An Iterative Method for Multi-class Cost-sensitive Learning

Naoki Abe
Mathematical Sciences Dept.
IBM T. J. Watson Res. Ctr.
Yorktown Heights, NY 10598
nabe@us.ibm.com

Bianca Zadrozny
Mathematical Sciences Dept.
IBM T. J. Watson Res. Ctr.
Yorktown Heights, NY 10598
zadrozny@us.ibm.com

John Langford
Toyota Technological Institute
at Chicago
427 East 60th Street, 2nd Flr.
Chicago, IL 60637
jl@tti-c.org

ABSTRACT
Cost-sensitive learning addresses the issue of classification in the presence of varying costs associated with different types of misclassification. In this paper, we present a method for solving multi-class cost-senstive learning problems using any binary classification algorithm. This algorithm is based on a number of techniques: iterative weighting, data space expansion, and gradient boosting with stochastic ensembles. We establish some theoretical performance guarantees concerning the convergence and optimality of a variant of the method and present empirical evidence of effectiveness for real-world problems in terms of predictive performance (cost minimization).

Keywords
cost-sensitive learning, multi-class classification, boosting, sampling, memory light learning

1. INTRODUCTION
Classification in the presence of varying costs associated with different types of misclassification is important for practical applications, including many data mining applications, such as targeted marketing, fraud and intrusion detection among others. A body of work on this subject has become known as *cost-sensitive learning*, in the areas of machine learning and data mining.

Research in cost-sensitive learning falls into three main categories. The first category is concerned with making particular classifier learners cost-sensitive, including methods specific for decision trees [12, 3], neural networks [11] and support vector machines [10]. The second category uses Bayes risk theory to assign each example to its lowest expected cost class [7, 16].This requires classifiers to output class membership probabilities and sometimes requires estimating costs [16] (when the costs are unknown at classification time). The third category concerns methods that modify the distribution of training examples before applying the classifier learning method, so that the classifier learned from the modified distribution is cost-sensitive. We call this approach *cost-sensitive learning by example weighting*. Work in this area includes stratification methods [6, 5] and the *costing* algorithm [17]. This approach is very general since it reuses arbitrary classifier learners and does not require accurate class probability estimates from the classifier. Empirically this approach attains similar or better cost-minimization performance.

Unfortunately, current methods in this category suffer from a major limitation: they are well-understood only for two-class problems. In the two-class case, it is easy to show that each example should be weighted proportionally to the difference in cost between predicting correctly or incorrectly [17]. However, in the multi-class case there is more than one way in which a classifier can make a mistake, breaking the application of this simple formula. Heuristics, such as weighting examples by the average misclassification cost, have been proposed [5, 13], but they are not well-motivated theoretically and do not seem to work very well in practice when compared to methods that use Bayes risk minimization [7].

In this paper, we propose a method for multi-class cost-sensitive learning based on iterative example weighting schemes. There are a number of key techniques we employ in this method: (1) an iterative process which empirically adjusts the example weighting according to the performance of learning; (2) data space expansion for multi-class labels; and (3) gradient boosting [14] with stochastic ensembles. We establish theoretical performance guarantee for a variant of this algorithm, given that the component classification algorithm satisfies a certain *weak learning assumption*. We test this and related algorithms on several datasets and discover good performance in terms of predictive performance (cost minimization).

2. PRELIMINARIES
We begin by introducing some general concepts and notation we use in the rest of the paper.

2.1 Cost-sensitive learning and related problems
A popular formulation of the cost-sensitive learning problem is via the use of a cost matrix. A cost matrix, $C(y_1, y_2)$, specifies how much cost is incurred when misclassifying an

example labelled y_2 as y_1, and the goal of a cost-sensitive learning method is to minimize the expected cost. Zadrozny and Elkan [16] noted that this formulation is not applicable in situations in which misclassification costs depend on particular instances, and proposed a more general form of cost function, $C(x, y_1, y_2)$, that allows dependence on the instance x. Here we adopt a slightly more general formulation.

Once we allow the costs to depend on each example, it is natural to assume that the costs are generated according to some distribution, along with the examples, which leads to the following formulation. In (multiclass) cost sensitive classification, examples of the form $(x, \vec{C})$ are drawn from a distribution D over a domain $X \times {R^+}^k$. (Throughout the paper, we will let k denote $|Y|$.) Given a set of examples, $S = (x_i, \vec{C})^m$, the goal is to find a classifier $h : X \to \{1, ..., k\}$ which minimizes the expected cost of the classifier:

$$\arg\min_h \mathop{\mathrm{E}}_{(x,\vec{C})\sim D}[C_{h(x)}] \quad (1)$$

We can assume without loss of generality that the costs are normalized so that

$$\forall x \in X \ \min_{y\in Y} C_y = 0.$$

Note that with this normalization, the above formulation of cost is equivalent to the common formulation in terms of misclassification cost, i.e.,

$$\min_h \mathop{\mathrm{E}}_{(x,\vec{C})\sim D}[C_h(x) I(h(x) \neq \arg\min_y C_y]$$

Normally a learning method attempts to do this by minimizing the empirical cost in the given training data, given some hypothesis class $\mathcal{H}$:

$$\arg\min_{h\in\mathcal{H}} \mathop{\hat{\mathrm{E}}}_{(x,\vec{C})\sim S}[C_{h(x)}] = \frac{1}{|S|} \sum_{(x,\vec{C})\in S} C_{h(x)} \quad (2)$$

Here the empirical expectation notation, $\widehat{E}$ refers to the averaged empirical cost.

