Submitted 12/21/02

Learning when Training Data are Costly: The Effect of Class Distribution on Tree Induction

Gary M. Weiss GMWEISS@ATT.COM
AT&T Labs, 30 Knightsbridge Road
Piscataway, NJ 08854 USA

Foster Provost FPROVOST@STERN.NYU.EDU
New York University, Stern School of Business
44 W. 4th St., New York, NY 10012 USA

Abstract

For large, real-world inductive learning problems, the number of training examples often must be limited due to the costs associated with procuring, preparing, and storing the data and/or the computational costs associated with learning from the data. One question of practical importance is: if *n* training examples are going to be selected, in what proportion should the classes be represented? In this article we analyze the relationship between the marginal class distribution of training data and the performance of classification trees induced from these data, when the size of the training set is fixed. We study twenty-six data sets and, for each, determine the best class distribution for learning. Our results show that, for a fixed number of training examples, it is often possible to obtain improved classifier performance by training with a class distribution other than the naturally occurring class distribution. For example, we show that to build a classifier robust to different misclassification costs, a balanced class distribution generally performs quite well. We also describe and evaluate a budget-sensitive progressive-sampling algorithm that selects training examples such that the resulting training set has a good (near-optimal) class distribution for learning.

1. Introduction

In many real-world situations the amount of training data must be limited, because obtaining data in a form suitable for learning may be costly and/or learning from large amounts of data may be costly. The costs associated with forming a useful training set include the costs of obtaining the raw data, cleaning the data, transporting/storing the data, and transforming the data into a representation suitable for learning. The costs associated with learning from the data involve the cost of computer hardware, the "cost" associated with the time it takes to learn from the data, and the "opportunity cost" associated with suboptimal learning from extremely large data sets due to limited computational resources. Turney (2000) provides a more complete description of these costs.

When the costs of preparing and learning from the data are substantial, so that it is necessary to limit the amount of training data, an important question is: in what proportion should the classes be represented in the training data? Some practitioners believe that the naturally occurring marginal class distribution should be used for learning, so that new examples will be classified using a model built from the same underlying distribution. Other practitioners believe that the training set should contain an increased percentage of minority-class examples, because otherwise the induced classifier will perform poorly at classifying minority-class examples. For example, “if the sample size is fixed, a balanced sample will usually produce more accurate predictions than an unbalanced 5%/95% split” (SAS, 2001). However, we are aware of no thorough empirical study of the relationship between the class distribution of the training set and classifier performance, so neither of these views has been validated and the choice of class distribution often is made arbitrarily—and with little understanding of the consequences. In this article we provide a thorough study of the relationship between class distribution and classifier performance, and provide guidelines and a budget-sensitive progressive-sampling algorithm for determining a proper class distribution to use for learning.

There are two basic situations where the research described in this article is of direct practical use. When the training data must be limited due to the cost of learning from the data, then our results—and the guidelines we establish—can help to determine the class distribution that should be used for the training data. In this case, these guidelines effectively determine how many examples of each class to *omit* from the training set so that the cost of learning is acceptable. The second scenario is where the data are costly to procure so that the amount of training data must be limited. In this case this article describes how to determine the proportion of training examples belonging to each class that should be procured to maximize classifier performance. Note that this assumes that one can select examples belonging to a specific class. There are many real world situations where this is the case. This occurs when 1) the examples belonging to each class come from separate sources, or 2) when the cost of procuring the example is due to the cost of forming a useful training example rather than the cost of obtaining the raw, labeled, data.

Fraud detection (Fawcett & Provost, 1997) provides an example where training instances belonging to each class come from different sources and may be procured independently by class. Typically, after a bill has been paid, any transactions credited as being fraudulent are stored separately from legitimate transactions. Furthermore, as described by Fawcett and Provost, transactions credited to a customer as being fraudulent may in fact have been legitimate, and so these transactions must undergo a verification process before being used as training data.

In other situations labeled raw data can be obtained very cheaply, but it is the process of forming usable training examples from the raw data that is expensive. As an example, consider the *phone* data set, one of the twenty-six data sets analyzed in this article. This data set is constructed from low-level call-detail records that describe a phone call, where each call-detail record includes the originating and terminating phone numbers, the connection time, the day of week and duration of each call. For each phone line, hundreds or even thousands of call-detail records associated with that line must be summarized into a single training example. Billions of call-detail records, covering hundreds of millions of phone lines, potentially are available for learning. Because of the effort associated with loading data from dozens of computer tapes, disk-space limitations, and the enormous processing time required to summarize the raw data, it is not feasible to construct a data set using all available raw data. Consequently, the number of usable training examples to be constructed must be limited. This can be done based on the class associated with each phone line, which is known. Due to the costs outlined above, the phone data set was limited to approximately 650,000 training examples, which were generated from approximately 600 million call-detail records. Given the number of huge transaction-oriented databases that are now being used for learning, combined with the fact that forming usable training examples from these transactions is often expensive, there is often a need to limit the number of usable training examples.

To provide some initial insight to why one might want to modify the class distribution given a fixed training-set size, consider Figure 1, which displays a set of learning curves for one of the data sets analyzed in this article. In addition to showing how the overall accuracy of the classifier varies in response to changes in training-set size, this figure also shows how classifier accuracy varies for the minority-class and majority-class test examples.

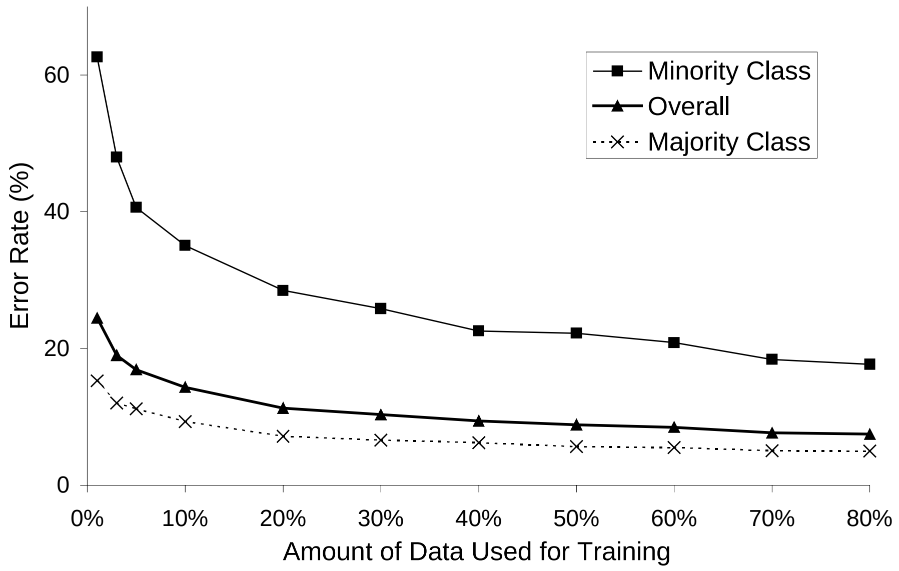


Figure 1: Learning Curves for the Letter-Vowel Data Set

Figure 1 shows us that, while each class certainly is not learned independently, providing additional training examples affects the performance of the minority and majority classes differently (i.e., the slopes of the corresponding learning curves are different). This difference in behavior suggests that adding examples from each class in a proportion that is *different* from the naturally occurring class distribution may lead to improved overall classifier performance.

The remainder of this article is organized as follows. Section 2 introduces terminology that will be used throughout this article. Section 3 describes how to adjust a classifier to compensate for changes made to the class distribution of the training set, so that the generated classifier is not improperly biased. Our experimental methodology and the twenty-six benchmark data sets used in our experiments are described in Section 4. In Section 5 we analyze the performance of the classifiers induced from the twenty-six unbalanced data sets to show how differences in class distribution affect the behavior and performance of the induced classifiers. Section 6 includes our main empirical results, in which we analyze the effect that different class distributions have on classifier performance. In Section 7 we describe a budget-sensitive progressive-sampling algorithm that selects training examples such that the resulting class distribution yields classifiers that generally perform quite well. Related research is described in Section 8 and limitations of our research and future research directions are discussed in Section 9. The main lessons learned from our research are summarized in Section 10.

It should be noted up front that our study utilizes only a decision-tree learner, and therefore, strictly speaking, our conclusions only apply to this class of learners. However, in Section 9, where we discuss this limitation, we describe why our results may hold for other learners as well.

We view this article as making two main contributions. The first contribution is that it addresses the practical problem of how to select the class distribution of the training data when the amount of training data must be limited. The second contribution is that by providing a detailed empirical study of the effect of class distribution on classifier performance, it provides a better understanding of the role of class distribution in learning.

2. Terminology

For simplicity, throughout this article we consider only two-class learning problems. The two classes will be referred to as the positive class, p, and the negative class, n. The minority class will always correspond to the positive class and the majority class will always correspond to the negative class.

Formally, let x be an instance drawn from some fixed distribution D. Every instance x is mapped (perhaps probabilistically) to a class C $\in$ {p, n} by the function c, where c represents the true, but unknown, classification function. Let ρ be the marginal probability of membership of x in the positive class. The marginal probability of membership in the negative class equals $1 - \rho$. These marginal probabilities sometimes are referred to as the "class priors" or the "base rate."

A classifier t is a mapping from instances x to classes {p, n} and is an approximation of c. For notational convenience, let $t(x) \in \{P, N\}$ so that it is always clear whether a class value is an actual (lower case) or predicted (upper case) value. The expected accuracy of a classifier t, α_t, is defined as $\alpha_t = Pr(t(x) = c(x))$, or, equivalently as:

$$\alpha_t = \rho \bullet Pr(t(x) = P \mid c(x) = p) + (1 - \rho) \bullet Pr(t(x) = N \mid c(x) = n) \qquad [1]$$

Many classifiers produce not only a classification, but also estimates of the probability that x will take on each class value. Let $\mathrm{Post}_t(x)$ be classifier t's estimated (posterior) probability that for instance x, $c(x)$=p. Classifiers that produce class-membership probabilities produce a classification by applying a numeric threshold to the posterior probabilities. For example, a threshold value of .5 may be used so that $t(x) = P$ *iff* $\mathrm{Post}_t(x) > .5$; otherwise $t(x) = N$.

A variety of classifiers function by partitioning the input space into a set $\mathcal{L}$ of disjoint regions (a region being defined by a set of potential instances). For example, for a classification tree, the regions are described by conjoining the conditions leading to the leaves of the tree. Each region $L \in \mathcal{L}$ will contain some number of training instances, λ_L. Let λ_{Lp} and λ_{Ln} be the numbers of positive and negative training instances in region L, such that $\lambda_L = \lambda_{Lp} + \lambda_{Ln}$. Such classifiers often estimate $\mathrm{Post}_t(x \mid x \in L)$ as $\lambda_{Lp}/(\lambda_{Lp} + \lambda_{Ln})$ and assign a classification for all instances $x \in L$ based on this estimate and a numeric threshold, as described earlier. Now, let $\mathcal{L}_P$ and $\mathcal{L}_N$ be the sets of regions that predict the positive and negative classes, respectively, such that $\mathcal{L}_P \cup \mathcal{L}_N = \mathcal{L}$. For each region $L \in \mathcal{L}$, t has an associated accuracy, $\alpha_L = Pr(c(x) = t(x) \mid x \in L)$. Let $\alpha_{\mathcal{L}_P}$ represent the expected accuracy for $x \in \mathcal{L}_P$ and $\alpha_{\mathcal{L}_N}$ the expected accuracy for $x \in \mathcal{L}_N$.[1]

3. Correcting for Changes to the Class Distribution of the Training Set

Many classifier induction algorithms assume that the training and test data are drawn from the same fixed, underlying, distribution D. In particular, these algorithms assume that r_{train} and r_{test}, the fractions of positive examples in the training and test sets, approximate ρ, the true "prior" probability of encountering a positive example. These induction algorithms use the estimated class priors based on r_{train}, either implicitly or explicitly, to construct a model and to assign classifications. If the estimated value of the class priors is not accurate, then the posterior probabilities of the model will be improperly biased. Specifically, "increasing the prior probability of a class increases the posterior probability of the class, moving the classification boundary for that class so that more cases are classified into the class" (SAS 2001). Thus, if the training-set data

[1] For notational convenience, we treat $\mathcal{L}_P$ and $\mathcal{L}_N$ as the union of the sets of instances in the corresponding regions.

are selected so that r_{train} does not approximate ρ, then the posterior probabilities should be adjusted based on the differences between ρ and r_{train}. If such a correction is not performed, then the resulting bias will cause the classifier to classify the preferentially sampled class more accurately, but the overall accuracy of the classifier will almost always suffer (we discuss this further in Section 4 and provide the supporting evidence in Appendix A). Thus, it is critical that the classifier be adjusted to eliminate this bias.[2]

In most of the experiments described in this article, the class distribution of the training set is purposefully altered so that r_{train} does not approximate ρ. In the context of our research, the purpose for modifying the class distribution of the training set is to evaluate how this change affects the overall performance of the classifier. We do not want the biased posterior probability estimates to affect our results. In the remainder of this section we describe our method for adjusting the posterior probabilities to account for the difference between r_{train} and ρ. This method, which we reported previously (Weiss and Provost, 2001), is justified informally, using a simple, intuitive, argument. Elkan (2001) presents an equivalent method for adjusting the posterior probabilities, including a formal derivation.

Our experiments utilize a decision tree learner. Differences between r_{train} and ρ will normally result in biased posterior class-probability estimates at the leaves of the decision tree. To remove this bias, we adjust the probability estimates to take these differences into account. Two simple, common probability estimation formulas are listed in Table 1. For each, let λ_{Lp} (λ_{Ln}) represent the number of minority-class (majority-class) training examples at a leaf L of a decision tree (or, more generally, within any partition L). The uncorrected estimates, which are in common use, estimate the probability of seeing a minority-class (positive) example in L. These uncorrected estimates are based on the assumption that the training and test sets are drawn from the same population and therefore both approximate ρ. The uncorrected frequency-based estimate is straightforward and requires no explanation. However, this estimate does not perform well when the sample size, $\lambda_{Lp}+\lambda_{Ln}$, is small—and is not even defined when the sample size is 0. For these reasons the Laplace estimate often is used instead. We consider a version based on the Laplace law of succession (Good, 1965). This probability estimate will always be closer to 0.5 than the frequency-based estimate, but the difference between the two estimates will be negligible for large sample sizes.

Estimate Name	**Uncorrected**	**Corrected**
Frequency-Based	$\lambda_{Lp}/(\lambda_{Lp}+\lambda_{Ln})$	$\lambda_{Lp}/(\lambda_{Lp}+o\,\lambda_{Ln})$
Laplace (law of succession)	$(\lambda_{Lp}+1)/(\lambda_{Lp}+\lambda_{Ln}+2)$	$(\lambda_{Lp}+1)/(\lambda_{Lp}+o\,\lambda_{Ln}+2)$

Table 1: Probability Estimates for Observing a Minority-Class Example

The corrected versions of the estimates in Table 1 account for differences between r_{train} and ρ by factoring in the over-sampling ratio *o*, which measures the degree to which the minority class is over-sampled in the training set relative to the naturally occurring distribution. The value of *o* is computed as the ratio of minority-class examples to majority-class examples in the training set divided by the same ratio in the naturally occurring class distribution. If the ratio of minority to

[2] In situations where it is more costly to misclassify minority-class examples than majority-class examples, practitioners sometimes introduce this bias on purpose. We would argue that in this case a more appropriate approach would be to use a probabilistic or cost-sensitive learning method, so that all available training data can be used for training.

majority examples were 1:2 in the training set and 1:6 in the naturally occurring distribution, then *o* would be 3. A learner can account properly for differences between r_{train} and ρ by using the corrected estimates to calculate the posterior probabilities at L.

