } k > j.
\end{aligned} \tag{31}$$

[2]If, in the primal problem, we regularize the b_j's as well (i.e., include the extra cost term $\sum_j b_j^2/2$) then the b_j's are guaranteed to be unique. Lemma 1 still holds in this case.

The *dual* problem (30)–(31) is a convex quadratic programming problem. The size of the optimization problem is $(r-1)n$ where $n = \sum_{k=1}^{r} n^k$ is the total number of training samples. The discriminant function value for a new input vector x is

$$f(x) = \langle \boldsymbol{w} \cdot \phi(x) \rangle = \sum_{k,i} \left(\sum_{j=1}^{k-1} \alpha_{ki}^{*j} - \sum_{j=k}^{r-1} \alpha_{ki}^{j} \right) \mathcal{K}(x_i^k, x). \tag{32}$$

The predictive ordinal decision function is given by $\arg\min_{i} \{i : f(x) < b_i\}$.

4.3 Optimality Conditions for the Dual

The Lagrangian for the *dual* problem is

$$\begin{array}{l}
\mathcal{L}_d = \\
\frac{1}{2} \sum_{k,i} \sum_{k',i'} \left(\sum_{j=1}^{k-1} \alpha_{ki}^{*j} - \sum_{j=k}^{r-1} \alpha_{ki}^{j} \right) \left(\sum_{j=1}^{k'-1} \alpha_{k'i'}^{*j} - \sum_{j=k'}^{r-1} \alpha_{k'i'}^{j} \right) \mathcal{K}(x_i^k, x_{i'}^{k'}) \\
- \sum_{k,i} \left(\sum_{j=1}^{k-1} \alpha_{ki}^{*j} + \sum_{j=k}^{r-1} \alpha_{ki}^{j} \right) + \sum_{j=1}^{r-1} \beta_j \left(\sum_{k=1}^{j} \sum_{i=1}^{n^k} \alpha_{ki}^{j} - \sum_{k=j+1}^{r} \sum_{i=1}^{n^k} \alpha_{ki}^{*j} \right) \\
- \sum_{j=1}^{r-1} \sum_{k=1}^{j} \sum_{i=1}^{n^k} \eta_{ki}^{j} \alpha_{ki}^{j} - \sum_{j=1}^{r-1} \sum_{k=j+1}^{r} \sum_{i=1}^{n^k} \eta_{ki}^{*j} \alpha_{ki}^{*j} \\
- \sum_{j=1}^{r-1} \sum_{k=1}^{j} \sum_{i=1}^{n^k} \pi_{ki}^{j} (C - \alpha_{ki}^{j}) - \sum_{j=1}^{r-1} \sum_{k=j+1}^{r} \sum_{i=1}^{n^k} \pi_{ki}^{*j} (C - \alpha_{ki}^{*j})
\end{array} \tag{33}$$

where the Lagrangian multipliers η_{ki}^{j}, η_{ki}^{*j}, π_{ki}^{j} and π_{ki}^{*j} are non-negative, while β_j can take any value. The KKT conditions associated with β_j can be given as follows:

$$\begin{array}{l}
\frac{\partial \mathcal{L}_d}{\partial \alpha_{ki}^{j}} = -f(x_i^k) - 1 - \eta_{ki}^{j} + \pi_{ki}^{j} + \beta_j = 0, \\
\pi_{ki}^{j} \geq 0, \eta_{ki}^{j} \geq 0, \pi_{ki}^{j} (C - \alpha_{ki}^{j}) = 0, \eta_{ki}^{j} \alpha_{ki}^{j} = 0, \text{for } k \leq j \text{ and } \forall i; \\
\frac{\partial \mathcal{L}_d}{\partial \alpha_{ki}^{*j}} = f(x_i^k) - 1 - \eta_{ki}^{*j} + \pi_{ki}^{*j} - \beta_j = 0, \\
\pi_{ki}^{*j} \geq 0, \eta_{ki}^{*j} \geq 0, \pi_{ki}^{*j} (C - \alpha_{ki}^{*j}) = 0, \eta_{ki}^{*j} \alpha_{ki}^{*j} = 0, \text{for } k > j \text{ and } \forall i;
\end{array} \tag{34}$$

where $f(x)$ is as defined in (32). The conditions in (34) can be regrouped into the following six cases:

$$\begin{array}{lll} \text{case 1}: & \alpha_{ki}^j = 0 & f(x_i^k) + 1 \leq \beta_j \\ \text{case 2}: & 0 < \alpha_{ki}^j < C & f(x_i^k) + 1 = \beta_j \\ \text{case 3}: & \alpha_{ki}^j = C & f(x_i^k) + 1 \geq \beta_j \\ \text{case 4}: & \alpha_{ki}^{*j} = 0 & f(x_i^k) - 1 \geq \beta_j \\ \text{case 5}: & 0 < \alpha_{ki}^{*j} < C & f(x_i^k) - 1 = \beta_j \\ \text{case 6}: & \alpha_{ki}^{*j} = C & f(x_i^k) - 1 \leq \beta_j \end{array} \qquad (35)$$