As a building block of our method, we make use of methods for solving importance weighted classification problems, which we define below. In importance weighted classification, examples of the form (x, y, c) are drawn from a distribution D over a domain $X \times Y \times R^+$. Given a set of examples $S = (x, y, c)^m$, the goal is to find a classifer $h : X \to Y$ having minimum importance-weighted misclassificaton error:

$$\arg\min_h \mathop{\mathrm{E}}_{(x,y,c)\sim D} [c \cdot I(h(x) \neq y)]$$

Again, usually, a learning method attempts to meet this goal by minimizing the empirical weighted error in some hypothesis class $\mathcal{H}$:

$$\arg\min_{h\in\mathcal{H}} \mathop{\hat{\mathrm{E}}}_{(x,y,c)\sim S} [c \cdot I(h(x) \neq y)] \quad (3)$$

We note that importance weighted classification can be solved very well with a classification method, by use of weighted rejection sampling techniques [17].

2.2 Hypothesis representations and other notation

In the above, we assumed that the hypotheses output by a cost-sensitive learner is a functional hypothesis h, i.e. $h : X \to Y$. It is also possible to allow hypotheses that are *stochastic*, namely

$$h : Y \times X \to [0, 1]$$

subject to the stochastic condition:

$$\forall x \in X \sum_{y\in Y} h(y|x) = 1.$$

With stochastic hypotheses, *stochastic* cost-sensitive learning is defined as that of minimizing the following expected cost:

$$\arg\min_h \mathop{\mathrm{E}}_{(x,\vec{C})\sim D} \mathop{\mathrm{E}}_{y\sim h(y|x)} C_y$$

Note that in the special case that h is deterministic, this formulation is equivalent to the definition given in Eq. 1. Also, this is a *convexification* of the standard objective function that we usually expect a stochastic cost-sensitive learner to minimize, i.e.

$$\mathop{\mathrm{E}}_{(x,\vec{C})\sim D}[C_{h(x)}]$$

We also consider a variant of cost-sensitive learning in which *relational* hypotheses are allowed. Here relational hypotheses h are relations over $X \times Y$, i.e. $h : X \times Y \to \{0, 1\}$. In general h is neither functional nor stochastic, and in particular it may violate the stochasticity: $\sum_{y\in Y} h(x, y) = 1$.

3. THE METHODOLOGY

Our methodology can be interpreted as a *reduction*, which translates a multi-class cost-sensitive learning problem to a classifier learning problem.

This methodology is derived using three key ideas: 1) iterative weighting; 2) expanding data space; and 3) gradient boosting with stochastic ensembles. The first two ideas are combined in a unified framework given by the third.

Below we will explain the first two key ideas by exhibiting a prototypical method based on each, and then derive our main learning method based on these ideas.

3.1 Iterative cost weighting

We note that the weighting scheme proposed in [17], called *costing*, exploits the following observation: For the binary class case, the above formulation in terms of per example costs for each class can be further reduced to a formulation in terms of a single importance number per example. This is possible by associating a number indicating the importance of an example (x, y), given by $|C_0 - C_1|$. This conversion allows us to reduce the cost-sensitive learning problem to a weighted classifier learning problem, but it is not immediately obvious how that would be done for the multi-class scenario. It is therefore natural to consider iterative weighting schemes, in which example weights are iteratively modified in search for the optimal weighting. The technique, which we call *IW (Iterative Weighting)*, presented in Figure 1, is an example of such a scheme. We can show that the final weights of IW are the optimal weights in some sense, *provided* the algorithm converges.

- An input sample $S = (x, \vec{C})^m$.
- A component learner A that takes an importance-weighted sample $S' = (x, y, c)^m$ and outputs a functional hypothesis h, possibly by weighted sampling.
- An integer T specifying the number of iterations to be performed.

1. Let S' =
$$\{(x, \arg\min_y C_y, \max_y C_y) | (x, \vec{C}) \in S\}$$
2. **For** $t := 1$ to T **do**
   (a) Let $h_t := A(S')$.
   (b) Choose $\alpha_t \in [0, 1)$, for example $\alpha_t := 1/t$.
   (c) For each example $(x, y, c) \in S'$ with an error $h_t(x) \neq y$ update the importance weight by
   $$c := \alpha_t C_{h(x)} + (1 - \alpha_t)c$$
3. Return h_T.