As an example, if the ratio of minority-class examples to majority-class examples in the naturally occurring class distribution is 1:5 but the training distribution is modified so that the ratio is 1:1, then *o* is 1.0/0.2, or 5. For L to be labeled with the minority class the probability must be greater than 0.5, so, using the corrected frequency-based estimate, $\lambda_{Lp}/(\lambda_{Lp}+5\lambda_{Ln}) > 0.5$, or, $\lambda_{Lp} > 5\ \lambda_{Ln}$. Thus, L is labeled with the minority class only if it covers *o* times as many minority-class examples as majority-class examples. Note that in calculating *o* above we use the class ratios and not the fraction of examples belonging to the minority class (if we mistakenly used the latter in the above example, then *o* would be one-half divided by one-sixth, or 3). Using the class ratios substantially simplifies the formulas and leads to more easily understood estimates. Elkan (2001) provides a more complicated, but equivalent, formula that uses fractions instead of ratios. In this discussion we assume that a good approximation of the true base rate is known. In some real-world situations this is not true and different methods are required to compensate for changes to the training set (Provost et al. 1998; Saerens et al. 2002).

4. Experimental Setup

In this article we empirically evaluate the effect that the class distribution of the training set has on the performance of learned classification trees. In this section we describe the data sets employed in our study, the sampling strategy used to alter the class distribution of the training data, the induction program used to generate our results, and finally, the metrics used to evaluate the performance of the induced classifiers.

4.1 The Data Sets and the Method for Generating the Training Data

The twenty-six data sets used throughout this article are described in Table 2. This collection includes twenty data sets from the UCI repository (Blake & Merz, 1998), five data sets, identified with a "+", from previously published work by researchers at AT&T (Cohen & Singer, 1999), and one new data set, the phone data set, generated by the authors. The data sets in Table 2 are listed in order of decreasing class imbalance, a convention used throughout this article.

	% Minority	Dataset			% Minority	Dataset
Dataset	Examples	Size	#	Dataset	Examples	Size
letter-a*	3.9	20,000	14	network2	27.9	3,826
pendigits*	8.3	13,821	15	yeast*	28.9	1,484
abalone*	8.7	4,177	16	network1+	29.2	3,577
sick-euthyroid	9.3	3,163	17	car*	30.0	1,728
connect-4*	9.5	11,258	18	german	30.0	1,000
optdigits*	9.9	5,620	19	breast-wisc	34.5	699
covertype*	14.8	581,102	20	blackjack+	35.6	15,000
solar-flare*	15.7	1,389	21	weather+	40.1	5,597
phone	18.2	652,557	22	bands	42.2	538
letter-vowel*	19.4	20,000	23	market1+	43.0	3,181
contraceptive*	22.6	1,473	24	crx	44.5	690
adult	23.9	48,842	25	kr-vs-kp	47.8	3,196
splice-junction*	24.1	3,175	26	move+	49.9	3,029

Table 2: Description of Data Sets

In order to simplify the presentation and the analysis of our results, data sets with more than two classes were mapped to two-class problems. This was accomplished by designating one of the original classes, typically the least frequently occurring class, as the minority class and then mapping the remaining classes into the majority class. The data sets that originally contained more than 2 classes are identified with an asterisk (*) in Table 2. Except for the letter-recognition data set, all of the original data set names are used. The letter-a data set was created from the letter-recognition data set by assigning the examples labeled with the letter "a" to the minority class; the letter-vowel data set was created by assigning the examples labeled with any vowel to the minority class.

In order to investigate the effect that different training-set class distributions have on classifier performance, given a fixed training-set size, we need a method for generating training sets with a variety of class distributions. This is accomplished, for each experimental run, as follows. First, the test set is formed by randomly selecting 25% of the minority-class examples and 25% of the majority-class examples from the original data set, without replacement (the resulting test set therefore conforms to the original class distribution). The remaining data are available for training. To ensure that all experiments for a given data set have the same training-set size—no matter what the class distribution of the training set—the training-set size, S, is made equal to the total number of minority-class examples still available for training (i.e., 75% of the original number). This makes it possible, without replicating any examples, to generate any class distribution for training-set size S. Each training set is then formed by random sampling from the remaining data, without replacement, in a manner such that the desired class distribution is achieved. For the experiments described in this article, the class distribution of the training set is varied so that the minority class accounts for between 2% and 95% of the training data.

4.2 The Induction Program: C4.5

The experiments in this article use C4.5, a program for inducing decision trees from labeled examples (Quinlan, 1993). C4.5 uses the uncorrected frequency-based estimate to label the leaves of the decision tree, since it assumes that the training data approximate the true, underlying distribution. Given that we modify the class distribution of the training set, it is essential that we use the corrected estimates to re-label the leaves of the induced tree. The results presented in the body of this article are based on the use of the corrected versions of the frequency-based and Laplace estimates (described in Table 1), using a probability threshold of .5 to label the leaves of the induced decision trees.

In order to demonstrate the importance of using the corrected estimates, Appendix A presents results comparing decision trees labeled using the uncorrected frequency-based estimate with trees using the corrected frequency-based estimate. This comparison shows that for a particular modification of the class distribution of the training sets (they are modified so that the classes are balanced), using the corrected estimates yields classifiers that substantially outperform classifiers labeled using the uncorrected estimate. In particular, over the twenty-six data sets used in our study, the corrected frequency-based estimate yields a relative reduction in error rate of 10.6%. Furthermore, for only one of the twenty-six data sets does the corrected estimate perform worse. Consequently it is critical to take the differences in the class distributions into account when labeling the leaves. Previous work on modifying the class distribution of the training set (Catlett, 1991; Chan & Stolfo, 1998; Japkowicz, 2002) did not take these differences into account and this undoubtedly affected the results.

C4.5 does not factor in the differences between the class distributions of the training and test sets—we adjust for this as a post-processing step. If C4.5's pruning strategy, which attempts to minimize error rate, were allowed to execute, it would prune based on a false assumption (viz., that the test distribution matches the training distribution). Since this may negatively affect the generated classifier, except where otherwise indicated all results are based on C4.5 without pruning. This decision is supported by recent research, which indicates that when target misclassification costs (or class distributions) are unknown then standard pruning should be avoided (Provost & Domingos, 2001; Zadrozny & Elkan, 2001; Bradford et al, 1998; Bauer & Kohavi, 1999). Indeed, Bradford et al. (1998) found that even if the pruning strategy is adapted to take misclassification costs and class distribution into account, this does not generally improve the performance of the classifier. Nonetheless, in order to justify the validity of the results when using C4.5 without pruning, we also present the results of C4.5 with pruning, when the training set uses the natural distribution. In this situation C4.5's assumption about r_{train} approximating ρ is valid and hence its pruning strategy will perform properly. Looking forward, these results show that in this situation C4.5 without pruning performs competitively with C4.5 with pruning. Based on this we conclude that C4.5 without pruning is a reasonable learner.

4.3 Evaluating Classifier Performance

We now consider a variety of metrics for assessing classifier performance with respect to the positive (minority) and negative (majority) classes. These metrics are based upon the terms listed in the confusion matrix, shown below.

		$t(x)$	
		Positive Prediction	Negative Prediction
$c(x)$	Actual Positive	*tp* (true positive)	*fn* (false negative)
	Actual Negative	*fp* (false positive)	*tn* (true negative)

Some of the classifier performance metrics used in this article are described in Table 3. The metrics described in the first two rows of Table 3 measure the ability of a classifier to classify positive and negative *examples* correctly and are often incorporated into other metrics to assess the overall effectiveness of a classifier. The metrics described in the last two rows of Table 3 measure the effectiveness of the *predictions* made by a classifier. For example, the positive predictive value (PPV), or precision, of a classifier measures the fraction of positive predictions that are correctly classified.

$\text{TP} = Pr(\text{P}\vert\text{p}) \approx \frac{tp}{tp+fn}$	*True Positive Rate (recall or sensitivity)*	$\text{FN} = Pr(\text{N}\vert\text{p}) \approx \frac{fn}{tp+fn}$	*False Negative Rate*
$\text{TN} = Pr(\text{N}\vert\text{n}) \approx \frac{tn}{tn+fp}$	*True Negative Rate (specificity)*	$\text{FP} = Pr(\text{P}\vert\text{n}) \approx \frac{fp}{tn+fp}$	*False Positive Rate*
$\text{PPV} = Pr(\text{p}\vert\text{P}) \approx \frac{tp}{tp+fp}$	*Positive Predictive Value (precision)*	$\overline{PPV} = Pr(\text{n}\vert\text{P}) \approx \frac{fp}{tp+fp}$	
$\text{NPV} = Pr(\text{n}\vert\text{N}) \approx \frac{tn}{tn+fn}$	*Negative Predictive Value*	$\overline{NPV} = Pr(\text{y}\vert\text{N}) \approx \frac{fn}{tn+fn}$	

Table 3: Classifier Performance Metrics

The metrics described in the last two rows of Table 3 are used throughout this article to evaluate how various training-set class distributions affect the predictions made by the induced classifiers. This is important because we wish to understand not only how the class distribution of the training data affects the ability of a classifier to label test examples correctly, but also how it affects the "behavior" of the classifiers. Finally, the metrics in the second column of Table 3 are "complements" of the corresponding metrics in the first column, and can alternatively be computed by subtracting the value in the first column from 1. Note that in two cases we are aware of no generally accepted terms—these metrics will be referred to using the complement of the names that appear in the first column (e.g., $\overline{PPV}$ and $\overline{NPV}$).

The metrics in Table 3 can be summarized intuitively as follows. Proceeding from row 1 through 4, the metrics in column 1 (column 2) represent: 1) the accuracy (error rate) when classifying positive/minority examples, 2) the accuracy (error rate) when classifying negative/minority examples, 3) the accuracy (error rate) of the positive/minority predictions, and 4) the accuracy (error rate) of the negative/majority predictions. When describing the results in Section 5 we use the terms in column 2 rather than those in column 1, since it often is more revealing to compare error rates than accuracies (e.g., an error rate of 2% versus 1% corresponds to a relative increase in error rate of 100% but a relative decrease in accuracy of only about 1%).

We use two performance measures in this article to gauge the *overall* performance of a classifier: classification accuracy and AUC (the area under the ROC curve). Classification accuracy is defined as $(tp + fp)/(tp + fp + tn + fn)$. This formula, which represents the fraction of test examples that are correctly classified, is an estimate of the expected accuracy, α_t, previously defined in equation 1. Throughout this article we specify classification accuracy in terms of error rate, which is defined as 1 – accuracy.

We consider classification accuracy in part because it is the most common evaluation metric in machine-learning research. However, using accuracy as a performance measure assumes that the target (marginal) class distribution is known and unchanging and, more importantly, that the error costs—the costs of a false positive and false negative—are equal. These assumptions are unrealistic in many domains (Provost et al. 1998). Accuracy is particularly suspect as a performance measure when studying the effect of class distribution on learning since, as we discuss in Section 5.2, it is heavily biased to favor the majority class. Furthermore, highly unbalanced data sets typically have highly non-uniform error costs that favor the minority class, which, as in the case of medical diagnosis and fraud detection, is the class of primary interest. Classifiers that optimize for accuracy for these problems are of questionable value since they rarely predict the minority class.

An alternative method for evaluating classifier performance is Receiver Operating Characteristic (ROC) analysis (Swets et al. 2000), which represents the false-positive rate (FP) on the x-axis of a graph and the true-positive rate (TP) on the y-axis. ROC *curves* are produced by varying the threshold on a classification model's numeric output—in our case by varying the threshold on the class-probability estimates at the leaves of a decision tree. For example, one point on an ROC curve may correspond to the case where the leaves of the decision tree are labeled with the minority class only if the probability of an example at a leaf belonging to the minority class is greater than .5; another point on the curve may correspond to a probability threshold of .1. We use the Laplace estimate to estimate the probabilities at the leaves for generating ROC curves because it has been shown to yield consistent improvements (Provost & Domingos, 2001). The use of ROC analysis for machine learning is described in detail elsewhere (Bradley 1997; Provost & Fawcett, 2001). One advantage of ROC analysis over accuracy is that it evaluates the performance of a classifier independent of the class distribution of the test set or the error costs. Several ROC curves are displayed in Section 6.3.

To assess the *overall* quality of a classifier we measure the fraction of the total area that falls under the ROC curve, which is equivalent to several other statistical measures for evaluating classification and ranking models (Hand, 1997). Larger AUC values indicate generally better classifier performance and, in particular, indicate a better ability to rank cases by likelihood of class membership. It should be kept in mind that if one ROC curve does not dominate the rest, then for *specific* cost and class distributions the best model may not be the one that maximizes AUC. If there is not a single dominating ROC curve, multiple classifiers can be combined to form a classifier that performs optimally for all costs and distributions (Provost & Fawcett, 2001).

5. Learning from Unbalanced Data Sets

We now are ready to analyze the classifiers induced from the twenty-six naturally unbalanced data sets described in Table 2. In this section we focus on the differences between the performance of the minority and majority classes. Before addressing these differences, it is important to discuss an issue that may lead to confusion if left untreated. Practitioners have noted that learning performance often is unsatisfactory when learning from data sets where the minority class is considerably underrepresented. In particular, they observe that there is a large error rate for the minority class. As should be clear from Table 3 and the associated discussion, there are two different notions of "error rate for the minority class": the minority-class predictions could have a high error rate (large $\overline{PPV}$) or the minority-class test examples could have a high error rate (large FN). When practitioners observe that the error rate is unsatisfactory for the minority class, they are usually referring to the fact that the minority-class *examples* have a high error rate (large FN). The analysis in this section will show that the error rate associated with the minority-class predictions and the minority-class test examples are both much larger than their majority class counterparts. We discuss several explanations for these observed differences.

5.1 Experimental Results

The performances of the classifiers induced from the twenty-six "unbalanced" data sets are described in Table 4. This table warrants some explanation. The first column specifies the data set name while the second column, which for convenience has been copied from Table 2, specifies the percentage of minority-class examples in natural class distribution. The third column specifies the percentage of the total test errors that can be attributed to the test examples that belong to the minority class. By comparing the values in columns two and three we see that in all cases a disproportionately large percentage of the errors come from the minority-class examples. For example, minority-class examples make up only 3.9% of the letter-a data set but contribute 58.3% of the errors. Furthermore, for 22 of 26 data sets a *majority* of the errors can be attributed to minority-class examples.

