Learning with Cost Intervals

Xu-Ying Liu and Zhi-Hua Zhou
National Key Laboratory for Novel Software Technology
Nanjing University, Nanjing, 210093, China
{liuxy, zhouzh}@lamda.nju.edu.cn

ABSTRACT

Existing cost-sensitive learning methods require that the unequal misclassification costs should be given as precise values. In many real-world applications, however, it is generally difficult to have a precise cost value since the user maybe only knows that one type of mistake is much more severe than another type, yet it is infeasible to give a precise description. In such situations, it is more meaningful to work with a *cost interval* instead of a precise cost value. In this paper we report the first study along this direction. We propose the CISVM method, a support vector machine, to work with cost interval information. Experiments show that when there are only cost intervals available, CISVM is significantly superior to standard cost-sensitive SVMs using any of the minimal cost, mean cost and maximal cost to learn. Moreover, considering that in some cases other information about costs can be obtained in addition to cost intervals, such as the distribution of costs, we propose a general approach CODIS for using the distribution information to help improve performance. Experiments show that this approach can reduce 60% more risks than the standard cost-sensitive SVM which assumes the expected cost is the true value.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning; H.2.8 [**Database Management**]: Database Applications—*Data mining*

General Terms

Algorithms; Design; Experimentation

Keywords

Cost-sensitive learning, misclassification costs, cost intervals

1. INTRODUCTION

In real-world applications, such as medical diagnosis, direct marketing and intrusion detection, different classification errors often lead to different losses. For example, in medical diagnosis, the loss of misdiagnosing a patient as healthy is much more serious than misclassifying a healthy person as sick, because the former may lead to the loss of a life. Unfortunately, traditional research assumes that all the classification errors will result in the same loss. Thus, standard classification methods try to minimize the number of errors rather than the total cost. To deal with unequal costs, cost-sensitive learning has attracted much attention in recent years [14, 8, 19, 20, 17, 21, 13, 7, 12, 11] .

Existing cost-sensitive learning methods work with fixed cost values. The cost information is provided by domain knowledge and is *taken as* a group of precise values. The classifiers will then be well tuned to reduce the total cost w.r.t. these particular cost values. However, in many real-world situations, although the user knows that one type of mistake is more severe than another type, it may be difficult to specify a precise cost value. The aspects that can cause imprecise costs include but not limited to:

- Inherent impreciseness. Some information is naturally stochastic, e.g., one may donate $1 or $5 randomly.
- Unknown information. Sometimes we can't know everything about a system. For example, customers' SSN and salary information can't be obtained because of privacy.
- Variations in modeling. In the process of modeling risk, the approach may sample data, transform input space, and use different methods or parameters. Due to these variations, the model may not provide precise assessment for costs.
- Expert opinions. Experts may have different opinions. And sometimes, experts can just give fair estimates of real risks.
- Dynamic environments. Environments always change. The risk may change with time, e.g., due to the appearance of a new competitor. While in some applications, the same model will be used in different scenarios.

In many situations, though precise information is not available, other useful information can be obtained. One of the most common and practical form to represent an imprecise value is to bound it with an interval. We call cost information represented by intervals *cost intervals*. There are at least three possible ways to obtain cost intervals:

- Natural upper and lower bounds. In some applications, costs naturally have upper and lower bounds. For example, the buying price of a stock is $3, and someone determines to hold it when the price is between $5 and $8, then the profit is between $2 and $5 if the decision is right.

KDD'10, July 25–28, 2010, Washington, DC, USA.

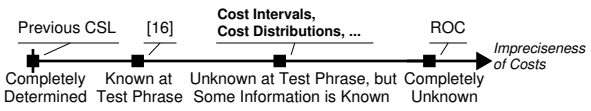


Figure 1: Related Work of Imprecise Costs.

- Transforming from confidence intervals. When there are no clear upper and lower bounds of cost, we can use a confidence interval of cost instead. For example, the 95% confidence interval indicates cost will appear in the interval with a probability of 0.95. Then it's reasonable to use the lower and upper bounds of this interval to approximate the real situation. To extreme, the 100% confidence interval indicates the true cost value is definitely in it and it can be used as cost interval if its range is appropriate. Confidence intervals are easy to get in many applications. In medicine, the inclusion of confidence interval in analysis results has been a standard [9, 18]. For example, it was reported in 2007 that [18], compared with women having non-proliferative lesions, the risk for breast cancer is greater for women with atypical ductal hyperplasia of (IRR=5.0; 95% CI=2.26-11.0). It means that the incidence rate ratio is *about* 5.0 and the 95% confidence interval is $[2.26, 11.0]$.

- Expert opinions. Sometimes, we can ask experts to provide cost intervals according to their experiences. It will be much easier for experts to give a cost interval than a "precise" cost value.

In this paper we report the first study along the direction of learning with cost intervals. Current cost-sensitive methods can be applied only when precise costs are given. To the best of our knowledge, there is no method learning with cost intervals. A related work is [16], which considers the situation that costs change over time. But it assumes that the true cost values are known at the time of classification. In our assumption, the true cost values are always unknown. ROC curve [2] can evaluate classifiers under imprecise class distributions or misclassification costs. Larger AUC (area under ROC curve) indicates better performance. This essentially assumes that nothing whatsoever is known about the relative severity of costs, which is a very rare situation in real-world problems. In our problem settings, cost intervals are known. The relationship to related work is shown in Figure 1.

An intuitive way to handle cost intervals is that, taking some value in a cost interval as the true cost, such as the minimal value, mean value, or maximal value, and then applying standard cost-sensitive learning methods. We will show theoretically that, they are not the best solutions. We propose CISVM (Cost-Interval-sensitive SVM), a support vector machine, to work with cost intervals. Experiments show that CISVM is significantly superior to all of them.

We also consider the situation in which more cost information is known in addition to cost intervals. For example, we can get distribution information of costs in some applications. We call them *cost distributions*. We propose a general approach CODIS to exploit the COst DIStribution information to help improve performance. We will show theoretically that CODIS can minimize the expected risk, while standard cost-sensitive methods taking the expected cost value as the true cost could not. Experiments show that CODIS can reduce 60% more risks than standard cost-sensitive SVM.

Section 2 formulates the problem of learning with cost intervals. Section 3 presents the CISVM method and Section 4 gives some analysis. Section 5 considers the situation in which cost distributions are known. Section 6 reports empirical results and Section 7 concludes the paper.

2. PROBLEM FORMULATION

In binary classification problems, suppose X is an input space and Y is a class label space with $y \in \{-1, +1\}$. Data are drawn i.i.d. from distribution $Pr(X, Y)$ and a training set is $S = \{(x_i, y_i)\}_{i=1}^n$. We consider *class-dependent costs*. That is, costs are only dependent on classes. The cost for any example of class y being misclassified to class y' is $cost(y, y')$. Thus, costs can be represented by a cost matrix. Since there are only 2 classes, assume that correct prediction costs 0, and let c_+ and c_- denote the cost of misclassifying a positive and negative example, respectively. Assume that positive class has higher cost, i.e., $c_+ \geq c_-$. Since the optimal decisions are unchanged when a cost matrix is multiplied by a positive constant [8], we can simplify the costs by fixing the cost of negative class so that we only need to consider the cost of positive class, i.e., $c_- = 1$, $c_+ = c$ $(c \geq 1)$.

For a classifier $h : X \to Y$, the loss of misclassifying an example (x, y) (denoted by 0-c loss) is:

$$l(c, h(x), y) = cI(h(x) \neq y \wedge y = +) + I(h(x) \neq y \wedge y = -), \quad (1)$$

where $I(a) = 1$ if $a = true$ and 0 otherwise.

Previous cost-sensitive learning assumes that cost c is a fixed value. The goal is to find a classifier h^* minimizing the empirical risk

$$\tilde{R}(h, c) = \sum\nolimits_{i=1}^{n} l(c, h(x), y). \quad (2)$$

When the true cost is unknown and a cost interval is available, we assume that the unique true cost C^* is a random value in the interval $[C_{min}, C_{max}]$. The goal is to learn a classifier H^* minimizing the *true risk* $\tilde{R}^* = \tilde{R}(h, C^*)$. Unfortunately, since the true cost is unknown, $\tilde{R}^*$ can't be obtained to guide the learning process. To overcome this difficulty, some risk $\tilde{R}_s$ can be used instead to learn a classifier, which is in fact determined by some cost C_s, i.e., $\tilde{R}_s = \tilde{R}(h, C_s)$. Since in general $\tilde{R}_s$ and C_s are different from the true risk $\tilde{R}^*$ and the true cost C^*, respectively, we call them "surrogate risk" and "surrogate cost". By minimizing surrogate risk $\tilde{R}_s$, the optimal classifier h_s^* is expected to minimize the true risk $\tilde{R}^*$. But this is infeasible since $\tilde{R}^*$ is unknown. However, since the true cost can be any value in the cost interval, it is expected that any possible risk of h_s^* should be small enough. Obviously, not all surrogate risk will be good enough for this purpose, so an appropriate surrogate cost C_s must be carefully chosen. Thus, in order to learn a classifier making any possible risks small enough, we can formulate the problem of learning with cost intervals as Eq. 3, by considering C_s as a variable for learning.

$$\begin{aligned} \min_{h, C_s} \quad & \tilde{R}(h, C_s) \quad (3) \\ \text{s.t.} \quad & p(\tilde{R}(h, c) < \epsilon) > 1 - \delta, \ \forall c \in [C_{min}, C_{max}] \\ & C_{min} \leq C_s \leq C_{max}. \end{aligned}$$

Note that, it doesn't matter for C_s to be different from C^* as long as $\tilde{R}(h, c)$ is small enough for all possible c values.

3. THE CISVM APPROACH

Let $C_\mu = 0.5(C_{min}+C_{max})$ denote the mean cost, and $d = 0.5(C_{max} - C_{min})$. Then, a random value $c \in [C_{min}, C_{max}]$ can be represented as $c = C_\mu + \varepsilon$, where ε is a random value in interval $[-d, d]$ and $|\varepsilon| \leq d$.

Since there are infinite constrains in Eq. 3, it is intractable to get optimal solutions. To overcome this difficulty, CISVM tries to solve a relaxation with a small number of informative constraints. Considering the worst case risk is the upper bound of the risks w.r.t. any c in $[C_{min}, C_{max}]$, the optimal solution of minimizing it can make all the constraints in Eq. 3 hold. That is, if $h_s^* = \arg\min_h \sup_c \tilde{R}(h, c)$ and $\sup_c \tilde{R}(h_s^*, c) = \epsilon$, then $\tilde{R}(h_s^*, c) < \epsilon$ holds for any c. So, the worst case risk is appropriate to be used as surrogate risk $\tilde{R}(h, C_s)$ to guide the learning process. However, the worst case risk could be far away from the true risk. When $C^* = C_{min}$, it overestimates the risk the most. So the optimal solution of minimizing the worst case risk could not make the true risk small enough sometimes. That is, if $H^* = \arg\min_h \tilde{R}(h, C^*)$ and $\tilde{R}(H^*, C^*) = \epsilon^*$, then $\epsilon > \epsilon^*$ could happen. CISVM overcomes this difficulty by minimizing another risk $\tilde{R}(h, C')$ in the meanwhile to avoid overfitting to surrogate risk, where $C' \neq C_s$. If the distribution of cost were known, the expected risk could be suitable to evaluate a classifier in imprecise environments. Unfortunately, only cost intervals are known. However, in this situation, the risk w.r.t. the mean cost C_μ (called "the mean risk"[1]) has the smallest maximal distortion of the true risk (see Section 4), so it is the best choice to reflect how good a classifier performs on the entire interval. Therefore, CISVM works by minimizing the worst case risk and the mean risk in the meanwhile. Though similar to the worst case risk, the best case risk is often used for analysis in imprecise environments, it is too optimistic to guarantee other possible risks are small enough, so it is not a good candidate for either of $\tilde{R}(h, C_s)$ or $\tilde{R}(h, C')$.

