Bootstrap Methods for the Cost-Sensitive Evaluation of Classifiers

Dragos D. Margineantu MARGINDR@CS.ORST.EDU
Thomas G. Dietterich TGD@CS.ORST.EDU
Department of Computer Science, Oregon State University, Corvallis, OR 97331-3202, USA

Abstract

Many machine learning applications require classifiers that minimize an asymmetric cost function rather than the misclassification rate, and several recent papers have addressed this problem. However, these papers have either applied no statistical testing or have applied statistical methods that are not appropriate for the cost-sensitive setting. Without good statistical methods, it is difficult to tell whether these new cost-sensitive methods are better than existing methods that ignore costs, and it is also difficult to tell whether one cost-sensitive method is better than another. To rectify this problem, this paper presents two statistical methods for the cost-sensitive setting. The first constructs a confidence interval for the expected cost of a single classifier. The second constructs a confidence interval for the expected difference in costs of two classifiers. In both cases, the basic idea is to separate the problem of estimating the probabilities of each cell in the confusion matrix (which is independent of the cost matrix) from the problem of computing the expected cost. We show experimentally that these bootstrap tests work better than applying standard z tests based on the normal distribution.

1. Introduction

Most existing classification methods were designed to minimize the number of mistakes that are made, also called the 0/1 loss. However, real-world applications often require classifiers that minimize the total cost of misclassifications given that each error has an associated cost. For example, the cost of false positive errors (e.g., giving chemotherapy to a patient that does not have cancer) is typically different from the cost of false negative errors (e.g., failing to treat a patient that does have cancer). This problem is widely recognized, and many researchers have tried to incorporate cost information into learning algorithms.

We address one of the most studied settings for cost-sensitive learning, which is defined by a set of labeled training examples $\{\langle x_\ell, y_\ell\rangle\}$ and a cost matrix C. The contents of $C(i,j)$ specify the cost incurred when an example is predicted to be in class i when in fact it belongs to class j. The goal of the learning algorithm is to output a classifier γ whose expected cost is minimized when classifying new examples drawn from the same distribution as the training examples. There are two main approaches to cost-sensitive learning. The first approach assumes that C is available at learning time, and several methods have been developed to incorporate C into the learning process (see, for example, Knoll & al., 1994; Bradford & al., 1998; Kukar & Kononenko, 1998; Domingos, 1999; Fan & al., 1999; Margineantu & Dietterich, 1999). The second approach is to assume that C is not available and to seek learning algorithms that produce hypotheses that have good performance over a wide range of different cost matrices (e.g. any method that outputs the optimal decision-theoretic prediction based on the class probability estimates of the test examples).

For either approach, cost-sensitive statistical methods are needed to help choose algorithms for a particular application and to guide the development of new algorithms. This paper proposes two statistical methods for the case where C is available. We do not treat the case where C is not available, although one strategy might be to apply the methods of this paper to several different C matrices drawn by sampling from some distribution of likely cost matrices.

Our statistical methods address two questions. First, given a classifier, we would like to use test data to estimate a confidence interval for the mean cost of that classifier. This is useful for estimating the costs that are likely to be incurred on a given practical task and also for determining whether the available test data set is large enough to estimate the expected cost accurately. Second, given two classifiers, we would like to decide whether one classifier has lower expected cost than another. This is useful for comparing classifiers for a particular application, and it can also help guide research on cost-sensitive learning. We do not address the more difficult statistical question of deciding whether one learning *algorithm* is better than another within a particular domain (i.e., over all training sets of fixed size m). In other words, our statistical tests control for variation in the test set, but they do not control for variation in the training set or any internal randomness inside the learning algorithm. See (Dietterich, 1998) for a discussion of these issues.

This paper is organized as follows. The next section introduces the cost-sensitive classification problem and then discusses possible statistical methods and their drawbacks. The third section presents the new statistical methods for cost-sensitive evaluation of classifiers. In Section 4, we evaluate these two statistical methods through a simulation study, and discuss the results. Section 5 concludes the paper.

2. An Analysis of the Cost-Sensitive Classification Problem

To illustrate the cost-sensitive learning problem, consider the 4×4 cost matrix shown in Table 1. In this table, there are two errors that are very expensive (predicting class 2 when the true class is 4 costs 100.0, and predicting class 3 when the true class is 1 costs 40.5). Other errors are much less expensive (e.g., predicting class 4 when the true class is 2 only costs 0.1). As a consequence, a learning algorithm should not treat all costs equally—it should avoid committing expensive errors, possibly at the cost of committing a larger number of inexpensive errors. For example, if the algorithm believes that there is any chance that a test example might belong to class 4, then it should predict class 4, because the costs of predicting any of the other classes in this case are high.

