One-sided Support Vector Regression for Multiclass Cost-sensitive Classification

Han-Hsing Tu r96139@csie.ntu.edu.tw
Hsuan-Tien Lin htlin@csie.ntu.edu.tw
Department of Computer Science and Information Engineering, National Taiwan University

Abstract

We propose a novel approach that reduces cost-sensitive classification to one-sided regression. The approach stores the cost information in the regression labels and encodes the minimum-cost prediction with the one-sided loss. The simple approach is accompanied by a solid theoretical guarantee of error transformation, and can be used to cast any one-sided regression method as a cost-sensitive classification algorithm. To validate the proposed reduction approach, we design a new cost-sensitive classification algorithm by coupling the approach with a variant of the support vector machine (SVM) for one-sided regression. The proposed algorithm can be viewed as a theoretically justified extension of the popular one-versus-all SVM. Experimental results demonstrate that the algorithm is not only superior to traditional one-versus-all SVM for cost-sensitive classification, but also better than many existing SVM-based cost-sensitive classification algorithms.

1. Introduction

Regular classification, which is a traditional and primary problem in machine learning, comes with a goal of minimizing the rate of misclassification errors during prediction. Many real-world applications, however, need different costs for different types of misclassification errors. For instance, let us look at a three-class classification problem of predicting a patient as {healthy, cold-infected, H1N1-infected}. Consider three different types of misclassification errors out of the six possibilities: (A) predicting a healthy patient as cold-infected; (B) predicting a healthy patient as H1N1-infected; (C) predicting an H1N1-infected patient as healthy. We see that (C) >> (B) > (A) in terms of the cost that the society pays. Many other applications in medical decision making and target marketing share similar needs, which can be formalized as the cost-sensitive classification problem. The problem is able to express any finite-choice and bounded-loss supervised learning problems (Beygelzimer et al., 2005), and thus has been attracting much research attention (Abe et al., 2004; Langford & Beygelzimer, 2005; Zhou & Liu, 2006; Beygelzimer et al., 2007).

While cost-sensitive classification is well-understood for the binary case (Zadrozny et al., 2003), the counterpart for the multiclass case is more difficult to analyze (Abe et al., 2004; Zhou & Liu, 2006) and will be the main focus of this paper. Many existing approaches for multiclass cost-sensitive classification are designed by reducing (heuristically or theoretically) the problem into other well-known problems in machine learning. For instance, the early MetaCost approach (Domingos, 1999) solved cost-sensitive classification by reducing it to a conditional probability estimation problem. Abe et al. (2004) proposed several approaches that reduce cost-sensitive classification to regular multiclass classification. There are also many approaches that reduce cost-sensitive classification to regular binary classification (Beygelzimer et al., 2005; Langford & Beygelzimer, 2005; Beygelzimer et al., 2007). Reduction-based approaches not only allow us to easily extend existing methods into solving cost-sensitive classification problems, but also broaden our understanding on the connections between cost-sensitive classification and other learning problems (Beygelzimer et al., 2005).

In this paper, we propose a novel reduction-based approach for cost-sensitive classification. Unlike existing approaches, however, we reduce cost-sensitive classification to a less-encountered problem: one-sided regression. Such a reduction is very simple but comes with solid theoretical properties. In particular, the re-

Appearing in *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel, 2010.

duction allows the total one-sided loss of a regressor to upper-bound the cost of the associated classifier. In other words, if a regressor achieves small one-sided loss on the reduced problem, the associated classifier would not suffer from much cost on the original cost-sensitive classification problem.

Although one-sided regression is not often seen in machine learning, we find that its regularized (and hyperlinear) form can be easily cast as a variant of the popular support vector machine (SVM, Vapnik, 1998). The variant will be named *one-sided support vector regression* (OSSVR). Similar to the usual SVM, OSSVR could solve both linear and non-linear one-sided regression via the kernel trick. By coupling OSSVR with our proposed reduction approach, we obtain a novel algorithm for cost-sensitive classification. Interestingly, the algorithm takes the common one-versus-all SVM (OVA-SVM, Hsu & Lin, 2002) as a special case, and is only a few lines different from OVA-SVM. That is, our proposed algorithm can be viewed as a simple and direct extension of OVA-SVM towards cost-sensitive classification. Experimental results demonstrate that the proposed algorithm is indeed useful for general cost-sensitive settings, and outperforms OVA-SVM on many data sets. In addition, when compared with other SVM-based algorithms, the proposed algorithm can often achieve the smallest average test cost, which makes it the leading SVM-based cost-sensitive classification algorithm.

The paper is organized as follows. In Section 2, we give a formal setup of the cost-sensitive classification problem. Then, in Section 3, we reduce cost-sensitive classification to one-sided regression and demonstrate its theoretical guarantees. OSSVR and its use for cost-sensitive classification is introduced in Section 4. Finally, we present the experimental results in Section 5 and conclude in Section 6.

