A Simple Methodology for Soft Cost-sensitive Classification

Te-Kang Jan
Institute of Information Science, Academia Sinica, Taipei, Taiwan
tekang@iis.sinica.edu.tw

Da-Wei Wang
Institute of Information Science, Academia Sinica, Taipei, Taiwan
wdw@iis.sinica.edu.tw

Chi-Hung Lin
Institute of Microbiology and Immunology, National Yang-Ming University, Taipei, Taiwan
linch@ym.edu.tw

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

ABSTRACT

Many real-world data mining applications need varying cost for different types of classification errors and thus call for cost-sensitive classification algorithms. Existing algorithms for cost-sensitive classification are successful in terms of minimizing the cost, but can result in a high error rate as the trade-off. The high error rate holds back the practical use of those algorithms. In this paper, we propose a novel cost-sensitive classification methodology that takes both the cost and the error rate into account. The methodology, called soft cost-sensitive classification, is established from a multi-criteria optimization problem of the cost and the error rate, and can be viewed as regularizing cost-sensitive classification with the error rate. The simple methodology allows immediate improvements of existing cost-sensitive classification algorithms. Experiments on the benchmark and the real-world data sets show that our proposed methodology indeed achieves lower test error rates and similar (sometimes lower) test costs than existing cost-sensitive classification algorithms.

Categories and Subject Descriptors

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

Keywords

Classification, Cost-sensitive learning, Multicriteria optimization, Regularization

KDD'12, August 12–16, 2012, Beijing, China.

1. INTRODUCTION

Classification is important for machine learning and data mining [16,17]. Traditionally, the regular classification problem aims at minimizing the rate of misclassification errors. In many real-world applications, however, different types of errors are often charged with different costs. For instance, in bacteria classification, mis-classifying a Gram-positive species as a Gram-negative one leads to totally ineffective treatments and is hence more serious than misclassifying a Gram-positive species as another Gram-positive one [24,31]. Similar application needs are shared by targeted marketing, information retrieval, medical decision making, object recognition and intrusion detection [1, 14, 15, 26, 33, 34], and can be formalized as the cost-sensitive classification problem. In fact, cost-sensitive classification can be used to express any finite-choice and bounded-loss supervised learning problems [5]. Thus, it has been attracting much research attention in recent years, in terms of both new algorithms and new applications [4, 6, 23, 24, 27, 34, 36].

Studies in cost-sensitive classification often reveal a trade-off between costs and error rates [23, 27, 36]. Mature regular classification algorithms can achieve significantly lower error rates than their cost-sensitive counterparts, but result in higher expected costs; state-of-the-art cost-sensitive classification algorithms can reach significantly lower expected cost than their regular classification counterparts, but are often at the expense of higher error rates. In addition, cost-sensitive classification algorithms are "sensitive" to large cost components and can thus be conservative or even "paranoid" in order to avoid making any big mistakes. The sensitivity makes cost-sensitive classification algorithms prone to overfitting the data or the costs. In fact, it has been observed that for some simpler classification tasks, cost-sensitive classification algorithms are inferior to regular classification ones in terms of even the expected test cost because of the overfitting [27, 36].

The expense of high error rates and the potential risk of overfitting holds back the practical use of cost-sensitive classification algorithms. Arguably, applications call for classifiers that can reach low costs *and* low error rates. The task of obtaining such a classifier has been studied for binary

cost-sensitive classifier [30], but the more general task for multi-class cost-sensitive classification is yet to be tackled.

In this paper, we propose a methodology to tackle the task. The methodology takes both the costs and the error rates into account and matches the realistic needs better. We name the methodology *soft cost-sensitive classification* to distinguish it from existing *hard cost-sensitive classification* algorithms that focus on only the costs. The methodology is designed by formulating the associated problem as a multicriteria optimization task [19]: one criterion being the cost and the other being the error rate. Then, the methodology solves the task by the weighted sum approach for multicriteria optimization [38]. The simplicity of the weighted sum approach allows immediate reuse of modern cost-sensitive classification algorithms as the core tool. In other words, with our proposed methodology, promising (hard) cost-sensitive classification algorithms can be immediately improved via soft cost-sensitive classification, with performance guarantees on costs and error rates supported by the theory behind multicriteria optimization.

We conduct experiments to validate the performance of the proposed methodology on the benchmark and the real-world data sets. Experimental results suggest that soft cost-sensitive classification can indeed achieve both low costs and low error rates. In particular, soft cost-sensitive classification algorithms out-perform regular ones in terms of the test costs on most of the data sets. In addition, soft cost-sensitive classification algorithms reach significantly lower test error rates than their hard siblings, while achieving similar (sometimes better) test costs. The observations are consistent across four different sets of tasks: the traditional benchmark tasks in cost-sensitive classification for balancing class influences [12], new benchmark tasks designed for examining the effect of using large cost components, the real-world medical task for classifying bacteria [24], and the real-world task for intrusion detection in KDD Cup 1999 [3].

The paper is organized as follows. We formally introduce the regular and the cost-sensitive classification problems in Section 2, and discuss related works on cost-sensitive classification. Then, we present the proposed methodology of soft cost-sensitive classification in Section 3. We discuss the empirical performance of the proposed methodology on the benchmark and the real-world data sets in Section 4. Finally, we conclude in Section 5.

2. COST-SENSITIVE CLASSIFICATION

We shall start by defining the regular classification problem and extend it to the cost-sensitive one. Then, we briefly review existing works on cost-sensitive classification.

In the regular classification problem, we are given a training set $\mathcal{S} = \{(\mathbf{x}_n, y_n)\}_{n=1}^N$, where the input vector $\mathbf{x}_n$ belongs to some domain $\mathcal{X} \subseteq \mathbb{R}^D$, the label y_n comes from the set $\mathcal{Y} = \{1, \dots, K\}$ and each example $(\mathbf{x}_n, y_n)$ is drawn independently from an unknown distribution $\mathcal{D}$ on $\mathcal{X} \times \mathcal{Y}$. The task of regular classification is to use the training set $\mathcal{S}$ to find a classifier $g\colon \mathcal{X} \to \mathcal{Y}$ such that the expected error rate $E(g) = \underset{(\mathbf{x},y)\sim\mathcal{D}}{\mathcal{E}} [\![y \neq g(\mathbf{x})]\!]$ is small,[1] where the expected error rate $E(g)$ penalizes every type of mis-classification error equally.

[1]The Boolean operation $[\![\cdot]\!]$ is 1 when the argument is true and 0 otherwise.