We can classify any variable into one of the following six sets:

$$\begin{aligned} I_{0a}^j &= \left\{ i \in \{1, \ldots, n^k\} : 0 < \alpha_{ki}^j < C, k \leq j \right\} \\ I_{0b}^j &= \left\{ i \in \{1, \ldots, n^k\} : 0 < \alpha_{ki}^{*j} < C, k > j \right\} \\ I_1^j &= \left\{ i \in \{1, \ldots, n^k\} : \alpha_{ki}^{*j} = 0, k > j \right\} \\ I_2^j &= \left\{ i \in \{1, \ldots, n^k\} : \alpha_{ki}^j = 0, k \leq j \right\} \\ I_3^j &= \left\{ i \in \{1, \ldots, n^k\} : \alpha_{ki}^j = C, k \leq j \right\} \\ I_4^j &= \left\{ i \in \{1, \ldots, n^k\} : \alpha_{ki}^{*j} = C, k > j \right\} \end{aligned} \qquad (36)$$

Let us denote $I_0^j = I_{0a}^j \cup I_{0b}^j$, $I_{up}^j = I_0^j \cup I_1^j \cup I_3^j$ and $I_{low}^j = I_0^j \cup I_2^j \cup I_4^j$. We further define F_{up}^j on the set I_{up}^j as

$$F_{up}^j = \begin{cases} f(x_i^k) + 1 & \text{if} \quad i \in I_{0a}^j \cup I_3^j \\ f(x_i^k) - 1 & \text{if} \quad i \in I_{0b}^j \cup I_1^j \end{cases} \qquad (37)$$

and F_{low}^j on the set I_{low}^j as

$$F_{low}^j = \begin{cases} f(x_i^k) + 1 & \text{if} \quad i \in I_{0a}^j \cup I_2^j \\ f(x_i^k) - 1 & \text{if} \quad i \in I_{0b}^j \cup I_4^j \end{cases} \qquad (38)$$

Then the optimality conditions can be simplified as

$$\beta_j \leq F_{up}^j \ \forall i \in I_{up}^j \quad \text{and} \quad \beta_j \geq F_{low}^j \ \forall i \in I_{low}^j, \qquad (39)$$

which can be compactly written as

$$b^j_{low} \leq \beta_j \leq b^j_{up} \tag{40}$$

where $b^j_{up} = \min\{F^j_{up} : i \in I^j_{up}\}$ and $b^j_{low} = \max\{F^j_{low} : i \in I^j_{low}\}$. By introducing the tolerance parameter $\tau > 0$, we get the approximate stopping condition

$$\max\{b^j_{low} - b^j_{up} : j = 1, \ldots, r-1\} \leq \tau \tag{41}$$

4.4 SMO Algorithm

The ideas for adapting SMO to (30)–(31) are similar to those in section 3. The resulting suboptimization problem is analogous to the case of *standard update* in section 3 where only one of the equality constraints from (31) is involved.

The index of *active threshold* can be determined as $J = \arg\max_j\{b^j_{low} - b^j_{up}\}$, and then the two multipliers associated with b^J_{low} and b^J_{up} are used as the *active variables* for the suboptimization problem. Let us denote these active variables as α_o and α_u, the corresponding indices as i_o and i_u, and the corresponding samples as x_o and x_u respectively. The suboptimization problem of (30)–(31) for α_o and α_u becomes

$$\min_{\alpha_o,\alpha_u} S(\alpha_o, \alpha_u) = \frac{1}{2}\sum_{k,i}\sum_{k',i'}\left(\sum_{j=1}^{k-1}\alpha^{*j}_{ki} - \sum_{j=k}^{r-1}\alpha^j_{ki}\right)\left(\sum_{j=1}^{k'-1}\alpha^{*j}_{k'i'} - \sum_{j=k'}^{r-1}\alpha^j_{k'i'}\right)\mathcal{K}(x^k_i, x^{k'}_{i'}) - \alpha_o - \alpha_u \tag{42}$$

subject to the bound constraints $0 \leq \alpha_o \leq C$, $0 \leq \alpha_u \leq C$, and the linear constraint $\alpha_o + s_o s_u \alpha_u = \rho$, where

$$\begin{aligned} s_o &= \begin{cases} -1 & \text{if} \quad i_o \in I^J_{0a} \cup I^J_2 \\ +1 & \text{if} \quad i_o \in I^J_{0b} \cup I^J_4 \end{cases} \\ s_u &= \begin{cases} -1 & \text{if} \quad i_u \in I^J_{0a} \cup I^J_3 \\ +1 & \text{if} \quad i_u \in I^J_{0b} \cup I^J_1 \end{cases} \end{aligned} \tag{43}$$