2. Problem Statement

We start by introducing the regular classification problem before we move to the cost-sensitive classification one. In the *regular classification* problem, we seek for a classifier that maps the input vector $\mathbf{x}$ to some discrete label y, where $\mathbf{x}$ is within an input space $\mathcal{X} \subseteq \mathbb{R}^D$, and y is within a label space $\mathcal{Y} = \{1, 2, \dots, K\}$. We assume that there is an unknown distribution $\mathcal{D}$ that generates examples $(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}$. Consider a training set $\mathcal{S} = \{(\mathbf{x}_n, y_n)\}_{n=1}^N$, where each training example $(\mathbf{x}_n, y_n)$ is drawn i.i.d from $\mathcal{D}$. Regular classification aims at using $\mathcal{S}$ to find a classifier $g\colon \mathcal{X} \to \mathcal{Y}$ that comes with a small $E(g)$, where $E(h) \equiv \mathop{\mathcal{E}}_{(\mathbf{x},y)\sim\mathcal{D}} [\![y \neq h(\mathbf{x})]\!]$ is the (expected) test error of a classifier h with respect to the distribution $\mathcal{D}$.[1]

The *cost-sensitive classification* problem extends regular classification by coupling a cost vector $\mathbf{c} \in \mathbb{R}^K$ with every example $(\mathbf{x}, y)$. The k-th component $\mathbf{c}[k]$ of the cost vector denotes the price to be paid when predicting $\mathbf{x}$ as class k. With the additional cost information, we now assume an unknown distribution $\mathcal{D}_c$ that generates cost-sensitive examples $(\mathbf{x}, y, \mathbf{c}) \in \mathcal{X} \times \mathcal{Y} \times \mathbb{R}^K$. Consider a cost-sensitive training set $\mathcal{S}_c = \{(\mathbf{x}_n, y_n, \mathbf{c}_n)\}_{n=1}^N$, where each cost-sensitive training example $(\mathbf{x}_n, y_n, \mathbf{c}_n)$ is drawn i.i.d from $\mathcal{D}_c$. Cost-sensitive classification aims at using $\mathcal{S}_c$ to find a classifier $g\colon \mathcal{X} \to \mathcal{Y}$ that comes with a small $E_c(g)$, where $E_c(h) \equiv \mathop{\mathcal{E}}_{(\mathbf{x},y,\mathbf{c})\sim\mathcal{D}_c} \mathbf{c}[h(\mathbf{x})]$ is the (expected) test cost of h with respect to $\mathcal{D}_c$.

Note that the label y is actually not needed for calculating $E_c(h)$. We keep the label in our setup to help illustrate the connection between cost-sensitive classification and regular classification. Naturally, we assume that $\mathbf{c}[y] = c_{\min} = \min_{1\le\ell\le K} \mathbf{c}[\ell]$.

We will often consider the *calibrated cost vector* $\bar{\mathbf{c}}$ where $\bar{\mathbf{c}}[k] \equiv \mathbf{c}[k] - c_{\min}$ for every $k \in \mathcal{Y}$. Thus, $\bar{\mathbf{c}}[y] = 0$. Define the calibrated test cost of h as

$$\bar{E}_c(h) \equiv \mathop{\mathcal{E}}_{(\mathbf{x},y,\mathbf{c})\sim\mathcal{D}_c} \bar{\mathbf{c}}[h(\mathbf{x})] = E_c(h) - \mathop{\mathcal{E}}_{(\mathbf{x},y,\mathbf{c})\sim\mathcal{D}_c} c_{\min}.$$

Because the second term at the right-hand-side is a constant that does not depend on h, finding a classifier g that comes with a small $E_c(g)$ is equivalent to finding a classifier that comes with a small $\bar{E}_c(g)$.

We put two remarks on our setup above. First, the setup is based on example-dependent cost vectors $\mathbf{c}\colon \mathcal{Y} \to \mathbb{R}$ rather than a class-dependent cost matrix $\mathbf{C}\colon \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$, where each entry $\mathbf{C}(y, k)$ denotes the price to be paid when predicting a class-y example as class k. The class-dependent setup allows one to use a complete picture of the cost information in algorithm design (Domingos, 1999; Zhou & Liu, 2006), but is not applicable when the cost varies in a stochastic environment. On the other hand, the example-dependent setup is more general (Abe et al., 2004), and includes the class-dependent setup as a special case by defining the cost vector in $(\mathbf{x}, y, \mathbf{c})$ as $\mathbf{c}[k] \equiv \mathbf{C}(y, k)$ for every $k \in \mathcal{Y}$. In view of generality, we focus on the example-dependent setup in this paper.

Secondly, regular classification can be viewed as a special case of cost-sensitive classification by replacing the

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

cost information in $\mathbf{c}$ with a naïve (insensitive) cost matrix of $\mathbf{C}_e(y,k) \equiv [\![y \neq k]\!]$. Thus, when applying a regular classification algorithm directly to general cost-sensitive classification problems, it is *as if* we are feeding the algorithm with inaccurate cost information, which intuitively may lead to unsatisfactory performance. We will see such results in Section 5.

3. One-sided Regression for Cost-sensitive Classification

From the setup above, the value of each cost component $\mathbf{c}[k]$ carries an important piece of information. Recent approaches that reduce cost-sensitive classification to regular classification encode the cost information in the weights (importance) of the transformed classification examples. Some of the approaches leads to more promising theoretical results, such as Sensitive Error Correcting Output Codes (SECOC, Langford & Beygelzimer, 2005), Filter Tree (FT, Beygelzimer et al., 2007) and Weighted All-Pairs (WAP, Beygelzimer et al., 2005). Nevertheless, it has been shown that a large number of weighted classification examples are often required to store the cost information accurately (Abe et al., 2004; Langford & Beygelzimer, 2005), or the encoding structure and procedure can be quite complicated (Langford & Beygelzimer, 2005; Beygelzimer et al., 2005; 2007). Because of those caveats, the practical use of those algorithms has not been fully investigated.