Dataset	% Minority Examples	% Errors from Min.	Leaves		Coverage		Prediction ER		Actuals ER	
			Min.	Maj.	Min.	Maj.	Min.	Maj.	Min.	Maj.
							($\overline{\text{PPV}}$)	($\overline{\text{NPV}}$)	(FN)	(FP)
letter-a	3.9	58.3	11	138	2.2	4.3	32.5	1.7	41.5	1.2
pendigits	8.3	32.4	6	8	16.8	109.3	25.8	1.3	14.3	2.7
abalone	8.7	68.9	5	8	2.8	35.5	69.8	7.7	84.4	3.6
sick-euthyroid	9.3	51.2	4	9	7.1	26.9	22.5	2.5	24.7	2.4
connect-4	9.5	51.4	47	128	1.7	5.8	55.8	6.0	57.6	5.7
optdigits	9.9	73.0	15	173	2.9	2.4	18.0	3.9	36.7	1.5
covertype	14.8	16.7	350	446	27.3	123.2	23.1	1.0	5.7	4.9
solar-flare	15.7	64.4	12	48	1.7	3.1	67.8	13.7	78.9	8.1
phone	18.2	64.4	1008	1220	13.0	62.7	30.8	9.5	44.6	5.5
letter-vowel	19.4	61.8	233	2547	2.4	0.9	27.0	8.7	37.5	5.6
contraceptive	22.6	48.7	31	70	1.8	2.8	69.8	20.1	68.3	21.1
adult	23.9	57.5	627	4118	3.1	1.6	34.3	12.6	41.5	9.6
splice-junction	24.1	58.9	26	46	5.5	9.6	15.1	6.3	20.3	4.5
network2	27.9	57.1	50	61	4.0	10.3	48.2	20.4	55.5	16.2
yeast	28.9	58.9	8	12	14.4	26.1	45.6	20.9	55.0	15.6
network1	29.2	57.1	42	49	5.1	12.8	46.2	21.0	53.9	16.7
car	30.0	58.6	38	42	3.1	6.6	14.0	7.7	18.6	5.6
german	30.0	55.4	34	81	2.0	2.0	57.1	25.4	62.4	21.5
breast-wisc	34.5	45.7	5	5	12.6	26.0	11.4	5.1	9.8	6.1
blackjack	35.6	81.5	13	19	57.7	188.0	28.9	27.9	64.4	8.1
weather	40.1	50.7	134	142	5.0	7.2	41.0	27.7	41.7	27.1
bands	42.2	91.2	52	389	1.4	0.3	17.8	34.8	69.8	4.9
market1	43.0	50.3	87	227	5.1	2.7	30.9	23.4	31.2	23.3
crx	44.5	51.0	28	65	3.9	2.1	23.2	18.9	24.1	18.5
kr-vs-kp	47.8	54.0	23	15	24.0	41.2	1.2	1.3	1.4	1.1
move	49.9	61.4	235	1025	2.4	0.6	24.4	29.9	33.9	21.2
Average	25.8	56.9	120	426	8.8	27.4	33.9	13.8	41.4	10.1
Median	26.0	57.3	33	67	3.9	6.9	29.9	11.1	41.5	5.9

Table 4: Behavior of Classifiers Induced from Unbalanced Data Sets

The fourth column specifies the number of leaves labeled with the minority and majority classes and shows that in all but two cases there are fewer leaves labeled with the minority class than with the majority class. The fifth column, “Coverage,” specifies the average number of training examples that each minority-labeled or majority-labeled leaf classifies (“covers”). These results indicate that the leaves labeled with the minority class are formed from far fewer training examples than those labeled with the majority class.

The “Prediction ER” column specifies the error rates associated with the minority-class and majority-class predictions, based on the performance of these predictions at classifying the test examples. The “Actuals ER” column specifies the classification error rates for the minority and majority class examples, again based on the test set. These last two columns are also labeled using the terms defined in Section 2 ($\overline{\text{PPV}}$, $\overline{\text{NPV}}$, FN, and FP). As an example, these columns show that for the letter-a data set the minority-labeled predictions have an error rate of 32.5% while the majority-labeled predictions have an error rate of only 1.7%, and that the minority-class test examples have a classification error rate of 41.5% while the majority-class test examples have an error rate of only 1.2%. In each of the last two columns we underline the higher error rate.

The results in Table 4 clearly demonstrate that the minority-class predictions perform much worse than the majority-class predictions and that the minority-class examples are misclassified much more frequently than majority-class examples. Over the twenty-six data sets, the minority predictions have an average error rate ($\overline{\text{PPV}}$) of 33.9% while the majority-class predictions have an average error rate ($\overline{\text{NPV}}$) of only 13.8%. Furthermore, for only three of the twenty-six data sets do the majority-class predictions have a higher error rate—and for these three data sets the class distributions are only slightly unbalanced. Table 4 also shows us that the average error rate for the minority-class test examples (FN) is 41.4% whereas for the majority-class test examples the error rate (FP) is only 10.1%. In every one of the twenty-six cases the minority-class test examples have a higher error rate than the majority-class test examples.

5.2 Discussion

So, why do the minority-class predictions have a higher error rate ($\overline{\text{PPV}}$) than the majority-class predictions ($\overline{\text{NPV}}$)? There are at least two reasons. First, consider a classifier t_{random} where the partitions $\mathcal{L}$ are chosen randomly and the assignment of each $L \in \mathcal{L}$ to $\mathcal{L}_P$ and $\mathcal{L}_N$ is also made randomly (recall that $\mathcal{L}_P$ and $\mathcal{L}_N$ represent the regions labeled with the positive and negative classes). For a two-class learning problem the expected overall accuracy, α_t, of this randomly generated and labeled classifier must be 0.5. However, the expected accuracy of the regions in the positive partition, $\alpha_{\mathcal{L}_P}$, will be ρ while the expected accuracy of the regions in the negative partition, $\alpha_{\mathcal{L}_N}$, will be $1 - \rho$. For a highly unbalanced class distribution where ρ=.01, $\alpha_{\mathcal{L}_P} = .01$ and $\alpha_{\mathcal{L}_N} = .99$. Thus, in such a scenario the negative/majority predictions will be much more "accurate." While this "test distribution effect" will be small for a well-learned concept with a low Bayes error rate (and non-existent for a perfectly learned concept with a Bayes error rate of 0), many learning problems are quite hard and have high Bayes error rates.[3]

The results in Table 4 suggest a second explanation for why the minority-class predictions are so error prone. According to the coverage results, minority-labeled predictions tend to be formed from fewer training examples than majority-labeled predictions. *Small disjuncts*, which are the components of disjunctive concepts (i.e., classification rules, decision-tree leaves, etc.) that cover few training examples, have been shown to have a much higher error rate than large disjuncts (Holte, et al., 1989; Weiss & Hirsh, 2000). Consequently, the rules/leaves labeled with the minority class have a higher error rate because they suffer more from this "problem of small disjuncts."

Next, why are minority-class examples classified incorrectly much more often than majority-class examples (FN > FP)—a phenomenon that has also been observed by others (Japkowicz & Stephen, 2002)? Consider the estimated accuracy, a_t, of a classifier t, where the test set is drawn from the true, underlying distribution D:

$$a_t = \text{TP} \cdot r_{\text{test}} + \text{TN} \cdot (1 - r_{\text{test}}) \qquad [2]$$

[3] The (optimal) Bayes error rate, using the terminology from Section 2, occurs when t(.)=c(.). Because c(.) may be probabilistic (e.g., when noise is present), the Bayes error rate for a well-learned concept may not always be low. The test distribution effect will only be small when the concept is well learned *and* the Bayes error rate is low.

Since the positive class corresponds to the minority class, $r_{\text{test}} < .5$ and for highly unbalanced data sets $r_{\text{test}} << .5$. Therefore, false-positive errors are more damaging to classification accuracy than false negative errors are. A classifier that is induced using an induction algorithm geared toward maximizing accuracy therefore should "prefer" false-negative errors over false-positive errors. This will cause negative/majority examples to be predicted more often and hence will lead to a higher error rate for minority-class examples. One straightforward example of how learning algorithms exhibit this behavior is provided by the common-sense rule: if there is no evidence favoring one classification over another, then predict the majority class. More generally, induction algorithms that maximize accuracy should be biased to perform better at classifying majority-class examples than minority-class examples, since the former component is weighted more heavily when calculating accuracy. This also explains why, when learning from data sets with a high degree of class imbalance, classifiers rarely predict the minority class.

A second reason why minority-class examples are misclassified more often than majority-class examples is that fewer minority-class examples are likely to be sampled from the distribution *D*. Therefore, the training data are less likely to include (enough) instances of all of the minority-class subconcepts in the concept space, and the learner may not have the opportunity to represent all truly positive regions in $\mathcal{L}_P$. Because of this, some minority-class test examples will be mistakenly classified as belonging to the majority class.

It is worth noting that $\overline{\text{PPV}} > \overline{\text{NPV}}$ does not imply that FN > FP. That is, having more error-prone minority predictions does *not* imply that the minority-class examples will be misclassified more often than majority-class examples. Indeed, a higher error rate for minority predictions means more majority-class test examples will be misclassified. The reason we generally observe a lower error rate for the majority-class test examples (FN > FP) is because the majority class is predicted far more often than the minority class.

Finally, we repeated these experiments using C4.5 with pruning. The results showed the same general trends but the magnitudes of the differences were altered. With pruning the error rate associated with the minority-class predictions ($\overline{\text{PPV}}$) decreases dramatically while the average error rate of the majority-class predictions ($\overline{\text{NPV}}$) decreases only slightly, so that even though $\overline{\text{PPV}} > \overline{\text{NPV}}$, the magnitude of the difference is cut in half (on average). Pruning causes the error rate for the minority-class test examples (FN) to increase and the error rate for the majority-class test examples (FP) to decrease, so that the average difference becomes slightly larger. Both of these differences can be explained by the effect that pruning has on small disjuncts. According to Weiss and Hirsh (2000), "pruning eliminates most of the small disjuncts and many of the emancipated examples are then classified by the larger disjuncts." Because minority-class leaves tend to have smaller-sized disjuncts, pruning tends to eliminate minority-labeled leaves disproportionately. This further increases the error rate for the minority-class test examples and decreases the error rate of the majority-class test examples (since an increased number of examples will be classified as belonging to the majority class). The error rate of the minority-class predictions is reduced because the most error-prone of these predictions, which tend to be associated with the smallest disjuncts, are eliminated.

6. The Effect of Training-Set Class Distribution on Classifier Performance

Let us now address the question of which class distribution is best for training. We begin by describing our methodology for determining which class distribution is best. Then, in the next two sections, we evaluate and analyze classifier performance for the twenty-six data sets using a variety of class distributions (we use error rate as the performance metric in Section 6.2 and AUC as the performance metric in Section 6.3).

6.1 Methodology for Determining the Optimum Training Class Distribution(s)

In order to evaluate the effect of class distribution on classifier performance, we vary the training-set class distributions for the twenty-six data sets using the methodology described in Section 4.1. We evaluate the following twelve class distributions (expressed as the percentage of minority-class examples): 2%, 5%, 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90%, and 95%. For each data set we also evaluate the performance using the naturally occurring class distribution.

Before we try to determine the "best" class distribution for a training set, there are several issues that must be addressed. First, because we do not evaluate every possible class distribution, we can only determine the best distribution among the 13 evaluated distributions. Beyond this concern, however, is the issue of statistical significance and, because we generate classifiers for 13 training distributions, the issue of multiple comparisons (Jensen & Cohen, 2000). Because of these issues we cannot always conclude that the distribution that yields the best performing classifiers is truly the best one for training.

We take several steps to address the issues of statistical significance and multiple comparisons. To enhance our ability to identify true differences in classifier performance with respect to changes in class distribution, all results presented in this section are based on 30 runs, rather than the 10 runs employed in Section 5. Also, rather than trying to determine *the* best class distribution, we adopt a more conservative approach, and instead identify an "optimal range" of class distributions—a range in which we are confident the best distribution lies. To identify the optimal range of class distributions, we begin by identifying, for each data set, the class distribution that yields the classifiers that perform best over the 30 runs. We then perform t-tests to compare the performance of these 30 classifiers with the 30 classifiers generated using each of the other twelve class distributions (i.e., 12 t-tests each with n=30 data points). If a t-test yields a probability $\leq .10$ then we conclude that the "best" distribution is different from the "other" distribution (i.e., we are at least 90% confident of this); otherwise we cannot conclude that the class distributions truly perform differently and therefore "group" the distributions together. These grouped distributions collectively form the "optimal range" of class distributions. As can be seen in the two tables that follow, in 50 of 52 cases the optimal ranges are contiguous, assuaging concerns that our conclusions are due to problems of multiple comparisons.

6.2 The Relationship between Class Distribution and Classification Error Rate

Table 5 displays the error rates of the classifiers induced for each of the twenty-six data sets using the class distributions specified in Section 6.1. The first column in Table 5 specifies the data set name and the next two columns specify the error rates that result from using the natural distribution, with and then without pruning. The next 12 columns present the error rate values for the 12 fixed class distributions (without pruning). For each data set, the "best" distribution (i.e., the one with the lowest error rate) is highlighted by underlining it and displaying it in boldface. The relative position of the natural distribution within the range of evaluated class distributions is denoted by the use of a vertical bar between columns. For example, for the letter-a data set the vertical bar indicates that the natural distribution falls between the 2% and 5% distributions (from Table 2 we see it is 3.9%).

Dataset	Error Rate when using Specified Training Distribution (training distribution expressed as % minority)															Relative % Improvement	
	Nat-Prune	Nat	2	5	10	20	30	40	50	60	70	80	90	95	best vs. nat	best vs. bal
letter-a	2.80 x	2.78	2.86	2.75	**2.59**	3.03	3.79	4.53	5.38	6.48	8.51	12.37	18.10	26.14	6.8	**51.9**
pendigits	3.65 +	3.74	5.77	3.95	3.63	**3.45**	3.70	3.64	4.02	4.48	4.98	5.73	8.83	13.36	7.8	**14.2**
abalone	10.68 x	10.46	**9.04**	9.61	10.64	13.19	15.33	20.76	22.97	24.09	26.44	27.70	27.73	33.91	13.6	**60.6**
sick-euthyroid	4.46 x	**4.10**	5.78	4.82	4.69	4.25	5.79	6.54	6.85	9.73	12.89	17.28	28.84	40.34	0.0	**40.1**
connect-4	10.68 x	10.56	**7.65**	8.66	10.80	15.09	19.31	23.18	27.57	33.09	39.45	47.24	59.73	72.08	**27.6**	**72.3**
optdigits	4.94 x	4.68	8.91	7.01	4.05	3.05	2.83	**2.79**	3.41	3.87	5.15	5.75	9.72	12.87	**40.4**	**18.2**
covertype	5.12 x	5.03	5.54	5.04	**5.00**	5.26	5.64	5.95	6.46	7.23	8.50	10.18	13.03	16.27	0.6	**22.6**
solar-flare	19.16 +	19.98	**16.54**	17.52	18.96	21.45	23.03	25.49	29.12	30.73	33.74	38.31	44.72	52.22	**17.2**	**43.2**
phone	12.63 x	12.62	13.45	12.87	**12.32**	12.68	13.25	13.94	14.81	15.97	17.32	18.73	20.24	21.07	**2.4**	**16.8**
letter-vowel	11.76 x	**11.63**	15.87	14.24	12.53	11.67	12.00	12.69	14.16	16.00	18.68	23.47	32.20	41.81	0.0	**17.9**
contraceptive	31.71 x	30.47	**24.09**	24.57	25.94	30.03	32.43	35.45	39.65	43.20	47.57	54.44	62.31	67.07	**20.9**	**39.2**
adult	17.42 x	17.25	18.47	17.26	**16.85**	17.09	17.78	18.85	20.05	21.79	24.08	27.11	33.00	39.75	**2.3**	**16.0**
splice-junction	8.30 +	8.37	20.00	13.95	10.72	8.68	8.50	**8.15**	8.74	9.86	9.85	12.08	16.25	21.18	2.6	6.8
network2	27.13 x	26.67	27.37	25.91	25.71	**25.66**	26.94	28.65	29.96	32.27	34.25	37.73	40.76	37.72	3.8	**14.4**
yeast	26.98 x	26.59	29.08	28.61	27.51	**26.35**	26.93	27.10	28.80	29.82	30.91	35.42	35.79	36.33	0.9	**8.5**
network1	27.57 +	27.59	27.90	27.43	26.78	**26.58**	27.45	28.61	30.99	32.65	34.26	37.30	39.39	41.09	3.7	**14.2**
car	9.51 x	8.85	23.22	18.58	14.90	10.94	8.63	8.31	7.92	**7.35**	7.79	8.78	10.18	12.86	**16.9**	7.2
german	33.76 x	33.41	**30.17**	30.39	31.01	32.59	33.08	34.15	37.09	40.55	44.04	48.36	55.07	60.99	**9.7**	**18.7**
breast-wisc	7.41 x	6.82	20.65	14.04	11.00	8.12	7.49	6.82	**6.74**	7.30	6.94	7.53	10.02	10.56	1.2	0.0
blackjack	28.14 +	**28.40**	30.74	30.66	29.81	28.67	28.56	28.45	28.71	28.91	29.78	31.02	32.67	33.87	0.0	1.1
weather	33.68 +	33.69	38.41	36.89	35.25	33.68	**33.11**	33.43	34.61	36.69	38.36	41.68	47.23	51.69	**1.7**	**4.3**
bands	32.26 +	32.53	38.72	35.87	35.71	34.76	33.33	**32.16**	32.68	33.91	34.64	39.88	40.98	40.80	1.1	1.6
market1	26.71 x	26.16	34.26	32.50	29.54	26.95	26.13	26.05	**25.77**	26.86	29.53	31.69	36.72	39.90	1.5	0.0
crx	20.99 x	**20.39**	35.99	30.86	27.68	23.61	20.84	20.82	21.48	21.64	22.20	23.98	28.09	32.85	0.0	**5.1**
kr-vs-kp	1.25 +	1.39	12.18	6.50	3.20	2.33	1.73	**1.16**	1.22	1.34	1.53	2.55	3.66	6.04	16.5	4.9
move	27.54 +	28.57	46.13	42.10	38.34	33.48	30.80	28.36	**28.24**	29.33	30.21	31.80	36.08	40.95	1.2	0.0

Table 5: Effect of Training Set Class Distribution on Error Rate

The error rate values that are not significantly different, statistically, from the lowest error rate (i.e., the comparison yields a t-test value > .10) are shaded. Thus, for the letter-a data set, the optimum range includes those class distributions that include between 2% and 10% minority-class examples—which includes the natural distribution. The last two columns in Table 5 show the relative improvement in error rate achieved by using the best distribution instead of the natural and balanced distributions. When this improvement is statistically significant (i.e., is associated with a t-test value ≤ .10) then the value is displayed in bold.