The index sets used above are as defined in (36). This is analogous to the case of *standard update* in section 3 where only one of the equality constraints from (12) is involved. Due to the linear constraint between α_o and α_u, the unbounded solution to (42) can be exactly determined as

$$\alpha_o^{new} = \alpha_o^{old} + s_o \frac{-f(x_o) + f(x_u) + s_o - s_u}{\mathcal{K}(x_o, x_o) + \mathcal{K}(x_u, x_u) - 2\mathcal{K}(x_o, x_u)} \tag{44}$$

where $f(x)$ is as defined in (32). Next, we check if α_o^{new} satisfies all the box constraints (see Figure 2 for an illustration) and, if it doesn't satisfy them, then we draw it back to the nearest boundary point of the box. Using the final value of α_o^{new}, α_u can be updated as $\alpha_u^{new} = s_u s_o(\rho - \alpha_o^{new})$ where $\rho = \alpha_o^{old} + s_o s_u \alpha_u^{old}$.

5 Numerical Experiments

We have implemented the two SMO algorithms for the ordinal regression formulations with explicit constraints (EXC) and implicit constraints (IMC),[3] along with the algorithm of Shashua and Levin (2003) for comparison purpose. The function caching technique and the double-loop scheme proposed by Keerthi et al. (2001) have been incorporated in the implementation for efficiency. We begin this section with a simple dataset to illustrate the typical behavior of the three algorithms, and then empirically study the scaling properties of our algorithms. Then we compare the generalization performance of our algorithms against standard support vector regression on eight benchmark datasets for ordinal regression. The following Gaussian kernel was used in these experiments:

$$\mathcal{K}(x, x') = \exp\left(-\tfrac{\kappa}{2} \textstyle\sum_{\varsigma=1}^{d} (x_\varsigma - x'_\varsigma)^2\right) \tag{45}$$

where $\kappa > 0$ and x_ς denotes the ς-th element of the input vector x. The tolerance parameter τ was set to 0.001 for all the algorithms. We have utilized two evaluation metrics which quantify the accuracy of predicted ordinal scales $\{\hat{y}_1, \dots, \hat{y}_t\}$ with respect to true targets $\{y_1, \dots, y_t\}$:

a) *Mean absolute error* is the average deviation of the prediction from the true target, i.e. $\frac{1}{t}\sum_{i=1}^{t} |\hat{y}_i - y_i|$, in which we treat the ordinal scales as consecutive integers;

b) *Mean zero-one error* is simply the fraction of incorrect predictions on individual samples.

[3] The source code of the two algorithms written in ANSI C can be found at http://www.gatsby.ucl.ac.uk/~chuwei/svor.htm.

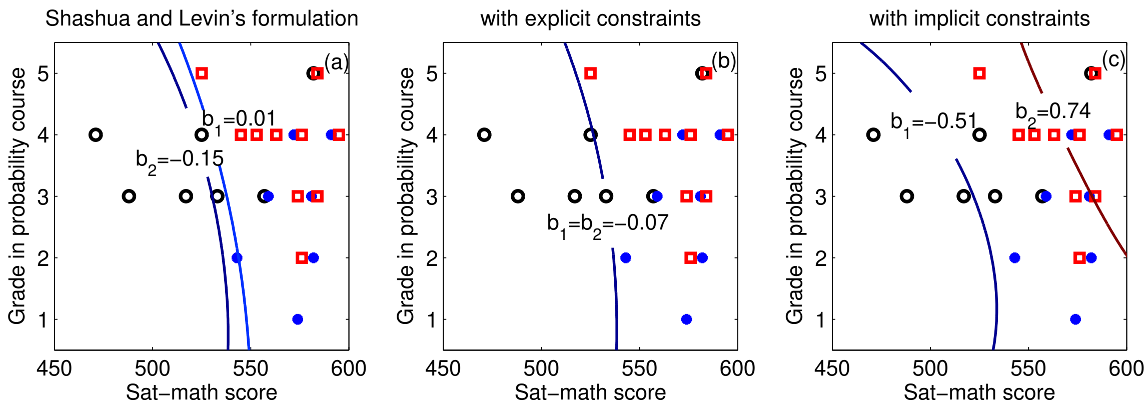


Figure 4: The training results of the three algorithms using a Gaussian kernel on the grading dataset. The discriminant function values are presented as contour graphs indexed by the two thresholds. The circles denote the students with grade D, the dots denote grade C, and the squares denote grade B.