Cost-sensitive classification extends regular classification by charging different costs for different types of classification errors. We adopt the example-dependent setting of cost-sensitive classification, which is rather general and can be used to express other popular settings [6, 23, 25, 27, 36]. The example-dependent setting couples each example $(\mathbf{x}, y)$ with a cost vector $\mathbf{c} \in [0, \infty)^K$, where the k-th component of $\mathbf{c}$ quantifies the cost for predicting the example $\mathbf{x}$ as class k. The cost $\mathbf{c}[y]$ of the intended class y is naturally assumed to be 0, the minimum cost. 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 independently from an unknown cost-sensitive distribution $\mathcal{D}_c$ on $\mathcal{X} \times \mathcal{Y} \times [0, \infty)^K$, the task of cost-sensitive classification is to use $\mathcal{S}_c$ to find a classifier $g\colon \mathcal{X} \to \mathcal{Y}$ such that the expected cost $E_c(g) = \underset{(\mathbf{x},y,\mathbf{c})\sim\mathcal{D}_c}{\mathcal{E}} \mathbf{c}[g(\mathbf{x})]$ is small.

One special case of the example-dependent setting is the class-dependent setting, in which the cost vectors $\mathbf{c}$ are taken from the y-th row of a cost matrix $\mathbf{C}\colon \mathcal{Y} \times \mathcal{Y} \to [0, \infty)^K$. Each entry $\mathbf{C}(y, k)$ of the cost matrix represents the cost for predicting a class-y example as class k. The special case is commonly used in some applications and some benchmark experiments [23, 24, 27].

Regular classification can be viewed as a special case of the class-dependent setting, which is in term a special case of the example-dependent setting. In particular, take a cost matrix that contains 0 in the diagonals and 1 elsewhere, which equivalently corresponds to the regular cost vectors $\bar{\mathbf{c}}_y$ with entries $\bar{\mathbf{c}}_y[k] = [\![y \neq k]\!]$. Then, the expected cost $E_c(g)$ with respect to $\{\bar{\mathbf{c}}_y\}$ is the same as the expected error rate $E(g)$. In other words, regular classification algorithms can be viewed as "wiping out" the given cost information and replacing it with a naïve cost matrix. Intuitively, such algorithms may not work well for cost-sensitive classification because of the wiping out.

The intuition leads to the past decade of studying cost-sensitive classification algorithms that respect the cost information during training and/or prediction. The cost-sensitive classification algorithms can be grouped into two categories: the binary ($K = 2$) cases and the multiclass ($K > 2$) cases. Binary cost-sensitive classification is well-understood in theory and in practice. In particular, every binary cost-sensitive classification problem can be reduced to a binary regular classification one by re-weighting the examples based on the costs [13, 39]. Multiclass cost-sensitive classification, however, is more difficult than the binary one, and is an ongoing research topic.

MetaCost [12] is one of the earliest multiclass cost-sensitive classification algorithms. It makes any regular classification algorithm cost-sensitive by re-labeling the training examples. Somehow the re-labeling procedure depends on an overly-ideal assumption, which makes it hard to rigorously analyze the performance of MetaCost in theory. Many other early approaches suffer from similar shortcomings [29].

In order to design multiclass cost-sensitive classification algorithms with stronger theoretical guarantees, modern cost-sensitive classification algorithms are mostly reduction-based, which allows not only reusing mature existing algorithms for cost-sensitive classification, but also extending existing theoretical results to the area of cost-sensitive classification. For instance, [1] reduces the multiclass cost-sensitive classification problem into several multiclass regular classification

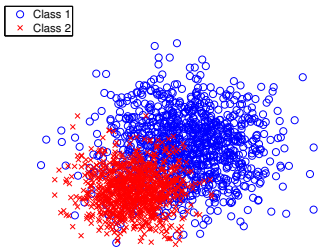


Figure 1: a two-dimensional artificial data set

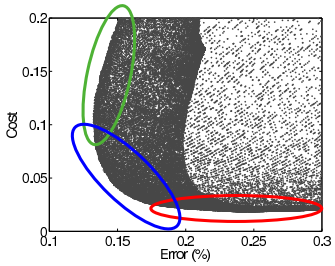


Figure 2: the different goals of regular (green), cost-sensitive (red) and soft cost-sensitive (blue) classification algorithms

problems using a boosting-style method and some intermediate traditional classifiers. The reduction is somehow too sophisticated for practical use. [40] derives another reduction approach from multiclass cost-sensitive classification to multiclass regular classification based on re-weighting with the solution to a linear system. The proposed reduction approach works with sound theoretical guarantees when the linear system attains a non-trivial solution; otherwise the approach decomposes the multiclass cost-sensitive classification problem to several binary classification problems to get an approximate solution [40].

There are many more studies on reducing multiclass cost-sensitive classification to binary cost-sensitive classification by decomposing the multiclass problem with a suitable structure and embedding the cost vectors into the weights for the re-weighted binary classification problems. For instance, cost-sensitive one-versus-one (CSOVO; [27]) and weighted all-pair (WAP; [5]) are based on pairwise comparisons of the classes. Another leading approach within the family is cost-sensitive filter tree (CSFT; [6]), which is based on a single-elimination tournament of competing classes.

Yet another family of approaches reduce the multiclass cost-sensitive classification problem into regression ones by embedding the cost vectors in the real-valued labels instead of the weights [35]. A promising representative of the family is to reduce to one-sided regression (OSR; [36]). Based on some earlier comparisons on general benchmark data sets [23, 36], OSR, CSOVO and CSFT are some of the leading algorithms that can reach state-of-the-art performance. Each algorithm corresponds to a popular sibling for regular classification. In particular, the common one-versus-all decomposition (OVA) [21] is the special case of OSR, the one-versus-one decomposition (OVO) [21] is the special case of CSOVO, and the modern filter tree decomposition (FT) [6] is the special case of CSFT. The regular classification algorithms, OVA, OVO and FT, do not consider any costs during their training. On the other hand, the cost-sensitive ones, OSR, CSOVO and CSFT, respect the costs faithfully during their training.

3. SOFT COST-SENSITIVE CLASSIFICATION

The difference between regular and cost-sensitive classification is illustrated with a binary and two-dimensional artificial data set shown in Figure 1. Class 1 is generated from a Gaussian distribution of standard deviation $\frac{4}{5}$; class 2 is generated from a Gaussian distribution of standard deviation $\frac{1}{2}$; the centers of the two classes are of $\sqrt{2}$ apart. We consider a cost matrix of $\begin{bmatrix} 0 & 1 \\ 30 & 0 \end{bmatrix}$. Then, we enumerate many linear classifiers in $\mathbb{R}^2$ and evaluate their average errors and average costs. The results are plotted in Figure 2. Each black point represents the achieved (error, cost) of one linear classifier.[2] We can see that there is a region of low-cost linear classifiers, as circled in red. There is also a region of low-error linear classifiers, as circled in green. Modern cost-sensitive classification algorithms are designed to seek for *something* in the red region, which contains classifiers with a wide range of different errors. Traditional regular classification algorithms, on the other hand, are designed to locate something in the green region (without using the cost information), which is far from the lowest achievable cost. In other words, there is a trade-off between the cost and the error, while cost-sensitive and regular classification each takes the trade-off to the extreme.

