Cost-Sensitive Learning vs. Sampling: Which is Best for Handling Unbalanced Classes with Unequal Error Costs?

Gary M. Weiss, Kate McCarthy, and Bibi Zabar
Department of Computer and Information Science
Fordham University
Bronx, NY, USA

Abstract - *The classifier built from a data set with a highly skewed class distribution generally predicts the more frequently occurring classes much more often than the infrequently occurring classes. This is largely due to the fact that most classifiers are designed to maximize accuracy. In many instances, such as for medical diagnosis, this classification behavior is unacceptable because the minority class is the class of primary interest (i.e., it has a much higher misclassification cost than the majority class). In this paper we compare three methods for dealing with data that has a skewed class distribution and non-uniform misclassification costs. The first method incorporates the misclassification costs into the learning algorithm while the other two methods employ oversampling or undersampling to make the training data more balanced. In this paper we empirically compare the effectiveness of these methods in order to determine which produces the best overall classifier—and under what circumstances.*

Keywords: Cost-sensitive learning, sampling, classification, decision trees, class imbalance.

1 Introduction

In many real-world domains, such as fraud detection and medical diagnosis, the class distribution of the data is skewed and the cost of misclassifying a minority-class example is substantially greater than the cost of misclassifying a majority-class example. In these cases it is important to create a classifier that minimizes the overall misclassification cost. This tends to cause the classifiers to perform better on the minority class than if the misclassification costs were equal. For highly skewed class distribution, this also ensures that the classifier does not always predict the majority class.

There are several methods that can be of use when dealing with skewed class distributions with unequal misclassification costs. The methods we analyze in this paper all can be considered a form of cost-sensitive learning. The most direct method is to use a learning algorithm that is itself cost-sensitive. What we mean by this is that the learning algorithm factors in the costs when building the classifier. Throughout this paper, the term "cost-sensitive learning algorithm" will refer to this type of learner.

An alternate strategy for dealing with skewed data with non-uniform misclassification costs is to use sampling to alter the class distribution of the training data. As we will show in Section 2, this method can be used to effectively impose, or simulate, non-uniform misclassification costs. Assuming that the cost of misclassifying a minority-class example is greater than the cost of misclassifying a majority-class example, the sampling method will make the class distribution of the training data more balanced (this effectively places more importance on the minority class).

There are two basic sampling methods that can be used: oversampling and undersampling. In this context oversampling replicates minority-class examples while undersampling discards majority-class examples. Note that sampling is a wrapper-based method that can make any learning algorithm cost-sensitive, whereas the cost-sensitive learning algorithm referred to earlier is not a wrapper-based method since the cost-sensitivity is embedded *in* the algorithm.

This paper compares the effectiveness of a cost-sensitive learning algorithm, oversampling, and undersampling. We use C5.0 [18], a more advanced version of Quinlan's popular C4.5 program [14], as our cost-sensitive learning algorithm. We believe that our results are noteworthy because all three methods are used in practice for handling imbalanced data sets. Our original conjecture was that a cost-sensitive learning algorithm should outperform both oversampling and undersampling because of the well-known problems (described in the next section) with these sampling methods—but our results do not support this conjecture. In this paper we also evaluate the efficacy of these three methods on data sets that are not skewed but may have non-uniform misclassification costs, in order to broaden the scope of our study.

2 Background

In this section we provide basic background information on cost-sensitive learning, sampling, and the connection between the two. Some related work is also described.

2.1 Cost-Sensitive Learning

The performance of a classifier for a two-class problem can be described by the confusion matrix described in Figure 1. Holding with the established practice, the minority class is designated the positive class and the majority class is designated the negative class.