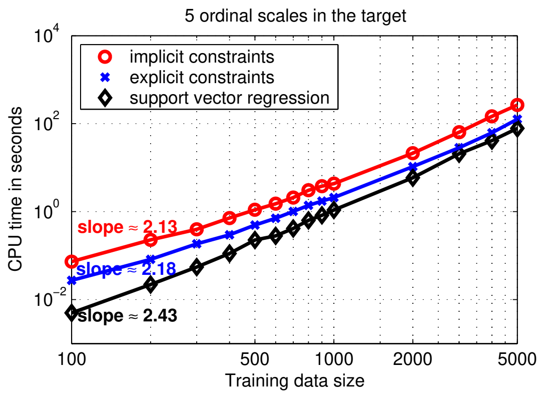
5.1 Grading Dataset

The grading dataset was used in chapter 4 of Johnson and Albert (1999) as an example of the ordinal regression problem.[4] There are 30 samples of students' score. The "sat-math score" and "grade in prerequisite probability course" of these students are used as input features, and their final grades are taken as the targets. In our experiments, the six students with final grade A or E were not used, and the feature associated with the "grade in prerequisite probability course" was treated as a continuous variable though it had an ordinal scale. In Figure 4 we present the solution obtained by the three algorithms using the Gaussian kernel (45) with $\kappa = 0.5$ and the regularization factor value of $C = 1$. In this particular setting, the solution to Shashua and Levin (2003)'s formulation has disordered thresholds $b_2 < b_1$ as shown in Figure 4 (left plot); the formulation with explicit constraints corrects this disorder and yields equal values for the two thresholds as shown in Figure 4 (middle plot).

5.2 Scaling

In this experiment, we empirically studied how the two SMO algorithms scale with respect to training data size and the number of ordinal scales in the target. The California Housing dataset

[4]The grading dataset is available at http://www.mathworks.com/support/books/book1593.jsp.



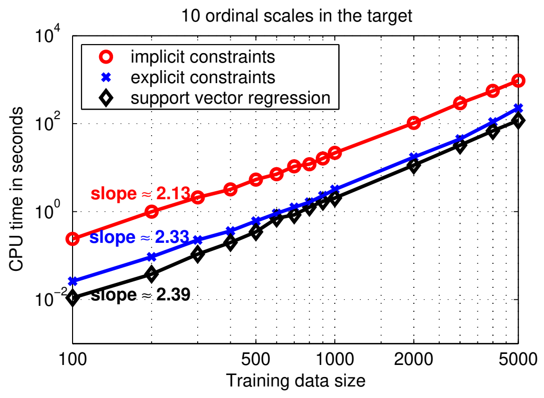


Figure 5: Plots of CPU time versus training data size on $\log - \log$ scale, indexed by the estimated slopes respectively. We used the Gaussian kernel with $\kappa = 1$ and the regularization factor value of $C = 100$ in the experiment.

was used in the scaling experiments.[5] Twenty-eight training datasets with sizes ranging from 100 to 5,000 were generated by random selection from the original dataset. The continuous target variable of the California Housing data was discretized to ordinal scale by using 5 or 10 equal-frequency bins. The standard support vector regression (SVR) was used as a baseline, in which the ordinal targets were treated as continuous values and ϵ=0.1. These datasets were trained by the three algorithms using a Gaussian kernel with κ=1 and a regularization factor value of C=100. Figure 5 gives plots of the computational costs of the three algorithms as functions of the problem size, for the two cases of 5 and 10 target bins. Our algorithms scale well with scaling exponents between 2.13 and 2.33, while the scaling exponent of SVR is about 2.40 in this case. This near-quadratic property in scaling comes from the sparseness property of SVMs, i.e., non-support vectors affect the computational cost only mildly. The EXC and IMC algorithms cost more than the SVR approach due to the larger problem size. For large sizes, the cost of EXC is only about 2 times that of SVR in this case. As expected, we also noticed that the computational cost of IMC is dependent on r, the number of ordinal scales in the target. The cost for 10 ranks is observed to be roughly 5 times that for 5 ranks, whereas the cost of EXC is nearly the same for the two cases. These observations are consistent with the size of the optimization problems. The problem size of IMC is $(r-1)n$ (which is heavily influenced by r) while the problem size of EXC is about $2n+r$ (which largely depends on n only since we usually have $n \gg r$). This factor of efficiency can be a key advantage for the EXC formulation.

[5]The California Housing dataset can be found at http://lib.stat.cmu.edu/datasets/.