Figure 2 motivates us to study the methodology for aiming at the blue region instead. The region does not take the trade-off between the cost and the error to the extreme, and contains classifiers that are of low cost *and* low error. Those classifiers match the real-world application needs better, with the cost being the subjective measure of performance and the error being the objective safety-check. The blue region improves the green one (regular) by taking the cost into account; the blue region also improves the red one (cost-sensitive) by keeping the error under control. The three regions, as depicted, are not meant to be disjoint. The blue region may contain the better cost-sensitive classifiers in its intersection with the green region, and the better regular classifiers in its intersection with the red region.

Figure 2 results from a simple artificial data set for the illustrative purpose. When applying more sophisticated classifiers on real-world data sets, the set of achievable (error, cost) may be of a more complicated shape—possibly non-convex, for instance. Somehow the essence of the problem remains the same: cost-sensitive classification only knocks down the cost and results in a red region at the bottom; regular classification only considers the error and lands on a green region at the left; our proposed methodology focuses on a blue region at the left-bottom, hopefully achieving the better for both criteria.

Formally speaking, regular classification algorithm is a process from $\mathcal{S}$ to g such that $E(g)$ is small. Cost-sensitive classification algorithm, on the other hand, is a process from $\mathcal{S}_c$ to g such that $E_c(g)$ is small. We now want a process

[2]Ideally, the points should be dense. The uncrowded part comes from simulating with a finite enumeration process.

from $\mathcal{S}_c$ to g such that both $E(g)$ and $E_c(g)$ are small, which can be written as

$$\min_g \mathbf{E}(g) = [E_c(g), E(g)] \text{ subject to all feasible } g. \quad (1)$$

The vector $\mathbf{E}$ represents the two criteria of interest.

Such a problem belongs to multicriteria optimization [19], which deals with multiple objective functions. The general form of multicriteria optimization is

$$\min_g \mathbf{F}(g) = [F_1(g), F_2(g), \ldots, F_M(g)] \quad \text{subject to all feasible } g, \quad (2)$$

where M is the number of criteria. For a multicriteria optimization problem (2), often there is no global optimal solution g^* that is the best in terms of every dimension (criterion) within $\mathbf{F}$. Instead, the goal of (2) is to seek for the set of "better" solutions, usually referred to as the Pareto-optimal front [20]. Formally speaking, consider two feasible candidates g_1 and g_2. The candidate g_1 is said to *dominate* g_2 if $F_m(g_1) \leq F_m(g_2)$ for all m while $F_i(g_1) < F_i(g_2)$ for some i. The Pareto-optimal front is the set of all non-dominated solutions [19].

Solving the multicriteria optimization problem is not an easy task, and there are many sophisticated techniques, including evolutionary algorithms like Non-dominated Sorting Genetic Algorithms [11] and Strength Pareto Evolutionary [9]. One important family of techniques is to transform the problem to a single-criterion optimization one that we are more familiar with. A simple yet popular approach of the family considers a non-negative linear combination of all the criteria F_m, which is called the weighted sum approach [38]. In particular, the weighted sum approach solves the following optimization problem:

$$\min_g \sum_{m=1}^{M} \alpha_m F_m(g) \text{ subject to all feasible } g, \quad (3)$$

where $\alpha_m \geq 0$ is the weight (importance) of the m-th criterion. By varying the values of α_m, the weighted sum approach identifies *some* of the solutions that are on the tangential of the Pareto-optimal front [19]. The drawback of the approach [10] is that *not all* the solutions within the Pareto-optimal front can be found when the achievable set of $\mathbf{F}(g)$ is non-convex.

We can reach the goal of getting a low-cost and low-error classifier by formulating a multicriteria optimization problem with $M = 2$, $F_1(g) = E_c(g)$ and $F_2(g) = E(g)$. Without loss of generality, let $\alpha_1 = 1 - \alpha$ and $\alpha_2 = \alpha$ for $\alpha \in [0, 1]$, the weighted sum approach solves

$$\min_g (1 - \alpha) E_c(g) + \alpha E(g), \quad (4)$$

which is the same as

$$\min_g \mathop{\mathcal{E}}_{(\mathbf{x},y,\mathbf{c})\sim\mathcal{D}_c} (1 - \alpha)\Big(\mathbf{c}[g(\mathbf{x})]\Big) + \alpha\Big(\bar{\mathbf{c}}_y[g(\mathbf{x})]\Big) \quad (5)$$

with the regular cost vectors $\bar{\mathbf{c}}_y$ defined in Section 2. For any given α, such an optimization problem is exactly a cost-sensitive classification one with modified cost vectors $\tilde{\mathbf{c}} = (1 - \alpha)\mathbf{c} + \alpha\bar{\mathbf{c}}_y$. Then, modern cost-sensitive classification algorithms can be applied to locate a decent g, which would belong to the Pareto-optimal front with respect to $E_c(g)$ and $E(g)$.

The weighted sum approach has also been implicitly taken by other algorithms in machine learning. For instance, [32] combines the pairwise ranking criterion and squared regression criterion and shows that the resulting algorithm achieves the best performance on both criteria. Our proposed methodology similarly utilizes the simplicity of the weighted sum approach to allow seamless reuse of modern cost-sensitive classification algorithms. If other techniques for multicriteria optimization (such as evolutionary computation) are taken instead, new algorithms need to be designed to accompany the techniques. Given the prevalence of promising cost-sensitive classification algorithms (see Section 2), we thus choose to study only the weighted sum approach.

The parameter α in (4) can be intuitively explained as a soft control of the trade-off between costs and errors, with $\alpha = 0$ and $\alpha = 1$ being the two extremes. The traditional (hard) cost-sensitive classification problem is a special case of soft cost-sensitive classification with $\alpha = 0$. On the other hand, the regular classification problem is a special case of soft cost-sensitive classification with $\alpha = 1$.

Another explanation behind (4) is regularization. From Figure 2, there are many low-cost classifiers in the red region. When picking one classifier using only the limited information in the training set $\mathcal{S}_c$, the classifier can be overfitting. The added term $\alpha E(g)$ can be viewed as restricting the number of low-cost classifiers by only favoring those with lower error rates. This similar explanation can be found from [30], which considers cost-sensitive classification in the binary case. Furthermore, the restriction is similar to common regularization schemes, where a penalty term on complexity is used to limit the number of candidate classifiers [2].