		ACTUAL	
PREDICTED		Positive class	Negative class
	Positive class	True positive (TP)	False positive (FP)
	Negative class	False negative (FN)	True negative (TN)

Figure 1: A Confusion Matrix

Corresponding to a confusion matrix is a cost matrix. The cost matrix will provide the costs associated with the four outcomes shown in the confusion matrix, which we refer to as C_{TP}, C_{FP}, C_{FN}, and C_{TN}. As is often the case in cost-sensitive learning, we assign no costs to correct classifications, so C_{TP} and C_{TN} are set to 0. Since the positive (minority) class is often more interesting than the negative (majority) class, typically $C_{FN} > C_{FP}$ (note that a false negative means that a positive example was misclassified). As discussed earlier, cost-sensitive learning can be implemented in a variety of ways, by using the cost information in the classifier-building process or by using a wrapper-based method such as sampling.

When misclassification costs are known the best metric for evaluating classifier performance is total cost. Total cost is the only evaluation metric used in this paper and is used to evaluate all three cost-sensitive learning methods. The formula for total cost is shown in equation 1.

$$\text{Total Cost} = (FN \times C_{FN}) + (FP \times C_{FP}) \qquad (1)$$

2.2 Sampling

Oversampling and undersampling can be used to alter the class distribution of the training data and both methods have been used to deal with class imbalance [1, 2, 3, 6, 10, 11]. The reason that altering the class distribution of the training data aids learning with highly-skewed data sets is that it effectively imposes non-uniform misclassification costs. For example, if one alters the class distribution of the training set so that the ratio of positive to negative examples goes from 1:1 to 2:1, then one has effectively assigned a misclassification cost ratio of 2:1. This equivalency between altering the class distribution of the training data and altering the misclassification cost ratio is well known and was formally described by Elkan [9].

There are known disadvantages associated with the use of sampling to implement cost-sensitive learning. The disadvantage with undersampling is that it discards potentially useful data. The main disadvantage with oversampling, from our perspective, is that by making exact copies of existing examples, it makes overfitting likely. In fact, with oversampling it is quite common for a learner to generate a classification rule to cover a single, replicated, example. A second disadvantage of oversampling is that it increases the number of training examples, thus increasing the learning time.

2.3 Why Use Sampling?

Given the disadvantages with sampling, it is worth asking why anyone would use it rather than a cost-sensitive learning algorithm for dealing with data with a skewed class distribution and non-uniform misclassification costs. There are several reasons for this. The most obvious reason is there are not cost-sensitive implementations of all learning algorithms and therefore a wrapper-based approach using sampling is the only option. While this is certainly less true today than in the past, many learning algorithms (e.g., C4.5) still do not directly handle costs in the learning process.

A second reason for using sampling is that many highly skewed data sets are enormous and the size of the training set must be reduced in order for learning to be feasible. In this case, undersampling seems to be a reasonable, and valid, strategy. In this paper we do not consider the need to reduce the training set size. We would point out, however, that if one needs to discard some training data, it still might be beneficial to discard some of the majority class examples in order to reduce the training set size to the required size, and then also employ a cost-sensitive learning algorithm, so that the amount of discarded training data is minimized.

A final reason that may have contributed to the use of sampling rather than a cost-sensitive learning algorithm is that misclassification costs are often unknown. However, this is *not* a valid reason for using sampling over a cost-sensitive learning algorithm, since the analogous issue arises with sampling—what should the class distribution of the final training data be? If this cost information is not known, a measure such as the area under the ROC curve could be used to measure classifier performance and both approaches could then empirically determine the proper cost ratio/class distribution.

3 Data Sets

We employed fourteen data sets in our experiments. Twelve of the data sets were obtained from the UCI Repository and two of the data sets came from AT&T and were used in previously published work done by Weiss and Hirsh [16]. A summary of these data sets is provided in Table 1. The data sets are listed in descending order according to the degree of class imbalance, with the most imbalanced data sets listed first. The data sets marked with an asterisk (*) were originally multi-class data sets that were previously mapped into two classes for work done by Weiss and Provost [17]. The letter-a and letter-vowel data sets are derived from the letter recognition data set that is available from the UCI Repository. In order to simplify the analysis of our results, all data sets contain only two classes.