Figure 1: Method *IW (Iterative Weighting)*

THEOREM 1. *Assume IW converges to hypothesis h. Then, the following holds.*

$$\widehat{E}_{(x,y,c)\sim S'} c \cdot I(h(x) \neq y) = \widehat{E}_{(x,\vec{C})\sim S} C_{h(x)}$$

This theorem says that, if the iterative algorithm converges, then the loss of the component learner with respect to the final weights (left hand side) is equal to the expected cost for the problem that we wish to solve (right hand side).

Proof

At convergence, for every example (x, y, c) that the h errs on, we must have:

$$c = \alpha_t C_{h(x)} + (1 - \alpha_t)c$$

and thus

$$c = C_{h(x)}$$

Noting that when h does not err on x, $I(h(x) \neq y) = 0$, it follows that

$$\widehat{E}_{(x,y,c)\sim S'}[c \cdot I(h(x) \neq y)] = \widehat{E}_{(x,\vec{C})\sim S}[C_{h_t(x)}]$$

Q.E.D

3.2 Data space expansion

One drawback to iterative weighting is an inability to directly take into account the different costs associated with multiple ways of misclassifying examples. This translates to non-convergence of the method in practice. We address this issue by the technique of expanding data space, as we describe below.

The objective of minimizing the empirical cost on the original training sample is equivalent to minimization on the following *expanded* sample. Given a labeled sample S consisting of $(x, \vec{C})$ of size m , we define a sample S' of size mk for classification, where k is the size of the label set, i.e. $k = |Y|$, as follows.

$$S' = \{(x, y, \max_{y'} C_{y'} - C_y) | (x, \vec{C}) \in S, y \in Y\}$$

Minimizing the importance weighted loss,

$$\hat{\mathrm{E}}_{(x,y,c)\in S'} c \cdot I(h(x) \neq y)$$

on this new dataset also minimizes the cost on our original sample. The algorithm DSE (Data Space Expansion), shown in Figure 2, takes advantage of this observation.

- An input sample $S = (x, \vec{C})^m$.
- A component learner A that takes an importance-weighted sample $S' = (x, y, c)^m$ and outputs a functional hypothesis h, possibly by weighted sampling.

1. Set $S' = \{(x, y, \max_{y'} C_{y'} - C_y) | (x, \vec{C}) \in S, y \in Y\}$
2. Return $h := A(S')$

Figure 2: Method *DSE (Data Space Expansion)*

3.3 Gradient Boosting with Stochastic Ensembles

Having described two key ideas, namely iterative weighting and data space expansion, we now apply them together to arrive at our main method. We do so by casting the *stochastic* multiclass cost-sensitive learning in the framework of gradient boosting [14], with the objective function defined as the expected cost of the *stochastic ensemble*, obtained as a mixture of individual hypotheses, on the *expanded* data set. As we stated in Section 2, a functional hypothesis of the form $h : X \to Y$ can be viewed as a special case of a stochastic hypothesis. We then define a stochastic ensemble hypothesis H, given multiple functional hypotheses, $h_t, t = 1, ..., T$, as the conditional distribution defined as the mixture of the component hypotheses, namely,

$$\forall x \in X, \forall y \in Y, H(y|x) = \frac{1}{T} \sum_{t=1}^{T} I(h_t(x) = y)$$

Let H_t denote the mixture hypothesis of the learning procedure at round t. The procedure is to update its current combined hypothesis by the mixture of the previous combined hypothesis and a new hypothesis, i.e. by setting

$$H_t(y|x) = (1 - \beta)H_{t-1}(y|x) + \beta I(h(x) = y)$$

Thus, the expected cost of H_t on x is

$$\mathop{\mathrm{E}}_{y\sim H_t(y|x)} C_y = (1 - \beta) \mathop{\mathrm{E}}_{y\sim H_{t-1}(y|x)} C_y + \beta C_{h_t(x)}$$

If we now take a derivative of this function with respect to β, we get:

$$\frac{\partial \, \mathrm{E}_{y\sim H_t(y|x)} C_y}{\partial \beta} = C_{h_t(x)} - \mathop{\mathrm{E}}_{y\sim H_{t-1}(y|x)} C_y$$

Note that this is the difference between the average cost of the current *ensemble* hypothesis and the new weak hypothesis assigning probability one to the specified label.

We then take the expectation of this derivative with respect to all data points (x, y) in the expanded data set S', and thus the gradient is mk-dimensional. The weak learner finds a hypothesis h whose inner-product with the negative gradient is large. That is, the output h of the weak learner seeks to maximize the following sum.

$$-\langle h, \nabla C \rangle = \hat{\mathrm{E}}_{(x,\vec{C})\sim S} \mathrm{E}_{y'\sim H_{t-1}(y|x)} C_{y'} - C_{h(x)} \tag{4}$$

Note that these weight updates are similar to those used in IW. Unlike the weights in IW, however, the weights $w_{x,y} := \mathrm{E}_{y'\sim H_{t-1}(y|x)} C_{y'} - C_y$ can be negative, since y is not necessarily the best (least cost) label. This means that the weak learner now receives both positive and negative weights. While the minimization of weighted misclassification with positive and negative weights makes perfect sense as an optimization problem, its interpretation as a classification problem is not immediately clear. In particular, it prohibits the use of weighted sampling as a means of realizing the weighted classification problem.