Cost-sensitive learning problems are made more difficult by the fact that in many domains the expensive errors correspond to the rare cases. In air traffic control, for example, crashing an aircraft is a very expensive error, but (fortunately) there are very few training examples of airplane crashes, compared to the number of safe flights. A cost-insensitive learner might therefore decide to ignore these rare events and always predict that the airplane is safe. This would give a low number of errors, but a very high cost.

The fact that different kinds of errors can have extremely different costs also means that we must take care when measuring the performance of cost-sensitive learning algorithms. In the ordinary 0/1 loss case, we can group together all of the different misclassification errors and estimate the total fraction of misclassifications. But in the cost-sensitive case, we need to treat each different kind of misclassification error separately, because they have different costs. The statistical problem is especially difficult if there is a rare but expensive kind of mistake. In that case, a larger test set is needed to ensure that we obtain enough data to be able to detect these rare, expensive cases.

Table 1. Example of cost matrix for a four-class problem.

Predicted	Correct Class			
Class	1	2	3	4
1	0.0	3.2	2.5	12.7
2	1.0	0.0	3.0	100.0
3	40.5	2.2	0.0	5.5
4	1.0	0.1	7.1	0.0

Let us consider the most straightforward approach to the statistical analysis of cost-sensitive classifiers. Given a set of n test examples, we can apply the learned classifier γ to evaluate each example. Let the cost of classifying example $(\mathbf{x}_\ell, y_\ell)$ be $c_\ell = C(\gamma(\mathbf{x}_\ell), y_\ell)$. We can compute the mean and standard deviation of the c_ℓ values and construct a confidence interval based on the t or z statistics. The central limit theorem assures us that the mean of n cost values will be normally distributed with a variance of $\sigma/\sqrt{n}$, where σ is the variance of the c_ℓ values.

Unfortunately if the values in the cost matrix C are very different, the variance σ will be very large, so the resulting confidence interval will be very wide. Consequently, this procedure will not give us a very tight confidence interval.

The alternative approach that we describe in this paper is to estimate the probability (on the test set) of each kind of mistake. Let $p(i,j)$ be the probability that a test example belonging to class j was predicted to be in class i. These probability values can be computed by constructing a confusion matrix and dividing each cell by n. Once these probabilities are computed, we can compute the expected cost of the classifier by multiplying each element $p(i,j)$ by the corresponding cell $C(i,j)$ and summing the results:

$$\overline{c} = \sum_{i,j} p(i,j) \cdot C(i,j).$$

Based on a small test set, there will be some uncertainty about each of the $p(i,j)$ values. If the corresponding cost $C(i,j)$ is very large, then this uncertainty will have a big impact on the estimated error. Fortunately, each probability $p(i,j)$ is a betterbehaved statistic than the average of the c values. It has a variance that depends only on its true value, and extreme values (near zero and one) have smaller variances than intermediate values. Moreover, when the test data set is small, we can use Laplace corrections (or equivalently, Dirichlet priors) to incorporate prior beliefs about the sizes of the $p(i,j)$ values (Good, 1965; Fienberg & Holland, 1968; Cestnik, 1991).

The approach that we take in this paper is to compute the $p(i,j)$ values, adjust them using Laplace corrections, and then apply bootstrap methods to construct confidence intervals for the expected cost of the classifier. We will see that this approach gives much better results than simply computing a confidence interval based on the t or z statistics.

3. New Statistical Methods

3.1 Estimating the Expected Cost of a Classifier: BCost

Let γ be a classifier, and $\mathcal{D}$ be a test set of n examples. Let M be the $k \times k$ confusion matrix computed from γ and $\mathcal{D}$, where element $M(i,j)$ is the number of test set examples predicted by γ to be in class i but actually belonging to class j.

Let $p(i,j)$ be the normalized, Laplace-corrected confusion matrix defined as follows:

$$p(i,j) = \frac{M(i,j) + \lambda}{k^2\lambda + n}.$$

Here, the constant $\lambda \geq 0$ determines the strength of the Laplace correction. The $p(i,j)$ values can be viewed as a multinomial probability distribution over the k^2 combinations of predicted and correct classes.