We illustrate the regularization property of soft-sensitive classification with the data set vowel as an example. The details of the experimental procedures will be introduced in Section 4. The test cost of soft cost-sensitive classification with various α when coupled with the one-sided regression (OSR) algorithm is shown in Figure 3. For this data set, the lowest test cost does not happen at $\alpha = 0$ (hard cost-sensitive) nor $\alpha = 1$ (non cost-sensitive). By choosing the regularization parameter α appropriately, some intermediate, non-zero values of α (soft cost-sensitive) could lead to better test performance. The figure reveals the potential of soft cost-sensitive classification not only to improve the test error with the added $\alpha E(g)$ term during optimization, but also to possibly improve the test cost with the effect of regularization.

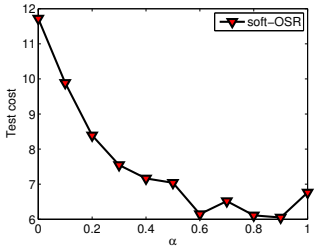


Figure 3: the effect of the regularization parameter α on soft cost-sensitive classification

4. EXPERIMENTS

In this section, we set up experiments to validate the usefulness of the proposed methodology of soft cost-sensitive classification in various scenarios. We take three state-of-the-art multiclass cost-sensitive classification algorithms (see Section 2). Then we examine if the proposed methodology can improve them. The three algorithms are one-sided regression (OSR), cost-sensitive one-versus-one (CSOVO) and cost-sensitive filter tree (CSFT). We also include their regular classification siblings, one-versus-all (OVA), one-versus-one (OVO), and filter tree (FT) for comparisons. The other state-of-the-art multiclass cost-sensitive classification algorithms would also be compared in the longer version of this paper.

We couple all the algorithms with the support vector machine (SVM) [37] with the perceptron kernel [28] as the internal learner for the reduced problem, and take LIBSVM [8] as the SVM solver.[3] The regularization parameter λ of SVM is chosen within $\{2^{10}, 2^7, \ldots, 2^{-2}\}$ and the parameter α for soft cost-sensitive classification is chosen within $\{0, 0.1, \ldots, 1\}$. For the hard or soft cost-sensitive classification algorithms, the best parameter setting is chosen by minimizing the 5-fold cross-validation cost. For the regular classification algorithms, which are not supposed to access any cost information in training or in validation, the best parameter λ is chosen by minimizing the 5-fold cross-validation error.

We consider four sets of tasks: the traditional benchmark tasks for balancing the influence of each class, new benchmark tasks for emphasizing some of the classes, a real-world biomedical task for classifying bacteria (see Section 1) and the KDD Cup 1999 task for the intrusion detection. The four sets of broad tasks will demonstrate that soft cost-sensitive classification is useful both as a general algorithmic methodology and as specific application tools.

4.1 Comparison on Benchmark Tasks

Twenty-two real-world data sets (iris, wine, glass, vehicle, vowel, segment, dna, satimage, usps, zoo ,yeast, pageblock, anneal, solar, splice, ecoli, nursery, soybean, arrhythmia, optdigits, mfeat, pendigit) are used in our experiment. To the best of our knowledge, our experiment is the most extensive empirical study on cost-sensitive classification in terms of the number of data sets taken. All data sets come from the UCI Machine Learning Repository [18] except usps [22]. In each run of the experiment, 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 data sets do not contain any cost information and we make them cost-sensitive by adopting the randomized proportional benchmark that was similarly used by [5, 27, 36]. In particular, the benchmark is class-dependent and is based on 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 $\left[0, \frac{|\{n:y_n=k\}|}{|\{n:y_n=y\}|}\right]$. This means that mis-classifying a rare class as a frequent one is of a high cost in expectation. In other words, the benchmark can be used to balance the influence of each class. We further scale every $\mathbf{C}(y, k)$ to $[0, 1]$ by dividing it with the largest component in $\mathbf{C}$. We then record the average test costs and their standard errors for all algorithms over 20 random runs in Table 1. We also report the average test errors in Table 2.

From Table 1, soft-OSR and soft-CSOVO usually result in the lowest test cost. Most importantly, soft-OSR is among the best algorithms (bold) on 17 of the 22 data sets, and achieves the lowest cost on 8 of them. The follow-ups, OSR and CSOVO, were the state-of-the-art algorithms in cost-sensitive classification and reach promising performance often. Filter-tree-based algorithms (FT, CSFT, soft-CSFT) are generally falling behind, and so are the regular classification algorithms (OVA, OVO, FT). The results justify that soft cost-sensitive classification can lead to similar and sometimes even better performance when compared with state-of-art cost-sensitive classification algorithms.

On the other hand, when we move to Table 2, regular classification algorithms like OVA and OVO generally achieve the lowest test errors. The hard cost-sensitive classification ones result in the highest test errors; soft ones lie in between.

Overall, soft cost-sensitive classification is better than the regular sibling in terms of the cost, the major criterion. It is similar to (sometimes better than) the hard sibling in terms of the cost, but usually better in terms of the error. We further justify the claims above by comparing the average test cost between soft cost-sensitive classification algorithms with their corresponding siblings for regular classification and hard cost-sensitive classification using a pairwise one-tailed t-test of significance level 0.1, as shown in Table 3. For each family of algorithms (OVA, OVO or FT), soft cost-sensitive classification algorithms are generally among the best of the three, and are significantly better than their regular siblings.

Table 4 shows the same t-test for comparing the test errors between soft cost-sensitive classification algorithms and their hard siblings. We see that soft-OSR improves OSR on 16 of the 22 data sets in terms of the test error; soft-CSOVO improves CSOVO on 13 of the 22; soft-CSFT improves CSFT on 14 of the 22. Given the similar test costs between soft and hard cost-sensitive classification algorithms in Table 3, the significant improvements on the test error justify that soft cost-sensitive classification algorithms are better choices for practical applications.

4.2 Comparison on New Benchmark Tasks: Emphasizing Cost

Next, we explore the usefulness of the algorithms with a different benchmark for generating the costs. Consider a situation where one hopes to indicate some of the classes is important. Traditionally, this task is done with re-weighting the examples of those classes, which corresponds to scaling the rows of the cost matrix. As discussed in Section 2, cost-sensitive classification is more sophisticated than re-weighting. In particular, it allows us to mark important classes by scaling up some *columns* of the cost matrix. In our benchmark, we scale up one random column of the regular cost matrix (that contains $\bar{\mathbf{c}}_y$) by an emphasis parameter w, and we call the benchmark *emphasizing cost*.