We deal with this problem by converting a relational version of the weighted multi-class classification problem (i.e. of finding h to maximize Eq. 4) in each iteration to a weighted binary classification problem. Specifically we convert each example pair (x, y) to $((x, y), l)$, and set $l = 1$ if the weight on (x, y) is positive, and $l = 0$ if the weight is negative. The output hypothesis of the binary classifier is in general relational, so it is converted to a stochastic hypothesis by the procedure **Stochastic** shown in Figure 4. (The particular way this procedure is defined is motivated by the theoretical guarantee, which will be shown in the next subsection.) The overall process, consisting of multiple iterations of such a reduction, constitues a reduction of the *stochastic* multi-class cost-sensitive classification to binary weighted classification.

With the foregoing definitions, we can now state our main method, *GBSE* (Gradient Boosting with Stochastic Ensembles), which is shown in Figure 3.

3.4 Theoretical Performance Guarantee on a Variant

It turns out that a strong theoretical performance guarantee can be proved on a variant of this method. The variant is obtained by simply replacing the weight updating rule of GBSE by the following:

$$w_{x,y} = \frac{C_{H_{t-1}(x)}}{k} - C_y$$

The resulting variant, which we call GBSE-T (Gradient Boosting with Stochastic Ensembles - Theoretical version), is summarized in Figure 5.

We can show that GBSE-T has a boosting property given a version of weak learning condition on the component classifier. This weak learning condition, which we make precise below, is one that is sensitive to class imbalance.

DEFINITION 1. *We say that an algorithm A for the binary importance weighted classification problem, as defined in Section 2, satisfies the* weak learning condition *for a given classification sample $S = (x, y, c)^m$, its output h satisfies the following, for some fixed $\gamma > 0$:*

$$\begin{aligned} & \hat{\mathrm{E}}_{(x,y,c)\sim S} cI(h(x) = y) \\ \geq & \hat{\mathrm{E}}_{(x,y,c)\sim S} cI(y = 0) + \gamma \hat{\mathrm{E}}_{(x,y,c)\sim S} cI(y = 1) \end{aligned} \tag{5}$$

THEOREM 2. *Suppose that the component learner A satisfies the weak learning condition for samples S created by GBSE-T. If we set $\alpha_t = \alpha$ for all t, the output of GBSE-T satisfies:*

$$\hat{\mathrm{E}}_{(x,\vec{C})\sim S} C_{H_T(x)} \leq \exp\{-\frac{\gamma\alpha}{k}T\} \hat{\mathrm{E}}_{(x,\vec{C})\sim S} C_{H_0(x)}$$

- An input sample $S = (x, \vec{C})^m$.
- A component learner A for (importance weighted) binary classification that takes a sample of the form $((x, y), l, w)^*$, and outputs a relational hypothesis h.
- A subprocedure **Stochastic**, as specified in Figure 4.

1. Initialize $S' = \{(x, y) | (x, \vec{C}) \in S, y \in Y\}$.
2. Initialize H_0 by $\forall x \in X, y \in Y$ $H_0(y|x) = 1/k$.
3. **For** $t := 1$ to T **Do**
   (a) $w_{x,y} := \mathrm{E}_{y'\sim H_t(x)} C_{y'} - C_y$ for all (x, y) in S.
   (b) $S_t = \{((x, y), I(w_{x,y} \geq 0), |w_{x,y}|) | (x, y) \in S'\}$.
   (c) Let $h_t := A(S_t)$
   (d) $f_t :=$ **Stochastic**(h_t, H_{t-1}).
   (e) Pick $\alpha_t \in [0, 1)$, set $H_t := (1 - \alpha_t)H_{t-1} + \alpha_t f_t$.
4. **End For**
5. Return H_T.

Figure 3: Method *GBSE (Gradient Boosting with Stochastic Ensembles)*

Stochastic(h: a relational hypothesis, H: a stochastic hypothesis)

1. Define f by setting for each $x \in X$:
   - (default) if $|\{y \in Y | h(x, y) = 1\}| = 0$ then define for all $y \in Y$, $f(y|x) = H(y|x)$.
   - else $f(y|x) = \frac{I(h(x,y)=1)}{|\{y\in Y | h(x,y)=1\}|}$.
2. Output f.

Figure 4: Sub-procedure Stochastic

Identical to Method *GBSE* of Figure 3, except for the following change:
3.(a) Set $w_{x,y} := \tilde{C}_{H_{t-1}(x)} - C_y$, for all (x, y) in S.

Figure 5: Method *GBSE-T (Gradient Boosting with Stochastic Ensembles - Theoretical variant)*

This theorem shows that the output error rate of GBSE-T (left side) converges to an ensemble hypothesis with small expected error (right side).