Assuming that the prediction function is $f = w^{\mathrm{T}}x + b$, CISVM utilizes a surrogate loss in the following form:

$$L(C_p, f(x), y) = I_{y=+}[C_p - yf(x)]_+ + I_{y=-}[1 - yf(x)]_+, \quad (4)$$

where $[a]_+ = max(a, 0)$ and $I_a = I(a)$. It means that, the loss for a negative and positive example is $L^- = [1 - yf(x)]_+$ and $L^+ = [C_p - yf(x)]_+$, respectively. We use $L(C_p)$ for $L(C_p, f(x), y)$ when this will not cause confusion.

As we know, for fixed costs, existing methods gain cost sensitivity by introducing a parameter to bias toward expensive class. For example, sampling-based methods [8, 21] sample more examples of the expensive class. Instance-weighting-based methods [15] give higher weights to the expensive class examples. Thresholding-based methods [6, 8, 21] move decision threshold toward the inexpensive class. Other methods such as cost-sensitive SVMs [3] and cost-sensitive boosting methods [12] also have such parameters to control the tradeoff between different classes. We call this parameter *the cost parameter*. For fixed costs, it is reasonable to let the cost parameter equal to the true cost. Some work also indicated the best value of the cost parameter is usually unequal to the true cost [5, 13]. Chawla et al. [4] further proposed a method to automatically find the best value according to the results on a validation set. Similarly, a cost parameter is needed in the cost interval learning problem to achieve cost sensitivity, which is C_p in Eq. 4 to make tradeoff between different classes.

Let $\tilde{R}_L(C_p)$ denote the empirical loss on training set S:

$$\tilde{R}_L(C_p) = \sum\nolimits_{i=1}^{n} L(C_p, f(x_i), y_i). \quad (5)$$

For the surrogate loss in form of Eq. 4, the worst case risk is $\tilde{R}_L(C_{max})$, which is guaranteed by Theorem 1. And the mean risk is $\tilde{R}_L(C_\mu)$. In addition, the best case risk is $\tilde{R}_L(C_{min})$.

THEOREM 1. *Suppose C_p is a random value in interval $[C_{min}, C_{max}]$, then the worst case risk w.r.t. loss function $L(C_p)$ is achieved when $C_p = C_{max}$:*

$$\tilde{R}_L(C_{max}) = \sup_{C_p} \tilde{R}_L(C_p). \quad (6)$$

PROOF.

$$\begin{aligned}
&\sup_{C_p} \tilde{R}_L(C_p) = \sup_{C_p} \sum\nolimits_{i=1}^{n} L(C_p, f(x_i), y_i) \\
&= \sup_{C_p} \sum\nolimits_{y_i=+} [C_p - y_i f(x_i)]_+ + \sum\nolimits_{y_i=-} [1 - y_i f(x_i)]_+ \\
&= \sup_{\varepsilon} \sum\nolimits_{y_i=+} [C_\mu + \varepsilon - y_i f(x_i)]_+ + \sum\nolimits_{y_i=-} [1 - y_i f(x_i)]_+ \\
&= \sum\nolimits_{y_i=+} [C_\mu + \sup(\varepsilon) - y_i f(x_i)]_+ + \sum\nolimits_{y_i=-} [1 - y_i f(x_i)]_+ \\
&= \sum\nolimits_{y_i=+} [C_\mu + d - y_i f(x_i)]_+ + \sum\nolimits_{y_i=-} [1 - y_i f(x_i)]_+ \\
&= \sum\nolimits_{y_i=+} [C_{max} - y_i f(x_i)]_+ + \sum\nolimits_{y_i=-} [1 - y_i f(x_i)]_+ \\
&= \sum\nolimits_{i=1}^{n} L(C_{max}, x_i, y_i) = \tilde{R}_L(C_{max}). \qquad \square
\end{aligned}$$

COROLLARY 1. *The best case risk w.r.t. loss function $L(C_p)$ is achieved when $C_p = C_{min}$, which is $\tilde{R}_L(C_{min})$.*

If the optimal solutions of minimizing $\tilde{R}(h, C_s)$ and $\tilde{R}(h, C')$ were identical, the relaxed problem that CISVM tries to solve would be degenerated to minimizing $\tilde{R}(h, C_s)$ only, and thus its optimal solution would not be good enough for the original learning problem in Eq. 3. In fact, minimizing the worst case risk $\tilde{R}_L(C_{max})$ is not equivalent to minimizing the mean risk $\tilde{R}_L(C_\mu)$.

COROLLARY 2. *Suppose n_+ is the size of positive class. Then, $\tilde{R}_L(C_\mu) + d \cdot n_+ \geq \sup_{C_p} \tilde{R}_L(C_p)$. Minimizing $\tilde{R}_L(C_\mu)$ is equivalent to minimizing an upper bound of $\tilde{R}_L(C_{max})$, since $d \cdot n_+$ is a constant.*

PROOF.

$$\begin{aligned}
&\sup_{C_p} \tilde{R}_L(C_p) \\
&= \sum\nolimits_{y_i=+} [C_\mu + d - y_i f(x_i)]_+ + \sum\nolimits_{y_i=-} [1 - y_i f(x_i)]_+ \\
&\leq \sum\nolimits_{y_i=+} \left([C_\mu - y_i f(x_i)]_+ + d \right) + \sum\nolimits_{y_i=-} [1 - y_i f(x_i)]_+ \\
&= \tilde{R}_L(C_\mu) + d \cdot n_+. \qquad \square
\end{aligned}$$

Here, equality holds only when $yf(x) \leq C_\mu$ for all positive examples. Since this requires almost all positive examples be support vectors, equality does not hold in general.

To minimize the worst case risk $\tilde{R}_L(C_{max})$ and the mean risk $\tilde{R}_L(C_\mu)$ in the meanwhile, CISVM firstly learns a variation of support vector machine to minimize $\tilde{R}_L(C_{max})$, and

[1]It is not exactly the mean risk since the distribution is unknown. We call it "the mean risk" for convenience.

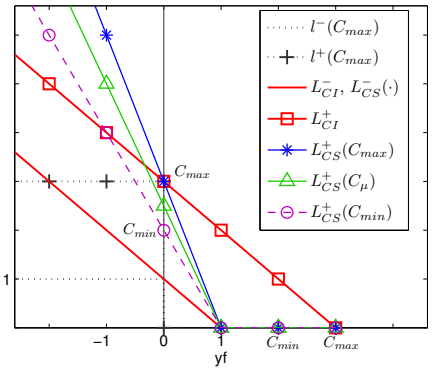


Figure 2: 5 loss functions l, L_{CI}, $L_{CS}(C_{max})$, $L_{CS}(C_\mu)$, $L_{CS}(C_{min})$ are shown with each is plotted for positive example (marked by "+") and negative example (marked by "−") separately. $L_{CS}(\cdot)$ represents for any of $L_{CS}(C_{max})$, $L_{CS}(C_\mu)$, and $L_{CS}(C_{min})$. L_{CI} is a convex upper bound of the true loss l.

then minimizes $\tilde{R}_L(C_\mu)$ by parameter selection. In particular, to minimize the worst case risk, we set the surrogate loss of CISVM as:

$$L_{CI}(f(x),y) = L(C_{max}, f(x), y) \quad (7)$$
$$= I_{y=+}[C_{max} - yf(x)]_+ + I_{y=-}[1 - yf(x)]_+.$$

L_{CI} is a convex upper bound of 0-c real loss function $l(c)$ in Eq. 1 for any cost c. See Figure 2 for an illustration. Then, CISVM minimizes the regularized loss of L_{CI}:

$$\min_{w,b,\xi\geq 0} \quad \|w\|^2/2 + \lambda \sum_{i=1}^n \xi_i \quad (8)$$
$$\text{s.t.} \quad y_i(w^{\mathrm{T}}\phi(x_i) + b) \geq C_{max} - \xi_i, \ \forall i : y_i = +1$$
$$y_i(w^{\mathrm{T}}\phi(x_i) + b) \geq 1 - \xi_i, \ \forall i : y_i = -1$$

where $\phi(x)$ is a feature map induced by a kernel function. The dual form can be easily derived.

To minimize the mean risk $\tilde{R}_L(C_\mu)$, CISVM selects λ and kernel parameters according to $\tilde{R}_L(C_\mu)$. That is, among the classifiers achieving the optimal worst case risk, CISVM selects the one minimizing $\tilde{R}_L(C_\mu)$. When a validation set is available, 0-c loss $l(C_\mu)$ can be used instead of $L(C_\mu)$ to assess the mean risk on the validation set.

Therefore, CISVM involves two parts: (1) minimizing the regularized worst case risk (Eq. 8) by learning a variation of SVM; (2) minimizing the mean risk by parameter selection on a validation set. The pseudo code is shown in Alg. 1. A cost interval is determined when C_{max} and C_μ are fixed, so CISVM provides a unique solution for each cost interval.

4. ANALYSIS

This section gives some analysis to compare existing cost-sensitive methods with CISVM on handling cost intervals.

Cost-sensitive SVM (CSSVM) [3] can only work with fixed costs. Its loss function for class-dependent cost is:

$$L_{CS}(C_p, f(x), y) = I_{y=+}C_p[1-yf(x)]_+ + I_{y=-}[1-yf]_+, \quad (9)$$

where, C_p is the cost parameter. For a given cost c, $C_p = c$.

Intuitively, we can enable CSSVM to handle cost intervals by assuming that the true cost is a particular value in interval $[C_{min}, C_{max}]$. In this paper, we consider three versions of CSSVM: CSMIN, CSMEAN and CSMAX, by setting C_p as C_{min}, C_μ and C_{max}, respectively. Their loss functions are shown in Figure 2, which are different from L_{CI}, the loss function of CISVM.

Algorithm 1 CISVM Algorithm

Input:
- $S = \{(x_i, y_i)\}_{i=1}^n$: training set, with $y_i \in \{-1, +1\}$
- V: validation set
- $[C_{min}, C_{max}]$: cost interval
- $\{(\lambda, k)\}$: a set of parameters, where k is the kernel parameters

1: For each (λ_i, k_i), train a SVM $f_i = w^{\mathrm{T}}\phi(x)+b$ by solving the optimization problem in Eq. 8.
2: Let $C_\mu = 0.5(C_{min} + C_{max})$ be the mean cost.
3: Evaluate the mean risk on validation set V for each f_i :

$$\tilde{R}_V(f_i, C_\mu) = \sum\nolimits_{(x,y)\in V} l(C_\mu, f_i(x), y)), \ l \text{ in Eq. 1}$$

Output: $H(x) = \arg\min_{f_i} \tilde{R}_V(f_i, C_\mu)$

Let $\tilde{R}_{\text{LCS}}(C_p)$ be the empirical loss on training set S:

$$\tilde{R}_{\text{LCS}}(C_p) = \sum\nolimits_{i=1}^n L_{CS}(C_p, f(x), y). \quad (10)$$

Then similar to the analysis in the previous section, for the loss function in form of Eq. 9, the worst case risk is $\tilde{R}_{\text{LCS}}(C_{max})$, the mean risk is $\tilde{R}_{\text{LCS}}(C_\mu)$, and the best case risk is $\tilde{R}_{\text{LCS}}(C_{min})$. Hence, CSMAX minimizes the worst case risk $\tilde{R}_{\text{LCS}}(C_{max})$, CSMEAN minimizes the mean risk $\tilde{R}_{\text{LCS}}(C_\mu)$, and CSMIN minimizes the best case risk $\tilde{R}_{\text{LCS}}(C_{min})$. The optimal solutions of minimizing the worst case risk and the mean risk are not equivalent.