The results in Table 5 show that for 9 of the 26 data sets we are confident that the natural distribution is not within the range of optimal class distributions. For most of these 9 data sets, using the best distribution rather than the natural distribution yields a remarkably large decrease in error rate. We feel that this is sufficient evidence to conclude that for accuracy, when the training-set size must be limited, it is not appropriate simply to assume that the natural distribution should be used. Inspection of the error-rate results in Table 5 also shows that the best distribution does not differ from the natural distribution in any consistent manner—sometimes it includes more minority-class examples (e.g., optdigits, car) and sometimes fewer (e.g., connect-4, solar-flare). However, it is clear that for data sets with a substantial amount of class imbalance (the ones in the top half of the table), a balanced class distribution also is not the best class distribution for training, to minimize undifferentiated error rate. More specifically, none of the top-12 most skewed data sets have the balanced class distribution within their respective optimal ranges, and for these data sets the relative improvements over the balanced distributions are striking.

Let us now consider the error-rate values for the remaining 17 data sets for which the t-test results do not permit us to conclude that the best observed distribution truly outperforms the natural distribution. In these cases we see that the error rate values for the 12 training-set class distributions usually form a unimodal, or nearly unimodal, distribution. This is the distribution one would expect if the accuracy of a classifier progressively degrades the further it deviates from the best distribution. This suggests that "adjacent" class distributions may indeed produce classifiers that perform differently, but that our statistical testing is not sufficiently sensitive to identify these differences. Based on this, we suspect that many of the observed improvements shown in the last column of Table 5 that are not deemed to be significant statistically are nonetheless meaningful.

Figure 2 shows the behavior of the learned classifiers for the adult, phone, covertype, and letter-a data sets in a graphical form. In this figure the natural distribution is denoted by the "X" tick mark and the associated error rate is noted above the marker. The error rate for the best distribution is underlined and displayed below the corresponding data point (for these four data sets the best distribution happens to include 10% minority-class examples).

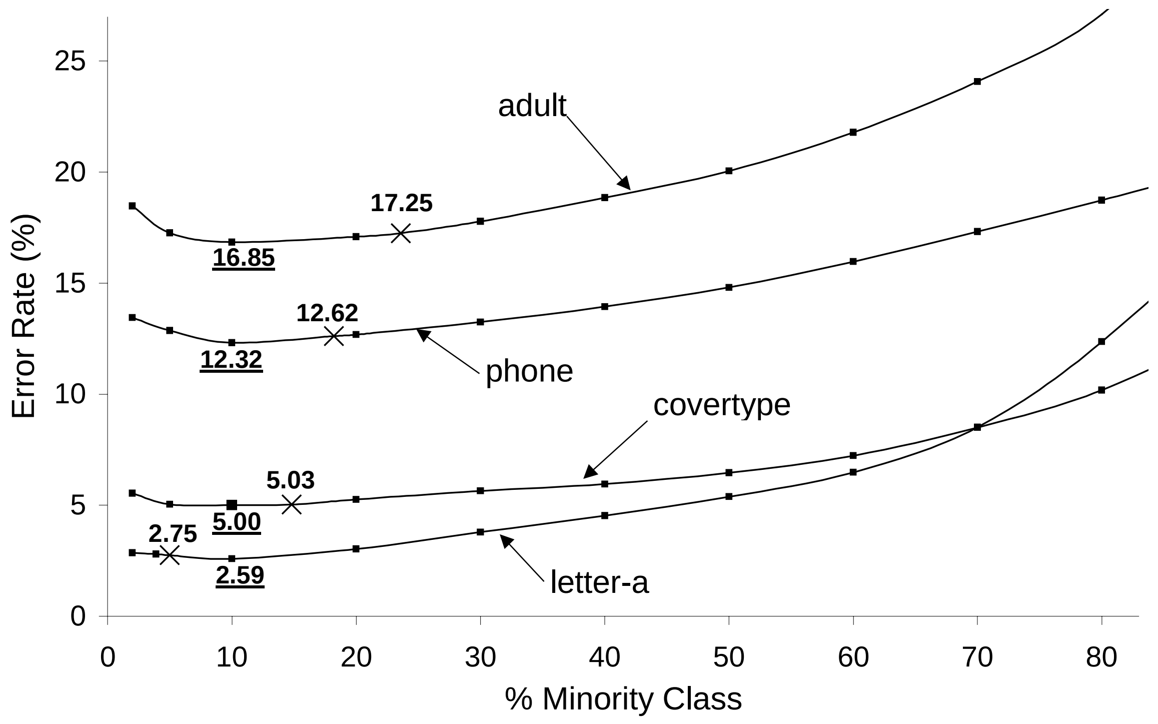


Figure 2: Effect of Class Distribution on Error Rate for Select Data Sets

Note that two of the curves are associated with data sets (adult, phone) for which we are >90% confident that the best distribution performs better than the natural distribution, while for the other two curves (covertype, letter-a) we are not. Note that all four curves are perfectly unimodal. It is also clear that near the distribution that minimizes error rate, changes to the class distribution yield only modest changes in the error rate—far more dramatic changes occur elsewhere. This is also evident for most data sets in Table 5. This is a convenient property given the common goal of minimizing error rate. This property would be far less evident if the correction described in Section 3 were not performed, since then classifiers induced from class distributions deviating from the naturally occurring distribution would be improperly biased.

Finally, to assess whether pruning would have improved performance, consider the second column in Table 5, which displays the error rates that result from using C4.5 with pruning on the natural distribution (recall from Section 4.2 that this is the only case when C4.5's pruning strategy will give unbiased results). A "+"/"x" in the second column indicates that C4.5 with pruning outperforms/underperforms C4.5 without pruning, when learning from the natural distribution. Note that C4.5 with pruning *underperforms* C4.5 without pruning for 17 of the 26 data sets, which leads us to conclude that C4.5 without pruning is a reasonable learner. Furthermore, in no case does C4.5 with pruning generate a classifier within the optimal range when C4.5 without pruning does not also generate a classifier within this range.

6.3 The Relationship between Class Distribution and AUC

The performance of the induced classifiers, using AUC as the performance measure, is displayed in Table 6. When viewing these results, recall that for AUC *larger* values indicate improved performance. The relative improvement in classifier performance is again specified in the last two columns, but now the relative improvement in performance is calculated in terms of the area above the ROC curve (i.e., 1 – AUC). We use the area above the ROC curve because it better reflects the relative improvement—just as in Table 5 relative improvement is specified in terms of the change in error rate instead of the change in accuracy. As before, the relative improvements are shown in bold only if we are more than 90% confident that they reflect a true improvement in performance (i.e., t-test value $\leq .10$).

In general, the optimum ranges appear to be centered near, but slightly to the right, of the balanced class distribution. For 12 of the 26 data sets the optimum range does not include the natural distribution (i.e., the third column is not shaded). Note that for these data sets, with the exception of the solar-flare data set, the class distributions within the optimal range contain more minority-class examples than the natural class distribution. Based on these results we conclude even more strongly for AUC (i.e., for cost-sensitive classification and for ranking) than for accuracy that it is not appropriate simply to choose the natural class distribution for training. Table 6 also shows that, unlike for accuracy, a balanced class distribution generally performs very well, although it does not always perform optimally. In particular, we see that for 19 of the 26 data sets the balanced distribution is within the optimal range. This result is not too surprising since AUC, unlike error rate, is unaffected by the class distribution of the test set, and effectively factors in classifier performance over *all* class distributions.

Dataset	AUC when using Specified Training Distribution (training distribution expressed as % minority)														Relative % Improv. (1-AUC)	
	Nat-prune	Nat	2	5	10	20	30	40	50	60	70	80	90	95	best vs. nat	best vs. bal
letter-a	.500 x	.772	.711	.799	.865	.891	.911	.938	.937	.944	.951	**.954**	.952	.940	**79.8**	**27.0**
pendigits	.962 x	.967	.892	.958	.971	.976	.978	**.979**	**.979**	.978	.977	.976	.966	.957	36.4	0.0
abalone	.590 x	.711	.572	.667	.710	.751	.771	.775	.776	**.778**	.768	.733	.694	.687	**25.8**	0.9
sick-euthyroid	.937 x	.940	.892	.908	.933	.943	.944	.949	.952	.951	**.955**	.945	.942	.921	**25.0**	6.3
connect-4	.658 x	.731	.664	.702	.724	.759	.763	.777	.783	**.793**	**.793**	.789	.772	.730	**23.1**	4.6
optdigits	.659 x	.803	.599	.653	.833	.900	.924	.943	.948	.959	.967	.965	**.970**	.965	**84.8**	**42.3**
covertype	.982 x	**.984**	.970	.980	**.984**	**.984**	.983	.982	.980	.978	.976	.973	.968	.960	0.0	**20.0**
solar-flare	.515 x	.627	.614	.611	.646	.627	.635	.636	.632	.650	**.662**	.652	.653	.623	**9.4**	**8.2**
phone	.850 x	.851	.843	.850	.852	.851	.850	.850	.849	.848	.848	.850	**.853**	.850	1.3	2.6
letter-vowel	.806 +	.793	.635	.673	.744	.799	.819	.842	.849	.861	**.868**	**.868**	.858	.833	**36.2**	**12.6**
contraceptive	.539 x	.611	.567	.613	.617	.616	.622	.640	.635	.635	.640	**.641**	.627	.613	**7.7**	1.6
adult	.853 +	.839	.816	.821	.829	.836	.842	.846	.851	.854	.858	**.861**	**.861**	.855	**13.7**	**6.7**
splice-junction	.932 +	.905	.814	.820	.852	.908	.915	.925	.936	.938	.944	**.950**	.944	.944	**47.4**	21.9
network2	.712 +	.708	.634	.696	.703	.708	.705	.704	.705	.702	.706	.710	**.719**	.683	3.8	4.7
yeast	.702 x	.705	.547	.588	.650	.696	**.727**	.714	.720	.723	.715	.699	.659	.621	10.9	2.5
network1	.707 +	.705	.626	.676	.697	.709	.709	.706	.702	.704	.708	**.713**	.709	.696	2.7	3.7
car	.931 +	.879	.754	.757	.787	.851	.884	.892	.916	.932	.931	**.936**	.930	.915	**47.1**	**23.8**
german	.660 +	.646	.573	.600	.632	.615	.635	**.654**	.645	.640	.650	.645	.643	.613	2.3	2.5
breast-wisc	.951 x	.958	.876	.916	.940	.958	.963	**.968**	.966	.963	.963	.964	.949	.948	23.8	5.9
blackjack	.682 x	.700	.593	.596	.628	.678	.688	.712	.713	**.715**	.700	.678	.604	.558	**5.0**	0.7
weather	.748 +	.736	.694	.715	.728	.737	.738	**.740**	.736	.730	.736	.722	.718	.702	1.5	1.5
bands	.604 x	**.623**	.522	.559	.564	.575	.599	.620	.618	.604	.601	.530	.526	.536	0.0	1.3
market1	.815 +	.811	.724	.767	.785	.801	.810	.808	.816	**.817**	.812	.805	.795	.781	3.2	0.5
crx	.889 +	.852	.804	.799	.805	.817	.834	.843	.853	.845	.857	.848	.853	**.866**	9.5	8.8
kr-vs-kp	.996 x	.997	.937	.970	.991	.994	.997	**.998**	**.998**	**.998**	.997	.994	.988	.982	33.3	0.0
move	.762 +	.734	.574	.606	.632	.671	.698	.726	.735	.738	**.742**	.736	.711	.672	3.0	2.6

Table 6: Effect of Training Set Class Distribution on AUC

If we look at the results with pruning, we see that for 15 of the 26 data sets C4.5 with pruning underperforms C4.5 without pruning. Thus, with respect to AUC, C4.5 without pruning is a reasonable learner. However, note that for the car data set the natural distribution with pruning falls into the optimum range, whereas without pruning it does not.

Figure 3 shows how class distribution affects AUC for the adult, covertype, and letter-a data sets (the phone data set is not displayed as it was in Figure 2 because it would obscure the adult data set). Again, the natural distribution is denoted by the “X” tick mark. The AUC for the best distribution is underlined and displayed below the corresponding data point. In this case we also see that near the optimal class distribution the AUC curves tend to be flatter, and hence less sensitive to changes in class distribution.