Table 2: Test results of the three algorithms using a Gaussian kernel. The targets of these benchmark datasets were discretized into 5 equal-frequency bins. d denotes the input dimension and "training/test" denotes the partition size. The results are the averages over 20 trials, along with the standard deviation. We use bold face to indicate the cases in which the average value is the lowest among the results of the three algorithms. The symbols $\star$ are used to indicate the cases significantly worse than the winning entry; A p-value threshold of 0.01 in Wilcoxon rank sum test was used to decide this.

	Partition		Mean zero-one error			Mean absolute error		
Dataset	d	training/test	SVR	EXC	IMC	SVR	EXC	IMC
Pyrimidines	27	50 /24	0.552±0.102	0.525±0.095	**0.517±0.086**	0.673±0.160	0.623±0.120	**0.615±0.127**
MachineCPU	6	150/59	0.454±0.054	**0.423±0.060**	0.431±0.054	0.495±0.067	**0.458±0.067**	0.462±0.062
Boston	13	300/206	0.354±0.033	0.336±0.033	**0.332±0.024**	0.376±0.033	0.362±0.036	**0.357±0.024**
Abalone	8	1000/3177	0.555±0.016*	**0.522±0.015**	0.527±0.009	0.679±0.014*	0.662±0.005	**0.657±0.011**
Bank	32	3000/5182	0.590±0.004*	**0.528±0.004**	0.537±0.004*	0.712±0.006*	0.674±0.006*	**0.661±0.005**
Computer	21	4000/4182	0.289±0.007*	**0.270±0.006**	0.273±0.007	0.306±0.008*	**0.288±0.007**	0.289±0.007
California	8	5000/15640	0.451±0.004*	**0.424±0.003**	0.426±0.004	0.515±0.004*	0.494±0.004	**0.491±0.005**
Census	16	6000/16784	0.522±0.005*	**0.473±0.003**	0.478±0.004*	0.619±0.006*	0.576±0.003	**0.573±0.005**

Table 3: Test results of the four algorithms using a Gaussian kernel. The partition sizes of these benchmark datasets were same as that in Table 2, but the targets were discretized by 10 equal-frequency bins. The results are the averages over 20 trials, along with the standard deviation. We use bold face to indicate the lowest average value among the results of the four algorithms. The symbols $\star$ are used to indicate the cases significantly worse than the winning entry; A p-value threshold of 0.01 in Wilcoxon rank sum test was used to decide this.

	Mean zero-one error				Mean absolute error			
Dataset	SVR	SLA	EXC	IMC	SVR	SLA	EXC	IMC
Pyrimidines	0.777±0.068*	0.756±0.073	0.752±0.063	**0.719±0.066**	1.404±0.184	1.400±0.255	1.331±0.193	**1.294±0.204**
MachineCPU	0.693±0.056*	**0.643±0.057**	0.661±0.056	0.655±0.045	1.048±0.141	1.002±0.121	**0.986±0.127**	0.990±0.115
Boston	0.589±0.025*	**0.561±0.023**	0.569±0.025	0.561±0.026	0.785±0.052	0.765±0.057	0.773±0.049	**0.747±0.049**
Abalone	0.758±0.017*	0.739±0.008*	0.736±0.011	**0.732±0.007**	1.407±0.021*	1.389±0.027*	1.391±0.021*	**1.361±0.013**
Bank	0.786±0.004*	0.759±0.005*	**0.744±0.005**	0.751±0.005*	1.471±0.010*	1.414±0.012*	1.512±0.017*	**1.393±0.011**
Computer	0.494±0.006*	0.462±0.006	**0.462±0.005**	0.473±0.005*	0.632±0.011*	0.597±0.010	0.602±0.009	**0.596±0.008**
California	0.677±0.003*	**0.640±0.003**	**0.640±0.003**	0.639±0.003	1.070±0.008*	1.068±0.006*	1.068±0.005*	**1.008±0.005**
Census	0.735±0.004*	**0.699±0.002**	**0.699±0.002**	0.705±0.002*	1.283±0.009*	1.271±0.007*	1.270±0.007*	**1.205±0.007**

5.3 Benchmark datasets

Next, we compared the generalization performance of the two approaches against the naive approach of using standard support vector regression (SVR) and the method (SLA) of Shashua and Levin (2003). We collected eight benchmark datasets that were used for metric regression problems.[6] The target values were discretized into ordinal quantities using equal-frequency binning. For each dataset, we generated two versions by discretizing the target values into five or ten ordinal scales respectively. We randomly partitioned each dataset into training/test splits as specified in Table 2. The partitioning was repeated 20 times independently. The input vectors