To avoid the caveats of encoding the cost information in the weights, we place the cost information in the labels of regression examples instead. Such an approach emerged in the derivation steps of SECOC (Langford & Beygelzimer, 2005), but its direct use has not been thoroughly studied. *Regression*, like regular classification, is a widely-studied problem in machine learning. Rather than predicting the discrete label $y \in \mathcal{Y}$ with a classifier g, regression aims at using a regressor r to estimate the real-valued labels $Y \in \mathbb{R}$. We propose to train a joint regressor $r(\mathbf{x},k)$ that estimates the cost values $\mathbf{c}[k]$ directly. Intuitively, if we can obtain a regressor $r(\mathbf{x},k)$ that estimates each $\mathbf{c}[k]$ perfectly for any cost-sensitive example $(\mathbf{x},y,\mathbf{c})$, we can use the estimate to choose the best prediction

$$g_r(\mathbf{x}) \equiv \operatorname*{argmin}_{1\le k\le K} r(\mathbf{x},k). \tag{1}$$

What if the estimate $r(\mathbf{x},k)$ cannot match the desired value $\mathbf{c}[k]$ perfectly? In the real world, it is indeed the common case that $r(\mathbf{x},k)$ would be somewhat different from $\mathbf{c}[k]$, and the inexact $r(\mathbf{x},k)$ may lead to misclassification (i.e. more costly prediction) in (1).

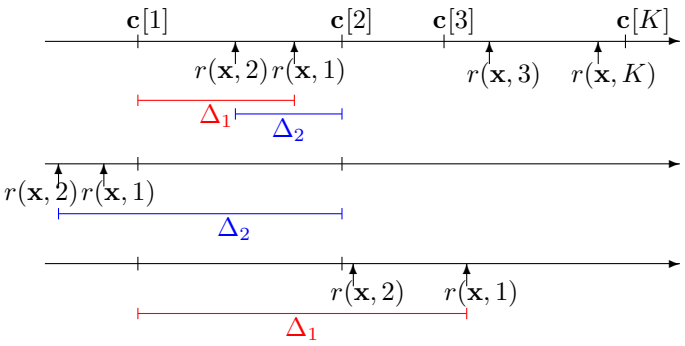


Figure 1. intuition behind one-sided regression

We illustrate such misclassification cases by Figure 1. Without loss of generality, assume that $\mathbf{c}$ is ordered such that $\mathbf{c}[1] \le \mathbf{c}[2] \le \cdots \le \mathbf{c}[K]$. We shall further assume that $\mathbf{c}[1] < \mathbf{c}[2]$ and thus the correct prediction $y = 1$ is unique. Now, if $g_r(\mathbf{x}) = 2$, which means $r(\mathbf{x},2) \le r(\mathbf{x},k)$ for every $k \in \mathcal{Y}$. More specifically, $r(\mathbf{x},2) \le r(\mathbf{x},1)$. Define $\Delta_1 \equiv r(\mathbf{x},1) - \mathbf{c}[1]$ and $\Delta_2 \equiv \mathbf{c}[2] - r(\mathbf{x},2)$. Because $\mathbf{c}[1] < \mathbf{c}[2]$ and $r(\mathbf{x},2) \le r(\mathbf{x},1)$, the terms Δ_1 and Δ_2 cannot both be negative. Then, there are three possible cases.

1. At the top of Figure 1, $\Delta_1 \ge 0$ and $\Delta_2 \ge 0$. Then, $\bar{\mathbf{c}}[2] = \mathbf{c}[2] - \mathbf{c}[1] \le \Delta_1 + \Delta_2$.
2. At the middle of Figure 1, $\Delta_1 \le 0$ and $\Delta_2 \ge 0$. Then, $\bar{\mathbf{c}}[2] \le \Delta_2$.
3. At the bottom of Figure 1, $\Delta_1 \ge 0$ and $\Delta_2 \le 0$. Then, $\bar{\mathbf{c}}[2] \le \Delta_1$.

In all the above cases in which a misclassification $g_r(\mathbf{x}) = 2$ happens, the calibrated cost $\bar{\mathbf{c}}[2]$ is no larger than $\max(\Delta_1,0) + \max(\Delta_2,0)$. This finding holds true even when we replace the number 2 with any k between 2 and K, and will be proved in Theorem 1.

A conceptual explanation is as follows. There are two different kinds of cost components $\mathbf{c}[k]$. If the component $\mathbf{c}[k]$ is the smallest within $\mathbf{c}$ (i.e., $\mathbf{c}[k] = c_{\min}$), it is acceptable and demanded to have an $r(\mathbf{x},k)$ that is no more than $\mathbf{c}[k]$ because a smaller $r(\mathbf{x},k)$ can only lead to a better prediction $g_r(\mathbf{x})$. On the other hand, if $\mathbf{c}[k] > c_{\min}$, it is acceptable and demanded to have an $r(\mathbf{x},k)$ that is no less than $\mathbf{c}[k]$. If all the demands are satisfied, no cost would incur in g_r; otherwise the calibrated cost would be upper-bounded by the total deviations on the wrong "side." Thus, for any cost-sensitive example $(\mathbf{x},y,\mathbf{c})$, we can define a special regression *loss*

$$\begin{aligned} \xi_k(r) &\equiv \max(\Delta_k(r),0), \text{ where} \\ \Delta_k(r) &\equiv \Big(2\big[\!\big[\mathbf{c}[k] = c_{\min}\big]\!\big] - 1\Big)\Big(r(\mathbf{x},k) - \mathbf{c}[k]\Big). \end{aligned} \tag{2}$$

When $r(\mathbf{x}, k)$ is on the correct side, $\xi_k(r) = 0$. Otherwise $\xi_k(r)$ represents the deviation between the estimate $r(\mathbf{x}, k)$ and the desired $\mathbf{c}[k]$. We shall use the definitions to prove a formal statement that connects the cost paid by g_r with the loss of r.