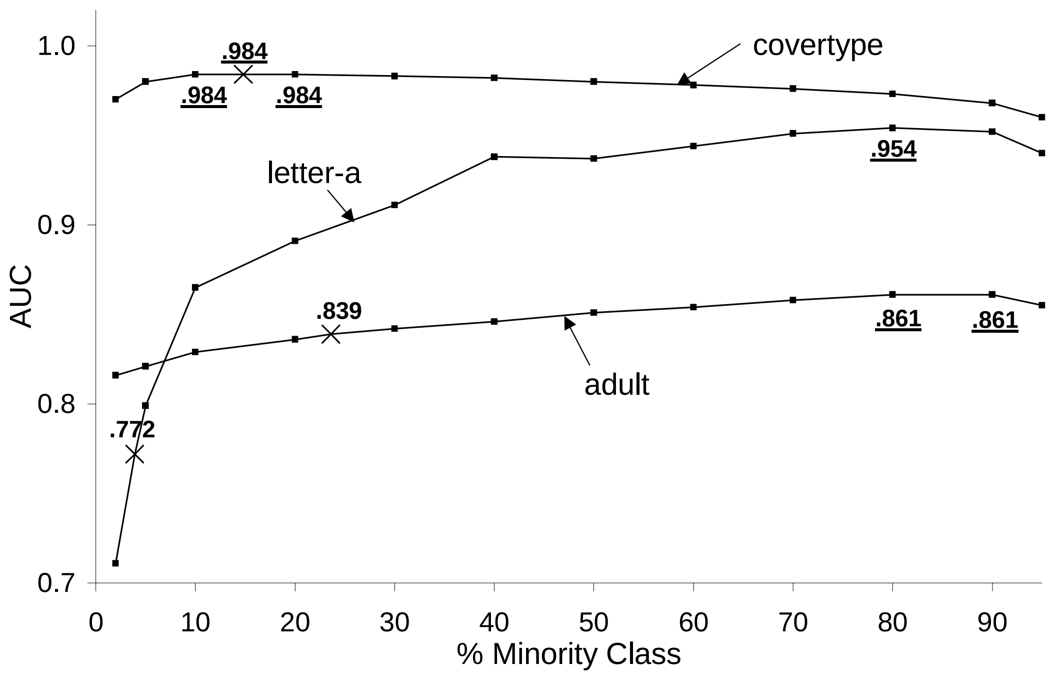
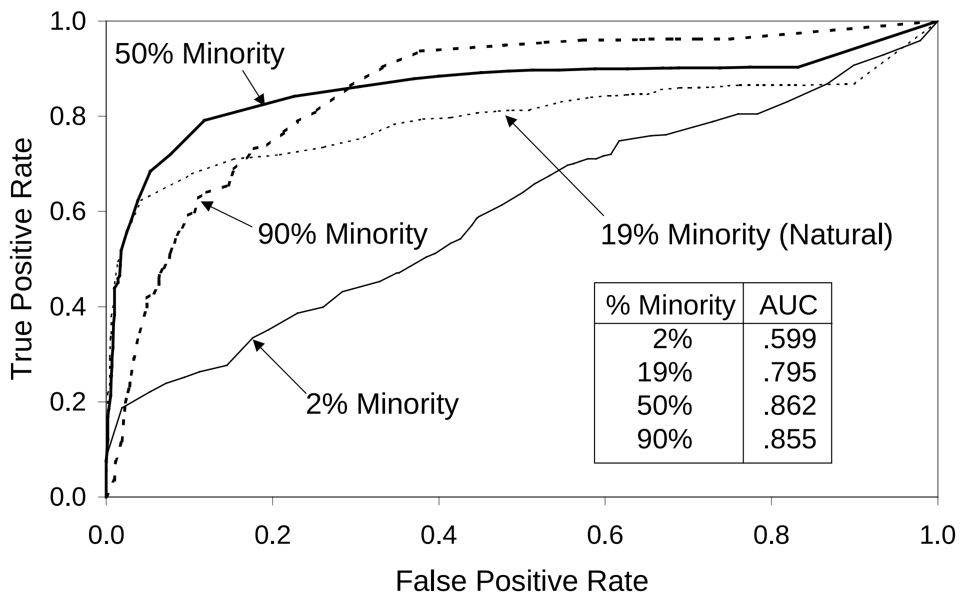


Figure 3: Effect of Class Distribution on AUC for Select Data Sets

Figure 4 shows several ROC curves associated with the letter-vowel data set. These curves each were generated from a single run of C4.5 (which is why the AUC values do not exactly match the values in Table 6). In ROC space, the point (0,0) corresponds to the strategy of never making a positive/minority prediction and the point (1,1) to always predicting the positive/minority class. Points to the "northwest" indicate improved performance.



% Minority	AUC
2%	.599
19%	.795
50%	.862
90%	.855

Figure 4: ROC Curves for the Letter-Vowel Data set

Observe that different training distributions perform better in different areas of ROC space. Specifically note that the classifier trained with 90% minority-class examples performs substantially better than the classifier trained with the natural distribution for high true-positive rates and that the classifier training with 2% minority-class examples performs fairly well for low true-positive rates. These results are easily explained. With only a small sample of minority-class examples (2%) a classifier can identify only a few minority-labeled "rules" with high confidence. However, with a much larger sample of minority-class examples (90%) it can identify many more such minority-labeled rules. However, for this data set a balanced distribution has the largest AUC and performs best overall. One interesting thing to note is that the curve generated using the balanced class distribution almost always outperforms the curve associated with the natural distribution (for low false-positive rates the natural distribution performs slightly better).

7. Forming a Good Class Distribution with Sensitivity to Procurement Costs

The results from the previous section demonstrate that some marginal class distributions yield classifiers that perform substantially better than the classifiers produced by other training distributions. Unfortunately, forming the thirteen training sets of size *n*, each with a different class distribution, requires nearly 2*n* examples. When it is costly to obtain training examples in a form suitable for learning, then this approach is not feasible—and ultimately is self-defeating. Ideally, one would select a total of *n* training examples—all of which would be used in the final training set—and the associated class distribution would yield classifiers that perform better than those generated from any other class distribution (given *n* training examples). In this section we describe and then evaluate a heuristic, budget-sensitive, progressive-sampling algorithm for selecting training data that approximates this ideal.

Before presenting this new algorithm, in Section 7.1 we measure classifier performance for a variety of training-set sizes and class distributions. This is done for the three largest data sets listed in Table 2: the phone data set (652,557 examples), the covertype data set (581,102 examples) and the adult data set (48,842 examples). These large data sets allow us to vary the training-set size and still have sufficient examples to yield meaningful results. These results show that it is possible to achieve competitive—or even superior—classifier performance using fewer training examples, if the class distribution is selected carefully. The algorithm for selecting training data is then described in Section 7.2 and the results of applying this algorithm to the three large data sets are presented in Section 7.3. The results from Section 7.1 are used when applying the algorithm to the three data sets, so that no additional experiments are required.

7.1 The Effect of Class Distribution and Training-Set Size on Classifier Performance

Experiments were run to establish the relationship between training-set size, class distribution and classifier performance for the phone, covertype, and adult data sets. The results are summarized in Table 7. Classifier performance is reported for the thirteen class distributions specified in Section 6.1 using the following nine training-set sizes, expressed in terms of the fraction of available training data: 1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2, 3/4, and 1. The natural class distribution is denoted by an asterisk and, for each training-set size, the class distribution that yields the best performance is displayed in bold and is underlined.

Data Set	Size	Metric	2	5	10	18.2*	20	30	40	50	60	70	80	90	95
PHONE	1/128		.641	.737	.784	**.793**	.792	.791	.791	.789	.788	.786	.785	.774	.731
|	1/64		.707	.777	.784	**.803**	.803	**.803**	.801	.802	.801	.798	.799	.788	.744
|	1/32		.762	.794	.809	**.812**	**.812**	.811	.811	.811	.810	**.812**	.811	.805	.778
|	1/16		.784	.813	.816	.823	.823	**.824**	.818	.821	.822	.821	.822	.817	.805
|	1/8	AUC	.801	.823	.828	.830	.830	.830	.830	.829	.830	.829	**.831**	.828	.818
|	1/4		.819	.835	.837	.839	.839	.837	.837	.836	.836	.836	**.838**	.836	.832
|	1/2		.832	.843	**.846**	**.846**	.845	.845	.843	.843	.843	.843	.844	**.846**	.844
|	3/4		.838	.847	.849	.849	.849	.848	.846	.847	.846	.847	.848	**.851**	.848
|	1		.843	.850	.852	.851	.851	.850	.850	.849	.848	.848	.850	**.853**	.850
|	1/128		17.47	16.42	**15.71**	16.10	16.25	17.52	18.81	21.21	22.87	26.40	30.43	33.26	37.27
|	1/64		17.01	15.75	15.21	**15.12**	15.20	16.39	17.59	19.60	22.11	24.80	27.34	30.21	26.86
|	1/32		16.22	15.02	14.52	**14.50**	14.75	15.41	16.81	18.12	20.02	21.77	24.86	25.31	28.74
|	1/16	Error	15.78	14.59	**14.01**	14.02	14.18	14.70	16.09	17.50	18.68	20.70	22.46	24.15	24.52
|	1/8	Rate	15.17	14.08	**13.46**	13.61	13.71	14.27	15.30	16.51	17.66	19.66	21.26	23.23	23.33
|	1/4		14.44	13.55	**13.12**	13.23	13.27	13.85	14.78	15.85	17.09	18.94	20.43	22.28	22.90
|	1/2		13.84	13.18	**12.81**	12.83	12.95	13.47	14.38	15.30	16.43	17.88	19.57	21.68	21.68
|	3/4		13.75	13.03	**12.60**	12.70	12.74	13.35	14.12	15.01	16.17	17.33	18.82	20.43	21.24
|	1		13.45	12.87	**12.32**	12.62	12.68	13.25	13.94	14.81	15.97	17.32	18.73	20.24	21.07
			2	**5**	**10**	**20**	**23.9***	**30**	**40**	**50**	**60**	**70**	**80**	**90**	**95**
ADULT	1/128		.571	.586	.633	.674	.680	.694	.701	.704	.723	.727	**.728**	.722	.708
|	1/64		.621	.630	.657	.702	.714	.711	.722	.732	.739	.746	**.755**	.752	.732
|	1/32		.638	.674	.711	.735	.742	.751	.755	.766	.762	.765	**.772**	.766	.759
|	1/16		.690	.721	.733	.760	.762	.778	.787	.791	**.794**	.787	.785	.780	.771
|	1/8	AUC	.735	.753	.768	.785	.787	.793	.799	.809	.812	**.816**	.813	.803	.797
|	1/4		.774	.779	.793	.804	.809	.813	.820	.827	.831	.832	**.834**	.824	.811
|	1/2		.795	.803	.812	.822	.825	.829	.834	.838	.841	.847	**.849**	.847	.834
|	3/4		.811	.814	.823	.830	.833	.837	.843	.845	.849	.853	**.856**	.855	.848
|	1		.816	.821	.829	.836	.839	.842	.846	.851	.854	.858	**.861**	**.861**	.855
|	1/128		23.80	23.64	**23.10**	23.44	23.68	23.90	25.22	26.94	29.50	33.08	37.85	46.13	48.34
|	1/64		23.32	22.68	22.21	**21.77**	21.80	23.08	24.38	26.29	28.07	31.45	36.41	43.64	47.52
|	1/32		22.95	22.09	21.12	**20.77**	20.97	21.11	22.37	24.41	27.08	30.27	34.04	42.40	47.20
|	1/16	Error	22.66	21.34	20.29	**19.90**	20.07	20.37	21.43	23.18	25.27	28.67	33.41	40.65	46.68
|	1/8	Rate	21.65	20.15	19.13	**18.87**	19.30	19.67	20.86	22.33	24.56	27.14	31.06	38.35	45.83
|	1/4		20.56	19.08	**18.20**	18.42	18.70	19.12	20.10	21.39	23.48	25.78	29.54	36.17	43.93
|	1/2		19.51	18.10	**17.54**	**17.54**	17.85	18.39	19.38	20.83	22.81	24.88	28.15	34.71	41.24
|	3/4		18.82	17.70	**17.17**	17.32	17.46	18.07	18.96	20.40	22.13	24.32	27.59	33.92	40.47
|	1		18.47	17.26	**16.85**	17.09	17.25	17.78	18.85	20.05	21.79	24.08	27.11	33.00	39.75
			2	**5**	**10**	**14.8***	**20**	**30**	**40**	**50**	**60**	**70**	**80**	**90**	**95**
COVERTYPE	1/128		.767	.852	.898	.909	**.916**	.913	**.916**	**.916**	.909	.901	.882	.854	.817
|	1/64		.836	.900	.924	.932	**.937**	.935	.936	.932	.928	.922	.913	.885	.851
|	1/32		.886	.925	.942	.947	**.950**	.947	.948	.948	.944	.939	.930	.908	.876
|	1/16		.920	.944	.953	.957	**.959**	**.959**	**.959**	.957	.955	.951	.945	.929	.906
|	1/8	AUC	.941	.955	.963	.965	.967	.968	**.969**	.968	.967	.963	.957	.948	.929
|	1/4		.953	.965	.970	.973	.975	**.976**	.975	.973	.972	.970	.965	.956	.943
|	1/2		.963	.972	.979	**.981**	**.981**	.980	.978	.977	.975	.972	.970	.961	.953
|	3/4		.968	.976	.982	.982	**.983**	.982	.980	.979	.976	.975	.971	.966	.958
|	1		.970	.980	**.984**	**.984**	**.984**	.983	.982	.980	.978	.976	.973	.968	.960
|	1/128		**10.44**	10.56	10.96	11.86	13.50	16.16	18.26	20.50	23.44	26.95	31.39	37.92	44.54
|	1/64		9.67	**9.29**	10.23	11.04	12.29	14.55	16.52	18.58	21.40	24.78	27.65	34.12	41.67
|	1/32		8.87	**8.66**	9.44	10.35	11.29	13.59	15.34	17.30	19.31	21.82	24.86	28.37	33.91
|	1/16	Error	8.19	**7.92**	8.93	9.67	10.37	11.93	13.51	15.35	17.42	19.40	22.30	25.74	28.36
|	1/8	Rate	7.59	**7.32**	7.87	8.65	9.26	10.31	11.63	13.06	14.68	16.39	18.28	22.50	26.87
|	1/4		6.87	**6.44**	7.04	7.49	8.01	9.05	9.86	10.56	11.45	12.28	14.36	18.05	22.59
|	1/2		6.04	**5.71**	5.97	6.45	6.66	7.14	7.53	8.03	8.80	9.94	11.44	14.85	18.37
|	3/4		5.81	**5.31**	5.48	5.75	5.87	6.25	6.57	6.89	7.58	8.72	10.69	13.92	16.29
|	1		5.54	5.04	**5.00**	5.03	5.26	5.64	5.95	6.46	7.23	8.50	10.18	13.03	16.27

Table 7: The Effect of Training-Set Size and Class Distribution on Classifier Performance

The results in Table 7 show that while the position of the best class distribution varies somewhat with training-set size, in many cases, especially with error rate, the variation is small. This gives support to the notion that there is a "best" marginal class distribution for a learning task. The results also indicate that, for any fixed class distribution, increasing the size of the training set *always* leads to improved classifier performance. Note that classifier performance has not reached a plateau for any of the three data sets, for either error rate or AUC. This is important because if a plateau had been reached (i.e., learning had stopped), then it would be possible to reduce the size of the training set without degrading classifier performance. Because this is not the case, the results in Table 7 indicate that, for these three data sets (and C4.5), it may be profitable to select the training examples carefully when forming the training set. This provides one practical motivation for the research described in this article.

In Table 7 there are six cases highlighted (by using a line to connect pairs of data points) where competitive or improved performance is achieved from fewer training examples. In each of these six cases, the data point corresponding to the smaller data-set size performs as well or better than the data point that corresponds to the larger data-set size (the latter being either the natural distribution or a balanced one). As an example, consider the adult data set and the AUC performance metric. Table 7 shows that one can achieve a better AUC value (.849 vs. .839) by using, instead of all available training data and the natural class distribution, one-half of the available training data with a class distribution that includes 80% minority-class examples. As a second example, one can achieve a competitive error rate on the phone data set (12.60% vs. 12.62%) by using ¾ of the available data rather than the full data set, if one uses a class distribution that includes 10% minority-class examples rather than the natural class distribution.

Figures 5 and 6 provide a graphical representation of the data, to show how class distribution and training-set size interact to affect classifier performance for the adult data set. In these figures each performance curve is associated with a single training-set size. Because the performance curves always improve with increasing data-set size, in both figures only the curves corresponding to the smallest and largest training-set sizes are explicitly labeled. Note that for a given x-value the space between successive y-values indicates the change in performance, so it is clear that classifier performance does begin to flatten out as more training data become available. Also note that as the training-set size increases the choice of class distribution becomes somewhat less important, since the range of error-rate values for each curve diminishes.