Table 1: Data Set Summary

Data Set	% Minority	Total Examples
Letter-a*	4%	20,000
Pendigits*	8%	13,821
Connect-4*	10%	11,258
Bridges1	15%	102
Letter-vowel*	19%	20,000
Hepatitis	21%	155
Contraceptive	23%	1,473
Adult	24%	21,281
Blackjack	36%	15,000
Weather	40%	5,597
Sonar	47%	208
Boa1	50%	11,000
Promoters	50%	106
Coding	50%	20,000

The data sets were chosen on the basis of their class distributions and data set sizes. Although the main focus of our research concerns classifying rare classes with unequal misclassification costs, in order to broaden the scope of our study we also include several data sets with relatively balanced class distributions. The boa1, promoters, and coding data sets each have an evenly balanced "50-50" distribution, so they are used for the sake of comparison. We used data sets of varying sizes to see how this would affect our results. One conjecture to be evaluated is that undersampling will do relatively poorly for small data sets, since discarding data in these cases should be extremely harmful (i.e., more so than for large data sets).

4 Experimental Methodology

The experiments conducted in our study are described in this section. All experiments utilize C5.0 [18], which is a more advanced version of Quinlan's popular C4.5 and ID3 decision tree induction programs[14, 15]. Unlike its predecessors, C5.0 is a cost-sensitive learning algorithm, which considers the cost information when building and pruning the induced decision tree.

The experiments in this paper assume that cost information is provided. Since the data sets described in Table 1 do not have this cost information, we instead investigate a variety of cost ratios. This actually increases the generality of our results since we evaluate more than one cost ratio per data set. Because we are primarily interested in the case where the cost of misclassifying minority-class (positive) examples is higher than that of misclassifying majority-class examples, we set $C_{FN} > C_{FP}$. For our experiments, a false positive prediction, C_{FP}, is assigned a unit cost of 1. For the majority of experiments C_{FN} is evaluated for the values: 1, 2, 3, 4, 6, and 10, although for some experiments the costs were allowed to increase beyond this point.

Oversampling and undersampling were also employed to implement the desired misclassification cost ratios, by altering the class distribution of the training data as described in Section 2.2. When this was done, no cost information was passed to C5.0 since we were not relying on the algorithm to implement the cost-sensitive learning. Since C5.0 does not provide support for sampling, we used scripts to implement the sampling prior to invoking C5.0.

For all experiments, 75% of the data is made available for training and 25% for testing. However, when using undersampling to implement cost-sensitive learning, some of the training examples are discarded. All experiments were run ten times, using random sampling to partition the data into the training and test sets. All results shown in this paper are the averages of these ten runs and all classifiers are evaluated using total cost, which was defined earlier in equation 1.

5 Results

Classifiers were generated for each data set and for a variety of misclassification cost ratios, using oversampling, undersampling, and C5.0's cost-sensitive learning capabilities. A figure was generated for each of the fourteen data sets, showing how the total cost varies when implementing cost-sensitive learning using the three schemes. Many of these figures are included in this section, although some are omitted due to space limitations. After presenting some of these detailed results, we provide summary statistics which make it easy to compare and contrast the performance of the three cost-sensitive learning schemes.

The results in Figure 2 for the letter-a data set show that the cost-sensitive learning algorithm and oversampling methods perform similarly, whereas undersampling performs much worse in essentially all cases (all methods will always perform identically for the 1:1 cost ratio). The results for the letter-vowel data set (not shown) are nearly identical, except that the cost-sensitive algorithm performs slightly better than oversampling for most cost ratios (both still outperform undersampling).