Proof

We first establish the following simple correspondence between the *weak learning conditions* on the relational multi-class classification problem that we wish to solve in each iteration, and the binary classification problem that is given to the component algorithm to solve it.

DEFINITION 2. *Let S be a weighted sample of the form $S = (x, y, c)^m$, where weights c can be both positive and negative.*

1. *The relational weighted multi-class classification problem for S is to find a relational hypothesis $h : X \times Y \to \{0, 1\}$ that maximizes the following sum:*

$$\hat{\mathrm{E}}_{(x,y,c)\sim S} c \cdot h(x, y)$$

2. *The weighted binary classification problem for S is to find a hypothesis $h : X \times Y \to \{0, 1\}$ that maximizes the following weighted classification accuracy:*

$$\hat{\mathrm{E}}_{(x,y,c)\sim S} |c| \cdot I(h(x, y) = I(c > 0))$$

LEMMA 1. *For all h:*

$$\hat{\mathrm{E}}_{(x,y,c)\sim S} c \cdot h(x, y)$$

$$= \hat{\mathrm{E}}_{(x,y,c)\sim S'} |c| \left[I(h(x, y) = I(c > 0)) - I(c < 0)\right]$$

Proof of Lemma 1

$$\begin{aligned}
& \hat{\mathrm{E}}_{x,y,c)\in S} |c| \cdot I(h(x, y) = I(c > 0)) \\
= & \hat{\mathrm{E}}_{(x,y,c)\sim S} cI(h(x, y) = 1 \text{ and } c \geq 0) \\
& + \hat{\mathrm{E}}_{(x,y,c)\sim S} -c \cdot I(h(x, y) = 0 \text{ and } c < 0) \\
= & \hat{\mathrm{E}}_{(x,y,c)\sim S} ch(x, y)I(c \geq 0) + c(h(x, y) - 1)I(c < 0) \\
= & \hat{\mathrm{E}}_{(x,y,c)\sim S} c \cdot h(x, y) + \hat{\mathrm{E}}_{(x,y,c)\sim S} |c|I(c < 0)
\end{aligned}$$

Hence the lemma follows. **Q.E.D.**

Proof of Theorem 2

First, note that applying **Stochastic** to h_t can increase the expected cost only for x's such that $|\{y|h_t(x, y) = 1\}| = 0$, and for such x's the cost of f equals that of H_{t-1} by the definition of **Stochastic**. Hence, the average empirical cost of f on the original sample S, satisfies the following:

$$\hat{\mathrm{E}}_{(x,\vec{C})\sim S} C_{f(x)} - \sum_y h_t(x, y)C_y \leq \hat{\mathrm{E}}_{(x,\vec{C})\sim S} C_{H_{t-1}}(x)I(\forall y h(x, y) = 0) \tag{6}$$

Now recall that the expected empirical cost of H_t equals the following, where we drop the subscript t from α_t.

$$\hat{\mathrm{E}}_{(x,\vec{C})\sim S} C_{H_t(x)} = \hat{\mathrm{E}}_{(x,\vec{C})\sim S} (1 - \alpha)C_{H_{t-1}(x)} + \alpha C_{f(x)}$$

Hence, by combining Eq. 6 and Eq. 7, we can show the following bound on the decrease in empirical cost in each iteration. Here, we also drop the subscript t on h.

$$\begin{aligned}
& \hat{\mathrm{E}}_{(x,\vec{C})\sim S} C_{H_{t-1}(x)} - \hat{\mathrm{E}}_{(x,\vec{C})\sim S} C_{H_t(x)} \\
= & \hat{\mathrm{E}}_{(x,\vec{C})\sim S} \alpha(C_{H_{t-1}}(x) - C_{f(x)}) \\
= & \alpha \hat{\mathrm{E}}_{(x,\vec{C})\sim S} C_{H_{t-1}}(x) - \sum_y h(x, y)C_y \\
& + \sum_y h(x, y)C_y - C_{f(x)} \\
\geq & \alpha \hat{\mathrm{E}}_{(x,\vec{C})\sim S} C_{H_{t-1}}(x) \\
& - \sum_y h(x, y)C_y - C_{H_{t-1}}(x)I(\forall y\, h(x, y) = 0) \\
\geq & \alpha \hat{\mathrm{E}}_{(x,\vec{C})\sim S} \sum_{y:h(x,y)=1} h(x, y) \left(\frac{C_{H_{t-1}(x)}}{k} - C_y\right) \\
& + \sum_{y:h(x,y)=0} \frac{C_{H_{t-1}(x)}}{k} - C_{H_{t-1}}(x)I(\forall y\, h(x, y) = 0) \\
= & \alpha \hat{\mathrm{E}}_{(x,\vec{C})\sim S} \sum_y h(x, y) \left(\frac{C_{H_{t-1}(x)}}{k} - C_y\right) \\
& + C_{H_{t-1}}(x) \left(\frac{|\{y : h(x, y) = 0\}|}{k} - I(\forall y\, h(x, y) = 0)\right) \\
\geq & \alpha \hat{\mathrm{E}}_{(x,\vec{C})\sim S} \sum_y h(x, y) \left(\frac{C_{H_{t-1}(x)}}{k} - C_y\right) \\
= & \alpha \hat{\mathrm{E}}_{(x,\vec{C})\sim S} \sum_y \left|\frac{C_{H_{t-1}(x)}}{k} - C_y\right| \\
& \left[I\left(h(x, y) = I\left(\frac{C_{H_{t-1}(x)}}{k} > C_y\right)\right) - I\left(\frac{C_{H_{t-1}(x)}}{k} < C_y\right)\right]
\end{aligned} \tag{7}$$