We vary the the emphasis parameter w between $\{10^2, 10^3, \ldots, 10^6\}$ to examine the stability of the algorithms when using large cost components. The results are shown in Figure 4. Due to the page limits, we only report the results of OSR and soft-OSR on iris, vehicle, and segment. Results on other data sets are similar and will be reported in a longer

[3] We use the cost-sensitive SVM implementation at `http://www.csie.ntu.edu.tw/~htlin/program/cssvm/`

Table 1: average test cost ($\cdot 10^{-3}$) on benchmark data sets

data set	OVA	OSR	soft-OSR	OVO	CSOVO	soft-CSOVO	FT	CSFT	soft-CSFT
iris	**18.34±4.48**	**17.21±3.84**	**18.79±3.72**	21.93±4.99	20.74±4.32	19.89±4.24	23.80±5.21	19.54±4.67	**15.94±3.26***
wine	**12.98±3.37**	**13.42±2.55**	**14.34±2.76**	15.04±4.05	**11.45±3.53***	**12.95±4.15**	15.21±3.49	**11.87±3.09**	**13.66±4.14**
glass	159.19±10.37	**126.84±9.71***	**129.42±9.51**	145.90±10.36	**128.56±9.77**	**132.69±9.62**	151.06±10.20	143.78±8.66	143.22±9.85
vehicle	114.14±9.08	**95.33±10.29***	**97.81±10.85**	112.31±8.82	**103.63±11.17**	**97.34±11.16**	112.48±7.71	**105.58±10.90**	106.74±11.27
vowel	**6.76±0.93**	11.72±1.44	**6.43±1.11**	**6.29±0.94***	9.58±1.08	**6.82±0.90**	9.53±1.31	13.71±1.58	11.87±1.47
segment	**14.02±1.17**	**13.84±0.94**	**13.03±1.08***	14.15±1.18	**14.00±1.11**	**14.10±1.31**	15.01±1.33	14.17±1.15	15.36±1.26
dna	24.43±1.26	24.40±1.55	**22.76±1.47***	24.51±1.37	28.26±2.04	24.51±1.52	27.94±2.34	31.49±2.09	29.23±2.28
satimage	40.20±2.08	**35.04±2.16**	**34.86±2.11***	40.43±1.92	**36.49±2.27**	**36.46±2.31**	41.98±2.08	40.16±2.10	39.63±2.23
usps	6.87±0.28	7.32±0.23	**6.58±0.27***	7.08±0.27	7.20±0.26	6.98±0.25	9.05±0.29	8.97±0.40	8.59±0.27
zoo	8.59±1.81	10.14±1.29	7.22±1.16	9.35±1.87	**5.91±1.15**	**6.56±1.37**	8.68±1.77	**6.56±1.27**	8.70±1.71
yeast	36.66±3.37	**0.58±0.07**	**0.58±0.07**	39.71±3.62	**0.55±0.08***	**0.55±0.08**	38.97±3.88	**0.62±0.09**	0.64±0.09
pageblock	2.80±0.48	0.18±0.04	0.19±0.04	2.59±0.45	**0.16±0.03**	**0.16±0.03**	2.78±0.48	**0.16±0.03**	**0.16±0.03***
anneal	0.85±0.23	**0.35±0.12***	**0.38±0.13**	0.83±0.23	0.61±0.16	0.67±0.17	0.85±0.23	0.58±0.16	0.65±0.16
solar	46.08±6.53	25.35±4.06	25.32±4.05	44.51±6.31	**18.04±1.94**	**17.89±1.95***	47.18±7.14	20.54±2.64	20.43±2.06
splice	14.01±0.84	**12.59±1.11***	**12.85±0.71**	13.97±0.76	17.06±1.26	**13.28±0.88**	16.64±0.79	18.19±1.62	16.06±1.17
ecoli	17.11±2.85	1.27±0.31	**0.92±0.18**	19.93±2.61	1.35±0.49	1.11±0.41	20.43±4.49	**0.85±0.14***	1.96±1.13
nursery	0.62±0.20	**0.00±0.00**	**0.00±0.00***	0.07±0.06	0.00±0.00	0.00±0.00	1.42±0.45	0.00±0.00	0.39±0.34
soybean	9.84±1.60	2.78±0.36	2.99±0.43	11.41±1.85	**2.13±0.29**	**2.08±0.30***	9.61±1.57	3.07±0.52	3.97±0.55
arrhythmia	6.46±1.23	0.55±0.08	0.63±0.08	7.32±1.48	**0.36±0.05***	**0.37±0.05**	8.69±1.78	0.57±0.19	0.55±0.17
optdigits	5.33±0.34	5.64±0.26	**4.90±0.35***	**4.98±0.26**	6.12±0.32	**5.23±0.31**	6.23±0.34	7.67±0.43	6.57±0.35
mfeat	**7.99±0.55**	9.27±0.74	**7.56±0.55***	8.74±0.59	8.36±0.61	8.70±0.64	11.74±0.76	11.23±0.89	10.87±0.83
pendigit	1.99±0.11	2.46±0.12	**1.88±0.09***	**1.88±0.10**	**1.95±0.08**	**1.95±0.08**	2.12±0.11	2.36±0.11	2.43±0.19
# bold	5	10	17	3	12	15	0	6	3