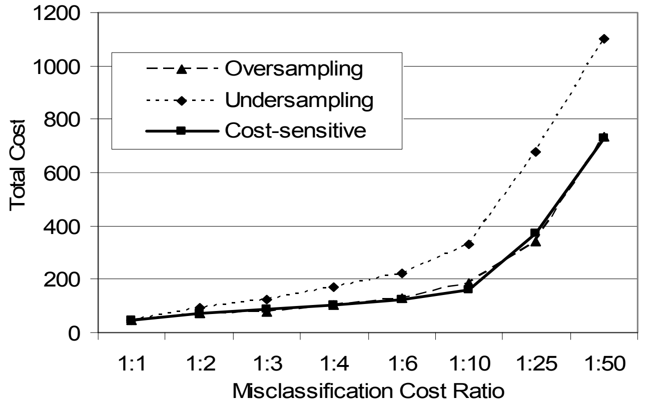

Figure 2: Results for Letter-a

The results for the weather data set, provided in Figure 3, show that oversampling consistently performs much worse than undersampling and the cost-sensitive algorithm, both of which performed similarly. This *exact* same pattern occurs in the results (not shown) for the adult and boa1 data sets.

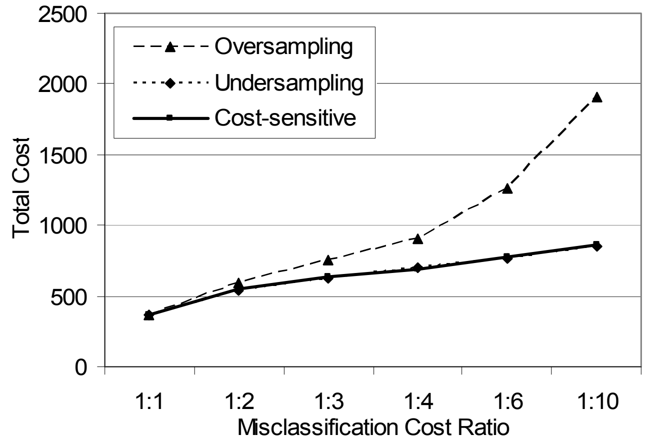

Figure 3: Results for Weather

The results for the coding data set in Figure 4 show that cost-sensitive learning outperforms both sampling methods, although the difference in total cost is much greater when compared to oversampling. However, as we shall see shortly in Figure 7, the cost-sensitive algorithm still outperforms undersampling by about 9%, a substantial amount (it outperforms oversampling by about 28%).

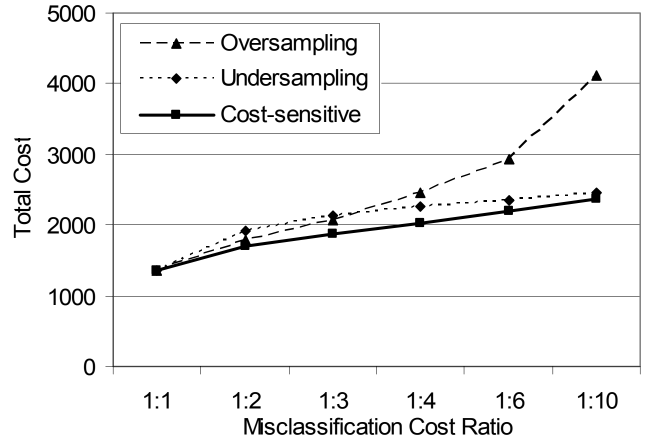

Figure 4: Results for Coding

The blackjack data set, shown in Figure 5, is the only data set for which all three methods yielded nearly identical performance for all cost ratios. The three methods also yielded nearly identical performance for the connect-4 data set (not shown), except for the highest cost ratio, 1:25, in which case oversampling performed the worst.