Theorem 1 (per-example loss bound)**.** *For any cost-sensitive example* $(\mathbf{x}, y, \mathbf{c})$*,*

$$\bar{\mathbf{c}}[g_r(\mathbf{x})] \leq \sum_{k=1}^{K} \xi_k(r). \tag{3}$$

Proof. Let $\ell = g_r(\mathbf{x})$. When $\mathbf{c}[\ell] = c_{\min}$, the left-hand-side is 0 while the right-hand-side is non-negative because all $\xi_k(r) \geq 0$ by definition.

On the other hand, when $\mathbf{c}[\ell] > c_{\min} = \mathbf{c}[y]$, by the definition in (2),

$$\xi_\ell(r) \geq \mathbf{c}[\ell] - r(\mathbf{x}, \ell), \tag{4}$$
$$\xi_y(r) \geq r(\mathbf{x}, y) - \mathbf{c}[y]\,. \tag{5}$$

Because $\ell = g_r(\mathbf{x})$,

$$r(\mathbf{x}, y) - r(\mathbf{x}, \ell) \geq 0. \tag{6}$$

Combining (4), (5), and (6), we get

$$\bar{\mathbf{c}}[\ell] \leq \xi_\ell(r) + \xi_y(r) \leq \sum_{k=1}^{K} \xi_k(r),$$

where the last inequality holds because $\xi_k(r) \geq 0$. □

Theorem 1 says that for any given cost-sensitive example $(\mathbf{x}, y, \mathbf{c})$, if a regressor $r(\mathbf{x}, k)$ closely estimates $\mathbf{c}[k]$ under the specially designed *linear one-sided loss* $\xi_k(r)$, the associated classifier $g_r(\mathbf{x})$ only pays a small calibrated cost $\bar{\mathbf{c}}[g_r(\mathbf{x})]$. We can prove a similar theorem for the *quadratic one-sided loss* $\xi_k^2(r)$, but the details are omitted because of page limits.

Based on Theorem 1, we could achieve the goal of finding a low-cost classifier by learning a low-one-sided-loss regressor first. We formalize the learning problem as *one-sided regression*, which seeks for a regressor that maps the input vector $\mathbf{X} \in \hat{\mathcal{X}}$ to some real label $Y \in \mathbb{R}$ with the loss evaluated by some direction $Z \in \{-1, +1\}$. We use $Z = -1$ to indicate that there is no loss at the left-hand-side $r(\mathbf{X}) \leq Y$, and $Z = +1$ to indicate that there is no loss at the right-hand-side $r(\mathbf{X}) \geq Y$. Assume that there is an unknown distribution $\mathcal{D}_o$ that generates one-sided examples $(\mathbf{X}, Y, Z) \in \hat{\mathcal{X}} \times \mathbb{R} \times \{-1, +1\}$. We consider a training set $\mathcal{S}_o = \{(\mathbf{X}_n, Y_n, Z_n)\}_{n=1}^N$, where each training example $(\mathbf{X}_n, Y_n, Z_n)$ is drawn i.i.d from $\mathcal{D}_o$. Linear one-sided regression aims at using $\mathcal{S}_o$ to find a regressor $r \colon \hat{\mathcal{X}} \to \mathbb{R}$ that comes with a small $E_o(r)$, where

$$E_o(q) \equiv \mathop{\mathcal{E}}_{(\mathbf{X},Y,Z)\sim\mathcal{D}_o} \max\left(Z\big(q(\mathbf{X}) - Y\big), 0\right).$$

is the expected linear one-sided loss of the regressor q with respect to $\mathcal{D}_o$.

With the definition above, we are ready to solve the cost-sensitive classification problem by reducing it to one-sided regression, as shown in Algorithm 1.

Algorithm 1 reduction to one-sided regression

1. Construct $\tilde{\mathcal{S}}_o = \left\{\left(\mathbf{X}_{n,k}, Y_{n,k}, Z_{n,k}\right)\right\}$ from $\mathcal{S}_c$,

   $$\text{where} \quad \mathbf{X}_{n,k} = (\mathbf{x}_n, k);\ Y_{n,k} = \mathbf{c}_n[k]\,;$$
   $$Z_{n,k} = 2\Big[\!\Big[\mathbf{c}_n[k] = \mathbf{c}_n[y_n]\Big]\!\Big] - 1.$$

2. Train a regressor $r(\mathbf{x}, k) \colon \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ from $\tilde{\mathcal{S}}_o$ with a one-sided regression algorithm.

3. Return the classifier g_r in (1).

Note that we can define a distribution $\mathcal{D}_o$ that draws an example $(\mathbf{x}, y, \mathbf{c})$ from $\mathcal{D}_c$, chooses k uniformly at random, and then generates $(\mathbf{X}, Y, Z)$ by

$$\mathbf{X} = (\mathbf{x}, k), Y = \mathbf{c}[k]\,, Z = 2\Big[\!\Big[\mathbf{c}[k] = c_{\min}\Big]\!\Big] - 1.$$

We see that $\tilde{\mathcal{S}}_o$ consists of (dependent) examples from $\mathcal{D}_o$ and contains (many) subsets $\mathcal{S}_o \sim \mathcal{D}_o^N$. Thus, a reasonable one-sided regression algorithm should be able to use $\tilde{\mathcal{S}}_o$ to find a regressor r that comes with a small $E_o(r)$. By integrating both sides of (3) with respect to $\mathcal{D}_c$, we get the following theorem.