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

Table 2: average test error (%) on benchmark data sets

data set	OVA	OSR	soft-OSR	OVO	CSOVO	soft-CSOVO	FT	CSFT	soft-CSFT
iris	**4.21±0.78***	6.71±0.98	**4.74±0.73**	**4.74±0.80**	10.66±2.32	5.26±0.70	**4.61±0.79**	7.11±1.24	**4.47±0.77**
wine	**1.78±0.43**	4.00±0.62	2.44±0.38	**2.11±0.51**	**1.78±0.51**	**1.67±0.52***	2.22±0.47	**1.67±0.44**	**2.00±0.49**
glass	**28.52±0.82***	32.22±1.11	31.94±1.21	**28.89±0.84**	44.26±2.73	45.28±2.52	29.81±0.96	39.17±2.35	36.02±2.52
vehicle	**20.66±0.62**	24.15±0.83	22.78±0.73	**20.31±0.67***	28.73±2.19	25.14±1.57	**20.75±0.64**	29.88±2.92	30.40±3.04
vowel	**1.27±0.17***	5.38±0.47	1.88±0.27	**1.29±0.18**	5.93±0.63	**1.43±0.17**	1.94±0.24	6.25±1.43	2.74±0.39
segment	**2.60±0.16***	3.69±0.27	**2.76±0.15**	**2.60±0.15**	5.57±0.95	4.11±0.59	2.78±0.15	4.30±0.62	3.43±0.35
dna	**4.20±0.14**	6.96±0.65	4.87±0.27	**4.19±0.13***	7.90±0.80	5.81±0.85	4.81±0.24	9.14±1.52	5.32±0.30
satimage	**7.19±0.10***	9.52±0.30	9.01±0.34	**7.24±0.09**	12.55±0.66	12.51±0.68	7.55±0.11	10.58±0.63	9.85±0.75
usps	**2.19±0.07***	3.82±0.13	2.66±0.11	2.28±0.06	5.27±0.70	3.53±0.17	2.79±0.06	6.26±0.86	3.50±0.10
zoo	**5.19±0.83**	15.38±1.61	13.08±1.52	6.15±1.03	10.77±1.71	8.27±1.77	**4.81±0.81***	12.69±2.54	6.35±1.47
yeast	40.38±0.64	73.76±0.55	73.68±0.55	**39.27±0.56***	76.58±0.68	76.70±0.67	40.20±0.52	77.02±0.92	76.70±0.81
pageblock	3.22±0.09	39.25±4.36	38.54±4.74	**3.06±0.08***	76.75±6.18	76.75±6.18	**3.10±0.10**	78.25±6.10	81.82±5.81
anneal	**1.40±0.15***	8.78±0.94	6.98±1.13	**1.51±0.15**	19.02±4.24	10.60±4.53	**1.47±0.17**	11.31±1.94	9.47±4.40
solar	27.27±0.42	34.83±1.16	35.22±1.75	**26.61±0.43***	47.49±3.30	47.83±3.12	27.27±0.46	46.15±3.12	43.48±2.85
splice	**3.86±0.15***	7.68±1.16	5.21±0.56	**3.92±0.12**	13.34±2.69	8.13±2.60	4.62±0.18	9.59±1.46	6.52±0.74
ecoli	15.12±0.99	32.68±1.67	33.63±1.61	**14.05±0.75***	37.80±3.30	38.45±3.19	16.85±1.14	36.73±2.72	40.89±3.85
nursery	0.11±0.02	33.33±0.17	31.02±1.54	**0.02±0.01***	37.62±2.17	3.31±2.21	0.32±0.08	33.89±0.44	20.04±3.61
soybean	**6.55±0.32***	24.53±0.82	21.67±1.42	7.46±0.34	39.06±3.51	40.12±3.76	7.13±0.38	35.41±2.48	28.48±3.40
arrhythmia	**28.41±0.93**	66.37±2.25	66.42±2.11	**27.92±0.74***	85.18±2.49	83.05±3.37	30.40±0.62	88.81±2.47	86.15±3.12
optdigits	1.09±0.06	1.85±0.06	1.15±0.07	**1.04±0.05***	2.25±0.09	1.36±0.12	1.35±0.05	2.14±0.24	1.55±0.05
mfeat	**1.69±0.09***	3.10±0.18	1.84±0.11	1.86±0.08	4.32±0.53	2.50±0.22	2.45±0.10	3.89±0.37	2.99±0.38
pendigit	**0.40±0.02**	0.85±0.04	**0.39±0.02**	**0.38±0.02***	0.65±0.03	0.42±0.02	0.45±0.02	0.62±0.04	0.52±0.03

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

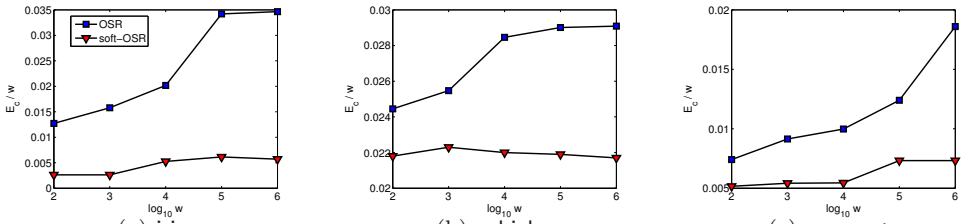
version of this paper. The figures plot the scaled test cost E_c/w on different values of $\log_{10} w$. From the three figures, we see that soft-OSR is better than OSR across all w. When the emphasis is very high (like 10^6), OSR can be conservative and "paranoid." It avoids classifying any of the test examples as the emphasized class, which results in the worse performance. On the other hand, the curves of soft-OSR remain mostly flat, which demonstrate that soft cost-sensitive classification is less sensitive (paranoid) to large cost components. The results again justify the superiority of soft-OSR, a promising representative of soft cost-sensitive classification, over its hard sibling.

4.3 Comparison on a Real-world Biomedical Task

To test the validity of our proposed soft cost-sensitive classification methodology on true applications, we use two real-world data sets for our experiments. The first one is a biomedical task [24], and the other one to be introduced later is from KDDCup 1999 [3]. Both data sets go through similar splitting and scaling procedures, as we did for the benchmark data sets.

The biomedical task is on classifying the bacterial meningitis, which is a serious and often life-threatening form of the meningitis infection. The inputs are the spectra of bacterial pathogens extracted by the Surface Enhanced Raman Scattering (SERS) platform [7]. In this paper, we call the task SERS, which contains 79 clinical samples of ten meningitis-causing bacteria species collected in the National Taiwan University Hospital and 17 standard bacteria samples from American Type Culture Collection. The cost matrix of SERS is shown in Table 5, which is specified by two human physicians who are specialized in infectious diseases.

The results are shown in Table 6. Among the nine algorithms, soft-CSOVO gets the lowest cost. If we compare the other eight algorithms with soft-CSOVO using a pairwise one-tailed t-test of significance level 0.1, we see that soft-CSOVO is significantly better than all other algorithms. The results confirm the usefulness of soft cost-sensitive classification for this real-world task.



(a) iris (b) vehicle (c) segment

Figure 4: test E_c/w of OSR and soft-OSR with the emphasizing cost for different emphasis parameter w

Table 3: comparisons on the test costs between the algorithms and their soft cost-sensitive classification sibling using a pairwise one-tailed t-test of significance level 0.1

data set	OVA	OSR	OVO	CSOVO	FT	CSFT
iris	≈	≈	≈	≈	◯	≈
wine	≈	≈	≈	≈	≈	≈
glass	◯	≈	≈	≈	≈	≈
vehicle	◯	≈	◯	◯	≈	≈
vowel	≈	◯	≈	◯	≈	◯
segment	◯	◯	≈	≈	≈	≈
dna	◯	◯	≈	◯	≈	◯
satimage	◯	≈	◯	≈	◯	≈
usps	≈	◯	≈	≈	◯	≈
zoo	≈	◯	◯	≈	≈	≈
yeast	◯	≈	◯	≈	◯	≈
pagblock	◯	≈	◯	≈	◯	≈
anneal	◯	≈	≈	≈	≈	≈
solar	◯	≈	◯	≈	◯	≈
splice	◯	≈	≈	◯	≈	◯
ecoli	◯	≈	◯	◯	◯	≈
nursery	◯	≈	≈	≈	≈	≈
soybean	◯	≈	◯	≈	◯	≈
arrhythmia	◯	≈	◯	≈	◯	≈
optdigits	◯	◯	≈	◯	≈	◯
mfeat	≈	◯	≈	≈	≈	≈
pendigit	≈	◯	≈	≈	×	≈

◯: soft cost-sensitive algorithms significantly better
×: soft cost-sensitive algorithms significantly worse
≈: otherwise