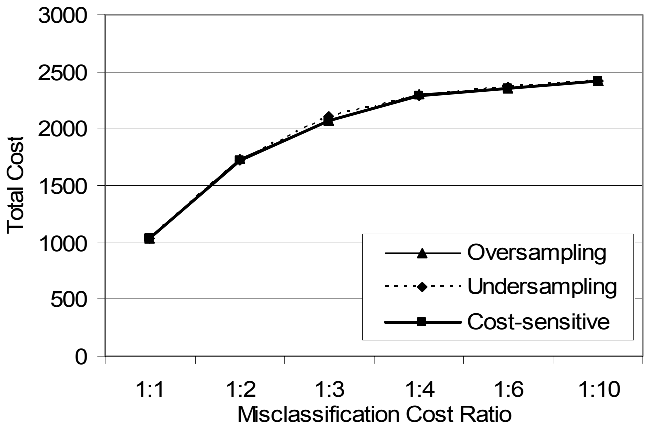

Figure 5: Results for Blackjack

There were three data sets for which the cost-sensitive method underperformed the two sampling methods for most cost ratios. This occurred for the contraceptive, hepatitis, and bridges1 data sets. The results for the contraceptive data set are shown in Figure 6.

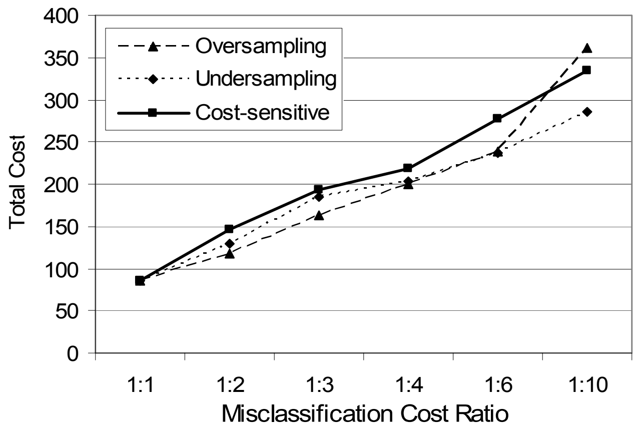

Figure 6: Results for Contraceptive

The charts for the promoters, sonar, and pendigits data sets are not provided, although their performance will be summarized shortly (in Table 2 and Figure 7). The results for the promoters data set are notable in that it is the only data set for which oversampling outperforms the other two methods for all misclassification cost ratios above 1:1 (significantly, this is a very small data set).

Table 2 summarizes the performance of the three methods over all fourteen data sets. This table specifies the first/second/third place finishes over the five cost ratios which were evaluated for each data set and method. For example, the entry for the letter-a data set in Table 2 shows that oversampling generates the best results for 3 of the 5 evaluated cost ratios, the second best results once, and the worst results once. The last row of the table totals the first/second/third place finishes for each method.

Table 2: First/Second/Third Place Finishes

Data Set	Over Sampling	Under Sampling	Cost-Sensitive
Letter-a	3/1/1	0/1/4	2/3/0
Pendigits	3/1/1	0/1/4	2/3/0
Connect-4	2/0/3	0/3/2	3/2/0
Bridges1	5/0/0	0/2/3	0/3/2
Letter-vowel	4/1/0	0/0/5	1/4/0
Hepatitis	3/1/1	2/2/1	0/2/3
Contraceptiv	3/1/1	2/3/0	0/1/4
Adult	2/0/3	3/1/1	0/4/1
Blackjack	1/1/3	1/2/2	3/2/0
Weather	0/0/5	4/1/0	1/4/0
Sonar	2/1/2	3/2/0	0/2/3
Boa1	0/0/5	3/2/0	2/3/0
Promoters	5/0/0	0/2/3	0/3/2
Coding	0/2/3	0/3/2	5/0/0
Total	**33/9/28**	**18/25/27**	**19/36/15**

Table 2 shows that it is quite rare—even for a single data set—for one method to consistently outperform, or "dominate", the other two. We do see that it does occur occasionally, since oversampling dominates in two cases (for Bridges1 and Promoters) and the cost-sensitive algorithm dominates in one case (for Coding). The last row of Table 2 indicates that undersampling performs the worst, but does not make it clear whether oversampling or the cost-sensitive algorithm performs best, since that would depend on the relative value of a first versus second place finish.