The BCost procedure computes a confidence interval for the expected cost of classifier γ as shown in Table 2. This pseudo-code generates 1000 simulated confusion matrices $\tilde{M}_u$ by generating a sample of size n according to the distribution $p(i,j)$. For each simulated confusion matrix, it then computes the cost $\tilde{c}_u$ by taking a dot product between $\tilde{M}_u$ and the cost matrix C. This gives 1000 simulated costs. For a 95% confidence interval, it then outputs the 26th and 975th simulated costs (after sorting them into ascending order). This is a bootstrap confidence interval for the mean cost after applying a Laplace correction to the original confusion matrix.

3.2 Comparing the Expected Cost of Two Classifiers: BDeltaCost

The second statistical problem is to compare the costs of two different classifiers to decide which is better. Given two classifiers γ_1 and γ_2, we want to test the null hypothesis H_0 that the two classifiers have the same expected cost (on new test data) against the alternative hypothesis H_a that the two classifiers have different costs. We want a test that will accept the null hypothesis with probability ρ if the null hypothesis is true (i.e., a test that has a Type I error of $1 - \rho$).

Table 2. Pseudo-code for the BCost method of constructing a confidence interval.

```
BCost(confidence value $\rho$, cost matrix $C$,
        distribution $p$, number of test examples $n$)
    for $u$ from 1 to 1000 do
        Let $\tilde{M}_u = 0$, a simulated confusion matrix.
        for $v$ from 1 to $n$ do
            draw a pair $(i,j)$ according to $p(i,j)$
            increment $\tilde{M}_u(i,j)$.
        end // for $v$
        Let $\tilde{c}_u = \tilde{M}_u \cdot C$ be the cost of $\tilde{M}_u$
    end // for $u$
    Sort the $\tilde{c}_u$ values into ascending order.
    Let $lb = \lfloor \frac{1-\rho}{2} \times 1000 \rfloor + 1$
    Let $ub = 1001 - lb$
    The confidence interval is $[\tilde{c}_{lb}, \tilde{c}_{ub}]$
end BCost
```

We follow essentially the same approach as for BCost. The key is to define a new kind of three-dimensional confusion matrix M. The contents of cell $M(i_1, i_2, j)$ is the number of test set examples for which γ_1 predicted that they belong to class i_1, γ_2 predicted that they belong to class i_2, and their true class was j. Analogously, we can define a three-dimensional cost matrix Δ such that $\Delta(i_1, i_2, j) = C(i_1, j) - C(i_2, j)$. In other words, the value of $\Delta(i_1, i_2, j)$ is the amount by which the cost of classifier γ_1 is greater than the cost of classifier γ_2 when γ_1 predicts class i_1, γ_2 predicts class i_2, and the true class is j. Given a 3-D confusion matrix measured over a test set, we can compute the difference in the costs of γ_1 and γ_2 by taking the "dot product":

$$M \cdot \Delta = \sum_{i_1, i_2, j} M(i_1, i_2, j)\Delta(i_1, i_2, j).$$

We can obtain our statistical test by computing a confidence interval for $M \cdot \Delta$ and rejecting the null hypothesis H_0 if this confidence interval does not include zero. If the confidence interval does contain zero, we cannot reject the null hypothesis. The confidence interval is constructed in exactly the same way as for BCost. We take the 3-D confusion matrix measured on the test set, normalize it, and apply a Laplace correction to get a multinomial probability distribution $p(i_1, i_2, j)$. We then simulate 1000 confusion matrices

$\tilde{M}_u$ by drawing (i_1, i_2, j) triples according to this distribution. We compute the cost $\tilde{c}_u$ of each simulated confusion matrix (by computing $\tilde{M}_u \cdot \Delta$), sort them, and choose the appropriate elements to construct the confidence interval.

4. Experimental Evaluation of the Proposed Methods

Now that we have described the two new statistical methods, we present some experimental tests to verify that the methods are working properly and to compare them to the confidence intervals based on the normal distribution described above. In addition, we must address the question of how to set the Laplace correction parameter λ and evaluate the sensitivity of the methods to this parameter.

4.1 Experimental Evaluation of BCost

We begin with an experiment to evaluate the BCost procedure. The purpose of this experiment is to construct a situation in which we have a classifier γ whose true expected average cost is known. We can then simulate a large number of different test sets, apply the BCost procedure to each of them, and see whether the resulting confidence interval does indeed capture the true mean in fraction ρ of the trials. In our experiments, we set $\rho = 0.95$, so we are seeking a 95% confidence interval.