Theorem 2 (error guarantee of Algorithm 1)**.** *Consider any* $\mathcal{D}_c$ *and its associated* $\mathcal{D}_o$*. For any regressor* r*,* $\bar{E}_c(g_r) \leq K \cdot E_o(r)$

Thus, if we can design a good one-sided regression algorithm that learns from $\tilde{\mathcal{S}}_o$ and returns a regressor r with a small $E_o(r)$, the algorithm can be cast as a cost-sensitive classification algorithm that returns a classifier g_r with a small $\bar{E}_c(g_r)$. That is, we have reduced the cost-sensitive classification problem to a one-sided regression problem with a solid theoretical guarantee. The remaining question is, how can we design a good one-sided regression algorithm? Next, we propose a novel, simple and useful one-sided regression algorithm that roots from the support vector machine (SVM, Vapnik, 1998).

4. One-sided Support Vector Regression

From Theorem 2, we intend to find a decent regressor r with respect to $\mathcal{D}_o$. Nevertheless, $\mathcal{D}_o$ is defined from an unknown distribution $\mathcal{D}_c$ and hence is also unknown. We thus can only rely on the training set $\tilde{\mathcal{S}}_o$ on hand. Consider an empirical risk minimization paradigm (Vapnik, 1998) that finds r by minimizing an in-sample version of $E_o(g)$. That is, $r = \operatorname{argmin}_q \sum_{k=1}^{K} \sum_{n=1}^{N} \xi_{n,k}$, where $\xi_{n,k}$ denotes $\xi_k(q)$ on the training example $(\mathbf{x}_n, y_n, \mathbf{c}_n)$. We can decompose the problem of finding a joint regressor $r(\mathbf{x}, k)$ to K sub-problems of finding individual regressors $r_k(\mathbf{x}) \equiv r(\mathbf{x}, k)$. In other words, for every given k, we can separately solve

$$r_k = \operatorname*{argmin}_{q_k} \sum_{n=1}^{N} \xi_{n,k} \quad . \tag{7}$$

Let us look at linear regressors $q_k(\mathbf{x}) = \langle \mathbf{w}_k, \phi(\mathbf{x}) \rangle + b_k$ in a Hilbert space $\mathcal{H}$, where the transform $\phi \colon \mathcal{X} \to \mathcal{H}$, the weight $\mathbf{w}_k \in \mathcal{H}$, and the bias $b_k \in \mathbb{R}$. Adding a regularization term $\frac{\lambda}{2}\langle \mathbf{w}_k, \mathbf{w}_k \rangle$ to the objective function, each sub-problem (7) becomes

$$\begin{aligned} \min_{\mathbf{w}_k, b_k, \xi_{n,k}} \quad & \frac{\lambda}{2}\langle \mathbf{w}_k, \mathbf{w}_k \rangle + \sum_{n=1}^{N} \xi_{n,k} \\ \text{s.t.} \quad & \xi_{n,k} \geq Z_{n,k}\Big(\langle \mathbf{w}_k, \phi(\mathbf{x_n}) \rangle + b_k - \mathbf{c}[k]\Big), \\ & \xi_{n,k} \geq 0, \text{ for all } n. \end{aligned} \tag{8}$$

where $Z_{n,k}$ is defined in Algorithm 1. Note that (8) is a simplified variant of the common support vector regression (SVR) algorithm. Thus, we will call the variant *one-sided support vector regression* (OSSVR).

Similar to the original SVR, we can solve (8) easily in the dual domain with the kernel trick $\mathcal{K}(\mathbf{x}_n, \mathbf{x}_m) \equiv \langle \phi(\mathbf{x}_n), \phi(\mathbf{x}_m) \rangle$. The dual problem of (8) is

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m Z_{n,k} Z_{m,k} \mathcal{K}(\mathbf{x}_n, \mathbf{x}_m) \\ & + \sum_{n=1}^{N} Z_{n,k} \mathbf{c}_n[k]\, \alpha_n \\ \text{s.t.} \quad & \sum_{m=1}^{n} Z_{m,k} \alpha_m = 0; \; 0 \leq \alpha_n \leq \frac{1}{\lambda}, \text{ for all } n. \end{aligned} \tag{9}$$

Coupling Algorithm 1 with OSSVR, we obtain the following novel algorithm for cost-sensitive classification: RED-OSSVR, as shown in Algorithm 2.

Note that the common OVA-SVM (Hsu & Lin, 2002) algorithm has exactly the same steps, except that the $(Z_{n,k}\mathbf{c}_n[k])$ terms in (8) and (9) are all replaced by -1. In other words, OVA-SVM can be viewed as a special case of RED-OSSVR by considering the cost vectors $\mathbf{c}_n[k] = 2\,[\![y_n \neq k]\!] - 1$. Using those cost vectors is the same as considering the insensitive cost matrix $\mathbf{C}_e$ (see Section 2) by scaling and shifting. That is, OVA-SVM equivalently "wipes out" the original cost information and replaces it by the insensitive costs.

Algorithm 2 reduction to OSSVR

1. Training: For $k = 1,2, \ldots, K$, solve the primal problem in (8) or the dual problem in (9). Then, obtain a regressor $r_k(\mathbf{x}) = \langle \mathbf{w}_k, \phi(\mathbf{x}) \rangle + b_k$.
2. Prediction: Return $g_r(\mathbf{x}) = \operatorname*{argmin}_{1 \leq k \leq K} r_k(\mathbf{x})$.