An issue with Table 2 is that it does not quantify the improvements in total cost—it treats all "wins" as equal even if the difference in total cost between methods is quite small. Figure 7 remedies this by displaying the relative reduction in total costs. This figure compares the performance of both sampling methods to the cost-sensitive learning algorithm. The figure was generated as follows. First, the total costs for each method, for a specific data set, are summed over the five misclassification cost ratios common to all of the experiments. These sums, for each of the two sampling methods, are then divided by the summed total cost for the cost-sensitive learning algorithm. This yields a normalized total cost, where a value greater than 1.0 indicates that the sampling method performs worse than the cost-sensitive algorithm (i.e., has a higher total cost) and a value less than 1.0 indicates that it performs better than the cost-sensitive algorithm.

As an example, Figure 7 indicates that for the letter-a data set, undersampling yields a total cost that is about 1.7 times that of the cost-sensitive algorithm whereas oversampling performs just slightly worse than the cost-sensitive algorithm. Because many of the data points are further above the line y=1.0 than below, the figure suggests that overall the cost-sensitive learning algorithm beats each of the other two methods. If we average the values in Figure 7 for each of the 14 datasets, we find that the average value for oversampling is 1.05 and the average value for undersampling is 1.04, which confirms the fact that the cost-sensitive learning algorithm has an edge over the other two methods.

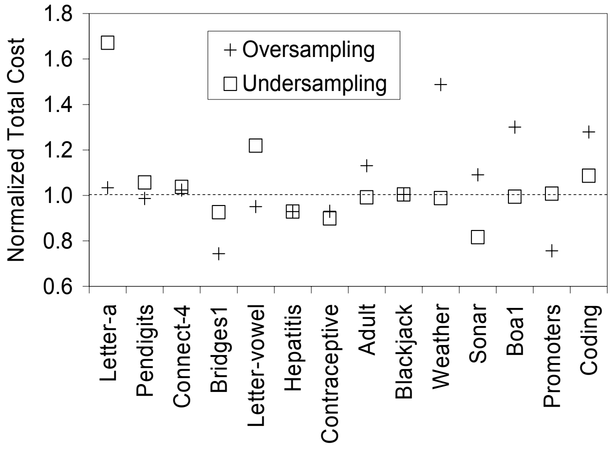

Figure 7: Performance Comparison for the Three Methods

Figure 7 can also be used to compare the performance of the two sampling methods, since one can compare the relative position of the relevant data points for each of the data sets. Overall, there does not seem to be a consistent winner. If we compute the total cost over all 14 data sets for oversampling versus undersampling, we find that on average oversampling has a total cost 1.03 times that of undersampling.

The results from Table 2 and Figure 7 show that the cost-sensitive learning algorithm does not consistently beat both, or either, of the sampling methods, although overall it does perform better (in the next section we shall see that there are some circumstances under which the advantage is relatively clear). One interesting thing to note is that the cost-sensitive algorithm rarely is the worst method. There is also no consistent winner between the two sampling methods, with undersampling performing better on some data sets and oversampling performing better on others.

6 Discussion

Based on the results from all of the data sets, there is no definitive winner between cost-sensitive learning, oversampling and undersampling. Given this, the logical question to ask is whether we can characterize the circumstances under which each method performs best.

We begin by analyzing the impact of data set size. Our study included four data sets (bridges1, hepatitis, sonar, and promoters) that are substantially smaller than the rest. If we compute the first/second/third place records for these four data sets from Table 2, we get the following results: oversampling *15/5/0*, undersampling *5/12/3* and cost-sensitive learning algorithm *0/13/7*. Based on the data underlying Figure 7, we see that for these four data sets oversampling and undersampling perform 12% and 8% better than the cost-sensitive learning algorithm and that oversampling outperforms undersampling by 3%. It makes sense that oversampling would outperform undersampling in these situations, since undersampling discards training examples, which would seem a poor strategy when dealing with very small data sets. However, it is not apparent why oversampling outperforms the cost-sensitive learning algorithm in these cases.