LEMMA 1. *Suppose C_p is a random value in $[C_{min}, C_{max}]$, then the worst case risk w.r.t. loss function $L_{CS}(C_p)$ is achieved when $C_p = C_{max}$:*

$$\tilde{R}_{\text{LCS}}(C_{max}) = \sup_{C_p} \tilde{R}_{\text{LCS}}(C_p). \quad (11)$$

COROLLARY 3. *The best case risk w.r.t. loss function $L_{CS}(C_p)$ is achieved when $C_p = C_{min}$, which is $\tilde{R}_{\text{LCS}}(C_{min})$.*

COROLLARY 4. $\tilde{R}_{\text{LCS}}(C_\mu) \leq \sup_{C_p} \tilde{R}_{\text{LCS}}(C_p)$. *Equality holds only when $yf(x) \geq 1$ for all positive examples. Since this requires many positive examples being non-support-vectors, equality does not hold in general.*

The difference between CSMAX, CSMEAN CSMIN and CISVM is that they use different surrogate risks to guide the learning process. However, no matter what surrogate risk is used, it is generally unequal to the true risk, so there is always distortion of the true risk. In a good cost interval learning method, the maximal distortion of the true risk should be as small as possible. Based on this idea, we define the robustness of a cost-sensitive loss function on a cost interval as follows:

DEFINITION 1. *Suppose $\mathcal{L}(C_p, h(x), y)$ is a cost-sensitive surrogate loss, where C_p is the cost parameter. $l(C_p, h(x), y)$ is 0-c real loss in Eq. 1. Examples are drawn from $Pr(X, Y)$. Then, the distortion of $\mathcal{L}(C_p)$ on cost interval $[C_{min}, C_{max}]$*

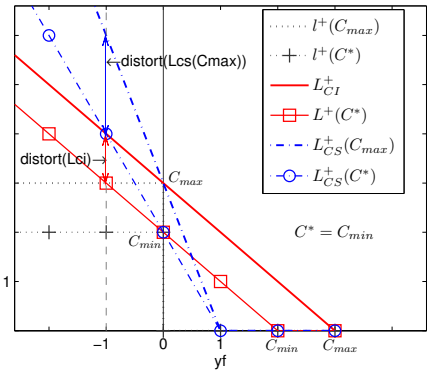


Figure 3: Comparison of robustness of CISVM and CSMAX. There are 3 loss functions: 0-c loss $l(C_{max})$, $L(C_{max})$ (i.e., L_{CI}), $L_{CS}(C_{max})$. The figure shows the loss for positive examples: $l^+(C_{max})$, L^+_{CI}, $L^+_{CS}(C_{max})$. When the true cost $C^* = C_{min}$, the distortion of the true risk is largest for all of these loss functions, $\text{distort}(L_{CI}) = C_{max} - C_{min}$ **and** $\text{distort}(L_{CS}(C_{max})) = 2(C_{max} - C_{min})$**.**

is defined as the worst case distortion of $\mathcal{L}(C_p)$. Let $C^* \in [C_{min}, C_{max}]$ denote the true cost, then:

$$\text{distort}(\mathcal{L}(C_p)) = \sup_{C^*,h(x),y} |\mathcal{L}(C^*, h(x), y) - \mathcal{L}(C_p, h(x), y)|.$$

$\inf_{C_p} \text{distort}(l(C_p))$ *can be used as a baseline when quantifying robustness, since l is the real loss and will not introduce additional distortion of risk.*

The robustness of $\mathcal{L}(C_p)$ is then defined as

$$\text{robust}(\mathcal{L}(C_p)) = \frac{\inf_{C_p} \text{distort}(l(C_p))}{\text{distort}(\mathcal{L}(C_p))}. \tag{12}$$

The larger the robust value, the better the robustness. robust $= +\infty$ *when there is no distortion.*

THEOREM 2. *The robustness of loss functions of CISVM and* CSMAX *is 0.5 and 0.25, respectively. Therefore, CISVM is more robust than* CSMAX.

PROOF.

$$\begin{aligned}\inf_{C_p} \text{distort}(l(C_p)) &= \inf_{C_p} \sup_{C^*} |C_p - C^*| \\ &= 0.5(C_{max} - C_{min}) = d.\end{aligned}$$

$$\begin{aligned}\text{distort}(\text{LCI}) &= \sup_{C^*,f(x),y} |[C^* - yf(x)]_+ - [C_{max} - yf(x)]_+| \\ &= \sup_{C^*} |C^* - C_{max}| = C_{max} - C_{min} = 2d.\end{aligned}$$

$$\begin{aligned}\text{distort}(\text{LCS}(C_{max})) &= \sup_{C^*,f(x),y} |(C^* - C_{max})[1 - yf(x)]_+| \\ &= 2(C_{max} - C_{min}) = 4d.\end{aligned}$$ □

Both CISVM and CSMAX try to minimize the worst case risk, but CISVM is more robust than CSMAX, which is shown in Theorem 2. The robustness of loss functions of CISVM and CSMAX is 0.5 and 0.25, respectively. See Figure 3 for an illustration. Similarly, for CSMEAN and CSMIN, the robust value is 0.5 and 0.25, respectively. CISVM also tries to minimize the mean risk whose loss function $L(C\mu)$ has a robust value of 1.

Thus, we contribute the success of CISVM to two aspects: (1) it minimizes the worst case risk and the mean risk in the meanwhile; (2) it is very robust.

5. UTILIZING COST DISTRIBUTION

In additional to cost intervals, sometimes we can get more information on the costs. In some applications, experts can provide the cost distribution information base on their experiences. The normal and uniform distributions are very popular and can be easily recognized from experience. For complex distributions, we can build models to assess costs. Then the values provided by different models can be regarded as samples from the underlying cost distribution. When cost distributions are known, many difficulties encountered in learning with cost intervals no more exist.

Let $R(h, c)$ be the expected risk of loss l for some cost c:

$$\begin{aligned}R(h, c) &= E_{Pr(X,Y)}[l(c, h(x), y)] \\ &= p_+ \cdot fn \cdot c + p_- \cdot fp,\end{aligned} \tag{13}$$

where $p_+ = p(y = +)$ and $p_- = p(y = -)$. fn is false negative rate, and fp is false positive rate.

Assume that cost c is independently drawn from distribution v with domain C[2], which is independent of X. Then the goal is to find a classifier h minimizing the expected risk over v:

$$\begin{aligned}R_{CD}(h, v) &= E_{c\sim v}[R(h, c)] \\ &= E_{c\sim v} E_{Pr(X,Y)}[l(c, h(x), y)].\end{aligned} \tag{14}$$

Note that, this is different from learning with example-dependent costs [19, 20, 3, 11], where examples $\{(x_i, y_i, c_i)\}$ are drawn from distribution D with domain $X \times Y \times C$, and costs are often assumed as $c_i = cost(x_i, y_i, y'_i)$, where y'_i is the prediction for x_i. The goal of example-dependent cost-sensitive learning is to minimize:

$$E_{(x_i,y_i,c_i)\sim D}[c_i I(h(x_i) \neq y_i)]. \tag{15}$$

In our setting, costs are independent of examples. That is, distribution v is not related to X. When c changes to c', (e.g., due to time change), the cost of any misclassified positive example will be c'.

When a classifier h is learned, R_{CD} should be used for evaluation. It is equivalent to $R(h, E[c])$, see Eq. 16. That is, the expected cost $E[c]$ should be used to evaluate risk.

$$\begin{aligned}R_{CD}(h, v) &= E_{c\sim v}[R(h, c)] \\ &= E_{c\sim v}[p_+ \cdot fn \cdot c + p_- \cdot fp] \\ &= p_+ \cdot fn \cdot E[c] + p_- \cdot fp = R(h, E[c]).\end{aligned} \tag{16}$$

In the above we assume known cost distribution. When there are only samples of the underlying cost distribution available, let $S_c = \{c_i\}_{i=1}^{|c|}$ be the sample set of costs, then the goal is to minimize the empirical form of R_{CD}:

$$\tilde{R}_{CD}(h, S_c) = \frac{1}{|c|} \sum_{i=1}^{|c|} E_{Pr(X,Y)}[l(c_i, h(x), y)].$$

Eq. 16 shows *for a fixed classifier h*, the expected risk over distribution v is the risk w.r.t. the expected cost $E[c]$. An intuitive way to utilize cost distributions is to take $E[c]$ as the true cost and then exploit standard cost-sensitive methods. Unfortunately, there is no guarantee that this solution can minimize the expected risk. $E_{c\sim v}[R(h, c)] = R(h, E[c])$ holds only when h is independent of c. A learned classifier h is fixed and is therefore independent of c, so Eq. 16 holds.

[2]There may be other domains, e.g., time. Here, we only consider the marginal distribution of C.

Algorithm 2 CODIS Algorithm

Input:
- $S = \{(x_i, y_i)\}_{i=1}^n$: training set, with $y_i \in \{-1, +1\}$
- v: cost distribution
- A: learning algorithm for example-dependent costs
- T: learning rounds

for $t = 1$ to T **do**
  $\hat{S} = \emptyset$
  for $i = 1$ to n **do**
    Draw a cost c_i independently from v
    $\hat{S} = \hat{S} \cup (x_i, y_i, c_i)$.
  end for
  $h_t = A(\hat{S})$: learn a classifier using A and $\hat{S}$.
end for

Output: $H(x) = \sum_{t=1}^T h_t(x)$

But in the learning process, h is a function in hypothesis space $\mathcal{H}$, so it is dependent on cost c.

Recall the cost parameter C_p described in Section 3. Cost-sensitive methods use C_p to make tradeoff between different classes according to their costs. Therefore, C_p can be regarded as a function dependent on the true cost c: $C_p = g(c)$. Since h is dependent on C_p in the learning process, it is a function of c: $h = h(C_p) = h(g(c))$. We then have:

PROPOSITION 1. *v is a cost distribution. h is a function in hypothesis space $\mathcal{H}$ to be learned. It has cost parameter C_p: $h(x) = h(x, C_p)$. C_p is dependent on the true cost c: $C_p = g(c)$. Then $h(x) = h(x, g(c))$, and the following holds when $g(c) \neq c$ or h is a nonlinear function of $g(c)$.*

$$E_{c\sim v}[R(h(g(c)), c)] \neq R(h(E[c]), E[c]). \tag{17}$$

Inequality holds in general, since h is generally a nonlinear function of $g(c)$, and there is no guarantee that $g(c)$ should be c even for fixed costs.