[6]These regression datasets are available at http://www.liacc.up.pt/∼ltorgo/Regression/DataSets.html.

were normalized to zero mean and unit variance, coordinate-wise. The Gaussian kernel (45) was used for all the algorithms. 5-fold cross validation was used to determine the optimal values of model parameters (the Gaussian kernel parameter κ and the regularization factor C) involved in the problem formulations, and the test error was obtained using the optimal model parameters for each formulation. The initial search was done on a 7×7 coarse grid linearly spaced in the region $\{(\log_{10} C, \log_{10} \kappa) | -3 \leq \log_{10} C \leq 3, -3 \leq \log_{10} \kappa \leq 3\}$, followed by a fine search on a 9×9 uniform grid linearly spaced by 0.2 in the $(\log_{10} C, \log_{10} \kappa)$ space. The ordinal targets were treated as continuous values in standard SVR, and the predictions for test cases were rounded to the nearest ordinal scale. The insensitive zone parameter, ϵ of SVR was fixed at 0.1. The test results of these algorithms are recorded in Table 2 and 3. It is very clear that the generalization capabilities of the three ordinal regression algorithms are better than that of the approach of SVR. The performance of Shashua and Levin's method is similar to our EXC approach, as expected, since the two formulations are pretty much the same. Our ordinal algorithms are comparable on the mean zero-one error, but the results also show the IMC algorithm yields much more stable results on mean absolute error than the EXC algorithm.[7] From the view of the formulations, EXC only considers the extremely worst samples between successive ranks, whereas IMC takes all the samples into account. Thus the outliers may affect the results of EXC significantly, while the results of IMC are relatively more stable in both validation and test.

5.4 Information Retrieval

Ranking learning arises frequently in information retrieval. Hersh et al. (1994) generated the OHSUMED dataset,[8] which consists of 348566 references and 106 queries with their respective ranked results. The relevance level of the references with respect to the given textual query were assessed by human experts, using a three rank scale: definitely, possibly, or not relevant. In our experiments, we used the results of query 1 which contain 107 assessed references (19 definitely relevant, 14 possibly relevant and 74 irrelevant) taken from the whole database. In order

[7]As a plausible argument, $\xi_i^j + \xi_i^{*j+1}$ in (2) of EXC is an upper bound on the zero-one error of the i-th example, while, in (18) of IMC, $\sum_{k=1}^{j} \xi_{ki}^j + \sum_{k=j+1}^{r} \xi_{ki}^{*j}$ is an upper bound on the absolute error. Note that, in all the examples we use consecutive integers to represent the ordinal scales.

[8]This dataset is publicly available at ftp://medir.ohsu.edu/pub/ohsumed/.

to apply our algorithms the bag-of-words representation was used to translate these reference documents into vectors. We computed, for all documents the vector of "term frequencies"(TF) components scaled by "inverse document frequencies"(IDF). The TFIDF is a weighted scheme for the bag-of-words representation which gives higher weights to terms which occur very rarely in all documents. We used the "rainbow" software released by McCallum (1998) to scan the title and abstract of these references for the bag-of-words representation. In the preprocessing, we skipped the terms in the "stoplist",[9] and restricted ourselves to terms that appear in at least 3 of the 107 documents. This results in 462 distinct terms. So each document is represented by its TFIDF vector with 462 elements. To account for different document lengths, we normalized the length of each document vector to unity (Joachims, 1998).

We randomly selected a subset of the 107 references (with size chosen from $\{20, 30, \ldots, 60\}$) for training and then tested on the remaining references. For each size, the random selection was repeated 100 times. The generalization performance of the two support vector algorithms for ordinal regression were compared against the naive approach of using support vector regression. The linear kernel $\mathcal{K}(x_i, x_j) = \langle x_i \cdot x_j \rangle$ was employed for all the three algorithms. The test results of the three algorithms are presented as boxplots in Figure 6. In this case, the zero-one error rates are almost at same level, but the naive approach yields much worse absolute error rate than the two ordinal SVM algorithms.

5.5 Collaborative Filtering

Collaborative filtering is to predict a person's rating on new items given the person's past ratings on similar items and the ratings of other people on all the items (including the new item). This is a typical ordinal regression problem (Shashua and Levin, 2003), since the ratings given by the users are usually discrete and ordered. We carried out ordinal regression on a subset of the EachMovie data.[10] The ratings given by the user with ID number 52647 on 449 movies were used as the targets, in which the numbers of zero-to-five stars are 40, 20, 57, 113, 145 and 74

[9]The "stoplist" is the SMART systems' list of 524 common words, like "the" and "of".

[10]The Compaq System Research Center ran the EachMovie service for 18 months. 72916 users entered a total of 2811983 ratings of zero-to-five stars on 1628 movies.