Next we look at the eight data sets with over 10,000 examples each (letter-a, pendigits, connect-4, letter-vowel, adult, blackjack, boa1, and coding). For these "large" data sets our results are as follows for first/second/third place finishes: oversampling *16/11/17*, undersampling *10/11/2*, and cost-sensitive *20/23/1*. The data underlying Figure 7 shows that over these eight data sets the average increase in total cost when using the sampling methods versus the cost-sensitive learning algorithm is 9% for oversampling and 13% for undersampling. Furthermore, in only one case out of these 16 comparisons does either sampling method outperform the cost-sensitive method by more than 1% (for the letter-vowel data set oversampling provides a 5% improvement). Thus, for the large data sets, the cost-sensitive learning algorithm *does* consistently yield the best results. Why might the cost-sensitive learning algorithm perform poorly for small data sets and well for good data sets? One possible explanation is that with very little training data the classifier cannot accurately estimate the class-membership probabilities—something that is critical in order to properly assign the correct classification based on the cost information. This explanation warrants further study.

Another factor worth considering is the degree to which the class distribution of the data set is unbalanced. This will impact the extent to which sampling must be used to get the desired distribution. However, the results in Tables 2 and Figure 7, which are ordered by decreasing class imbalance, show no obvious pattern and hence we cannot conclude that the degree of class imbalance favors one method over another.

7 Related Work

Previous research has compared cost-sensitive learning algorithms and sampling. The experiments that we performed are similar to the work that was done by Chen, Liaw, and Breiman [6], who proposed two methods of dealing with highly-skewed class distributions based on the Random Forest algorithm. Balanced Random Forest (BRF) uses undersampling of the majority class to create a training set with a more equal distribution between the two classes, whereas Weighted Random Forest (WRF) uses the idea of cost-sensitive learning. By assigning a higher misclassification cost to the minority class, WRF improves classification performance of the minority class and also reduces the total cost. However, although both BRF and WRF outperform existing methods, the authors found that neither one is consistently superior to the other. Thus, the cost-sensitive version of the Random Forest does not outperform the version than employs undersampling.

Drummond and Holte [8] found that undersampling outperforms oversampling for skewed class distributions and non-uniform cost ratios. Their results indicate that this is because oversampling shows little sensitivity to changes in misclassification cost, while undersampling shows reasonable sensitivity to these changes. Breiman et al. [2] analyzed classifiers produced by sampling and by varying the cost matrix and found that these classifiers were indeed similar. Japkowicz and Stephen [10] found that cost-sensitive learning algorithms outperform under-sampling and over-sampling, but only on artificially generated data sets. Maloof [12] also compared cost-sensitive learning algorithms to sampling but found that the cost-sensitive learning algorithm, oversampling and undersampling performed nearly identically. However, because only a single data set was analyzed, one can not draw any general conclusions from those results. Since we analyzed fourteen real-world data sets, we believe our research extends this earlier work and gives more weight to our conclusions.

Recent research [7] has analyzed C5.0's implementation of cost-sensitive learning and has shown that it does not always produce the desired, and expected, results. Specifically, this research showed that one can achieve lower total cost by passing into C5.0 cost information that differs from the "actual" cost information used to evaluate the classifier. In this case, the "best" cost ratio to use for learning was determined empirically, using a validation set. Clearly these results are surprising since one would expect the actual cost ratio to produce the best results. This seems to indicate that C5.0's cost-sensitive learning implementation may not be operating optimally. However, we suspect a similar phenomenon would exist with sampling—that the best class distribution for learning would not always be the one that effectively "imposes" the actual misclassification costs. This is supported by some empirical results that show that the best class distribution for learning is typically domain dependent [17].