Therefore, standard methods taking the expected cost $E[c]$ as the true cost and then minimizing the corresponding risk can not minimize $E_{c\sim v}[R(h(g(c)), c)]$. However, it can be easily derived that cost distribution learning problem can be reduced to a special case of example-dependent cost-sensitive learning problem, which is guaranteed by Theorem 3.

THEOREM 3. *Suppose example (x, y) is i.i.d. drawn from $Pr(X, Y)$, and cost c is i.i.d. drawn from $v(C)$. Suppose Pr and v are independent. Then new example (x, y, c) can be considered to be i.i.d. drawn from a joint distribution D with domain $X \times Y \times C$, with $p_D(x, y, c) = p_{Pr}(x, y)p_v(c)$, where $p_A(a)$ denotes the pdf of sample a drawn from distribution A. We have the following holds for any function h with cost parameter $C_p = g(c)$:*

$$E_{c\sim v}[R(h(g(c)), c)] = E_{(x,y,c)\sim D}[l(c, h(x, g(c)), y)]. \tag{18}$$

PROOF.

$$\begin{aligned} E_{c\sim v}[R(h(g(c)), c)] &= E_{c\sim v}E_{(x,y)\sim Pr}[l(c, h(x, g(c)), y)] \\ &= E_{(x,y)\sim Pr, c\sim v}[l(c, h(x, g(c)), y)] \\ &= E_{(x,y,c)\sim D}[l(c, h(x, g(c)), y)]. \end{aligned}$$ □

Cost distribution v is independent of X by definition, so Theorem. 3 holds. Based on this theorem, CODIS, a general method exploiting example-dependent cost-sensitive methods is proposed to utilize cost distributions. Firstly, a cost sample c_i is drawn from v (or provided by a risk model) for each example (x_i, y_i) in training set S to form a new example set $\hat{S} = \{(x_i, y_i, c_i)\}_{i=1}^n$. Secondly, a standard example-dependent cost-sensitive method A is called to learn a classifier minimizing the risk of $\hat{S}$. Furthermore, to reduce the variance caused by sampling from v, we can draw cost multiple times from v for a single example (x_i, y_i) since v and Pr are independent. Therefore, we can repeat the first two steps several times and ensemble all the classifiers. The pseudo code is shown in Alg. 2. We use 0-1 voting for ensemble. It is possible to use other kinds of ensemble to get even better performance and which will be studied in the future. The learning algorithm A can be any standard cost-sensitive method which works with example-dependent costs, such as Costing [20] and example-dependent cost-sensitive SVM [3], which both work by assigning each example a weight proportional to its own cost c_i.

Table 1: Cost Intervals

Width	Counts	Cost Intervals
3	10	$[1,3], [4,6], [7,9], [10,12], [13,15]$ $[16,18], [19,21], [22,24], [25,27], [28,30]$
5	6	$[1,5], [6,10], [11,15], [16,20], [21,25], [26,30]$
11	5	$[1,11], [5,15], [10,20], [15,25], [20,30]$
15	4	$[1,16], [5,20], [10,25], [15,30]$

6. EXPERIMENTS

6.1 Settings

We compare CISVM and CSMIN, CSMEAN, CSMAX, with standard SVM as baseline on 15 binary UCI data sets [1]: *breast-cancer*, *breast-w*, *credit-a*, *credit-g*, *diabetes*, *german*, *haberman*, *heart-statlog*, *ionosphere*, *liver*, *sonar*, *spambase*, *spect*, *spectf* and *tic-tac-toe*. We treat each class of a data set as positive class in turn. Thus, we actually have 30 tasks[3]. Since class imbalance may affect cost-sensitive methods [10], a balanced subset is used for each task to focus on the difficulty caused by imprecise costs. 25 cost intervals are used, which are shown in Table 1. 30 times stratified hold-out tests are carried out, with 2/3 data as training set and 1/3 data as test set. All methods use RBF kernels. Parameters are selected for every cost interval by 5-fold cross validation on training data from $\lambda \in \{0.01, 0.1, 1, 10, 100\}$, and kernel parameter σ is selected as $\{1/10, 1/2, 1, 2, 10\}$ times the mean squared distance of training set. There is no guidelines on how to select parameters for standard cost-sensitive methods to handle cost intervals. To test CISVM rigorously, we use the mean risk to select parameters for *all methods*. We use standard SVM-style implementation by invoking MOSEK[4].

Since the true cost is unknown, we need to evaluate how good a classifier performs in different situations. Here, quadrisection points of an interval are considered in turn as the true cost: $C_{min} + (C_{max} - C_{min}) \times \{0, 0.25, 0.5, 0.75, 1\}$. We call them test points P1-P5.

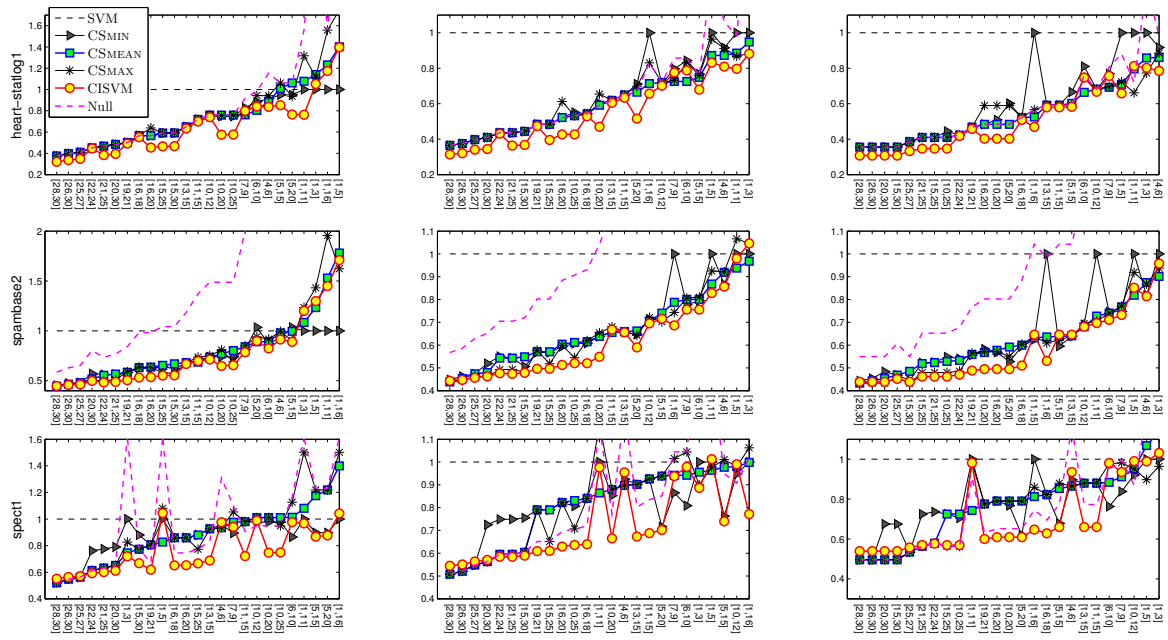
6.2 Results

Due to page limit, we can only show part of the results[5]. Figure 4 shows the performance of each method on two medical data (*heart-statlog1*, *spect1*) and a spam detection data

[3]Data sets marked by '1' use the first class as negative class and by '2' use the second class as negative class.

[4]The code is available at `http://lamda.nju.edu.cn/code_CISVM.ashx`

[5]More results will be reported in a longer version.



Test Point P1 (the true cost is C_{min}) Test Point P3 (the true cost is C_{μ}) Test Point P5 (the true cost is C_{max})

Figure 4: Some Results of Learning with Cost Intervals. X-axis shows cost intervals. Y-axis is the ratio of the loss of each method against that of SVM. The lower the value, the better the performance. A trivial classifier 'Null' which always predict positive is also shown.

Table 2: Win/tie/loss counts of method-in-row vs. method-in-column after t-tests at 95% significance level. Bold/Italic highlights significant better/worse of sign-tests results at 95% significance level.

		SVM	CSMIN	CSMEAN	CSMAX
P1	CSMIN	**457/271/22**			
	CSMEAN	**484/162/104**	*83/487/180*		
	CSMAX	**488/160/102**	*107/461/182*	*92/522/136*	
	CISVM	**521/96/133**	257/262/231	**316/248/186**	**303/259/188**
P2	CSMIN	**464/272/14**			
	CSMEAN	**533/175/42**	121/514/115		
	CSMAX	**525/186/39**	139/494/117	*82/542/126*	
	CISVM	**578/97/75**	**304/263/183**	**296/269/185**	**282/282/186**
P3	CSMIN	**466/273/11**			
	CSMEAN	**550/180/20**	**143/509/98**		
	CSMAX	**554/173/23**	**159/495/96**	*77/560/113*	
	CISVM	**593/93/64**	**304/271/175**	**276/292/182**	**260/309/181**
P4	CSMIN	**472/270/8**			
	CSMEAN	**554/180/16**	**150/503/97**		
	CSMAX	**561/176/13**	**168/493/89**	89/565/96	
	CISVM	**599/91/60**	**299/275/176**	**256/296/198**	232/327/191
P5	CSMIN	**475/268/7**			
	CSMEAN	**556/178/16**	**150/503/97**		
	CSMAX	**566/170/14**	**173/488/89**	100/556/94	
	CISVM	**606/85/59**	**299/270/181**	**252/294/204**	226/325/199
All	CSMIN	**2334/1354/62**			
	CSMEAN	**2677/875/198**	647/2516/587		
	CSMAX	**2694/865/191**	**746/2431/573**	*440/2745/565*	
	CISVM	**2897/462/391**	**1463/1341/946**	**1396/1399/955**	**1303/1502/945**

(*spambase2*) at P1, P3 and P5. Table 2 summaries the results of paired t-tests and sign-tests for each test point separately. Table 3 shows the loss of each method on the first set of data sets for some cost intervals, where we count the number of significantly best performance (SBP) [22] achieved by each method. SBPs are the best performances and there is no significant difference among them according to t-tests results. The SBP counts are further summarized in Table 4 for each cost interval and each test point.

• *At P1.* The true cost is C_{min}. CSMIN is significantly better than CSMEAN and CSMAX. It is not a surprise, since CSMIN learns with the true cost. CSMEAN and CSMAX use C_p larger than the true cost, which obviously causes the risk being overestimated. While CISVM is comparable to CSMIN, and beats CSMEAN and CSMAX significantly. Note that, P1 is the most far point from CISVM's cost parameter ($C_p = C_{max}$), but in general it does not prevent CISVM from effectively reducing risk. Some work indicated the best C_p value is usually unequal to the true cost [5, 13]. This might be the reason why CSMIN is not always the best at P1.

• *At P2.* None of the methods learns with the true cost. CSMIN is comparable to CSMEAN and CSMAX. CSMEAN performs better than CSMAX. While CISVM beats them all significantly.

• *At P3.* The true cost is C_{μ}, and CSMEAN learns with the true cost. It is not surprising that CSMEAN is significantly better than CSMIN and CSMAX. The latter two methods underestimates and overestimates the risk, respectively. While CSMEAN is significantly worse than CISVM.

• *At P4.* None of the methods learns with the true cost. CSMIN has the most unappropriate cost parameter among all methods, and the results show it is significantly worse than the others. CSMEAN and CSMAX are comparable. While CISVM is significantly better than CSMIN and CSMEAN, and comparable to or slightly better than CSMEAN.