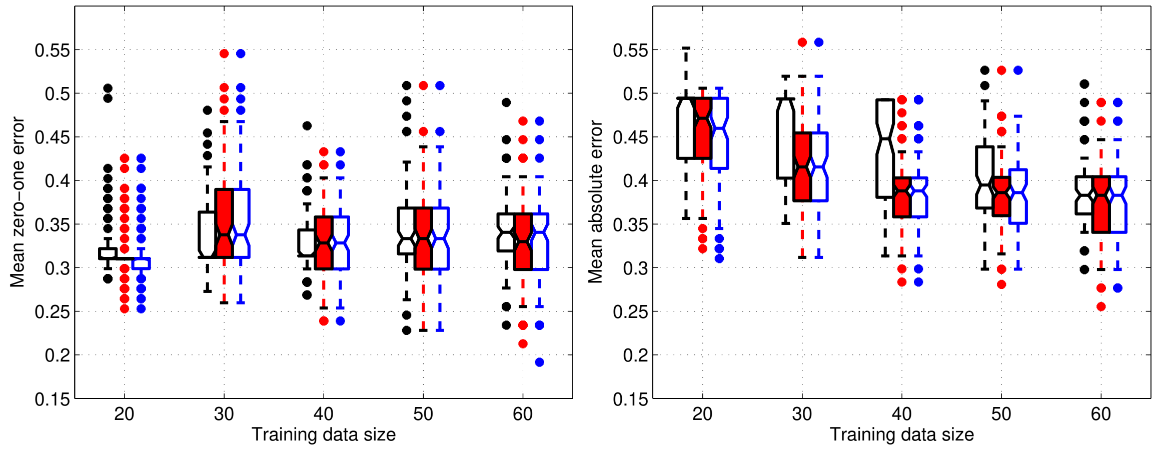


Figure 6: The test results of the three algorithms on the subset of the OHSUMED data relating to query 1, over 100 trials. The grouped boxes represent the results of SVR (left), EXC (middle) and IMC (right) at different training data sizes. The notched-boxes have lines at the lower quartile, median, and upper quartile values. The whiskers are lines extending from each end of the box to the most extreme data value within 1.5·IQR(Interquartile Range) of the box. Outliers are data with values beyond the ends of the whiskers, which are displayed by dots. The left graph gives mean zero-one error and the right graph gives mean absolute error.

respectively. We selected 1500 users who contributed the most ratings on these 449 movies as the input features, i.e. the ratings given by the 1500 users on each movie were used to form the input vector. In the 449×1500 input matrix, about 40% elements were observed. We randomly selected a subset (with size taking values from $\{50, 100, \ldots, 300\}$) of the 449 movies for training, and then tested on the remaining movies. For each size, the random selection was carried out 50 times.

Pearson correlation coefficient (a particular dot product between normalized rating vectors) is the most popular correlation measure in collaborative filtering (Basilico and Hofmann, 2004). Applied to the movies, we can define the so-called z-scores as $z(v,u) = \frac{r(v,u)-\mu(v)}{\sigma(v)}$, where u indexes users, v indexes movies, and $r(v,u)$ is the rating on the movie v given by the user u. $\mu(v)$ and $\sigma(v)$ are the movie-specific mean and standard deviation respectively. This correlation coefficient, defined as

$$\mathcal{K}(v,v') = \sum_u z(v,u)z(v',u) \tag{46}$$

where $\sum_u$ denotes summing over all the users, was used as the kernel function in our experiments for the three algorithms. As not all ratings are observed in the input vectors, we use mean imputation as a reasonable strategy to deal with missing values. Thus, unobserved values are