Where lemma 1 is applied for the last equality. Next apply the weak learning assumption on the induced measure over (x', y', c') defined by: $x' = (x, y)$, $y' = I\left(\frac{C_{H_{t-1}(x)}}{k} > C_y\right)$, and $c' = \left|\frac{C_{H_{t-1}(x)}}{k} - C_y\right|$ to get:

$$\begin{aligned}
\geq & \alpha\gamma \hat{\mathrm{E}}_{(x,\vec{C})\sim S} \sum_y \left|\frac{C_{H_{t-1}(x)}}{k} - C_y\right| I\left(\frac{C_{H_{t-1}(x)}}{k} > C_y\right) \\
\geq & \frac{\alpha\gamma}{k} \hat{\mathrm{E}}_{(x,\vec{C})\sim S} C_{H_{t-1}(x)}
\end{aligned}$$

The last inequality follows because all $C_y \geq 0$, there exists $C_y = 0$, and the sum is bouned by it's largest term.

Since the expected cost is convex (in fact linear), this implies convergence to the global optimum. Noting that in each iteration, the empirical cost is reduced at least by a factor of $1 - \frac{\gamma\alpha}{k}$, the theorem follows. **Q.E.D.**

Note that at earlier iterations, the binary classifier used as the component learner is likely to be given weighted sample with balanced positive and negative examples. As the number of iterations increases and progress is made, however, it will receive samples that are increasingly more negative. (This is because the positive examples correspond to labels that can further improve the current performance.) It therefore becomes easier to attain high weighted accuracy by simply classifying all examples to be negative. The weak learning condition of Eq. 5 appropriately deals with this issue, as it requires that the weak learner achieve better weighted accuracy than that attainable by assigning all examples to the negative class.

4. EXPERIMENTAL EVALUATION

We use the C4.5 decision tree learner [15] as the base classifier learning method, because it is a standard for empirical comparisons and it was used as the base learner by Domingos for the *MetaCost* method [7].

We compare our methods against three representative methods: Bagging [4], Averaging cost [6, 7] and MetaCost. The Averaging cost method was also used for comparison in [7]. Note that Bagging is a cost-insensitive learning method. Here we give a brief description of these methods, and refer the reader to [4, 7] for the details.

- *Bagging* obtains multiple sub-samples by sampling with replacement, feeds them to the base learner (C4.5), and takes the average over the ensemble of output hypotheses.
- *Averaging Cost* (AvgCost) obtains a subsample by weighted sampling with weights defined as the average cost for each x, and then feeds it to the base learner (C4.5).
- *MetaCost* uses bagging to obtain an ensemble of hypotheses, uses the ensemble to estimate the class probabilities, and then outputs a hypothesis that minimizes the expected risk with respect to these estimates.

There are some deviations from these methods in our implementation, which we clarify below. The main deviation is that we use *rejection* sampling for all methods, while other sampling schemes such as resampling with replacement are used in the original methods. We do this for two reasons: (1) inadequacy of resampling with replacement, especially for C4.5, has been noted by various authors [17]; (2) since our proposed methods use rejection sampling, we do the same for the other methods for fairness of comparison. We stress that this deviation should only improve their performance. Another deviation is that we use a variant of MetaCost that skips the last step of learning a classifier on a relabeled training data set. It has been observed that this variant performs at least as well as MetaCost, in terms of cost minimization. (This variant has been called *BagCost* by Margineantu [13].) Also, in our implementation of AvgCost, we perform weighted sampling multiple times to obtain an ensemble of hypotheses, then ouput their average as the final hypothesis. We note that, due to our normalization assumption that the minimum cost for each instance x is always zero, our version of AvgCost is identical to a more sophisticated variant in which the difference between the average cost and the minimum cost is used for sampling weights. Our experience shows that this variant of AvgCost performs better than the original method.

Dataset	# of examples	# of classes	Class ratio
Annealing	898	5	0.01316
KDD-99	197710	5	0.0001278
Letter	20000	26	0.9028
Satellite	6435	6	0.4083
Solar flare	1389	7	0.002562
Splice	3190	3	0.4634

Table 1: Data set characteristics: data size, number of classes, and the ratio between the frequency of the most common class to the least common.