• *At P5.* The true cost is C_{max}. Both CSMAX and CISVM learn with the true cost. Again, CSMIN is the worst one, due to its under-bias. CSMAX does not achieve to perform better than CSMEAN, but CISVM does. CISVM is comparable to or slightly better than CSMAX.

• Over all test points, CISVM beats all other methods significantly. The SBPs of CISVM are over 20% more than others (Table 4). CSMAX is better than CSMIN yet worse than CSMEAN. The reason might be that, though CSMEAN could not achieve the minimal worst case risk, it is more robust than CSMAX. The results of CSMIN show that min-

Table 3: The Loss of Compared Methods. '*' denotes the significantly best performance (SBP) according to paired t-tests at significance level 95%. $\#SBPs$ counts the number of SBPs of each method.

	P1					P3					P5				
[1,3]	SVM	CSmin	CSmean	CSmax	CISVM	SVM	CSmin	CSmean	CSmax	CISVM	SVM	CSmin	CSmean	CSmax	CISVM
breast-c1	*21.3	*21.3	27.4	27.4	*21.6	31.5	31.5	*27.6	*27.6	*28.1	41.8	41.8	*27.7	*27.8	34.5
breast-w1	10.0	10.0	4.3	*4.1	*4.0	10.0	10.0	4.9	4.5	*4.2	10.0	10.0	5.5	4.9	*4.4
credit-a1	*25.9	*25.9	28.4	36.7	34.6	41.0	41.0	*38.6	43.2	*39.3	56.1	56.1	48.9	49.7	*44.1
credit-g1	*59.9	*59.9	64.0	67.0	65.3	81.0	81.0	77.9	75.7	*73.9	102.2	102.2	91.7	*84.4	*82.5
diabetes1	*48.2	*48.2	*48.7	57.9	*47.9	64.5	64.5	58.3	62.3	*56.6	80.8	80.8	67.9	*66.6	*65.3
german1	*54.2	*54.2	62.8	63.9	58.3	80.0	80.0	75.9	76.4	*73.0	105.8	105.8	*89.1	*88.8	*87.7
haberman1	*23.1	*23.1	27.0	27.0	24.3	35.9	35.9	*27.0	*27.0	32.9	48.7	48.7	*27.0	*27.0	41.6
heart-s1	*14.7	*14.7	16.8	16.5	*15.5	23.1	23.1	21.9	*20.4	*20.4	31.5	31.5	27.0	*24.3	*25.2
ionosph1	6.5	6.5	*5.5	*5.5	8.5	7.6	7.6	*6.7	*6.6	9.2	8.6	8.6	*8.0	*7.7	9.8
liver1	32.0	32.0	31.5	35.8	*29.7	47.0	47.0	41.6	45.9	*35.9	62.0	62.0	51.7	55.9	*42.1
sonar1	10.3	10.3	14.3	*7.7	10.3	14.3	14.3	15.3	*10.7	14.3	18.3	18.3	16.4	*13.6	18.3
spambase1	*25.8	*25.8	28.4	28.6	28.4	42.0	42.0	42.4	38.3	*36.9	58.2	58.2	56.3	48.0	*45.5
spect1	11.2	11.2	*8.3	9.2	*8.0	12.0	12.0	11.5	*10.8	*10.7	*12.9	*12.9	14.7	*12.4	*13.3
spectf1	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2
tic-tac1	*4.7	*4.7	*4.7	*4.7	*4.7	*4.7	*4.7	*4.7	*4.7	*4.7	*4.7	*4.7	*4.7	*4.7	*4.7
#SBPs	**10**	**10**	**5**	**5**	**8**	**2**	**2**	**6**	**8**	**12**	**3**	**3**	**6**	**11**	**11**
[6,10]	SVM	CSmin	CSmean	CSmax	CISVM	SVM	CSmin	CSmean	CSmax	CISVM	SVM	CSmin	CSmean	CSmax	CISVM
breast-c1	106.0	*28.0	*28.0	*28.0	*28.2	138.0	*28.0	*28.0	*28.0	*28.6	170.0	*28.0	*28.0	*28.0	*28.9
breast-w1	10.0	*5.4	*5.4	5.5	6.8	10.0	*5.9	*6.0	6.0	*6.9	10.0	*6.4	*6.5	6.6	*6.9
credit-a1	113.8	62.9	62.2	62.8	*59.0	137.0	70.5	68.7	68.4	*63.7	160.3	78.1	75.2	74.0	*68.4
credit-g1	410.0	*99.1	*100.0	*100.0	*100.0	535.2	*99.4	*100.0	*100.0	*100.0	660.4	*99.7	*100.0	*100.0	*100.0
diabetes1	103.8	77.3	79.4	89.0	*72.8	120.0	81.9	80.3	89.0	*74.7	136.3	86.5	81.2	89.0	*76.6
german1	211.1	100.0	100.0	100.0	*91.3	260.6	100.0	100.0	100.0	*93.5	310.1	100.0	100.0	100.0	*95.7
haberman1	87.1	*27.0	*27.0	*27.0	35.1	112.7	*27.0	*27.0	*27.0	39.8	138.3	*27.0	*27.0	*27.0	44.4
heart-s1	40.7	36.0	*32.7	38.3	*34.2	46.7	39.3	*33.8	38.9	36.8	52.7	42.7	*35.0	39.4	39.4
ionosph1	11.7	9.9	*9.1	12.8	13.7	13.8	11.6	*10.5	15.1	14.6	15.8	13.2	*11.8	17.5	15.5
liver1	86.3	*48.0	*48.0	*48.0	*47.4	105.7	*48.0	*48.0	*48.0	*48.1	125.0	*48.0	*48.0	*48.0	*48.9
sonar1	30.4	*20.9	21.9	22.3	*19.8	38.5	*23.2	23.9	24.1	*21.7	46.6	*25.5	25.9	25.9	*23.7
spambase1	106.9	*75.0	*75.8	*77.4	*76.7	139.4	86.7	*81.4	*84.4	88.7	171.9	98.4	*86.9	91.4	100.7
spect1	15.5	*13.4	15.7	17.5	*15.1	17.2	*13.9	16.2	18.0	16.9	19.0	*14.5	16.8	18.5	18.6
spectf1	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2
tic-tac1	*4.7	*4.7	*4.7	*4.7	5.2	*4.7	*4.7	*4.7	*4.7	5.2	*4.7	*4.7	*4.7	*4.7	5.2
#SBPs	**2**	**10**	**10**	**7**	**11**	**2**	**9**	**10**	**7**	**9**	**2**	**9**	**10**	**6**	**9**
[5,15]	SVM	CSmin	CSmean	CSmax	CISVM	SVM	CSmin	CSmean	CSmax	CISVM	SVM	CSmin	CSmean	CSmax	CISVM
breast-c1	90.0	28.0	28.0	*26.9	28.1	170.0	*28.0	*28.0	*27.9	*28.9	250.0	*28.0	*28.0	*28.9	*29.7
breast-w1	10.0	10.0	*5.2	5.8	7.3	10.0	10.0	*6.6	7.2	*7.3	10.0	10.0	*7.9	*8.5	*7.3
credit-a1	102.1	*55.5	60.0	61.4	*56.9	160.3	77.2	74.0	78.2	*67.5	218.5	98.9	88.0	95.1	*78.2
credit-g1	347.4	100.0	100.0	100.0	*96.4	660.4	*100.0	*100.0	*100.0	*101.2	973.4	*100.0	*100.0	*100.0	*106.1
diabetes1	95.6	*72.0	89.0	89.0	*73.2	136.3	79.7	89.0	89.0	*76.5	177.0	87.4	89.0	89.0	*79.9
german1	186.4	93.4	100.0	100.0	*91.2	310.1	102.4	100.0	100.0	*94.8	433.7	111.4	*100.0	*100.0	*98.5
haberman1	74.3	*27.0	*27.0	*27.0	39.0	138.3	*27.0	*27.0	*27.0	57.9	202.3	*27.0	*27.0	*27.0	76.7
heart-s1	37.7	35.8	38.1	40.0	*32.1	52.7	40.4	39.4	40.0	*35.8	67.7	45.1	*40.7	*40.0	*39.4
ionosph1	*10.7	*9.8	11.6	11.8	14.2	*15.8	*14.8	17.5	17.1	*16.3	*21.0	*19.8	23.3	22.4	*18.5
liver1	76.7	*48.1	*48.0	*47.9	*47.6	125.0	*48.3	*48.0	*47.9	*49.3	173.3	*48.5	*48.0	*47.9	50.9
sonar1	26.4	23.3	21.4	22.8	*19.3	46.6	*24.5	25.9	26.2	*23.6	66.7	*25.7	30.4	29.5	*27.9
spambase1	119.0	78.2	73.9	86.4	*62.5	150.5	*89.4	*91.4	*93.2	*86.0	182.0	*100.6	108.9	*100.0	*109.5
spect1	14.6	13.1	17.2	17.8	*12.7	19.0	14.5	18.5	19.2	*14.0	23.3	15.8	19.9	20.5	*15.4
spectf1	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2
tic-tac1	*4.7	*4.7	*4.7	*4.7	6.4	*4.7	*4.7	*4.7	*4.7	6.4	*4.7	*4.7	*4.7	*4.7	6.4
#SBPs	**3**	**7**	**5**	**5**	**10**	**3**	**9**	**8**	**7**	**13**	**3**	**9**	**9**	**10**	**12**
[10,20]	SVM	CSmin	CSmean	CSmax	CISVM	SVM	CSmin	CSmean	CSmax	CISVM	SVM	CSmin	CSmean	CSmax	CISVM
breast-c1	170.0	*28.0	*27.9	*28.9	*28.9	250.0	*28.0	*28.9	*29.7	*29.7	330.0	*28.0	*29.9	30.6	30.6
breast-w1	10.0	*6.6	7.2	*7.1	*7.9	10.0	*7.9	*8.5	*8.1	*7.9	10.0	*9.2	*9.8	*9.1	*7.9
credit-a1	160.3	74.0	78.2	78.0	*67.5	218.5	88.0	95.1	91.0	*77.1	276.6	102.0	111.9	104.0	*86.8
credit-g1	660.4	100.0	100.0	*99.1	*101.6	973.4	100.0	100.0	*99.1	*106.1	1286.4	100.0	100.0	*99.1	*110.6
diabetes1	136.3	89.0	89.0	92.9	*76.7	177.0	89.0	89.0	104.0	*79.2	217.6	89.0	89.0	115.2	*81.7
german1	310.1	100.0	100.0	100.0	*95.1	433.7	*100.0	*100.0	*100.0	*98.3	557.4	*100.0	*100.0	*100.0	*101.5
haberman1	157.6	*27.0	*27.0	*27.0	50.0	230.6	*27.0	*27.0	*27.0	63.8	303.6	*27.0	*27.0	*27.0	77.7
heart-s1	52.7	39.4	40.0	39.8	*30.3	67.7	40.7	40.0	44.3	*31.8	82.7	42.1	40.0	48.8	*33.3
ionosph1	*15.8	17.5	17.1	*14.4	*16.4	*21.0	23.3	22.4	*18.2	*18.4	26.2	29.1	27.8	*22.1	*20.4
liver1	125.0	*48.0	*47.9	*48.8	50.2	173.3	*48.0	*47.9	*49.3	52.7	221.7	*48.0	*47.9	*49.8	55.2
sonar1	46.6	25.9	26.2	27.9	*24.0	66.7	*30.4	*29.5	*30.7	*28.4	86.9	*34.9	*32.8	*33.6	*32.7
spambase1	150.5	112.5	*93.2	98.2	100.8	182.0	113.6	*100.0	103.0	128.8	213.5	114.8	*106.9	*107.8	156.8
spect1	19.0	18.5	19.2	19.2	*14.2	23.3	19.9	20.5	20.5	*15.5	27.6	21.2	21.8	21.8	*16.8
spectf1	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2	*13.2
tic-tac1	*4.7	*4.7	*4.7	*4.7	8.9	*4.7	*4.7	*4.7	*4.7	8.9	*4.7	*4.7	*4.7	*4.7	8.9
#SBPs	**3**	**6**	**6**	**8**	**11**	**3**	**8**	**9**	**10**	**11**	**2**	**8**	**9**	**10**	**10**

Table 4: Significantly Best Performances. M1-M5: SVM, CSmin, CSmean, CSmax and CISVM. Boldface highlights the largest counts.