Figure 1 shows the decision boundaries for an artificial domain with two features and five classes. We drew one million examples uniformly randomly from this space, and labelled them according to the decision boundaries in the figure. To design a classifier, we used a version of the C4.5 algorithm (Quinlan, 1993) modified to accept weighted training examples. Each example was weighted in proportion to the average value of the column of the cost matrix C corresponding to the label of the example. This is the average cost (over the training set) of misclassifying examples of this class. Breiman (1984) suggests a similar method for building cost-sensitive decision trees; Margineantu & Dietterich (1999) compare this method against other methods for incorporating costs into the decision tree learning algorithm. We will call this cost-sensitive classifier the *C4.5-avg* tree. The classifier is trained on a separate set of one thousand examples drawn from the same distribution as the one million test examples. Table 3 shows the probabilities of each of the class labels computed from the test data.

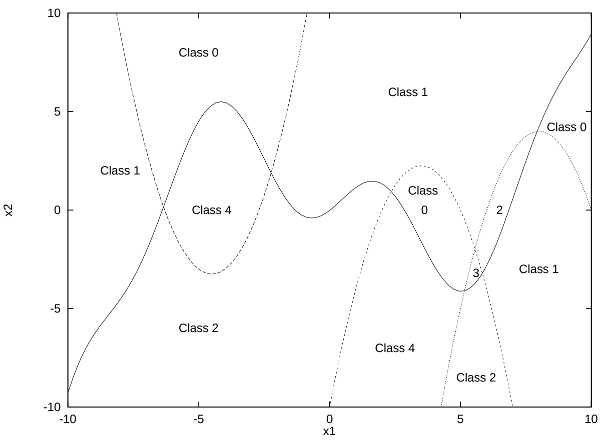

Figure 1. daDecision boundaries for the Expf5 data set.

Table 3. Class frequencies for the Expf5 domain.

Class	Frequency (%)
0	21.02
1	44.73
2	25.68
3	0.16
4	8.41

Each experiment involves testing several different cost matrices, C. These were generated randomly based on nine different cost models. Table 4 shows the underlying distributions for each of the cost models. The diagonal elements of C are always zero for cost models $M1$, $M2$, $M4$, $M5$, $M6$, and $M7$, and they are drawn from a uniform distribution over the range described in the third column for cost models $M3$, $M8$, and $M9$. For all cost models, the off-diagonal elements are drawn from a uniform distribution over the range described in the second column of the table. For cost matrices drawn according to models $M6$, $M7$, and $M9$, we set the cost using the frequency of the classes. In cost model $M6$, for example, the cost of mislabeling an example from class j as belonging to class i is determined by the ratio of the number of examples in class i to the number of examples in class j. In particular, if class i is very common, and class j is very rare, then this mistake will (on the average) be very expensive, because $P(i)/P(j)$ will be a large number. For cost model $M7$, we reversed this relationship, so that the least expensive errors are those that mislabel a rare class j as belong to a common class i. Finally, model $M9$ is like model $M6$ except that the costs are even larger, and there are non-zero costs on the diagonal of C. Cost models $M6$, $M7$ and $M9$ have a higher potential of generating cost matrices for which there are higher risks in making a decision. In the case of our experimental domain, a cost matrix based on cost model $M9$ will have the value of $C(1, 3)$ drawn from a uni-

Table 4. The cost models used for the experiments. Unif$[a, b]$ indicates a uniform distribution over the $[a, b]$ interval. $P(i)$ represents the prior probability of class i.

Cost Model	$C(i,j)$ $i \neq j$	$C(i,i)$
M1	Unif$[0, 10]$	0
M2	Unif$[0, 100]$	0
M3	Unif$[0, 100]$	Unif$[0, 10]$
M4	Unif$[0, 1000]$	0
M5	Unif$[0, 10000]$	0
M6	Unif$[0, 1000 \times P(i)/P(j)]$	0
M7	Unif$[0, 1000 \times P(j)/P(i)]$	0
M8	Unif$[0, 10000]$	Unif$[0, 1000]$
M9	Unif$[0, 2000 \times P(i)/P(j)]$	Unif$[0, 1000]$

Table 5. Results of running BCost on the 1000 1000-example test sets for nine different cost models. 10 cost matrices were used for each cost model. The second column shows the average number of runs of BCost for which the true cost was included in the confidence interval. Laplace correction $\lambda = 0.1$. True cost was computed as the average cost of the classifier for all 1,000,000 examples.

Cost Model	avg. # of cases for which the true cost was included in the c.i. (out of 1000)
M1	956.1
M2	955.5
M3	953.1
M4	954.9
M5	953.9
M6	994.1
M7	971.7
M8	952.9
M9	995.6

form distribution over the interval [0, 559125] (because $2000 \times P(1)/P(3) = 559125$) while other off-diagonal cells can be as low as 0.