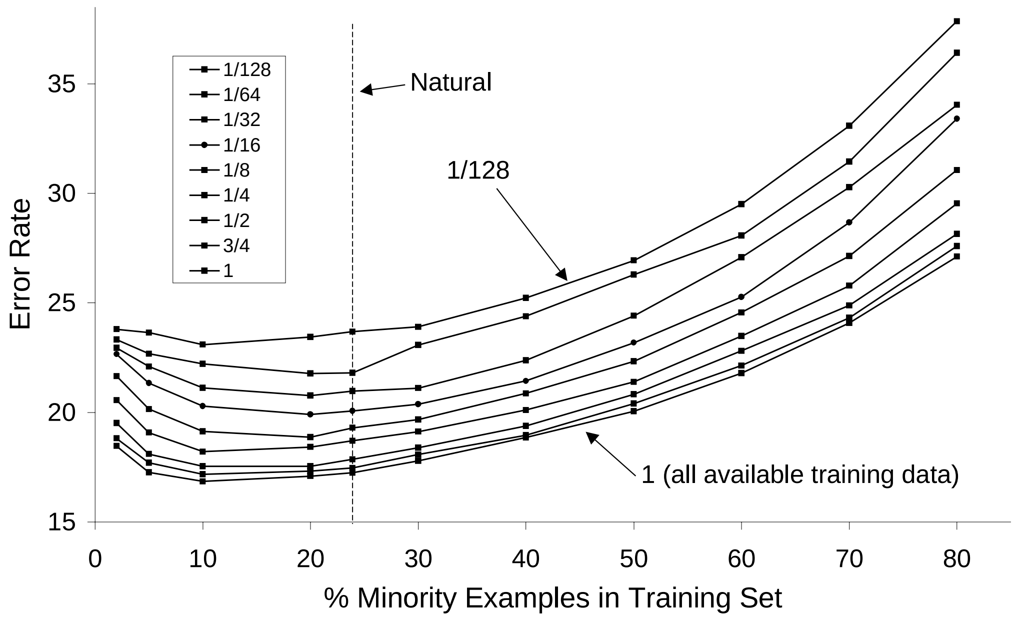


Figure 5: Effect of Class Distribution and Training-set Size on Error Rate (Adult Data Set)

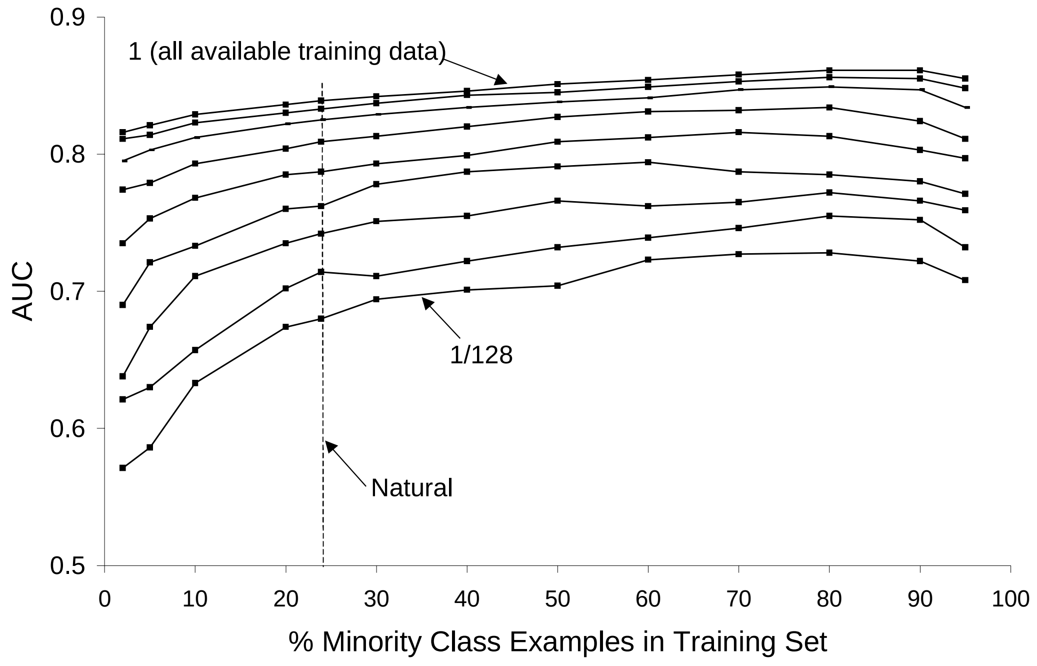


Figure 6: Effect of Class Distribution and Training-set Size on AUC (Adult Data Set)

7.2 A Budget-Sensitive Progressive-Sampling Algorithm for Selecting Training Data

As described earlier in this article, the size of the training set must sometimes be limited due to costs associated with procuring usable training examples. For simplicity, let us assume that there is a budget *n*, which permits one to procure *n* training examples. Further let us assume that the number of training examples that potentially can be procured is sufficiently large so that a training set of size *n* can be formed with any desired marginal class distribution. We would like a sampling strategy that selects *x* minority-class examples and *y* majority-class examples, where $x + y = n$, such that the resulting distribution yields the best possible classification performance for a training set of size *n*. Here we will assume that the cost of executing the learning algorithm is negligible compared to the cost of procuring examples, so that the learning algorithm may be run multiple times. We also assume that the cost of procuring examples is the same for each class and hence the budget represents both the number of examples that can be procured as well as the total cost. However, the algorithm described in this section could be extended to handle non-uniform procurement costs.

The algorithm described in this section selects minority-class and majority-class training examples such that the resulting class distribution will yield classifiers that tend to perform well. The algorithm begins with a small amount of training data and progressively adds training examples using a geometric sampling schedule (Provost, Jensen & Oates, 1999). The proportion of minority-class examples and majority-class examples added in each iteration of the algorithm is determined empirically by forming several class distributions from the currently available training data, evaluating the classification performance of the resulting classifiers, and then determining the class distribution that performs best. The algorithm implements a beam-search through the space of possible class distributions.

We say that the sampling algorithm is *budget-efficient* if all examples selected during any iteration of the algorithm are used in the final training set, which has the heuristically determined class distribution. The key is to constrain the search through the space of class distributions so that budget-efficiency is either guaranteed, or very likely. As we will show, the algorithm described in this section is guaranteed to be budget-efficient. Note, however, that the class distribution of the final training set, that is heuristically determined, is not guaranteed to be the best class distribution; however, as we will show, it performs well in practice.

The algorithm is outlined in Table 8, using pseudo-code, followed by a line-by-line explanation (a complete example is provided in Appendix B). The algorithm takes three user-specified input parameters: μ, the geometric factor used to determine the rate at which the training-set size grows, *n*, the budget, and *cmin*, the minimum fraction of minority-class examples and majority-class examples that are assumed to appear in the final training set in order for the budget-efficiency guarantee to hold.[4] For the results presented in this section, μ is set to 2, so that the training-set size doubles every iteration of the algorithm, and *cmin* is set to 1/32.

```
1.  minority = majority = 0;  # current number minority/majority examples requested
2.  $K = \left\lceil \log_{\mu}\left(\frac{1}{c\min}\right) \right\rceil$;                    # number of iterations is K+1

3.  for (j = 0; j ≤ K; j = j+1)                   # for each iteration (e.g., j = 0, 1,2,3,4)
4.  {
5.    size = $n(1/\mu)^{K-j}$                          # set training-set size for iteration j

6.    if (j = 0)
7.        beam_bottom = 0;   beam_top = 1;
8.    else
9.        $beam\_radius = \frac{\min(best, 1-best)}{\mu - 1/\mu + 1}$
10.       beam_bottom = best – beam_radius;  beam_top = best + beam_radius;

11.     min_needed = size • beam_top;              # number minority examples needed
12.     maj_needed = size • (1.0 – beam_bottom);   # number majority examples needed

13.     if (min_needed > minority)
14.         request (min_needed - minority) additional minority-class examples;
15.     if (maj_needed > majority)
16.         request (maj_needed - majority) additional majority-class examples;

17.     if (j ≠ K)
18.         eval(beam_bottom, beam_top, size); # evaluate distributions in the beam
19.     else
20.         eval(best, best, size);                # for last iteration just evaluate previous best
21. }
```

Table 8: The Algorithm for Selecting Training Data

[4] Consider the degenerate case where the algorithm determines that the best class distribution contains no minority-class examples or no majority-class examples. If the algorithm begins with even a single example of this class, then it will not be efficient.

The algorithm begins by initializing the values for the *minority* and *majority* variables, which represent the total number of minority-class examples and majority-class examples requested by the algorithm. Then, in line 2, the number of iterations of the algorithm is determined, such that the initial training-set size, which is subsequently set in line 5, will be at most $cmin \cdot n$. This will allow all possible class distributions to be formed using at most *cmin* minority-class examples and *cmin* majority-class examples. For example, given that μ is 2 and *cmin* is 1/32, in line 2 variable K will be set to 5 and in line 5 the initial training-set size will be set to 1/32 n.

Next, in lines 6-10, the algorithm determines the class distributions to be considered in each iteration by setting the boundaries of the beam. For the first iteration, all class distributions are considered (i.e., the fraction of minority-class examples in the training set may vary between 0 and 1). In subsequent iterations, the beam will be centered on the class distribution that performed best in the previous iteration. In line 9 the radius of the beam is set such that the ratio beam_top/beam_bottom will equal μ. For example, if μ is 2 and *best* is .15, then *beam_radius* is .05 and the beam will span from .10 to .20—which differ by a factor of 2 (i.e., μ).

In lines 11 and 12 the algorithm computes the number of minority-class examples and majority-class examples needed to form the class distributions that fall within the beam. These values are determined from the class distributions at the boundaries of the beam. In lines 13-16 additional examples are requested, if required. In lines 17-20 an evaluation procedure is called to form the class distributions within the beam and then to induce and to evaluate the classifiers. At a minimum this procedure will evaluate the class distributions at the endpoints and at the midpoint of the beam; however, this procedure may be implemented to evaluate additional class distributions within the beam. The procedure will set the variable *best* to the class distribution that performs best. If the best performance is achieved by several class distributions, then a resolution procedure is needed. For example, the class distribution for which the surrounding class distributions perform best may be chosen; if this still does not yield a unique value, then the best-performing class distribution closest to the center of the beam may be chosen. In any event, for the last iteration, only one class distribution is evaluated—in Table 8, the previous best. To ensure budget-efficiency, only one class distribution can be evaluated in the final iteration.

This algorithm is guaranteed to request only examples that are subsequently used in the final training set, which will have the heuristically determined class distribution. This guarantee can be verified inductively. First, the base case. The calculation for K in line 2 ensures that the initial training set will contain $cmin \cdot n$ training examples. Since we assume that the final training set will have at least *cmin* minority-class examples and *cmin* majority-class examples, all examples used to form the initial training set are guaranteed to be included in the final training set. Note that *cmin* may be set arbitrarily small—the smaller *cmin* the larger K and the smaller the size of the initial training set.

The inductive step is based on the observation that because the radius of the beam in line 9 is set so that the beam spans at most a factor of μ, all examples requested in each iteration are guaranteed to be used in the final training set. To see this, we will work backward from the final iteration, rather than working forward as is the case in most inductive proofs. Assume that the result of the algorithm is that the fraction of minority-class examples in the final training set is p, so that there are $p \cdot n$ minority-class examples in the final training set. This means that p was the best distribution from the previous iteration. Since p must fall somewhere within the beam for the previous iteration and the beam must span a factor μ, we can say the following: the fraction of minority-class examples in the previous iteration could range from p/μ (if p was at the top of the previous beam) to $\mu \cdot p$ (if p was at the bottom of the previous beam). Since the previous iteration contains n/μ examples, due to the geometric sampling scheme, then the previous iteration has at most (μ • *p*) • *n*/μ, or *p* • *n*, minority-class examples. Thus, in all possible cases all minority-class examples from the previous iteration can be used in the final interaction. This argument applies similarly to the majority-class examples and can be extended backwards to previous iterations.[5] Thus, because of the bound on the initial training-set size and the restriction on the width of the beam not to exceed the geometric factor μ, the algorithm guarantees that all examples requested during the execution of the algorithm will be used in the final training set.

A complete, detailed, iteration-by-iteration example describing the sampling algorithm as it is applied to a data set is provided in Appendix B, Table B1. In this example, the algorithm is applied to the phone data set and error rate is used to evaluate classifier performance. The description lists the class distributions that are evaluated during the execution of the algorithm. This "trajectory" is graphically depicted in Figure 7.

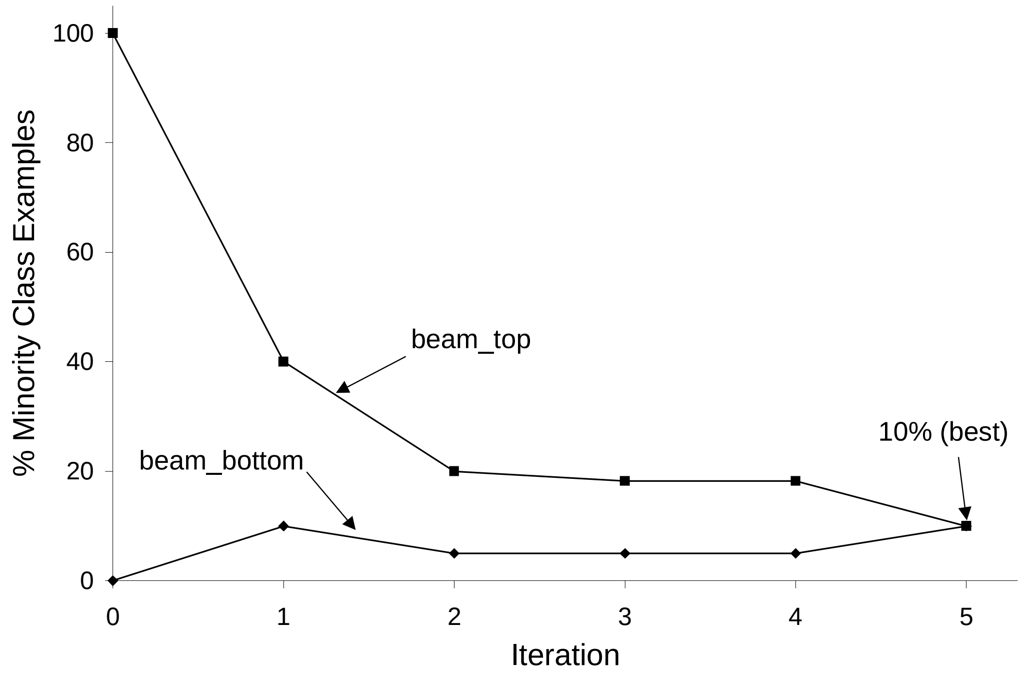


Figure 7: The Trajectory of the Algorithm through the Space of Class Distributions

7.3 Results for the Sampling Algorithm

The budget-sensitive progressive-sampling algorithm was applied to the phone, adult, and covertype data sets using both error rate and AUC to measure classifier performance. However, the method for setting the beam (described in lines 6-10 in Table 8) was modified so that the results from Section 7.1, which evaluate only the 13 listed class distributions, could be used. Specifically, the low end (high end) of the beam is set to the class distribution listed in Table 7 that is just below (above) the best performing class distribution. For example, if the best performing class distribution contains 30% minority-class examples, then the bottom of the beam is set to

[5] The only exception is for the first iteration of the algorithm, since in this situation the beam is unconditionally set to span all class distributions. This is the reason why the *cmin* value is required to provide the efficiency guarantee.

include 20% minority-class examples and the top of the beam to include 40% minority-class examples. Although this will sometimes allow the beam to span a range greater than μ (2), in practice this does not result in a problem—for all three data sets all examples requested by the algorithm are all included in the final training set. In addition, a slight improvement was made to the algorithm. Specifically, on any iteration, if the number of examples already in hand (procured in previous iterations) is sufficient to evaluate additional class distributions, then the beam is widened to include these additional class distributions (this can happen because during the first iteration the beam is set very wide).

The performance of this sampling algorithm is summarized in Table 9, along with the performance of three other strategies for selecting the class distribution of the training data. These additional strategies for selecting the class distribution are: 1) always pick the natural class distribution, 2) always pick the balanced class distribution, and 3) pick the class distribution that performs best over the thirteen class distributions in Table 7. Note that an iteration-by-iteration description of the algorithm, for all three data sets, is provided in Appendix B, Table B3. Table 9 also specifies the cost for each strategy, which is based on the number of training examples requested by the algorithm. This cost is expressed with respect to the budget n (each strategy yields a final training set with n examples). The strategies that involve either picking the natural or balanced class distributions require exactly *n* examples to be selected (and hence are budget-efficient). Given that Table 7 evaluates the class distributions using between 2% and 95% minority-class examples, the strategy that involves selecting the best of the thirteen class distributions requires that .95*n* minority-class examples and .98*n* majority-class examples be chosen. This yields a total cost of 1.93*n* and hence is not budget-efficient. Unlike the other three strategies, the cost of the sampling algorithm depends on the performance of the induced classifiers, and with the changes to the algorithm described in this section, is not guaranteed to be budget-efficient. Nonetheless, in all cases—for both error rate and AUC—the sampling algorithm has a cost of exactly *n* and hence is budget-efficient.