	M1	M2	M3	M4	M5	M1	M2	M3	M4	M5	M1	M2	M3	M4	M5	M1	M2	M3	M4	M5	M1	M2	M3	M4	M5	M1	M2	M3	M4	M5
[1, 3]	**20**	**20**	10	8	11	10	10	14	12	**16**	7	7	15	14	**20**	7	7	17	17	**19**	7	7	16	**18**	17	51	51	72	69	**83**
[4, 6]	7	19	14	14	**21**	7	19	15	14	**21**	7	19	15	16	**21**	7	**20**	16	16	**20**	6	**21**	16	17	**21**	34	98	76	77	**104**
[7, 9]	7	18	14	13	**22**	7	18	14	13	**21**	7	18	15	13	**21**	7	19	17	14	**20**	7	19	18	14	**20**	35	92	78	67	**104**
[10, 12]	7	18	16	14	**21**	7	19	16	15	**22**	7	19	18	16	**22**	7	18	18	16	**21**	7	18	19	17	**21**	35	92	87	78	**107**
[13, 15]	6	18	18	18	**22**	5	17	18	18	**22**	5	18	18	20	**22**	5	18	18	20	**22**	5	19	17	19	**22**	26	90	89	95	**110**
[16, 18]	4	17	16	14	**22**	4	17	16	14	**22**	4	17	16	14	**22**	4	18	17	15	**22**	4	19	17	16	**22**	20	88	82	73	**110**
[19, 21]	4	15	14	16	**21**	4	15	14	16	**21**	3	15	14	16	**21**	3	15	14	16	**21**	3	15	14	16	**21**	17	75	70	80	**105**
[22, 24]	4	16	19	19	**23**	4	17	20	20	**23**	4	17	20	20	**23**	4	18	21	21	**23**	4	18	21	21	**23**	20	86	101	101	**115**
[25, 27]	3	21	20	22	**23**	3	21	20	**22**	**22**	3	21	20	21	**22**	3	21	20	**22**	**22**	3	21	20	22	**23**	15	105	100	109	**112**
[28, 30]	3	**24**	**24**	23	21	3	**24**	**24**	23	21	3	**24**	**24**	23	21	3	**24**	**24**	23	21	3	**24**	**24**	23	21	15	**120**	**120**	115	105
[1, 5]	**18**	**18**	8	6	8	11	11	14	13	**17**	7	7	18	15	**19**	7	7	**20**	15	18	8	8	**20**	15	17	51	51	**80**	64	79
[6, 10]	6	15	14	11	**21**	6	17	15	12	**20**	6	17	16	12	**18**	6	18	17	13	**19**	6	**19**	17	14	**19**	30	86	79	62	**97**
[11, 15]	5	15	13	13	**21**	5	16	13	14	**21**	5	18	17	18	**21**	5	19	18	20	**21**	4	19	19	20	**21**	24	87	80	85	**105**
[16, 20]	6	17	17	16	**22**	4	16	15	16	**22**	3	17	15	16	**22**	3	16	15	16	**21**	3	16	15	16	**21**	19	82	77	80	**108**
[21, 25]	4	16	17	19	**23**	4	15	17	19	**23**	4	16	18	19	**23**	4	17	19	19	**23**	3	17	18	20	**22**	19	81	89	96	**114**
[26, 30]	3	22	22	**23**	21	3	22	**23**	**23**	21	3	22	**23**	**23**	21	3	21	**23**	**23**	21	3	20	**23**	**23**	21	15	107	114	**115**	105
[1, 11]	**22**	**22**	7	5	10	8	8	18	9	**21**	6	6	18	14	**20**	6	6	**21**	18	16	6	6	**21**	20	15	48	48	**85**	66	82
[5, 15]	7	15	12	12	**20**	7	17	12	14	**20**	7	20	16	15	**24**	6	19	19	16	**23**	6	19	19	18	**23**	33	90	78	75	**110**
[10, 20]	5	12	12	12	**22**	5	13	13	12	**22**	5	15	16	16	**22**	3	15	16	15	**21**	3	15	16	15	**20**	21	70	73	70	**107**
[15, 25]	5	16	14	19	**22**	5	16	15	20	**22**	3	15	15	20	**21**	3	17	16	20	**21**	3	18	17	21	**22**	19	82	77	100	**108**
[20, 30]	4	14	19	20	**21**	3	15	20	**21**	**21**	3	16	19	**22**	21	3	17	23	**24**	21	3	17	23	**24**	21	16	79	104	**111**	105
[1, 16]	**26**	**26**	8	5	6	7	7	11	13	**20**	7	7	16	16	**22**	7	7	18	19	**21**	6	6	18	18	**20**	53	53	71	71	**89**
[5, 20]	7	17	14	13	**18**	7	18	12	14	**21**	6	16	11	15	**22**	4	15	14	15	**20**	4	16	14	14	**19**	28	82	65	71	**100**
[10, 25]	5	15	13	15	**21**	5	16	15	18	**21**	5	18	16	18	**20**	3	**20**	15	19	19	3	**21**	17	19	20	21	90	76	89	**101**
[15, 30]	5	14	15	18	**20**	4	14	16	19	**20**	4	17	17	**20**	**20**	3	17	18	**24**	20	3	16	17	**22**	20	19	78	83	**103**	100
sum.	193	440	370	368	**483**	138	398	400	404	**523**	124	402	426	432	**531**	116	409	454	456	**516**	113	414	456	462	**512**	684	2063	2106	2122	**2565**
	P1					P2					P3					P4					P5					All				

Table 5: Grouped Results of t-tests: CISVM vs. Others. Results in bold is significant in sign-tests.

		$Width = 3$	$Width = 5$	$Width = 10$	$Width = 15$
size		450/600/450	300/300/300	300/300/150	150/300/150
mean	CSMIN	**33/43/23**	**44/32/25**	**45/32/23**	**61/15/23**
$\in$	CSMEAN	**38/38/24**	34/38/28	36/35/29	**48/25/27**
$[1, 10]$	CSMAX	**40/37/23**	**40/34/26**	**41/32/27**	45/25/31
mean	CSMIN	**39/34/27**	**40/31/29**	**41/32/26**	36/32/31
$\in$	CSMEAN	**40/35/25**	**44/29/27**	**44/32/23**	**48/26/26**
$(10, 20]$	CSMAX	**40/36/24**	**40/35/25**	**36/43/21**	**38/39/23**
mean	CSMIN	**30/48/22**	**35/42/23**	**45/33/23**	39/33/27
$\in$	CSMEAN	25/49/26	28/48/23	25/52/23	**37/40/23**
$(20, 30]$	CSMAX	22/52/26	23/52/26	19/51/30	21/49/30

imizing the best case risk performs well only when the true cost is very close to the minimal cost.

To study how CISVM's performance will change with different cost intervals, we split 25 cost intervals into 12 groups according to different width and means and show the t-tests results for each group in Table 5. Since there are different numbers of intervals contained in each group, we normalize the win/tie/lose counts for the ease of comparison, with total counts shown in the row *size*. It shows that, when the mean cost is larger than 20, the performance of CISVM decreases, yet it is still comparable to or better than other methods. While when the mean cost is smaller than 20, the performance of CISVM is very similar for intervals with different width and means.

From the experiments we conclude that: (1) The success of CISVM implies that learning with cost intervals should be solved specially, and standard cost-sensitive methods could not handle it well. (2) CISVM is generally the best method over the whole interval. Though we consider only 5 test points, it is very likely that CISVM is the best over the whole interval since it is better than or comparable to the methods learning with the true costs. Even at P1, where CISVM makes the most wrong guess of the true risk, CISVM still beats other methods sometimes. (3) Generally, CISVM is better than CSSVM with cost parameter being any value in a cost interval.

6.3 The Effect of Using Cost Distributions

To see whether more cost information will help improve performance, we consider two typical types of distributions, uniform and normal distributions, in our experiments: $U(1,3)$, $U(1,6)$, $U(1,11)$, $U(5,7)$, $U(5,10)$, $U(5,15)$, $N(5,0.5)$, $N(5,1)$, $N(5,2)$, $N(10,1)$, $N(10,2)$, $N(10,4)$, $N(20,1)$, $N(20,2)$, $N(20,4)$. There are 15 cost distributions altogether. For CODIS, we use example-dependent cost-sensitive SVM [3] as learning algorithm A and set $T = 9$. We compare CODIS with CSSVM taking $E[c]$ as the true cost on the first 15 data sets used above. The settings for the 30 trails experiment and parameter selection are as same as that in the previous section.

Table 6 shows the results for each cost distribution separately. The results of SVM and CSSVM for normal distributions with the same mean value are all the same since they just use the expected cost. It can be found that CODIS can reduce 60% more risks than CSSVM. Paired t-tests and a following sign-tests both at significance level 95% are also carried out. It shows that CODIS is significantly better than other methods for any cost distribution on any data set in all 225 cases (15×15). Moreover, CODIS is very robust to different cost distributions since its results are very similar. For a specific cost distribution $N(10,2)$, its 98.8% confidence interval is [5,15] which can be handled by CISVM. The expected cost is 10 which is used as the true cost in CSSVM. In this case, CSSVM is equivalent to CSMEAN. The results can be found in Table 3 at P3 since the true cost is the mean value. It shows CODIS is much better than CISVM, and CISVM is much better than CSSVM. All the results indicate that when more information can be obtained in addition to cost intervals, a great advantage can be obtained by using the additional information appropriately.

7. CONCLUSION

In many real-world applications, it is generally difficult to have precise costs, while relatively easier to get cost intervals. This paper reports the first study on learning with cost intervals. We propose the CISVM method and experiments show that it is significantly superior to standard cost-sensitive SVMs. We also consider the situation where distributions are known about costs in addition to cost intervals, and propose the CODIS approach which is able to exploit cost distributions to reduce 60% more risks than standard cost-sensitive SVM.

8. ACKNOWLEDGEMENT

This research was supported by the NSFC (60635030), the JiangsuSF (BK2008018), and the 973 Program (2010CB327903).

9. REFERENCES

[1] A. Asuncion and D. Newman. UCI machine learning repository, 2007.

[2] A. P. Bradley. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Patt. Recogn.*, 30(7):1145–1159, 1997.