5. Experiments

In this section, we conduct experiments to validate our proposed RED-OSSVR algorithm. In all the experiments, we use LIBSVM (Chang & Lin, 2001) as our SVM solver, adopt the perceptron kernel (Lin & Li, 2008), and choose the regularization parameter λ within $\{2^{17}, 2^{15}, \ldots, 2^{-3}\}$ by a 5-fold cross-validation procedure on only the training set (Hsu et al., 2003). Then, we report the results using a separate test set (see below). Following a common practice in regression, the labels $Y_{n,k}$ (that is, $\mathbf{c}_n[k]$) are linearly scaled to $[0, 1]$ using the training set.

5.1. Comparison with Artificial Data Set

We first demonstrate the usefulness of RED-OSSVR using an artificial data set in $\mathbb{R}^2$ with $K = 3$. Each class is generated from a Gaussian distribution of variance $\frac{1}{4}$ with centers at $(-1, 0)$, $(\frac{1}{2}, \frac{\sqrt{3}}{2})$, $(\frac{1}{2}, -\frac{\sqrt{3}}{2})$, respectively. The training set consists of 500 points of each class, as shown in Figure 2. We make the data set cost-sensitive by considering a fixed cost matrix $\mathbf{C}_{\text{rot}} = \begin{pmatrix} 0 & 1 & 100 \\ 100 & 0 & 1 \\ 1 & 100 & 0 \end{pmatrix}$. Figure 2(a) shows the Bayes optimal boundary with respect to $\mathbf{C}_{\text{rot}}$. Because there is a big cost in $\mathbf{C}_{\text{rot}}(1, 2)$, $\mathbf{C}_{\text{rot}}(2, 3)$, and $\mathbf{C}_{\text{rot}}(3, 1)$, the optimal boundary rotates in the counter-clockwise direction to avoid the huge costs. Figure 2(b) depicts the decision boundary obtained from OVA-SVM. The boundary separates the adjacent two Gaussian almost evenly. Although such a boundary achieves a small misclassification error rate, it pays for a big overall cost. On the other hand, the boundary obtained from RED-OSSVR in Figure 2(c) is more similar to the Bayes optimal one. Thus, for

cost-sensitive classification problems, it is important to respect the cost information (like RED-OSSVR) instead of dropping it (like OVA-SVM), and decent performance can be obtained by using the cost information appropriately.

5.2. Comparison with Benchmark Data Sets

Next, we compare RED-OSSVR with four existing algorithms, namely, FT-SVM (Beygelzimer et al., 2007), SECOC-SVM (Langford & Beygelzimer, 2005) WAP-SVM (Beygelzimer et al., 2005) and (cost-insensitive) OVA-SVM (Hsu & Lin, 2002). As discussed in Section 3, the first three algorithms reduce cost-sensitive classification to binary classification while carrying a strong theoretical guarantee. The algorithms not only represent the state-of-the-art cost-sensitive classification algorithms, but also cover four major multiclass-to-binary decompositions (Beygelzimer et al., 2005) that are commonly used in SVM: one-versus-all (RED-OSSVR, OVA), tournament (FT), error correcting (SECOC) and one-versus-one (WAP).

Ten benchmark data sets (iris, wine, glass, vehicle, vowel, segment, dna, satimage, usps, letter) are used for comparison. All data sets come from the UCI Machine Learning Repository (Hettich et al., 1998) except usps (Hull, 1994). We randomly separate each data set with 75% of the examples for training and the rest 25% for testing. All the input vectors in the training set are linearly scaled to $[0, 1]$ and then the input vectors in the test set are scaled accordingly.

The ten benchmark data sets were originally gathered for regular classification and do not contain any cost information. To make the data sets cost-sensitive, we adopt the randomized proportional setup that was used by Beygelzimer et al. (2005). In particular, we consider a cost matrix $\mathbf{C}(y, k)$, where the diagonal entries $\mathbf{C}(y, y)$ are 0, and the other entries $\mathbf{C}(y, k)$ are uniformly sampled from $\left[0, 2000\frac{|\{n:y_n=k\}|}{|\{n:y_n=y\}|}\right]$. Then, for each example $(\mathbf{x}, y)$, the cost vector $\mathbf{c}$ comes from the y-th row of $\mathbf{C}$ (see Section 2). Although such a setup has a long history, we acknowledge that it does not fully reflect the realistic needs. The setup is taken here solely for a general comparison on the algorithms.

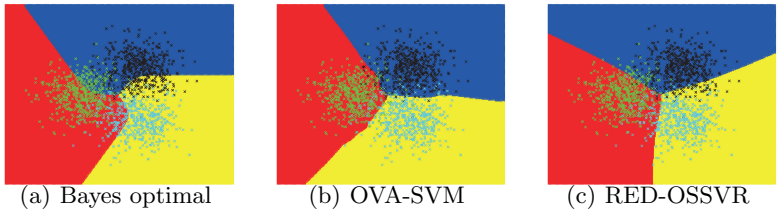
To test the validity of our proposed algorithm on more realistic cost-sensitive classification tasks, we take a random 40% of the huge 10%-training set of KDDCup 1999 (Hettich et al., 1998) as another data set (kdd99). We do not use the test set accompanied because of the known mismatch in training and test distributions, but we do take its original cost matrix for evaluation. The 40% then goes through similar 75%-25% splits and scaling, as done with other data sets.