	Sampling Algorithm			Pick Natural			Pick Balanced			Pick Best		
Data Set	**ER**	**AUC**	**Cost**	**ER**	**AUC**	**Cost**	**ER**	**AUC**	**Cost**	**ER**	**AUC**	**Cost**
phone	12.32%	0.851	*n*	12.32%	0.851	*n*	14.81%	0.849	*n*	12.32%	0.853	1.93*n*
adult	17.09%	0.861	*n*	17.25%	0.839	*n*	20.05%	0.851	*n*	16.85%	0.861	1.93*n*
covertype	5.04%	0.984	*n*	5.03%	0.984	*n*	6.46%	0.980	*n*	5.00%	0.984	1.93*n*

Table 9: Comparative Performance of the Sampling Algorithm

The results in Table 9 show that by using the budget-sensitive progressive-sampling algorithm to choose the training data it is possible to achieve results that are no worse and sometimes better than the strategies of always using the natural or balanced class distributions, without requiring that any extra examples to be procured. Specifically, note that in no case does the strategy of always using the balanced distribution outperform the sampling algorithm and that in only one case (for covertype using error rate) does the strategy of using the natural distribution outperform the sampling algorithm (and in this case the difference is minimal). The budget-sensitive algorithm does almost as well (essentially as well in 5 of 6 cases) as the "Pick Best" strategy, which is almost twice as costly. Based on these results, the budget-sensitive progressive-sampling algorithm is attractive—it incurs the minimum possible cost in terms of procuring examples while permitting the class distribution for training to be selected using some intelligence.

8. Related Work

Several researchers have considered the question of what class distribution to use for a fixed training-set size, and/or, more generally, how class distribution affects classifier performance. Both Catlett (1991) and Chan & Stolfo (1998) study the relationship between (marginal) training class distribution and classifier performance when the training-set size is held fixed, but focus most of their attention on other issues. These studies also analyze only a few data sets, which makes it impossible to draw general conclusions about the relationship between class distribution and classifier performance. Nonetheless, based on the results for three data sets, Chan & Stolfo (1998) show that when accuracy is the performance metric, a training set that uses the natural class distribution yields the best results. These results agree partially with our results—although we show that the natural distribution does not always maximize accuracy, we show that the optimal distribution generally is close to the natural distribution. Chan & Stolfo also show that when actual costs are factored in (i.e., the cost of a false positive is not the same as a false negative), the natural distribution does not perform best; rather a training distribution closer to a balanced distribution performs best. They also observe, as we did, that by increasing the percentage of minority-class examples in the training set, the induced classifier performs better at classifying minority examples. It is important to note, however, that neither Chan & Stolfo nor Catlett adjusted the induced classifiers to compensate for changes made to the class distribution of the training set. This means that their results are biased in favor of the natural distribution (when measuring classification accuracy) and that they could improve the classification performance of minority class examples simply by changing (implicitly) the decision threshold. As the results in Appendix A show, compensating for the changed class distribution can affect the performance of a classifier significantly.

Several researchers have looked at the general question of how to reduce the need for labeled training data by selecting the data intelligently, but without explicitly considering the class distribution. For example, Cohn et al. (1994) and Lewis and Catlett (1994) use "active learning" to add examples to the training set for which the classifier is least certain about the classification. Saar-Tsechansky and Provost (2001, 2003) provide an overview of such methods and also extend them to cover AUC and other non-accuracy based performance metrics. The setting where these methods are applicable is different from the setting we consider. In particular, these methods assume either that arbitrary examples can be labeled or that the descriptions of a pool of unlabeled examples are available and the critical cost is associated with labeling them (so the algorithms select the examples intelligently rather than randomly). In our typical setting, the cost is in procuring the descriptions of the examples—the labels are known beforehand.

There also has been some prior work on progressive sampling strategies. John and Langley (1996) show how one can use the extrapolation of learning curves to determine when classifier performance using a subset of available training data comes close to the performance that would be achieved by using the full data set. Provost et al. (1999) suggest using a geometric sampling schedule and show that it is often more efficient than using all of the available training data. The techniques described by John and Langley (1996) and Provost et al. (1999) do not change the distribution of examples in the training set, but rather rely on taking random samples from the available training data. Our progressive sampling routine extends these methods by stratifying the sampling by class, and using the information acquired during the process to select a good final class distribution.

There is a considerable amount of research on how to build "good" classifiers when the class distribution of the data is highly unbalanced and it is costly to misclassify minority-class examples (Japkowicz et al. 2000). This research is related to our work because a frequent approach for learning from highly skewed data sets is to modify the class distribution of the training set. Under these conditions, classifiers that optimize for accuracy are especially inappropriate because they tend to generate trivial models that almost always predict the majority class. A common approach for dealing with highly unbalanced data sets is to reduce the amount of class imbalance in the training set. This tends to produce classifiers that perform better on the minority class than if the original distribution were used. Note that in this situation the training-set size is not fixed and the motivation for changing the distribution is simply to produce a "better" classifier—not to reduce, or minimize, the training-set size.

The two basic methods for reducing class imbalance in training data are under-sampling and over-sampling. Under-sampling eliminates examples in the majority class while over-sampling replicates examples in the minority class (Breiman, et al. 1984; Kubat & Matwin, 1997; Japkowicz & Stephen, 2001). Neither approach consistently outperforms the other nor does any specific under-sampling or over-sampling rate consistently yield the best results. Estabrooks and Japkowicz (2001) address this issue by showing that a mixture-of-experts approach, which combines classifiers built using under-sampling and over-sampling methods with various sampling rates, can produce consistently good results.

Both under-sampling and over-sampling have known drawbacks. Under-sampling throws out potentially useful data while over-sampling *increases the size of the training set* and hence the time to build a classifier. Furthermore, since most over-sampling methods make exact copies of minority class examples, overfitting is likely to occur—classification rules may be induced to cover a single replicated example.[6] Recent research has focused on improving these basic methods. Kubat and Matwin (1997) employ an under-sampling strategy that intelligently removes majority examples by removing only those majority examples that are "redundant" or that "border" the minority examples—figuring they may be the result of noise. Chawla et al. (2000) combine under-sampling and over-sampling methods, and, to avoid the overfitting problem, form new minority class examples by interpolating between minority-class examples that lie close together. Chan and Stolfo (1998) take a somewhat different, and innovative, approach. They first run preliminary experiments to determine the best class distribution for learning and then generate multiple training sets with this class distribution. This is typically accomplished by including all minority-class examples and some of the majority-class examples in each training set. They then apply a learning algorithm to each training set and then combine the generated classifiers to form a composite learner. This method ensures that all available training data are used, since each majority-class example will be found in at least one of the training sets.

The research in this article can properly be viewed as research into under-sampling and its effect on classifier performance. However, given this perspective, our research performs under-sampling in order to reduce the training-set size, whereas in the research relating to skewed data sets the primary motivation is to improve classifier performance. For example, Kubat and Matwin (1997) motivate the use of under-sampling because "… adding examples of the majority class to the training set can have a detrimental effect on the learner's behavior." A consequence of these different motivations is that in our experiments we under-sample the minority and/or majority classes, while in the research concerned with learning from skewed distributions it is always the majority class that is under-sampled.

[6] This is especially true for methods such as C4.5, which stops splitting based on counting examples at the leaves of the tree.

The use of under-sampling for reducing the training-set size (and thereby reducing cost) may be the more practically useful perspective. Reducing the class imbalance in the training set effectively causes the learner to impose a greater cost for misclassifying minority-class examples (Breiman et al., 1984). Thus, when the cost of acquiring and learning from the data is not an issue, cost-sensitive or probabilistic learning methods are a more direct and arguably more appropriate way of dealing with class imbalance, because they do not have the problems, noted earlier, that are associated with under-sampling and over-sampling. Such approaches have been shown to outperform under-sampling and over-sampling (Japkowicz & Stephen, 2002). To quote one of the papers that considers this issue, "All of the data available can be used to produce the tree, thus throwing away no information, and learning speed is not degraded due to duplicate instances" (Drummond & Holte 2000, page 239).

9. Limitations and Future Research

One limitation with the research described in this article is that because all results are based on the use of a decision-tree learner, our conclusions may hold only for this class of learners. However, there are strong reasons to believe that our conclusions will hold for other learners as well. Namely, since the role that class distribution plays in learning—and the reasons, discussed in Section 5.2, for why a classifier will perform worse on the minority class—are not specific to decision-tree learners, one would expect other learners to behave similarly. One class of learners that may especially warrant further attention, however, are those learners that do not form disjunctive concepts. These learners will not suffer in the same way from the "problem of small disjuncts," which our results indicate is partially responsible for minority-class predictions having a higher error rate than majority-class predictions.[7] Thus, it would be informative to extend this study to include other classes of learners, to determine which results indeed generalize.

The program for inducing decision trees used throughout this article, C4.5, only considers the class distribution of the training data when generating the decision tree. The differences between the class distribution of the training data and the test data are accounted for in a post-processing step by re-computing the probability estimates at the leaves and using these estimates to re-label the tree. If the induction program had knowledge of the target (i.e., test) distribution during the tree-building process, then a different decision tree might be constructed. However, research indicates that this is not a serious limitation. In particular, Drummond and Holte (2000) showed that there are splitting criteria that are completely insensitive to the class distribution and that these splitting criteria perform as well or better than methods that factor in the class distribution. They further showed that C4.5's splitting criterion is relatively insensitive to the class distribution—and therefore to changes in class distribution.

We employed C4.5 without pruning in our study because pruning is sensitive to class distribution and C4.5's pruning strategy does not take the changes made to the class distribution of the training data into account. To justify this choice we showed that the observed differences in behavior between the minority and majority classes still exist with pruning (Section 5) and that C4.5 without pruning performs competitively with C4.5 with pruning anyway (Sections 6.2 and 6.3). Moreover, other research (Bradford et al. 1998) indicates that classifier performance does not generally improve when pruning takes class distribution and costs into account. Nevertheless

[7] However, many learners do form disjunctive concepts or something quite close. For example, Van den Bosch et al. (1997) showed that instance-based learners can be viewed as forming disjunctive concepts.

it would be worthwhile to see just how a "cost/distribution sensitive" pruning strategy would affect our results. We know of no pruning method that attempts to maximize AUC.

In this article we introduced a budget-sensitive algorithm for selecting training data when it is costly to obtain usable training examples. Because this topic is of great practical importance, we feel it warrants additional research. In particular, it would be interesting to extend our research in Section 7 to consider the case where it is more costly to procure examples belonging to one class than to another.

10. Conclusion

In this article we analyze, for a fixed training-set size, the relationship between the class distribution of training data and classifier performance with respect to accuracy and AUC. This is important for applications where data procurement is costly, but where data can be procured independently by class. Our results indicate that when accuracy is the performance measure, the best class distribution for learning tends to be near the natural distribution—although the use of a class distribution other than the naturally occurring distribution often leads to substantial improvements in classifier performance. Our results further indicate that when AUC is the performance measure, then a balanced class distribution will generally perform quite well. Thus, to generate a robust classifier that will perform well for a variety of misclassification costs or when the class distribution of the data is unknown or may change, our results suggest that a balanced class distribution should be used for training if a single class distribution must be chosen ex ante.

If it is possible to interleave data procurement and learning, we show that a budget-sensitive progressive sampling strategy can improve upon the use of either the natural distribution or a balanced distribution, and never does considerably worse. Furthermore, in our experiments, the budget-sensitive progressive strategy performs nearly optimally (with respect to what could be done with the given budget).

This article also provides a more comprehensive understanding of how class distribution affects learning and suggests answers to some fundamental questions, such as why classifiers almost always perform worse at classifying minority-class examples. We also describe how to adjust a classifier to compensate for changes made to the class distribution of the training set, so that the classifier is not unduly biased, and measure how this adjustment improves classifier performance. We consider this to be significant because previous related research did not make this adjustment (Catlett, 1991; Chan & Stolfo, 1998; Japkowicz & Stephen, 2002), which, as we show in Appendix A, has a major impact on classifier performance.

Practitioners often make changes to the class distribution of training data, especially when the classes are highly unbalanced. These changes are seldom done in a principled manner and the reasons for changing the distribution—and the consequences—are often not fully understood. We hope this article helps researchers and practitioners better understand the relationship between class distribution and classifier performance and permits them to learn more effectively when there is a need to limit the amount of training data.

Acknowledgments

We would like to thank Haym Hirsh for the comments and feedback he provided throughout this research. We would also like to thank the anonymous reviewers of this article, as well as IBM for a Faculty Partnership Award.

Appendix A: Impact of Class Distribution Correction on Classifier Performance

The results in the main body of the article were based on the use of the corrected frequency-based estimate, described in Section 3, to label the leaves of the induced decision trees. This corrected estimate ensures that the decision trees are not improperly biased by the changes made to the class distribution of the training set. In Table A1 below, we compare the performance of the decision trees labeled using the uncorrected frequency-based estimate (FB) with those labeled using the corrected frequency-based method (CT-FB). This comparison is based on the situation where the training set contains an equal number of minority-class and majority-class examples and the test set uses the natural class distribution. The results are based on 30 runs and the data sets are listed in order of decreasing class imbalance.

The error rate for each estimate is displayed in the second and third columns in Table A1, and, for each data set, the lowest error rate is underlined. The fourth column specifies the relative improvement that results from using the corrected frequency-based estimate. The fifth column specifies the percentage of the leaves in the decision tree that are assigned a different class label when the corrected estimate is used. The last two columns specify, for each estimate, the percentage of the total errors that are contributed by the minority-class test examples.

	Error Rate		% Rel.	% Labels	% Errors from Min.	
Dataset	**FB**	**CT-FB**	**Improv.**	**Changed**	**FB**	**CT-FB**
letter-a	9.79	5.38	45.0	39.0	2.7	7.2
pendigits	4.09	4.02	1.7	3.2	5.6	7.8
abalone	30.45	22.97	24.6	5.6	8.5	19.1
sick-euthyroid	9.82	6.85	30.2	6.7	8.8	14.6
connect-4	30.21	27.57	8.7	14.7	8.5	10.4
optdigits	6.17	3.41	44.7	42.5	6.0	21.2
covertype	6.62	6.46	2.4	2.4	7.0	8.5
solar-flare	36.20	29.12	19.6	20.4	19.3	30.7
phone	17.85	14.81	17.0	3.2	25.2	44.4
letter-vowel	18.89	14.16	25.0	44.1	15.9	30.2
contraceptive	40.77	39.65	2.7	11.1	20.6	27.6
adult	22.69	20.05	11.6	30.7	19.6	36.8
splice-junction	9.02	8.74	3.1	14.1	20.1	28.4
network2	30.80	29.96	2.7	1.2	32.9	40.1
yeast	34.01	28.80	15.3	4.6	29.4	47.0
network1	31.99	30.99	3.1	1.3	32.9	38.2
car	8.26	7.92	4.1	5.3	25.9	33.8
german	38.37	37.09	3.3	16.1	30.8	35.8
breast-wisc	6.76	6.74	0.3	0.4	38.5	38.7
blackjack	33.02	28.71	13.1	17.1	42.9	76.2
weather	34.62	34.61	0.0	0.0	40.5	40.5
bands	32.68	32.68	0.0	0.6	90.2	90.2
market1	25.77	25.77	0.0	23.9	46.0	48.6
crx	20.84	21.48	-3.1	17.2	46.2	51.4
kr-vs-kp	1.22	1.22	0.0	0.2	58.5	58.5
move	28.24	28.24	0.0	20.8	52.6	60.7
Average	21.89	19.90	10.6	13.3	28.3	36.4

Table A1: Impact of the Probability Estimates on Error Rate

Table A1 shows that by employing the corrected frequency-based estimate instead of the uncorrected frequency-based estimate, there is, on average, a relative 10.6% reduction in error rate. Furthermore, in only one case does the uncorrected frequency-based estimate outperform the corrected frequency-based estimate. The correction tends to yield a larger reduction for the most highly unbalanced data sets—in which cases it plays a larger role. If we restrict ourselves to the first 13 data sets listed in Table 2, for which the minority-class makes up less than 25% of the examples, then the relative improvement over these data sets is 18.2%. Note that because in this scenario the minority class is over-sampled in the training set, the corrected frequency-based estimate can only cause minority-labeled leaves to be labeled with the majority-class. Consequently, as the last column in the table demonstrates, the corrected version of the estimate will cause more of the errors to come from the minority-class test examples.