For each of the one thousand test sets, we ran BCost and computed the 95% confidence interval for the expected cost. The true expected cost was considered to be the average cost for the entire one million example data set. We set the Laplace correction parameter $\lambda = 0.1$. Table 5 shows the number of test sets for which BCost outputs a 95% confidence interval that includes the true expected cost. The values are averaged over ten cost matrices for each cost model. A perfect 95% confidence interval would include the true average cost exactly 950 times out of 1000. We see that the confidence intervals are all sufficiently wide, although the intervals for $M6$, $M9$, and $M7$ are somewhat too wide.

We also computed a normal confidence interval for each test set. For all five cost models, the true average cost was included in the normal 95% confidence interval 100% of the time (i.e., for all 1000 test sets).

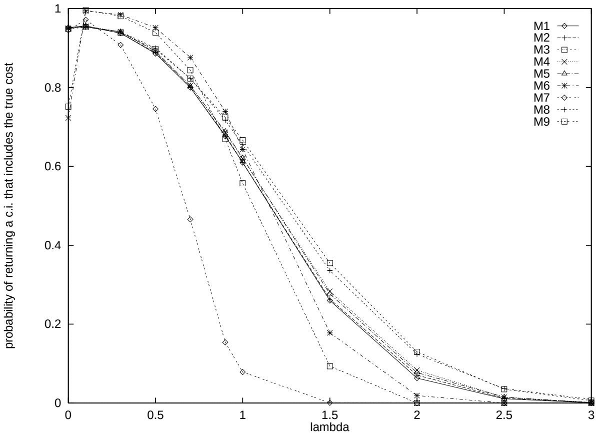

Figure 2. Plots that describe the sensitivity of BCost to the choice of λ for the nine cost models. For each cost model, 10 different matrices were tested, and each point in the plots the average result over the 10 matrices.

This means that the normal confidence intervals are definitely too wide.

The sensitivity of BCost to the choice of the Laplace correction parameter is shown in Figure 2. Each curve illustrates the probability that BCost returns a confidence interval that includes the true cost for one of the nine cost models, as λ varies from 0.0 to 3.0. For our experimental domain, all curves reach a maximum within the interval (0.0,0.5]. No Laplace correction ($\lambda = 0$) will cause BCost to generate confidence intervals that are too narrow while values of λ larger than 0.5 will produce intervals that are too wide and biased too high.

4.2 Experimental Evaluation of BDeltaCost

Any statistical test can commit two kinds of errors: Type I and Type II. We evaluate BDeltaCost on each of these kinds of errors.

A Type I error occurs when two classifiers have exactly the same cost, but the statistical test rejects the null hypothesis. If a statistical test is conducted with a confidence level of 0.95, then it should commit a Type I error with probability 0.05.

To evaluate the probability that BDeltaCost commits a Type I error, we faced a difficult experimental problem. We need two classifiers, γ_1 and γ_2, that are different and yet that have identical average costs. Furthermore, the worst case scenario for a statistical test usually occurs when the two classifiers are very different from each other (Dietterich, 1998), since this variability tends to "fool" the statistical test into thinking that the classifiers have different expected costs. So we want to design the two classifiers so that they misclassify different test examples and yet have the same average cost.

To do this, we first ran C4.5 as described above to produce γ_1. Then, we computed the confusion matrix M^* for γ_1 on the entire collection of one million test examples. To construct our second classifier, γ_2, we don't need to actually construct a decision tree (or any other real classifier)—all we need to do is assign a label to each test example, which we will declare to be the label assigned by γ_2. We want to assign these labels so that the cost of γ_2 (defined in this way) over the entire one million test examples is the same as the cost of γ_1. We will do this by ensuring that γ_2 has the same confusion matrix as γ_1. For each test example, $\langle \mathbf{x}_\ell, y_\ell \rangle$, we choose randomly a new label y_1 such that $M^*(y_1, y_\ell) > 0$, assign that label to example ℓ, and decrement $M^*(y_1, y_\ell)$. When we have labeled all of the test examples, every cell of M^* will be zero. This is equivalent to randomly permuting the γ_1 labels of all test examples belonging to the same true class. This makes the individual decisions of each classifier highly uncorrelated, but makes their confusion matrices identical, so their average cost on each of the 1000 test sets is the same.

Table 6 shows the results of running this procedure under each of our nine cost models with no Laplace correction. It plots the number of test sets for which the null hypothesis is not rejected. This number should be 950 if the test is working perfectly. The results show that the probability of Type I error is higher in the case of cost models that involve higher risks for making a decision.