The methods were applied to five benchmark datasets available from the UCI machine learning repository [2] and one dataset from the UCI KDD archive [1]. These datasets were selected by the criteria of having approximately 1,000 data or more, besides being multiclass problems. A summary of these datasets is given in Table 1. Except for the KDD-99 dataset, these datasets do not have standard misclassification costs associated with them. For this reason, we follow Domingos and generate cost matrices according to a model that gives higher costs for misclassifying a rare class as a frequent one, and inversely for lowest. (Note therefore that our experiments do not exploit the full generality of the instance-dependent cost formulation presented in Section 2.) This reflects a situation that is found in many practical data mining applications, including direct marketing and fraud detection, where the rare classes are the most valuable to identify correctly.

Our cost model is as follows: Let $\hat{P}(y_1)$ and $\hat{P}(y_2)$ be the empirical probabilities of occurence of classes y_1 and y_2 in the training data. We choose the non-diagonal entries of the cost matrix $C(y_1, y_2)$, $y_1 \neq y_2$ with uniform probability from the interval $[0, 2000\hat{P}(y_1)/\hat{P}(y_2)]$. In [7], the diagonal entries were then chosen from the interval $[0, 1000]$, which often leads to cost matrices in which the correct label is not the least costly one. Besides being unreasonable (see Elkan [8]), these cost matrices can give an unfair advantage to cost-sensitive methods over cost-insensitive ones. We therefore set the diagonal entries to be identically zero, which is consistent with our normalization assumption.

In all experiments, we randomly select 2/3 of the examples in the dataset for training and use the remaining 1/3 for testing. Also, for each training/test split we generate a different cost matrix according to the rules above. Thus, the standard deviations that we report reflect both variations in the data and in the misclassification costs.

We remark on certain implementation details of the proposed learning methods in our experimentation. First, we note that in all of the methods used for comparison, except IW, C4.5 was used as the component algorithm, and the final hypothesis is expressed as an ensemble of output decision tress of C4.5. IW, as a meta-method, does not use ensembles; instead we used an ensemble method of *costing* [17]

Dataset	Bagging	AvgCost	MetaCost	IW	DSE	GBSE
Annealing	1059 ± 174	127.4 ± 12.2	206.8 ± 42.8	67.38 ± 9.22	127.1 ± 14.9	$\mathbf{33.01 \pm 4.30}$
Solar	5403 ± 397	237.8 ± 37.5	5317 ± 390	174.2 ± 32.7	110.9 ± 28.7	$\mathbf{35.18 \pm 12.4}$
KDD-99	319.4 ± 42.2	42.43 ± 7.95	49.39 ± 9.34	50.43 ± 10.0	46.68 ± 10.16	$\mathbf{2.05 \pm 0.815}$
letter	151.0 ± 2.58	$\mathbf{91.90 \pm 1.36}$	129.6 ± 2.44	247.7 ± 4.15	114.0 ± 1.43	222.5 ± 8.7
Splice	56.98 ± 3.76	51.16 ± 2.99	$\mathbf{46.14 \pm 2.66}$	63.92 ± 5.51	55.25 ± 3.18	68.34 ± 4.64
Satellite	189.9 ± 9.57	107.8 ± 5.95	$\mathbf{104.4 \pm 6.43}$	140.1 ± 18.2	116.8 ± 6.28	162.36 ± 12.9

Table 2: Experimental results: the average cost and standard error.

Dataset	Bagging	AvgCost	MetaCost	IW	DSE	GBSE
Annealing	11991 ± 13.1	1002.8 ± 183	11987 ± 9.84	–	3795.5 ± 688	1260.2 ± 224
Solar	18499 ± 20.4	334.80 ± 37.5	18510 ± 14.4	–	2112.8 ± 276	486.45 ± 53.3
KDD-99	395310 ± 143	2551.9 ± 428.6	395580 ± 143	–	12512 ± 2450	4181 ± 783.6
letter	40037 ± 44.3	159720 ± 2028	40052 ± 41	–	479130 ± 2710	363001 ± 5557
Splice	42515 ± 26.6	33658 ± 1697	42501 ± 21	–	52123 ± 592	50284 ± 3659
Satellite	86136 ± 123	60876 ± 1641	85984 ± 127	–	218870 ± 6516	140810 ± 3335

Table 3: Experimental results: the average data size used by each method in 30 iterations, and standard error.

on C4.5, as the component algorithm. Its output hypothesis is therefore also an ensemble of decision trees. Next, the choice of the mixture weight α_t was unspecified in the algorithm descriptions. The choice of α_t was set at $\frac{1}{t}$ for most methods, except some optimization was performed in GBSE, as is common in the gradient boosting framework. In particular, the choice of α_t was optimized in GBSE by performing a simple line search to approximately minimize the empirical cost of the resulting ensemble hypothesis on (a part of) the training data. Here the optimization is done with respect to the objective of minimizing the empirical cost of the functional hypothesis derived from the resulting mixture hypothesis by outputting y with the maximum probability, namely,

$$\hat{\mathrm{E}}_{(x,\vec{C})\sim S}\left[C_{\arg\max_y (1-\alpha_t)\,\mathrm{E}_{y'\sim H_{t-1}(y|x)} + \alpha_t f_t(y|x)}\right]$$