Table 4: comparison on the test errors between the hard cost-sensitive classification algorithms and their soft sibling using a pairwise one-tailed t-test of significance level 0.1

data set	OSR	CSOVO	CSFT
iris	◯	◯	◯
wine	◯	≈	≈
glass	◯	≈	≈
vehicle	◯	◯	≈
vowel	◯	◯	◯
segment	◯	◯	◯
dna	◯	◯	◯
satimage	◯	◯	◯
usps	◯	◯	◯
zoo	◯	◯	◯
yeast	≈	≈	≈
pagblock	≈	≈	≈
anneal	≈	◯	◯
solar	≈	≈	≈
splice	◯	≈	◯
ecoli	◯	≈	≈
nursery	≈	◯	◯
soybean	◯	≈	◯
arrhythmia	≈	≈	≈
optdigits	◯	◯	◯
mfeat	◯	◯	◯
pendigit	◯	◯	◯

◯: soft cost-sensitive algorithms significantly better
×: soft cost-sensitive algorithms significantly worse
≈: otherwise

SERS is an interesting data set in which regular classification algorithms like OVO or FT can perform better than their hard cost-sensitive classification siblings like CSOVO or CSFT. Given the small number of examples in SERS, the phenomenon can be attributed to overfitting with respect to the cost—i.e. over-using the cost information. Soft cost-sensitive classification provides a balanced alternative between over-using (hard) or not using (regular) the cost. The balancing can lead to significantly lower test cost, as demonstrated by the promising performance of soft-CSOVO on this biomedical task.

4.4 Comparison on the KDD Cup 1999 Task

The KDDCup 1999 data set (kdd99) is another real-world cost-sensitive classification task [3]. The task contains an intrusion detection problem for distinguishing the "good" and "bad" connections. Following the usual procedure in literature [1], we extract a random 40% of the 10%-training set for our experiments. The test set accompanied is not used because of the known mismatch between training and test distributions [1]. We take the given cost matrix in the competition for our experiments.[4]

The results are listed in Table 7. While the cost-sensitive classification algorithm OSR achieves the lowest test cost, other algorithms (soft, hard, or regular) all result in similar performance. The reason of the similar performance is because all the algorithms are of error rate less than 1% and are thus of low costs. That is, the data set is easy to classify, and there is almost no room for improvements. The easiness is partly because the data set is highly imbalanced. In particular, the size of the majority class is over 8000 times more than the size of the minority class.

To further compare the performance of the algorithms, we consider a more challenging version of the real-world task. The version is called kdd99-balanced, which is generated by

[4] `http://www.kdd.org/kddcup/site/1999/files/awkscript.htm`

Table 5: cost matrix on SERS

real class \ classify to	Ab	Ecoli	HI	KP	LM	Nm	Psa	Spn	Sa	GBS
Ab	0	1	10	7	9	9	5	8	9	1
Ecoli	3	0	10	8	10	10	5	10	10	2
HI	10	10	0	3	2	2	10	1	2	10
KP	7	7	3	0	4	4	6	3	3	8
LM	8	8	2	4	0	5	8	2	1	8
Nm	3	10	9	8	6	0	8	3	6	7
Psa	7	8	10	9	9	7	0	8	9	5
Spn	6	10	7	7	4	4	9	0	4	7
Sa	7	10	6	5	1	3	9	2	0	7
Gbs	2	5	10	9	8	6	5	6	8	0

Table 6: experiment results on SERS, with t-test for cost

	error (%)	cost ($\cdot 10^0$)	t-test
OVA	23.0 ± 2.51	1.056 ± 0.097	◯
OSR	27.6 ± 2.27	0.986 ± 0.092	◯
soft-OSR	25.8 ± 2.80	1.024 ± 0.095	◯
OVO	23.2 ± 2.55	0.970 ± 0.106	◯
CSOVO	27.4 ± 1.53	1.150 ± 0.109	◯
soft-CSOVO	26.6 ± 2.55	0.906 ± 0.069	$*$
FT	23.0 ± 2.51	0.986 ± 0.092	◯
CSFT	27.6 ± 1.40	1.118 ± 0.090	◯
soft-CSFT	31.4 ± 4.09	1.054 ± 0.040	◯

$*$: best entry of cost
◯: best entry significantly better in cost
$\approx$: otherwise

scaling down the y-th row of the cost matrix by the size of the y-th class. The results on kdd99-balanced are shown in Table 8. OSR remains to be the best algorithm, with comparable test cost to soft-OSR. Nevertheless, when comparing the errors of OSR and soft-OSR, we see that soft-OSR can reach lower test error. Similar results hold (even more significantly) between CSOVO and soft-CSOVO, and between CSFT and soft-CSFT. The results again demonstrate the usefulness of soft cost-sensitive classification in reaching low cost *and* low error on this real-world task.

5. CONCLUSIONS

We have explored the trade-off between the cost and the error rate in cost-sensitive classification tasks, and have identified the practical needs to reach both low cost and low error rate. Based on the trade-off, we have proposed a simple and novel methodology between traditional regular classification and modern cost-sensitive classification. The proposed methodology, soft cost-sensitive classification, takes both the cost and the error into account by a multicriteria optimization problem. By using the weighted sum approach to solving the optimization problem, the proposed methodology allows immediate improvements of existing cost-sensitive classification algorithms in terms of similar or sometimes lower costs, and of lower errors. The significant improvements have been observed on a broad range of benchmark and real-world tasks in our extensive experimental study.

An immediate future work is to take more state-of-art algorithms for comparison. Furthermore, instead of treating the cost and error symmetrically in the methodology, an interesting future research direction is to consider them in an asymmetric way that treats the cost as the major criterion and the error as the minor one.

Our work reveals a new insight for cost-sensitive classification in machine learning and data mining: Feeding in the exact cost information for the machines to learn may not be the best approach, much like how fitting the provided data faithfully without regularization may lead to overfitting. Our work takes the error rates to "regularize" the cost information and leads to better performance. Another interesting direction for future research is to consider other types of regularization on the cost information.