We compare the test costs between RED-OSSVR and each individual algorithms over 20 runs using a pairwise one-tailed t-test of 0.1 significance level, as shown in Table 1. kdd99 takes longer to train and hence we only show the results over 5 runs. We then show the average test costs and their standard errors for all algorithms in Table 2. Furthermore, we list the average test error rate in Table 3.

OVA-SVM versus RED-OSSVR. We see that RED-OSSVR can often achieve lower test costs than OVA-SVM (Table 2), at the expense of higher error rates (Table 3). In particular, Tabel 1 shows that RED-OSSVR is significantly better on 5 data sets and significantly worse on only 2: vowel and letter. We can take a closer look at vowel. Table 3 suggests that OVA-SVM does not misclassify much on vowel. Hence, the resulting test cost is readily small. Then, it is hard for RED-OSSVR to make improvements using arbitrary cost information. On the other hand, for data sets like glass or vehicle, on which OVA-SVM suffers from large error and cost, RED-OSSVR can use the cost information appropriately to perform much better.

SECOC-SVM versus RED-OSSVR. SECOC-SVM is usually the worst among the five algorithms. Note that SECOC can be viewed as a reduction from cost-sensitive classification to regression coupled with a reduction from regression to binary classification (Langford & Beygelzimer, 2005). Nevertheless, the latter part of the reduction requires a thresholding step (for which we used the grid-based thresholding in the original paper). Theoretically, an infinite number of thresholds is needed, and hence any finite-sized threshold choices inevitably lead to loss of information. From the results, SECOC-SVM can suffer much from the loss of information. RED-OSSVR, on the other hand, only goes through the first part of the reduction, and hence could preserve the cost information accurately and achieves significantly better performance on 9 out of the 11 data sets, as shown in Table 1.

WAP-SVM versus RED-OSSVR. Both WAP-SVM and RED-OSSVR performs similarly well on 6 out of the 11 data sets. Nevertheless, note that WAP-SVM does pairwise comparisons, and hence needs needs $\frac{K(K-1)}{2}$ underlying binary SVMs. Thus, it takes much longer to train and does not scale well with the number of classes. For instance, on letter, training WAP-SVM would take about 13 times longer than training RED-OSSVR. With the similar performance, RED-OSSVR can be a preferred choice.



(a) Bayes optimal (b) OVA-SVM (c) RED-OSSVR

Figure 2. boundaries learned from a 2D artificial data set

FT-SVM versus RED-OSSVR. FT-SVM and RED-OSSVR both need only $O(K)$ underlying SVM classifier/regressors and hence scale well with K. Nevertheless, from Table 1, we see that RED-OSSVR performs significantly better than FT-SVM on 7 out of the 11 data sets. Note that FT-SVM is based on letting the labels compete in a tournament, and thus the design of the tournament can affect the resulting performance. From the results we see that the simple random tournament design, as Beygelzimer et al. (2007) originally used, is not as good as RED-OSSVR. The difference makes RED-OSSVR a better choice unless there is a strong demand on the $O(\log_2 K)$ prediction complexity of FT-SVM.

In summary, RED-OSSVR enjoys three advantages: using the cost-information accurately and appropriately, $O(K)$ training time, and strong empirical performance. The advantages suggest that it shall be the leading SVM-based algorithm for cost-sensitive classification nowadays.

Note that with the kernel trick in RED-OSSVR, we can readily obtain a wide range of classifiers of different complexity and thus achieve lower test costs than existing methods that focused mainly on decision trees (Abe et al., 2004; Beygelzimer et al., 2005; Zhou & Liu, 2006). The results from those comparisons are not included here because of page limits.

6. Conclusion

We proposed a novel reduction approach from cost-sensitive classification to one-sided regression. The approach is based on estimating the components of the cost vectors directly via regression, and uses a specifically designed regression loss that is tightly connected to the cost of interest. The approach is simple, yet enjoys strong theoretical guarantees in terms of error transformation. In particular, our approach allows any decent one-sided regression method to be cast as a decent cost-sensitive classification algorithm.

We modified the popular SVR algorithm to derive a new OSSVR method that solves one-sided regression problems. Then, we coupled the reduction approach with OSSVR for cost-sensitive classification. Our novel RED-OSSVR algorithm is a theoretically justified extension of the commonly used OVA-SVM algorithm. Experimental results demonstrated that RED-OSSVR is superior to OVA-SVM for cost-sensitive classification. Furthermore, RED-OSSVR can enjoy some advantages over three major SVM-based cost-sensitive classification algorithms. The findings suggest that RED-OSSVR is the best SVM-based algorithm for cost-sensitive classification nowadays.

Table 1. comparing the test costs of RED-OSSVR and each algorithm using a pairwise one-tailed t-test of 0.1 significance level

data set	FT	SECOC	WAP	OVA
iris	≈	≈	≈	◯
wine	≈	≈	≈	◯
glass	◯	◯	◯	◯
vehicle	◯	◯	◯	◯
vowel	≈	◯	×	×
segment	◯	◯	≈	≈
dna	◯	◯	◯	≈
satimage	◯	◯	◯	◯
usps	◯	◯	≈	≈
letter	◯	◯	≈	×
kdd99	≈	◯	≈	≈

◯: RED-OSSVR significantly better
×: RED-OSSVR significantly worse
≈: otherwise

Acknowledgments

We thank Chih-Jen Lin, Yuh-Jye Lee, Shou-De Lin and the anonymous reviewers for valuable suggestions. The project was partially supported by the National Science Council of Taiwan via NSC 98-2221-E-002-192 and 98-2218-E-002-019. We are grateful to the NTU Computer and Information Networking Center for the support of high-performance computing facilities.