[3] U. Brefeld, P. Geibel, and F. Wysotzki. Support vector machines with example dependent costs. In *Proc. ECML*, pages 23–34, 2003.

[4] N. V. Chawla, D. A. Cieslak, L. O. Hall, and A. Joshi. Automatically countering imbalance and its empirical relationship to cost. *DMKD*, 17(2):225–252, 2008.

[5] M. Ciraco, M. Rogalewski, and G. Weiss. Improving classifier utility by altering the misclassification cost ratio. In *Proc. Intl. Workshop on Utility-Based Data Mining*, pages 46–52, 2005.

[6] P. Domingos. MetaCost: A general method for making classifiers cost-sensitive. In *Proc. SIGKDD*, pages 155–164, 1999.

[7] C. Drummond and R. C. Holte. Cost curves: An improved method for visualizing classifier performance. *Mach. Learn.*, 65(1):95–130, 2006.

[8] C. Elkan. The foundations of cost-sensitive learning. In *Proc. IJCAI*, pages 973–978, 2001.

[9] M. J. Gardner and D. G. Altman. Confidence intervals rather than p values: Estimation rather than hypothesis testing. *Stat. in Medic.*, 292:746–750, 1986.

[10] X.-Y. Liu and Z.-H. Zhou. The influence of class imbalance on cost-sensitive learning: An empirical study. In *Proc. IEEE ICDM*, pages 970–974, 2006.

[11] A. C. Lozano and N. Abe. Multi-class cost-sensitive boosting with p-norm loss functions. In *Proc. SIGKDD*, pages 506–514, 2008.

[12] H. Masnadi-Shirazi and N. Vasconcelos. Asymmetric boosting. In *Proc. ICML*, pages 609–616, 2007.

[13] V. S. Sheng and C. X. Ling. Thresholding for making classifiers cost-sensitive. In *Proc. AAAI*, 2006.

[14] S. J. Stolfo, W. Fan, W. Lee, A. Prodromidis, and P. K. Chan. Cost-based modeling for fraud and intrusion detection: Results from the JAM project. *DARPA Information Survivability Conference and Exposition*, 2:1130–1144, 2000.

[15] K. M. Ting. An instance-weighting method to induce cost-sensitive trees. *IEEE TKDE*, 14(3):659–665, 2002.

Table 6: CODIS **results (mean ± std.). The minimal risks are in bold.** r **is the improvement ratio of** CODIS **vs. CSSVM.**