To test the sensitivity of BDeltaCost to the choice of the Laplace correction parameter, we ran the procedure described in the previous paragraph for different values of λ from the interval (0.0,0.5]. $\lambda \geq 0.1$ caused BDeltaCost to generate confidence intervals that were too wide in the case of all cost models. In particular, for models $M6$, $M7$, and $M9$, the probability of rejecting the null hypothesis for any $\lambda > 0$ was smaller than 0.005, which indicates that the confidence intervals were definitely too wide.

The second kind of error, the Type II error, occurs when the statistical test fails to reject the null hypothesis even when the two classifiers have different expected costs. The standard way of measuring Type II errors is to plot a *power function* which plots the probability that the null hypothesis will be rejected as a function of the amount by which the two classifiers differ in their expected costs.

Table 6. Results of running BDeltaCost on the 1000 1000-example test sets for the two classifiers that have the same expected average cost. $\lambda = 0$.

Cost Model	avg. # of cases for which H_0 is not rejected (out of 1000)
M1	948.83
M2	950.50
M3	948.26
M4	950.46
M5	951.16
M6	934.03
M7	943.76
M8	949.96
M9	932.06

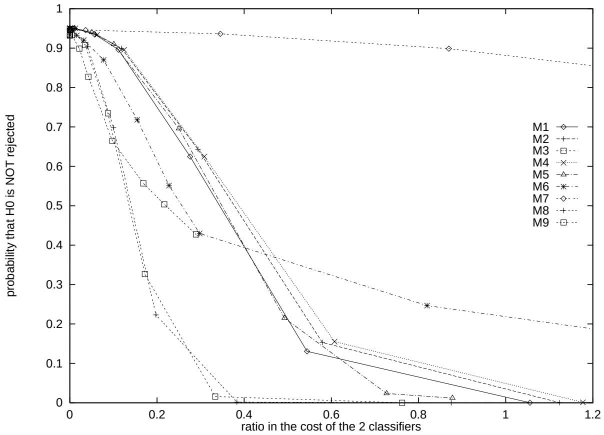

Figure 3. Power of BDeltaCost ($\lambda = 0$). Each curve plots the probability of rejecting H_0 as a function of the cost ratio r_c.

To generate the power curve, we must generate pairs of classifiers γ_1 and γ_2 that have a specified degree of difference in their expected costs. We started with the γ_1 and γ_2 classifiers constructed above. We then built a new classifier, γ_2' by randomly changing q of the labels that γ_2 had assigned to the test eamples. We varied q from 0 to 1,000,000, ran BDeltaCost 10 times for each value of q, and averaged the results for three matrices within each cost model.

Figure 4 plots these power functions for each of the nine cost models as a function of q (we have split up the nine curves for readability purposes) and Figure 3 plots the power functions as a function of the ratio in the cost of γ_2' to γ_1

$$r_c = \frac{|cost(\gamma_2') - cost(\gamma_1)|}{cost(\gamma_1)},$$

where $cost(\gamma)$ is the cost incurred by classifier γ when it classifies all the examples in a test set.