The results of these experiments are summarized in Table 2 and Table 3. Table 2 lists the average costs attained by each of these methods on the 6 data sets, and their stand errors. These results were obtained by averaging over 20 runs, each run consisting of 30 iterations of the respective learning method. Examining these results in conjunction with the data characteristics in Table 1 reveals a definite trend. First, note that the data sets are divided into two groups: those having very large skews, or very low class ratios (Annealing, KDD-99 and Solar flare), and those having moderate skews (Satelite, Splice and Letter). On the first group, GBSE's performance is much better than all comparison methods (and tend to be magnitudes better than MetaCost in particular). On the second group, the comparsion methods (MetaCost and AvgCost) perform somewhat better, but are almost tied with the other two proposed methods, DSE and IW.

Table 3 lists the average total data size used by each of the methods in 30 iterations. The data size for IW is not listed, since it consists of 30 iterations of 10 rounds of costing and the direct comparison of total data size does not seem to make as much sense for this method. Here again, the trend is clear: the methods based on example weighting (AvgCost, GBSE, DSE) use magnitudes smaller data sizes for the 3 data sets in the first group (i.e. with large skews), as compared to other methods, Bagging and MetaCost. For the data sets in the first group, the performance of GBSE is especially impressive, achieving much lower cost while requiring very small data sizes. It is worth mentioning that it is these data sets in the first group with large skews, that require *cost-sensitive* learning the most.

5. DISCUSSION

It is not the first time that the issue of incorporating cost-sensitivity to boosting has been addressed. For example, AdaCost [9] suggested a way of modifying AdaBoost's exponential loss using a function (called *cost adjustment function*) of the cost and confidence. The rational choice of this cost adjustment function, however, appears not to be well-understood. The stochastic ensemble that we employ in the prepsent paper provides a straightforward but reasonable way of incorporating cost and confidence, i.e. in terms of expected cost. An interesting future direction is to investigate the relationship between these alternative approaches to cost-sensitive boosting. Also note that AdaCost, being a modification of AdaBoost, is restricted to the two-class problems. Cost-sensitive extension of various multiclass versions of boosting is another interesting future research topic.

6. REFERENCES

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

[2] C. L. Blake and C. J. Merz. UCI repository of machine learning databases. Department of Information and Computer Sciences, University of California, Irvine, 1998. `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

[3] J. Bradford, C. Kunz, R. Kohavi, C. Brunk, and

C. Brodley. Pruning decision trees with misclassification costs. In *Proceedings of the European Conference on Machine Learning*, pages 131–136, 1998.

[4] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[5] L. Breiman, J. H. Friedman, R. A. Olsen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth International Group, 1984.

[6] P. Chan and S. Stolfo. Toward scalable learning with non-uniform class and cost distributions. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, pages 164–168, 1998.

[7] P. Domingos. MetaCost: A general method for making classifiers cost sensitive. In *Proceedings of the Fifth International Conference on Knowledge Discovery and Data Mining*, pages 155–164. ACM Press, 1999.

[8] C. Elkan. Magical thinking in data mining: Lessons from coil challenge 2000. In *Proceedings of the Seventh International Conference on Knowledge Discovery and Data Mining*, pages 426–431. ACM Press, 2001.

[9] W. Fan, S. J. Stolfo, J. Zhang, and P. K. Chan. AdaCost: Misclassification cost-sensitive boosting. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pages 97–105, 1999.

[10] G. Fumera and F. Roli. Cost-sensitive learning in support vector machines. In *VIII Convegno Associazione Italiana per L'Intelligenza Artificiale*, 2002.

[11] P. Geibel and F. Wysotzki. Perceptron based learning with example dependent and noisy costs. In *Proceedings of the Twentieth International Conference on Machine Learning*, 2003.

[12] U. Knoll, G. Nakhaeizadeh, and B. Tausend. Cost-sensitive pruning of decision trees. In *Proceedings of the Eight European Conference on Machine Learning*, pages 383–386, 1994.

[13] D. Margineantu. *Methods for Cost-Sensitive Learning*. PhD thesis, Department of Computer Science, Oregon State University, Corvallis, 2001.

[14] L. Mason, J. Baxter, P. Barlett, and M. Frean. Boosting algorithms as gradient descent. In *Advances in Neural Information Processing systems 12*, pages 512–158, 2000.

[15] J. Quinlan. *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann, 1993.

[16] B. Zadrozny and C. Elkan. Learning and making decisions when costs and probabilities are both unknown. In *Proceedings of the Seventh International Conference on Knowledge Discovery and Data Mining*, pages 204–213. ACM Press, 2001.

[17] B. Zadrozny, J. Langford, and N. Abe. Cost-sensitive learning by cost-proportionate example weighting. In *Proceedings of the Third IEEE International Conference on Data Mining*, pages 435–442, 2003.