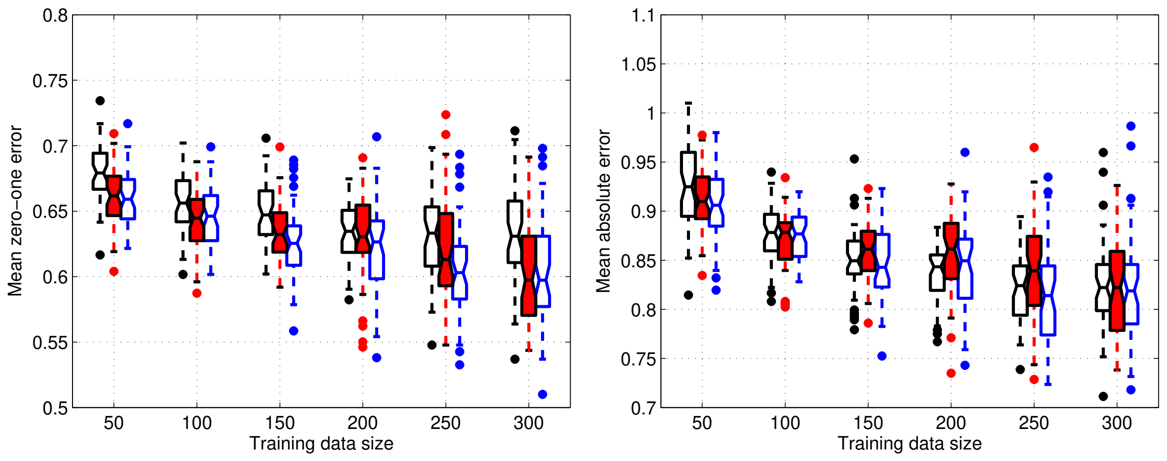


Figure 7: The test results of the three algorithms on a subset of EachMovie data, averaged over 50 trials. The grouped boxes represent the results of SVR (left), EXC (middle) and IMC (right) at different training data sizes. The left graph gives mean zero-one error and the right graph gives mean absolute error.

identified with the mean value, which means that their corresponding z-score is zero.

The test results are presented as boxplots in Figure 7. In this application, the naive approach yields much worse zero-one error rate than the two ordinal SVM algorithms and IMC performs better than EXC in many cases.

6 Conclusion

In this paper we proposed two new approaches to support vector ordinal regression that determine $r-1$ parallel discriminant hyperplanes for the r ranks by using $r-1$ thresholds. The ordinal inequality constraints on the thresholds are imposed explicitly in the first approach and implicitly in the second one. The problem size of the two approaches is linear in the number of training samples. We also designed SMO algorithms that scale only about quadratically with the problem size. The results of numerical experiments verified that the generalization capabilities of these approaches are much better than the naive approach of applying standard regression.

Acknowledgments

A part of the work was carried out at the Institute for Pure and Applied Mathematics (IPAM), University of California, from April to June 2004. WC was supported by a NIH Grant Number 1 P01 GM63208. The revision work was partly supported by a research contract from Consolidated Edison.

References

Basilico, J. and T. Hofmann. Unifying collaborative and content-based filtering. In *Proceedings of the 21th International Conference on Machine Learning*, pages 65–72, 2004.

Crammer, K. and Y. Singer. Pranking with ranking. In Dietterich, T. G., S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 641–647, Cambridge, MA, 2002. MIT Press.

Frank, E. and M. Hall. A simple approach to ordinal classification. In *Proceedings of the European Conference on Machine Learning*, pages 145–165, 2001.

Har-Peled, S., D. Roth, and D. Zimak. Constraint classification: A new approach to multiclass classification and ranking. In *Advances in Neural Information Processing Systems 15*, 2002.

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

Hersh, W., C. Buckley, T. Leone, and D. Hickam. Ohsumed: An interactive retrieval evaluation and new large test collection for research. In *Proceedings of the 17th Annual ACM SIGIR Conference*, pages 192–201, 1994.

Joachims, T. Text categorization with support vector machines: learning with many relevant features. In Nédellec, C. and C. Rouveirol, editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, pages 137–142. Springer Verlag, Heidelberg, DE, 1998.

Johnson, V. E. and J. H. Albert. *Ordinal Data Modeling (Statistics for Social Science and Public Policy)*. Springer-Verlag, 1999.

Keerthi, S. S. and E. G. Gilbert. Convergence of a generalized SMO algorithm for SVM classifier design. *Machine Learning*, 46:351–360, 2002.

Keerthi, S. S., S. K. Shevade, C. Bhattacharyya, and K. R. K. Murthy. Improvements to Platt's SMO algorithm for SVM classifier design. *Neural Computation*, 13:637–649, March 2001.

Kramer, S., G. Widmer, B. Pfahringer, and M. DeGroeve. Prediction of ordinal classes using regression trees. *Fundamenta Informaticae*, 47:1–13, 2001.

McCallum, A. Rainbow, Sept. 1998. URL `http://www.cs.umass.edu/~mccallum/bow/rainbow/`.

Platt, J. C. Fast training of support vector machines using sequential minimal optimization. In Schölkopf, B., C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 185–208. MIT Press, 1999.

Schölkopf, B. and A. J. Smola. *Learning with Kernels*. The MIT Press, 2002.

Shashua, A. and A. Levin. Ranking with large margin principle: two approaches. In S. Becker, S. T. and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 937–944, 2003.

Srebro, N., J. D. M. Rennie, and T. Jaakkola. Maximum-margin matrix factorization. In *Advances in Neural Information Processing Systems 17*, 2005.

Vapnik, V. N. *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.