Table 7: average test results on kdd99, with t-test for cost

	error (%)	cost ($\cdot 10^{-3}$)	t-test
OVA	0.11 ± 0.003	1.84 ± 0.179	$\approx$
OSR	0.11 ± 0.003	1.80 ± 0.171	$*$
soft-OSR	0.11 ± 0.003	1.92 ± 0.178	$\approx$
OVO	0.10 ± 0.002	1.85 ± 0.174	$\approx$
CSOVO	0.11 ± 0.003	1.81 ± 0.169	$\approx$
soft-CSOVO	0.11 ± 0.003	1.82 ± 0.169	$\approx$
FT	0.10 ± 0.002	1.84 ± 0.170	$\approx$
CSFT	0.11 ± 0.003	1.83 ± 0.171	$\approx$
soft-CSFT	0.11 ± 0.003	1.83 ± 0.171	$\approx$

$*$: best entry of cost
◯: best entry significantly better in cost
$\approx$: otherwise

Table 8: average test results on kdd99-balanced, with t-test for cost

	error (%)	cost ($\cdot 10^{-6}$)	t-test
OVA	0.11 ± 0.00	2.35 ± 0.167	◯
OSR	2.96 ± 0.63	1.80 ± 0.157	$*$
soft-OSR	2.51 ± 0.53	1.85 ± 0.160	$\approx$
OVO	0.10 ± 0.00	2.49 ± 0.176	◯
CSOVO	3.12 ± 0.64	1.81 ± 0.128	$\approx$
soft-CSOVO	2.28 ± 0.40	1.82 ± 0.140	$\approx$
FT	0.10 ± 0.00	2.46 ± 0.178	◯
CSFT	2.70 ± 0.58	1.90 ± 0.148	$\approx$
soft-CSFT	1.46 ± 0.46	2.11 ± 0.183	$\approx$

$*$: best entry of cost
◯: best entry significantly better in cost
$\approx$: otherwise

6. ACKNOWLEDGMENTS

The authors thank Yao-Nan Chen, Ku-Chun Chou, Chih-Han Yu and the anonymous reviewers for valuable comments. This work was supported by the National Science Council (NSC 100-2628-E-002-010 and NSC 100-2120-M-001-003-CC1) of Taiwan.

7. REFERENCES

[1] N. Abe, B. Zadrozny, and J. Langford. An iterative method for multi-class cost-sensitive learning. In *Proc. SIGKDD*, pages 3–11, 2004.

[2] Y. S. Abu-Mostafa, M. Magdon-Ismail, and H.-T. Lin. *Learning from Data: A Short Course*. AMLBook, 2012.

[3] S. D. Bay. UCI KDD archive. Department of Information and Computer Sciences, University of California, Irvine, 2000. Downloaded from `http://kdd.ics.uci.edu/`.

[4] A. Bernstein, F. Provost, and S. Hill. Toward intelligent assistance for a data mining process: An ontology-based approach for cost-sensitive classification. *IEEE TKDE*, 17(4):503–518, 2005.

[5] A. Beygelzimer, V. Daniand, T. Hayes, J. Langford, and B. Zadrozny. Error limiting reductions between classification tasks. In *Proc. ICML*, pages 49–56, 2005.

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

[7] A. Campion and P. Kambhampati. Surface enhanced Raman scattering. *Chem. Soc. Rev.*, 27(4):241–250, 1998.

[8] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM TIST*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[9] D. Corne, N. Jerram, J. Knowles, and M. Oates. PESA-II: Region-based selection in evolutionary multiobjective optimization. In *Proc. GECCO*, 2001.

[10] I. Das and J. E. Dennis. A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems. *Struct. Multidiscip. Opti.*, 14(1):63–69, 1996.

[11] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE TEC*, 6(2):182–197, 2002.

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

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

[14] W. Fan, W. Lee, S. Stolfo, and M. Miller. A multiple model cost-sensitive approach for intrusion detection. In *Proc. ECML*, pages 142–154, 2000.

[15] A. Freitas. Building cost-sensitive decision trees for medical applications. *AI Comm.*, 24(3):285–287, 2011.

[16] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. Witten. The WEKA data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18, 2009.

[17] J. Han, M. Kamber, and J. Pei. *Data mining: concepts and techniques*. Morgan Kaufmann, 2011.

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

[19] C. Hillermeier. *Nonlinear multiobjective optimization*. Birkhauser, 2001.

[20] J. Horn, N. Nafpliotis, and D. Goldberg. A niched Pareto genetic algorithm for multiobjective optimization. In *Proc. IEEE WCCI*, pages 82–87, 1994.

[21] C.-W. Hsu and C.-J. Lin. A comparison of methods for multi-class support vector machines. *IEEE TNN*, 13(2):415–425, 2002.

[22] J. J. Hull. A database for handwritten text recognition research. *IEEE TPAMI*, 16(5):550–554, 1994.

[23] T.-K. Jan. A comparison of methods for cost-sensitive support vector machines. Master's thesis, National Taiwan University, 2010.

[24] T.-K. Jan, H.-T. Lin, H.-P. Chen, T.-C. Chern, C.-Y. Huang, C.-Y. Huang, C.-W. Chung, Y.-J. Li, Y.-C. Chuang, L.-L. Li, Y.-J. Chan, J.-K. Wang, Y.-L. Wang, C.-H. Lin, and D.-W. Wang. Cost-sensitive classification on pathogen species of bacterial meningitis by surface enhanced Raman scattering. In *Proc. IEEE BIBM*, pages 406–409, 2011.

[25] J. Langford and A. Beygelzimer. Sensitive error correcting output codes. In *Proc. COLT*, pages 158–172, 2005.

[26] W. Lee, W. Fan, M. Miller, S. Stolfo, and E. Zadok. Toward cost-sensitive modeling for intrusion detection and response. *JCS*, 10(1/2):5–22, 2002.

[27] H.-T. Lin. A simple cost-sensitive multiclass classification algorithm using one-versus-one comparisons. Downloaded from `http://www.csie.ntu.edu.tw/~htlin/paper/doc/csovo.pdf`, 2010.

[28] H.-T. Lin and L. Li. Support vector machinery for infinite ensemble learning. *JMLR*, 9(2):285–312, 2008.

[29] D. D. Margineantu. *Methods for cost-sensitive learning*. PhD thesis, Oregon State University, 2001.

[30] S. Rosset. Value weighted analysis: Building prediction models for data with observation. Downloaded from `http://www.tau.ac.il/~saharon/`, 2002.

[31] K. Schleifer. Classification of bacteria and archaea: past, present and future. *Syst. Appl. Microbiol.*, 32(8):533–542, 2009.

[32] D. Sculley. Combined regression and ranking. In *Proc. SIGKDD*, pages 979–988, 2010.

[33] Y. Sun, M. Kamel, A. Wong, and Y. Wang. Cost-sensitive boosting for classification of imbalanced data. *PR*, 40(12):3358–3378, 2007.

[34] M. Tan. Cost-sensitive learning of classification knowledge and its applications in robotics. *ML*, 13(1):7–33, 1993.

[35] H.-H. Tu. Regression approaches for multi-class cost-sensitive classification. Master's thesis, National Taiwan University, 2009.

[36] H.-H. Tu and H.-T. Lin. One-sided support vector regression for multiclass cost-sensitive classification. In *Proc. ICML*, pages 1095–1102, 2010.

[37] V. N. Vapnik. *Statistical Learning Theory*. Wiley, 1998.

[38] L. Zadeh. Optimality and non-scalar-valued performance criteria. *IEEE TAC*, 8(1):59–60, 1963.

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

[40] Z.-H. Zhou and X.-Y. Liu. On multi-class cost-sensitive learning. In *Proc. AAAI*, pages 567–572, 2006.