Figure 4 shows that in the case of cost models $M6$ and

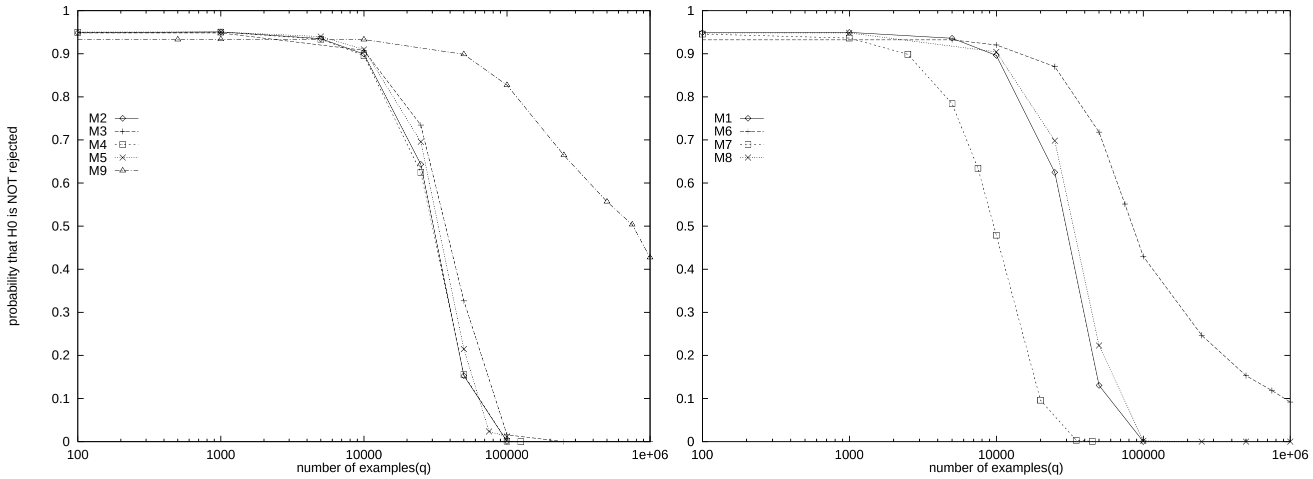

Figure 4. Power of BDeltaCost ($\lambda = 0$). Each curve plots the probability of rejecting H_0 as a function of q.

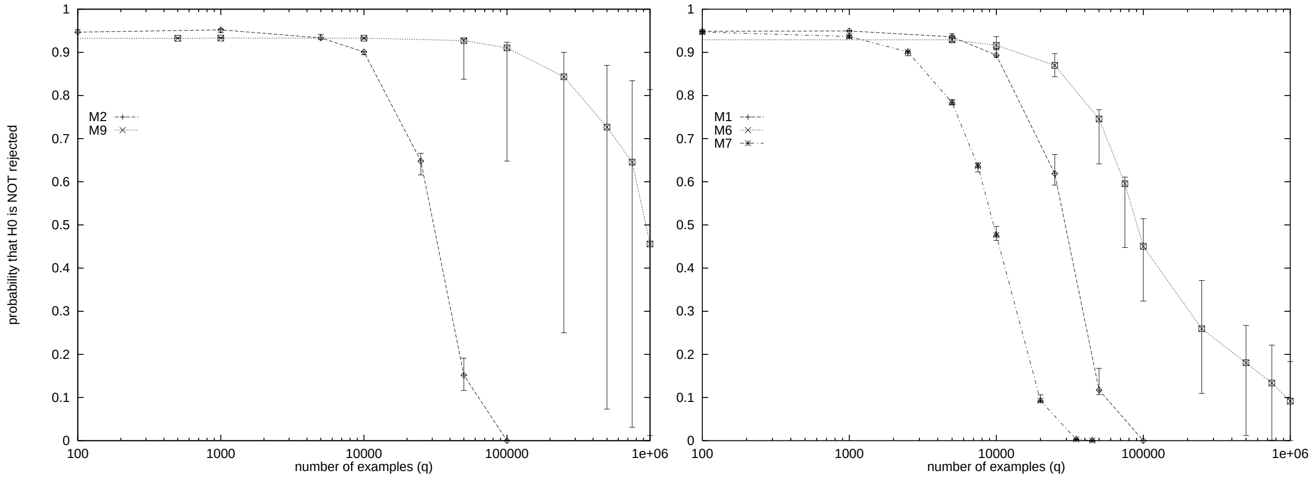

Figure 5. Enlarged plots from Figure 4. Error bars show range of observed probability of rejecting $H_0 . \lambda = 0$.

$M9$ the test has a higher probability of Type II errors. These two cost models assign higher costs for misclassifying instances that belong to rare classes, and we can observe that in these two cases the probability that H_0 is not rejected is greater than 0 (0.427 for cost model $M9$ and 0.092 for cost model $M6$) even when $q = 1,000,000$. Notice that while $M7$ looks best in Figure 4, it appears worst in Figure 3. This is because $M7$ assigns high costs to frequent errors, so even a small change in the number of such errors gives a high change in the cost ratio. For $M7$, BDeltaCost requires a large change in the cost ratio before it can discriminate between the two classifiers.

We have also looked at the variability of the probability of rejecting the null hypothesis, within each cost model. For readability purposes we have plotted this information only for cost models $M1$, $M2$, $M6$, $M7$, and $M9$, and we have split them up into two graphs shown in Figure 5. Each graph uses error bars to show the range of measured values. The variability is minor except for cost models $M9$ and $M6$ when q is large.

In addition to running BDeltaCost on each pair of classifiers, we also ran the paired-differences z test as follows. Let c^1_ℓ be the cost incurred by classifier γ_1 when it classifies example ℓ, and c^2_ℓ be the cost incurred by classifier γ'_2, when it classifies example ℓ. The paired-differences test constructs a normal 95% confidence interval for the differences $c^1_\ell - c^2_\ell$ and rejects the null hypothesis if this confidence interval does not contain zero.

When we ran this test for the various cost models and the various values of q, we found that in all cases, the null hypothesis was never rejected. In other words, this test has no power to detect when one classifier has different expected cost than another.