	SVM	CSSVM	CODIS	r(%)	SVM	CSSVM	CODIS	r(%)	SVM	CSSVM	CODIS	r(%)
	U(1,3)				U(1,6)				U(1,11)			
breast-c1	31.5±5.0	27.6±1.1	**11.8±2.5**	57.2	46.9±9.0	27.7±2.5	**12.2±2.7**	55.9	106.0±57.2	28.0±0.0	**11.9±2.5**	57.6
breast-w1	10.0±2.6	4.9±1.9	**2.7±1.2**	45.6	10.0±2.6	4.9±2.1	**2.7±1.2**	45.3	10.0±2.6	5.4±3.1	**3.6±1.5**	33.5
credit-a1	41.0±7.0	38.6±5.9	**7.5±2.5**	80.5	63.6±11.7	49.7±6.3	**17.6±4.1**	64.5	113.8±78.1	62.9±8.6	**18.5±4.3**	70.5
credit-g1	81.0±9.7	77.9±8.4	**33.8±5.3**	56.6	253.5±118.8	88.3±6.3	**34.1±5.1**	61.4	410.0±236.0	99.1±3.0	**36.6±5.5**	63.1
diabetes1	64.5±7.3	58.3±7.5	**29.9±6.3**	48.7	87.7±11.0	69.7±7.6	**31.3±6.5**	55.0	103.8±15.5	77.3±8.8	**33.8±6.5**	56.2
german1	80.0±10.6	75.9±6.4	**34.5±5.6**	54.5	149.3±71.8	93.2±11.3	**34.9±5.7**	62.5	211.1±141.7	100.0±0.0	**35.3±5.8**	64.7
haberman1	35.9±5.4	27.0±0.0	**13.8±2.7**	48.8	55.1±9.4	27.0±0.0	**12.5±2.8**	53.8	87.1±16.2	27.0±0.0	**12.7±2.8**	53.0
heart-s1	23.1±5.8	21.9±5.3	**8.6±2.6**	60.6	33.6±8.1	28.2±5.9	**9.4±2.9**	66.8	40.7±9.3	36.0±7.3	**10.6±2.7**	70.5
ionosph1	7.6±2.4	6.7±2.4	**0.3±0.5**	95.5	9.1±3.2	8.3±3.2	**0.3±0.5**	96.4	11.7±4.9	9.9±4.8	**0.3±0.5**	97.0
liver1	47.0±6.2	41.6±5.8	**16.6±2.6**	60.1	62.2±14.7	47.4±2.2	**18.1±2.7**	61.8	86.3±26.9	48.0±0.0	**20.2±3.6**	58.0
sonar1	14.3±4.3	15.3±4.1	**4.7±2.0**	69.1	20.4±6.8	20.3±4.0	**4.7±1.9**	76.7	30.4±11.2	20.9±6.4	**5.2±2.0**	75.2
spambase1	42.5±6.0	41.1±5.8	**12.0±3.1**	70.8	66.6±10.2	56.4±8.6	**14.8±3.4**	73.8	106.8±17.6	76.5±14.0	**13.6±3.6**	82.2
spect1	12.0±2.7	11.5±3.2	**4.8±1.8**	58.3	13.3±3.8	13.0±3.2	**5.3±2.0**	59.2	15.5±6.2	13.4±3.6	**6.2±1.9**	53.7
spectf1	13.2±2.9	13.2±2.9	**4.9±2.6**	63.2	13.2±2.9	13.2±2.9	**5.5±2.6**	58.7	13.2±2.9	13.2±2.9	**5.0±2.6**	62.2
tic-tac1	4.7±1.3	4.7±1.3	**1.1±1.4**	76.8	4.7±1.3	4.7±1.3	**1.5±1.5**	67.6	4.7±1.3	4.7±1.3	**1.4±1.4**	71.1
avg.	*33.9*	*31.1*	***12.5***	*63.1*	*59.3*	*36.8*	***13.7***	*64.0*	*90.1*	*41.5*	***14.3***	*64.6*
	U(5,7)				U(5,10)				U(5,15)			
breast-c1	106.0±57.2	28.0±0.0	**11.9±2.5**	57.6	130.0±74.1	28.0±0.0	**11.9±2.5**	57.6	170.0±102.3	28.0±0.0	**11.9±2.5**	57.6
breast-w1	10.0±2.6	5.4±3.1	**3.6±1.5**	32.9	10.0±2.6	5.8±3.7	**3.6±1.5**	38.9	10.0±2.6	6.6±4.8	**3.5±1.5**	47.2
credit-a1	113.8±78.1	62.9±8.6	**19.1±4.4**	69.6	131.2±100.1	66.9±10.7	**19.1±4.3**	71.4	160.3±136.8	74.0±11.6	**19.4±4.4**	73.8
credit-g1	410.0±236.0	99.1±3.0	**37.6±5.0**	62.0	503.9±306.4	100.0±0.0	**34.3±5.2**	65.7	660.4±423.6	100.0±0.0	**34.3±5.2**	65.7
diabetes1	103.8±15.5	77.3±8.8	**37.7±6.7**	51.2	116.0±19.6	82.3±8.8	**38.5±6.7**	53.3	136.3±26.8	89.0±0.0	**40.4±6.3**	54.6
german1	211.1±141.7	100.0±0.0	**35.5±5.8**	64.5	248.2±183.6	100.0±0.0	**35.7±5.7**	64.3	310.1±253.6	100.0±0.0	**35.7±5.7**	64.3
haberman1	87.1±16.2	27.0±0.0	**12.8±2.8**	52.6	106.3±20.3	27.0±0.0	**12.9±2.8**	52.2	138.3±27.3	27.0±0.0	**13.0±3.0**	51.9
heart-s1	40.7±9.3	36.0±7.3	**8.1±2.4**	77.5	45.2±11.8	38.1±6.2	**8.2±2.4**	78.4	52.7±16.0	39.4±5.2	**8.4±2.3**	78.8
ionosph1	11.7±4.9	9.9±4.8	**0.3±0.5**	97.0	13.3±6.0	10.1±5.9	**0.3±0.5**	97.0	15.8±8.0	17.5±9.2	**0.3±0.5**	98.3
liver1	86.3±26.9	48.0±0.0	**22.6±3.1**	52.8	100.8±34.3	48.0±0.0	**23.6±3.6**	50.8	125.0±46.6	48.0±0.0	**25.1±4.2**	47.8
sonar1	30.4±11.2	20.9±6.4	**5.2±2.0**	75.1	36.5±13.9	23.4±8.0	**6.3±2.2**	73.2	46.6±18.4	25.9±9.9	**6.6±2.3**	74.4
spambase1	106.8±17.6	76.5±14.0	**16.1±4.4**	79.0	130.9±22.1	83.4±15.6	**16.8±5.4**	79.9	156.1±37.4	87.5±13.2	**21.4±8.8**	75.6
spect1	15.5±6.2	13.4±3.6	**5.8±2.2**	56.7	16.8±7.7	15.3±3.7	**5.9±2.1**	61.7	19.0±10.2	18.5±2.8	**6.2±2.6**	66.7
spectf1	13.2±2.9	13.2±2.9	**5.0±2.6**	62.2	13.2±2.9	13.2±2.9	**5.0±2.6**	62.2	13.2±2.9	13.2±2.9	**5.0±2.6**	62.2
tic-tac1	4.7±1.3	4.7±1.3	**1.4±1.4**	71.1	4.7±1.3	4.7±1.3	**1.4±1.4**	71.1	4.7±1.3	4.7±1.3	**1.4±1.4**	71.1
avg.	*90.1*	*41.5*	***14.8***	*64.1*	*107.1*	*43.1*	***14.9***	*65.2*	*134.6*	*45.3*	***15.5***	*66.0*
	N(5,0.5)				N(5,1)				N(5,2)			
breast-c1	90.0±45.9	28.0±0.0	**11.9±2.5**	57.6	90.0±45.9	28.0±0.0	**11.9±2.5**	57.6	90.0±45.9	28.0±0.0	**11.9±2.5**	57.6
breast-w1	10.0±2.6	8.0±3.4	**3.7±1.5**	54.0	10.0±2.6	8.0±3.4	**3.7±1.5**	54.0	10.0±2.6	8.0±3.4	**3.7±1.5**	54.0
credit-a1	86.3±16.5	55.5±8.3	**19.2±4.2**	65.5	86.3±16.5	55.5±8.3	**18.9±4.3**	65.9	86.3±16.5	55.5±8.3	**18.7±4.2**	66.3
credit-g1	347.4±189.1	94.3±5.2	**36.5±5.4**	61.3	347.4±189.1	94.3±5.2	**36.3±5.4**	61.5	347.4±189.1	94.3±5.2	**12.6±31.8**	86.7
diabetes1	95.6±12.7	75.1±9.5	**36.2±6.8**	51.7	95.6±12.7	75.1±9.5	**35.8±6.7**	52.3	95.6±12.7	75.1±9.5	**34.2±6.8**	54.5
german1	186.4±113.7	93.4±5.9	**35.3±5.7**	62.2	186.4±113.7	93.4±5.9	**35.3±5.7**	62.2	186.4±113.7	93.4±5.9	**35.2±5.8**	62.3
haberman1	74.3±13.5	27.0±0.0	**12.6±2.8**	53.3	74.3±13.5	27.0±0.0	**16.2±2.2**	40.0	74.3±13.5	27.0±0.0	**12.5±2.8**	53.7
heart-s1	37.7±7.7	32.9±5.3	**8.1±2.4**	75.3	37.7±7.7	32.9±5.3	**8.1±2.4**	75.3	37.7±7.7	32.9±5.3	**10.6±2.8**	67.8
ionosph1	10.7±4.2	9.8±4.2	**0.3±0.5**	96.9	10.7±4.2	9.8±4.2	**0.3±0.5**	96.9	10.7±4.2	9.8±4.2	**0.3±0.5**	96.9
liver1	76.7±22.0	48.1±0.9	**21.4±3.2**	55.5	76.7±22.0	48.1±0.9	**21.2±3.0**	56.0	76.7±22.0	48.1±0.9	**20.4±2.8**	57.7
sonar1	26.4±9.4	23.3±3.4	**4.8±2.0**	79.3	26.4±9.4	23.3±3.4	**4.8±2.0**	79.3	26.4±9.4	23.3±3.4	**4.8±2.0**	79.4
spambase1	90.7±14.6	68.6±13.8	**15.9±4.6**	76.9	90.7±14.6	68.6±13.8	**15.3±4.7**	77.6	90.7±14.6	68.6±13.8	**14.1±4.2**	79.4
spect1	14.6±5.2	13.1±3.2	**6.3±1.9**	52.3	14.6±5.2	13.1±3.2	**5.7±2.1**	56.3	14.6±5.2	13.1±3.2	**5.7±2.2**	56.9
spectf1	13.2±2.9	13.2±2.9	**5.0±2.6**	62.2	13.2±2.9	13.2±2.9	**5.0±2.6**	62.2	13.2±2.9	13.2±2.9	**5.8±2.5**	55.9
tic-tac1	4.7±1.3	4.7±1.3	**2.0±1.5**	57.7	4.7±1.3	4.7±1.3	**2.0±1.5**	58.5	4.7±1.3	4.7±1.3	**0.2±0.8**	95.1
avg.	*77.7*	*39.7*	***14.6***	*64.1*	*77.7*	*39.7*	***14.7***	*63.7*	*77.7*	*39.7*	*12.7*	*68.3*
	N(10,1)				N(10,2)				N(10,4)			
breast-c1	170.0±102.3	28.0±0.0	**11.9±2.5**	57.6	170.0±102.3	28.0±0.0	**11.9±2.5**	57.6	170.0±102.3	28.0±0.0	**11.9±2.5**	57.6
breast-w1	10.0±2.6	6.6±4.8	**3.5±1.5**	47.2	10.0±2.6	6.6±4.8	**3.5±1.5**	47.2	10.0±2.6	6.6±4.8	**3.5±1.5**	47.2
credit-a1	160.3±136.8	74.0±11.6	**19.4±4.4**	73.8	160.3±136.8	74.0±11.6	**19.3±4.5**	73.9	160.3±136.8	74.0±11.6	**19.3±4.4**	73.9
credit-g1	660.4±423.6	100.0±0.0	**34.3±5.2**	65.7	660.4±423.6	100.0±0.0	**34.3±5.2**	65.7	660.4±423.6	100.0±0.0	**34.3±5.2**	65.7
diabetes1	136.3±26.8	89.0±0.0	**41.0±6.3**	53.9	136.3±26.8	89.0±0.0	**40.5±6.7**	54.5	136.3±26.8	89.0±0.0	**39.4±6.7**	55.7
german1	310.1±253.6	100.0±0.0	**35.8±5.7**	64.2	310.1±253.6	100.0±0.0	**35.7±5.7**	64.3	310.1±253.6	100.0±0.0	**35.7±5.7**	64.3
haberman1	138.3±27.3	27.0±0.0	**13.0±3.0**	51.9	138.3±27.3	27.0±0.0	**13.0±3.0**	51.9	138.3±27.3	27.0±0.0	**13.0±3.0**	51.9
heart-s1	52.7±16.0	39.4±5.2	**8.4±2.3**	78.7	52.7±16.0	39.4±5.2	**8.4±2.3**	78.8	52.7±16.0	39.4±5.2	**8.4±2.3**	78.8
ionosph1	15.8±8.0	17.5±9.2	**0.3±0.5**	98.3	15.8±8.0	17.5±9.2	**0.3±0.5**	98.3	15.8±8.0	17.5±9.2	**0.3±0.5**	98.3
liver1	125.0±46.6	48.0±0.0	**25.8±3.9**	46.3	125.0±46.6	48.0±0.0	**25.4±4.2**	47.2	125.0±46.6	48.0±0.0	**24.6±4.0**	48.8
sonar1	46.6±18.4	25.9±9.9	**6.7±2.3**	74.3	46.6±18.4	25.9±9.9	**6.6±2.3**	74.6	46.6±18.4	25.9±9.9	**6.6±2.3**	74.6
spambase1	156.1±37.4	87.5±13.2	**22.7±8.2**	74.1	150.5±37.4	91.4±13.2	**21.7±8.6**	75.2	156.1±37.4	87.5±13.2	**20.1±7.9**	77.1
spect1	19.0±10.2	18.5±2.8	**5.5±1.9**	70.1	19.0±10.2	18.5±2.8	**5.8±2.3**	68.5	19.0±10.2	18.5±2.8	**5.8±2.3**	68.5
spectf1	13.2±2.9	13.2±2.9	**5.0±2.6**	62.2	13.2±2.9	13.2±2.9	**5.0±2.6**	62.2	13.2±2.9	13.2±2.9	**5.0±2.6**	62.2
tic-tac1	4.7±1.3	4.7±1.3	**1.4±1.4**	71.1	4.7±1.3	4.7±1.3	**1.4±1.4**	71.1	4.7±1.3	4.7±1.3	**1.4±1.4**	71.1
avg.	*134.6*	*45.3*	***15.6***	*66.0*	*134.6*	*45.3*	***15.5***	*66.1*	*134.6*	*45.3*	***15.3***	*66.4*
	N(20,1)				N(20,2)				N(20,4)			
breast-c1	330.0±215.2	30.6±7.8	**11.9±2.5**	61.2	330.0±215.2	30.6±7.8	**11.9±2.5**	61.2	330.0±215.2	30.6±7.8	**11.9±2.5**	61.2
breast-w1	10.0±2.6	9.1±8.1	**3.5±1.5**	61.8	10.0±2.6	9.1±8.1	**3.5±1.5**	61.8	10.0±2.6	9.1±8.1	**3.4±1.5**	62.1
credit-a1	276.6±283.6	104.0±27.3	**20.1±4.4**	80.6	276.6±283.6	104.0±27.3	**20.1±4.4**	80.6	276.6±283.6	104.0±27.3	**20.1±4.4**	80.6
credit-g1	1286.4±892.7	99.1±0.6	**34.3±5.4**	65.4	1286.4±892.7	99.1±0.6	**34.3±5.4**	65.4	1286.4±892.7	99.1±0.6	**34.3±5.4**	65.4
diabetes1	217.6±55.8	89.0±0.0	**52.1±6.3**	41.5	217.6±55.8	89.0±0.0	**52.1±6.3**	41.5	217.6±55.8	89.0±0.0	**52.1±6.3**	41.5
german1	557.4±533.4	100.0±0.0	**37.4±4.9**	62.6	557.4±533.4	100.0±0.0	**37.3±4.9**	62.7	557.4±533.4	100.0±0.0	**37.4±4.8**	62.6
haberman1	303.6±52.6	27.0±0.0	**20.0±2.0**	26.0	303.6±52.6	27.0±0.0	**19.9±1.9**	26.2	303.6±52.6	27.0±0.0	**19.9±1.9**	26.2
heart-s1	82.7±32.8	40.0±0.0	**9.1±2.6**	77.3	82.7±32.8	40.0±0.0	**9.1±2.6**	77.3	82.7±32.8	40.0±0.0	**9.1±2.6**	77.3
ionosph1	26.2±15.9	22.1±16.9	**0.3±0.5**	98.6	26.2±15.9	22.1±16.9	**0.3±0.5**	98.6	26.2±15.9	22.1±16.9	**0.3±0.5**	98.6
liver1	776.6±167.8	49.8±11.0	**28.3±4.3**	43.2	776.6±167.8	49.8±11.0	**28.3±4.3**	43.2	776.6±167.8	49.8±11.0	**28.3±4.3**	43.2
sonar1	86.9±36.7	33.6±16.3	**4.8±2.1**	85.7	86.9±36.7	33.6±16.3	**4.8±2.1**	85.7	86.9±36.7	33.6±16.3	**4.8±2.1**	85.7
spambase1	180.3±20.8	107.2±16.6	**32.3±7.5**	69.9	180.3±20.8	107.2±16.6	**32.1±7.5**	70.0	180.3±20.8	107.2±16.6	**31.1±8.4**	71.0
spect1	27.6±20.5	21.8±6.5	**11.7±3.6**	46.4	27.6±20.5	21.8±6.5	**11.6±3.6**	46.7	27.6±20.5	21.8±6.5	**11.6±3.6**	46.9
spectf1	13.2±2.9	13.2±2.9	**5.0±2.6**	62.0	13.2±2.9	13.2±2.9	**5.0±2.6**	62.0	13.2±2.9	13.2±2.9	**5.0±2.6**	62.0
tic-tac1	4.7±1.3	4.7±1.3	**1.4±1.4**	71.1	4.7±1.3	4.7±1.3	**1.4±1.4**	71.1	4.7±1.3	4.7±1.3	**1.4±1.4**	71.1
avg.	*278.7*	*50.1*	***18.1***	*63.6*	*278.7*	*50.1*	***18.1***	*63.6*	*278.7*	*50.1*	***18.0***	*63.7*

[16] K. M. Ting and Z. Zheng. Boosting trees for cost-sensitive classifications. In *Proc. ECML*, pages 190–195, 1998.

[17] K. W. Wong, S. Zho, Q. Yang, and J. M. S. Yeung. Mining customer value: From association rules to direct marketing. *DMKD*, 11(1):57–79, 2005.

[18] M. J. Worsham, J. Abram, U. Raju, A. Kapke, M. Lu, J. Cheng, D. Mott, and S. R. Wolman. Breast cancer incidence in a cohort of women with benign breast disease from a multiethnic, primary health care population. *The Breast Journal*, 13(2):115–121, 2007.

[19] B. Zadrozny and C. Elkan. Learning and making decisions when costs and probabilities are both unknown. In *Proc. SIGKDD*, pages 204–213, 2001.

[20] B. Zadrozny, J. Langford, and N. Abe. Cost-sensitive learning by cost-proportionate example weighting. In *Proc. IEEE ICDM*, pages 435–442, 2003.

[21] Z.-H. Zhou and X.-Y. Liu. Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE TKDE*, 18(1):63–77, 2006.

[22] Z.-H. Zhou and Y. Yu. Ensembling local learners through multimodal perturbation. *IEEE TSMCB*, 35(4):725–735, 2005.