References

Abe, N., Zadrozny, B., and Langford, J. An iterative method for multi-class cost-sensitive learning. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 3–11. ACM, 2004.

Table 2. average test cost of SVM-based algorithms

data set	N	K	RED-OSSVR	FT-SVM	SECOC-SVM	WAP-SVM	OVA-SVM
iris	150	3	**23.82±5.52***	**31.75±5.53**	**29.30±5.53**	**28.13±6.20**	35.58±7.16
wine	178	3	**19.43±4.65***	**20.72±5.27**	**19.66±4.54**	**19.99±5.82**	28.00±5.16
glass	214	6	**228.54±15.61***	264.52±15.66	251.91±14.18	253.29±16.76	283.86±18.17
vehicle	846	4	**178.63±20.37***	**190.68±21.62**	**193.52±21.38**	**187.25±22.69**	216.94±17.83
vowel	990	11	23.07±2.89	25.96±2.76	88.57±7.25	17.63±2.01	**13.43±1.84***
segment	2310	7	**26.85±1.83***	31.01±2.24	61.48±9.73	**27.22±2.14**	**27.07±2.20**
dna	3186	3	**42.57±2.97***	56.92±3.98	54.86±5.67	50.27±3.38	**42.94±2.61**
satimage	6435	6	**68.62±4.20***	78.34±4.54	93.26±5.01	73.64±4.24	79.39±4.06
usps	9298	10	**24.22±0.94**	30.64±1.08	75.94±7.29	**23.71±0.78***	**23.76±0.96**
letter	20000	26	27.34±0.57	44.04±0.90	207.89±5.64	**26.61±0.51***	**26.62±0.72**
kdd99	197608	5	**0.0015±0.0001***	**0.0015±0.0001***	0.7976±0.0000	**0.0015±0.0001***	**0.0016±0.0001**

(those with the lowest mean are marked with *; those within one standard error of the lowest one are in bold)

Table 3. average test error (%) of SVM-based algorithms

data set	RED-OSSVR	FT-SVM	SECOC-SVM	WAP-SVM	OVA-SVM
iris	6.84±1.15	11.71±2.34	19.47±3.58	6.97±0.82	**4.34±0.75***
wine	**3.56±0.55**	**3.00±0.74**	7.00±2.10	**2.78±0.78**	**2.78±0.47***
glass	31.48±1.37	49.54±2.24	45.65±3.15	39.81±2.48	**29.91±0.63***
vehicle	26.18±2.46	29.13±2.90	29.46±2.82	29.13±2.94	**20.83±0.61***
vowel	5.26±0.49	4.09±0.52	42.64±2.89	6.92±0.79	**1.27±0.17***
segment	3.66±0.27	4.78±0.48	25.35±3.83	4.28±0.37	**2.59±0.15***
dna	7.00±0.62	10.79±1.66	13.24±3.31	7.74±0.70	**4.19±0.14***
satimage	9.50±0.30	13.05±0.97	29.94±3.85	11.06±0.60	**7.26±0.10***
usps	3.45±0.20	4.60±0.39	32.74±2.37	4.96±0.63	**2.19±0.06***
letter	3.84±0.22	9.65±0.61	76.66±1.72	7.73±0.21	**2.66±0.07***
kdd99	**0.075±0.004**	**0.074±0.004***	0.796±0.000	**0.078±0.004**	**0.077±0.004**

(those with the lowest mean are marked with *; those within one standard error of the lowest one are in bold)

Beygelzimer, A., Dani, V., , Hayes, T., Langford, J., and Zadrozny, B. Error limiting reductions between classification tasks. In *Machine Learning: Proceedings of the 22rd International Conference*, pp. 49–56. ACM, 2005.

Beygelzimer, A., Langford, J., and Ravikumar, P. Multiclass classification with filter trees. Downloaded from http://hunch.net/~jl, 2007.

Chang, C.-C. and Lin, C.-J. *LIBSVM: A Library for Support Vector Machines*. National Taiwan University, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Domingos, P. MetaCost: A general method for making classifiers cost-sensitive. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 155–164. ACM, 1999.

Hettich, S., Blake, C. L., and Merz, C. J. UCI repository of machine learning databases, 1998.

Hsu, C.-W. and Lin, C.-J. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425, 2002.

Hsu, C.-W., Chang, C.-C., and Lin, C.-J. A practical guide to support vector classification. Technical report, National Taiwan University, 2003.

Hull, J. J. A database for handwritten text recognition research. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):550–554, 1994.

Langford, J. and Beygelzimer, A. Sensitive error correcting output codes. In *Learning Theory: 18th Annual Conference on Learning Theory*, pp. 158–172. Springer-Verlag, 2005.

Lin, H.-T. and Li, L. Support vector machinery for infinite ensemble learning. *Journal of Machine Learning Research*, 9:285–312, 2008.

Vapnik, V. N. *Statistical Learning Theory*. Wiley, New York, NY, 1998.

Zadrozny, B., Langford, J., and Abe, N. Cost sensitive learning by cost-proportionate example weighting. In *Proceedings of the 3rd IEEE International Conference on Data Mining*, pp. 435, 2003.

Zhou, Z.-H. and Liu, X.-Y. On multi-class cost-sensitive learning. In *Proceedings of the 21st National Conference on Artificial Intelligence*, pp. 567–572. AAAI Press, 2006.