Appendix B: Detailed Results for the Budget-Sensitive Sampling Algorithm

This appendix describes the execution of the sampling algorithm that was presented in Table 8 of Section 7.2. First, in Table B1, a very detailed iteration-by-iteration description of the sampling algorithm is presented as it is applied to the phone data set using error rate to measure classifier performance. Table B2 provides a more compact version of this description, by reporting only the key variables as they change value from iteration to iteration. Finally, in Table B3, this compact description is used to describe the execution of the sampling algorithm for the phone, adult, and covertype data sets, using both error rate and AUC to measure performance. Note that for each of these tables, the column labeled "budget" refers to the budget used, or cost incurred. Note that in Table B3 in no case is the budget exceeded, which means that all examples requested during the execution of the algorithm are used in the final training set, with the heuristically-determined class distribution (i.e., the algorithm is budget-efficient).

The results that are described in this appendix, consistent with the results presented in Section 7, are based on a geometric factor, μ, of 2, and a value of *cmin* of 1/32. The total budget available for procuring training examples is *n*. Based on these values, the value of *K*, which determines the number of iterations of the algorithm and is computed on line 2 of Table 8, is set to 5. Note that since we use the values in Table 7 when applying the sampling algorithm to the data sets, *n* represents the number of training examples used when generating Table 7 (i.e., it corresponds to the training-set size when the value in column 2 of Table 7 is 1).

Below is the description of the sampling algorithm, as it is applied to the phone data set with error rate as the performance measure:

<u>j = 0</u> Training-set size = 1/32 *n*. Form 13 data sets, which will contain between 2% and 95% minority-class examples. This requires .0298*n (*95% of 1/32 *n)* minority-class examples and .0307*n* (100%-2% = 98% of 1/32 *n*) majority-class examples. Induce and then evaluate the resulting classifiers. Based on the results in Table 7, the natural distribution, which contains 18.2% minority-class examples, performs best. Total Budget: .0605*n* (.0298*n* minority, .0307*n* majority).

<u>j = 1</u> Training-set size = 1/16 *n*. Form data sets corresponding to the best-performing class distribution form the previous iteration (18.2% minority) and the adjoining class distributions used in the beam search, which contain 10% and 20% minority-class examples. This requires .0250*n* (20% of 1/16 *n*) minority-class examples and .0563*n* (90% of 1/16 *n*) majority-class examples. Since .0298*n* minority-class examples were previously obtained, class distributions containing 30% and 40% minority-class examples can also be formed without requesting additional examples. This iteration requires .0256*n* additional

majority-class examples. The best-performing distribution contains 10% minority-class examples. Total Budget: .0861 *n* (.0298*n* minority, .0563*n* majority).

j = 2 Training-set size = 1/8 *n*. Since the 10% distribution performed best, the beam search evaluates the 5%, 10%, and 18.2% minority-class distributions. The 20% class distribution is also evaluated since this requires only .0250*n* of the .0298*n* previously obtained minority-class examples. A total of .1188*n* (95% of 1/8 *n*) majority-class examples are required. The best performing distribution contains 10% minority-class examples. This iteration requires .0625*n* additional majority-class examples. Total Budget: .1486*n* (298*n* minority, 1188*n* majority).

j = 3 Training-set size = 1/4 *n*. The distributions to be evaluated are 5%, 10%, and 18.2%. There are no "extra" minority-class examples available to evaluate additional class distributions. This iteration requires .0455*n* (18.2% of 1/4 *n*) minority-class examples and .2375*n* (95% of 1/4 *n*) majority-class examples. The best-performing class distribution contains 10% minority-class examples. Total Budget: .2830*n* (.0455*n* minority, .2375*n* majority)

j = 4 Training-set size = 1/2 *n*. The 5%, 10%, and 18.2% class distributions are evaluated. This iteration requires .0910*n* (18.2% of 1/2 *n*) minority-class examples and .4750*n* (95% of 1/2 *n*) majority-class examples. The best-performing distribution contains 10% minority-class examples. Total Budget: .5660*n* (.0910*n* minority, .4750*n* majority).

j = 5 Training-set size = *n*. For this last iteration only the best class distribution from the previous iteration is evaluated. Thus, a data set of size *n* is formed, containing .1*n* minority-class examples and .9*n* majority-class examples. Thus .0090*n* additional minority-class examples and .4250*n* additional majority-class examples are required. Since all the previously obtained examples are used, there is no "waste" and the budget is not exceeded. Total Budget: 1.0*n* (.1000*n* minority, .9000*n* majority)

Table B1: A Detailed Example of the Sampling Algorithm (Phone Data Set using Error Rate)

				Expressed as a fraction on *n*				
j	size	class-distr	best	min-need	maj-need	minority	majority	budget
0	1/32 *n*	all	18.2%	.0298	.0307	.0298	.0307	.0605
1	1/16 *n*	10, *18.2*, 20, 30, 40	10%	.0250	.0563	.0298	.0563	.0861
2	1/8 *n*	5, *10*, 18.2, 20	10%	.0250	.1188	.0298	.1188	.1486
3	1/4 *n*	5, *10*, 18.2	10%	.0455	.2375	.0455	.2375	.2830
4	1/2 *n*	5, *10*, 18.2	**10%**	.0910	.4750	.0910	.4750	.5660
5	1 *n*	10		.1000	.9000	.1000	.9000	1.0000

Table B2: Compact Description of the Results in Table B1

Data set	Metric	j	size	class-distr	best	min-need	maj-need	minority	majority	budget
Phone	ER	0	1/32 *n*	all	18.2	.0298	.0307	.0298	.0307	.0605
		1	1/16 *n*	10, *18.2*, 20, 30, 40	10	.0250	.0563	.0298	.0563	.0861
		2	1/8 *n*	5, *10*, 18.2, 20	10	.0250	.1188	.0298	.1188	.1486
		3	1/4 *n*	5, *10*, 18.2	10	.0455	.2375	.0455	.2375	.2830
		4	1/2 *n*	5, *10*, 18.2	**10**	.0910	.4750	.0910	.4750	.5660
		5	1 *n*	10		.1000	.9000	.1000	.9000	1.0
Phone	AUC	0	1/32 *n*	all	20	.0298	.0307	.0298	.0307	.0605
		1	1/16 *n*	18.2, *20*, 30, 40	30	.0250	.0511	.0298	.0511	.0809
		2	1/8 *n*	20, *30*, 40	30	.0500	.1000	.0500	.1000	.1500
		3	1/4 *n*	20, *30*, 40	20	.1000	.2000	.1000	.2000	.3000
		4	1/2 *n*	18.2, *20*, 30	**18.2**	.1500	.4090	.1500	.4090	.5590
		5	1 *n*	18.2		.1820	.8180	.1820	.8180	1.0
Adult	ER	0	1/32 *n*	all	20	.0298	.0307	.0298	.0307	.0605
		1	1/16 *n*	10, *20*, 23.9, 30, 40	20	.0250	.0563	.0298	.0563	.0861
		2	1/8 *n*	10, *20*, 23.9	20	.0299	.1125	.0299	.1125	.1424
		3	1/4 *n*	10, *20*, 23.9	10	.0598	.2250	.0598	.2250	.2848
		4	1/2 *n*	5, *10*, 20	**20**	.1000	.4750	.1000	.4750	.5750
		5	1 *n*	20		.2000	.8000	.2000	.8000	1.0
Adult	AUC	0	1/32 *n*	all	80	.0298	.0307	.0298	.0307	.0605
		1	1/16 *n*	60, 70, *80*, 90	70	.0563	.0250	.0563	.0307	.0870
		2	1/8 *n*	60, *70*, 80	70	.1000	.0500	.1000	.0500	.1500
		3	1/4 *n*	60, *70*, 80	80	.2000	.1000	.2000	.1000	.3000
		4	1/2 *n*	70, *80*, 90	**80**	.4500	.1500	.4500	.1500	.6000
		5	1 *n*	80		.8000	.2000	.8000	.2000	1.0
Covertype	ER	0	1/32 *n*	all	5	.0298	.0307	.0298	.0307	.0605
		1	1/16 *n*	2, *5*, 10, 20, 30, 40	5	.0250	.0613	.0298	.0613	.0911
		2	1/8 *n*	2, *5*, 10, 20	5	.0250	.1225	.0298	.1225	.1523
		3	1/4 *n*	2, *5*, 10	5	.0250	.2450	.0298	.2450	.2748
		4	1/2 *n*	2, *5*, 10	**5**	.0500	.4900	.0500	.4900	.5400
		5	1 *n*	5		.0500	.9500	.0500	.9500	1.0
Covertype	AUC	0	1/32 *n*	all	20	.0298	.0307	.0298	.0307	.0605
		1	1/16 *n*	14.8, *20*, 30, 40	30	.0250	.0533	.0298	.0533	.0831
		2	1/8 *n*	20, *30*, 40	40	.0500	.1000	.0500	.1000	.1500
		3	1/4 *n*	30, *40*, 50	30	.1250	.1750	.1250	.1750	.3000
		4	1/2 *n*	20, *30*, 40	**20**	.2000	.4000	.2000	.4000	.6000
		5	1 *n*	20		.2000	.8000	.2000	.8000	1.0

Table B3: Complete Summary Results for the Sampling Algorithm

References

Bauer, E., & Kohavi, R. (1999). An empirical comparison of voting classification algorithms: bagging, boosting, and variants. *Machine Learning*, 36, 105-139.

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

Blake, C., & Merz, C. (1998). UCI Repository of Machine Learning Databases, (http://www.ics.uci.edu/~mlearn/MLRepository.html), Department of Computer Science, University of California.

Bradley, A. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7), 1145-1159.

Bradford, J.P., Kunz, C., Kohavi, R., Brunk, C., & Brodley, C. E. (1998). Pruning decision trees with misclassification costs. In *Proceedings of the European Conference on Machine Learning*, pp. 131-136.

Catlett, J. (1991). *Megainduction: machine learning on very large databases*. Ph.D. thesis, Department of Computer Science, University of Sydney.

Chan, P., & Stolfo, S. (1998). Toward scalable learning with non-uniform class and cost distributions: a case study in credit card fraud detection. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, pp. 164-168, Menlo Park, CA: AAAI Press.

Chawla, N., Bowyer, K., Hall, L., & Kegelmeyer, W. P. (2000). SMOTE: synthetic minority over-sampling technique. In *International Conference on Knowledge Based Computer Systems*.

Cohen, W., & Singer, Y. (1999). A simple, fast, and effective rule learner. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, pp. 335-342, Menlo Park, CA: AAAI Press.

Cohn, D., Atlas, L. and Ladner, R. (1994). Improved generalization with active learning. *Machine Learning*, 15:201-221.

Drummond, C., & Holte, R.C. (2000). Exploiting the cost (in)sensitivity of decision tree splitting criteria. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pp. 239-246.

Elkan, C. (2001). The foundations of cost-sensitive learning. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*.

Estabrooks, A., & Japkowicz, N. (2001). A Mixture-of-Experts Framework for Concept-Learning from Imbalanced Data Sets. *In Proceedings of the 2001 Intelligent Data Analysis Conference*.

Fawcett, T. & Provost, F. (1997). Adaptive Fraud Detection. *Data Mining and Knowledge Discovery* **1**(3): 291-316.

Good, I. J. (1965). *The Estimation of Probabilities*. Cambridge, MA: M.I.T. Press.

Hand, D. J. (1997). *Construction and Assessment of Classification Rules*. Chichester, UK: John Wiley and Sons.

Holte, R. C., Acker, L. E., & Porter, B. W. (1989). Concept learning and the problem of small disjuncts. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pp. 813-818. San Mateo, CA: Morgan Kaufmann.

Japkowicz, N. & Stephen, S. (2002). The Class Imbalance Problem: A Systematic Study. *Intelligent Data Analysis Journal, 6(5).*

Japkowicz, N., Holte, R. C., Ling, C. X., & Matwin S. (Eds.) (2000). In *Papers from the AAAI Workshop on Learning from Imbalanced Data Sets*. Tech, rep. WS-00-05, Menlo Park, CA: AAAI Press.

Jensen, D. D., & Cohen, P. R. (2000). Multiple comparisons in induction algorithms. *Machine Learning*, 38(3): 309-338.

John, G. H., & Langley, P. (1996). Static versus dynamic sampling for data mining. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pp. 367-370. Menlo Park, CA. AAAI Press.

Kubat, M., & Matwin, S. (1997). Addressing the curse of imbalanced training sets: one-sided selection. In *Proceedings of the Fourteenth International Conference on Machine Learning*, pp. 179-186.

Lewis, D. D., & Catlett, J. (1994). Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of the Eleventh International Conference on Machine Learning*, pp.148-156.

Provost, F., Fawcett, T., & Kohavi, R. (1998). The case against accuracy estimation for comparing classifiers. In *Proceedings of the Fifteenth International Conference on Machine Learning*. San Francisco, CA: Morgan Kaufmann.

Provost, F., Jensen, D., & Oates, T. (1999). Efficient progressive sampling. In *Proceedings of the Fifth International Conference on Knowledge Discovery and Data Mining*. ACM Press.

Provost, F., & Fawcett, T (2001). Robust classification for imprecise environments. *Machine Learning*, 42, 203-231.

Provost, F., & Domingos, P. (2001). Well-trained PETs: improving probability estimation trees. CeDER Working Paper #IS-00-04, Stern School of Business, New York University, New York, NY.

Quinlan, J.R. (1993). C*4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann.

Saar-Tsechansky, M., & Provost, F. (2001). Active learning for class probability estimation and ranking. *In Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, Seattle, WA.

Saar-Tsechansky, M. and F. Provost (2003). Active Sampling for Class Probability Estimation and Ranking. To appear in *Machine Learning*.

SAS Institute (2001). Getting Started With SAS Enterprise Miner. Cary, NC: SAS Institute Inc.

Saerens, M., Latinne, P., & Decaestecker, C. (2002). Adjusting the outputs of a classifier to new *a priori* probabilities: a simple procedure. *Neural Computation*, 14:21-41.

Swets, J., Dawes, R., & Monahan, J. (2000). Better decisions through science. *Scientific American*, October 2000: 82-87.

Turney P. (2000). Types of cost in inductive learning. In *Workshop on Cost-Sensitive Learning at the Seventeenth International Conference on Machine Learning*, 15-21, Stanford, CA.

Van den Bosch A., Weijters, A., Van den Herik, H.J., & Daelemans, W. (1997). When small disjuncts abound, try lazy learning: a case study. In *Proceedings of the Seventh Belgian-Dutch Conference on Machine Learning*, 109-118.

Weiss, G.M., & Hirsh, H. (2000). A quantitative study of small disjuncts, *In Proceedings of the Seventeenth National Conference on Artificial Intelligence*, 665-670. Menlo Park, CA: AAAI Press.

Weiss, G. M., & Provost, F (2001). The effect of class distribution on classifier learning: an empirical study. Tech rep. ML-TR-44, Department of Computer Science, Rutgers University, New Jersey.

Zadrozny, B., & Elkan, C. (2001). Learning and making decisions when costs and probabilities are both unknown. Tech. rep. CS2001-0664, Department of Computer Science and Engineering, University of California, San Diego.