5. Conclusions

We have presented two procedures for the cost-sensitive evaluation of k-class classifiers.

The first procedure, BCost, derives a bootstrap confidence interval for the expected cost of a classifier for a given domain. The experimental results presented in Table 5 show that BCost behaves very well, although it is slightly too conservative in some cases. The experiments showed that BCost becomes too aggressive if we set the Laplace correction parameter, λ, to zero. The greatest drawback of BCost is that we have no principled way of setting λ. Nonetheless, for a wide range of values for λ, BCost performs much better than the other method that we tested—a confidence interval based on the normal distribution. We recommend $\lambda = 0.1$.

The second procedure, BDeltaCost, derives a bootstrap confidence interval for the expected value of the difference in cost between two classifiers. This procedure also performed very well: For most cost matrices, it achieves a 50% H_0 rejection rate when only 3% of the training examples have different predicted classes. However, for cost matrices having very large differences between the maximum and minimum costs, the method performs more poorly, and larger test sets are probably required before BDeltaCost will perform well. An important question for future research is to establish a relationship between the range of costs in the cost matrix and the required size of the test set. We found that λ should be set to zero in all cases.

In both cases, confidence intervals and statistical tests based on the normal distribution performed so badly that they are worthless for guiding machine learning research. The statistical tests reported in this paper are the first usable methods for cost-sensitive learning with a known cost matrix.

Acknowledgements

The authors gratefully acknowledge the support of NSF grants IRI-9626584 and EIA-9818414.

References

Bradford, J. P., Kunz, C., Kohavi, R., Brunk, C., & Brodley, C. E. (1998). Pruning decision trees with misclassification costs. In *ECML-98. LNAI.* Springer.

Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pat. Rec.*, *30*(7), 1145–1159.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and Regression Trees.* Wadsworth.

Cestnik, B. (1991). Estimating probabilities: A crucial task in machine learning. In Aiello, L. C. (Ed.), *Proc. ECAI-91*, pp. 147–149.

DeGroot, M. H. (1975). *Probability and Statistics.* Addison-Wesley, Reading, Mass.

Dietterich, T. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. *Neur. Comp.*, *10* (7): 1895–1924.

Domingos, P. (1999). Metacost: A general method for making classifiers cost-sensitive. In *Proc. KDD-99.*

Fan, W., Stolfo, S. J., Zhang, J., & Chan, P. K. (1999). AdaCost: Misclassification cost-sensitive boosting. In *ICML-99.* Morgan Kaufmann.

Fienberg, S. E., & Holland, P. W. (1968). Methods for eliminating zero counts in contingency tables. In Patil, G. P. (Ed.), *Random Counts in Scientific Work: Volume 1.* Penn State Univ. Press.

Good, I. J. (1965). *The Estimation of Probabilities: An Essay on Modern Bayesian Methods.* MIT Press, Cambridge, Mass.

Knoll, U., Nakhaeizadeh, G., & Tausend, B. (1994). Cost-sensitive pruning of decision trees. In *ECML-94, LNAI, 784,* pp. 383–386. Springer Verlag.

Kukar, M., & Kononenko, I. (1998). Cost-sensitive learning with neural networks. In *Proc. ECAI-98.*

Margineantu, D., & Dietterich, T. G. (1999). Learning decision trees for loss minimization in multi-class problems. Tech. rep., Dept. Comp. Sci., Oregon St. Univ.

Provost, F., & Fawcett, T. (1997). Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. In *Proc. KDD-97*, pp. 43–48.

Provost, F., Fawcett, T., & Kohavi, R. (1998). The case against accuracy estimation for comparing classifiers. In *Proc. ICML-98.* Morgan Kaufmann, San Francisco, CA.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning.* Morgan Kaufmann, San Francisco, CA.

Rosen, D. B. (1995). How good were those probability predictions? The expected recommendation loss (ERL) scoring rule. In Heidbreder, G. (Ed.), *Maximum Entropy and Bayesian Methods. Proceedings of the Thirteenth International Workshop.* Kluwer.

Ting, K. M., & Zheng, Z. (1998). Boosting trees for cost-sensitive learning. In *Proc. ECML-98. LNAI.* Springer.

Turney, P. (1995). Data engineering for the analysis of semiconductor manufacturing data. In *Proc. IJCAI-95 Workshop on Data Engineering for Indctive Learning*, pp. 50–59.

Turney, P. (1997). Cost-sensitive learning bibliography. In *Online Bibliography. IIT, NRC* Ottawa, Canada. [`http://ai.iit.nrc.ca/bibliographies/cost-sensitive.html`].