8 Conclusion

The results from this study indicate that for data sets with class imbalance and unequal misclassification costs, there is no clear winner when comparing the performance of oversampling, undersampling and a cost-sensitive learning algorithm. However, if we focus exclusively on data sets with more than 10,000 examples, then the cost-sensitive learning algorithm consistently outperforms the sampling methods (oversampling appears to be the best method for small data sets). Note that in this study our focus was on using the cost information to improve the performance of the minority class, but in fact our results are much more general; they can be used to assess the relative performance of the three methods for implementing cost-sensitive learning. Our results also allow us to compare the performance of oversampling to undersampling, which is of significance because, as described in Section 7, previous research studies have come to contradictory conclusions about the relative effectiveness of these two sampling strategies. We found that which sampling method performs best is highly dependent on the data set, with neither method a clear winner over the other. This explains why previous studies, which typically only looked at a few data sets, came to contradictory conclusions.

There are a variety of enhancements that people have made to improve the effectiveness of sampling. Some of these enhancements include introducing new "synthetic" examples when oversampling [5], deleting less useful majority-class examples when undersampling [11] and using multiple sub-samples when undersampling such than each example is used in at least one sub-sample [3]. While these techniques have been compared to oversampling and undersampling, they generally have not been compared to cost-sensitive learning algorithms. This would be worth studying in the future.

In our research, we evaluated classifier performance using a variety of cost ratios. We did this based on the assumption that the actual cost information will be known or can be estimated. However, this is not always the case and it would be interesting to repeat our experiments and use other measures, such as the area under the ROC curve, to compare the effectiveness of the three methods when specific cost information is not known.

The implications of this research are significant. The fact that sampling, a wrapper-based approach, performs competitively—if not better—than a commercial tool that implements cost-sensitivity raises several important questions. These questions are: 1) why doesn't the cost-sensitive learning algorithm perform better given the known drawbacks with sampling, 2) are there ways to improve the effectiveness of cost-sensitive learning algorithms and 3) are we better off not using the cost-sensitivity features of a learner and using sampling instead. We hope to address these questions in future research.

9 References

[1] N. Abe, B. Zadrozny, and J. Langford. An iterative method for multi-class cost-sensitive learning. KDD '04, August 22-25, 2004, Seattle, Washington, USA, 2004.

[2] E. Breiman, J. Friedman, R. Olshen and C. Stone. Classification and Regression Trees. Belmont, CA: Wadsworth International Group, 1984.

[3] P. Chan and S. Stolfo. Toward scalable learning with non-uniform cost and class distributions: a case study in credit card fraud detection. American Association for Artificial Intelligence, 1998.

[4] N. Chawla. C4.5 and imbalanced datasets: investigating the effect of sampling method, probabilistic estimate, and decision tree structure. ICML 2003 Workshop on Imbalanced Datasets.

[5] N. Chawla, K. Bowyer, L. Hall, and W. Kegelmeyer. SMOTE: synthetic minority over-sampling technique. Journal of Artificial Intelligence Research, Volume 16, 321-357, 2002.

[6] C. Chen, A. Liaw, and L. Breiman. Using random forest to learn unbalanced data. Technical Report 666, Department of Statistics, University of California at Berkeley, 2004. <http://www.stat.berkeley.edu/users/chenchao/666.pdf>

[7] M. Ciraco, M. Rogalewski, and G. M. Weiss. Improving classifier utility by altering the misclassification cost ratio. Proceedings of the KDD-2005 Workshop on Utility-Based Data Mining.

[8] C. Drummond and R. Holte. C4.5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. Workshop on Learning from Imbalanced Data sets II, ICML, Washington DC, 2003.

[9] C. Elkan. The foundations of cost-sensitive learning. Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, 2001.

[10] N. Japkowicz and S. Stephen, The class imbalance problem: a systematic study. Intelligent Data Analysis Journal, 6(5), 2002.

[11] M. Kubat and S. Matwin. Addressing the curse of imbalanced training sets: one-sided selection. Proceedings of the Fourteenth International Conference on Machine Learning, 179-186, 1997.

[12] M. Maloof. Learning when data sets are imbalanced and when costs are unequal and unknown. ICML 2003 Workshop on Imbalanced Datasets.

[13] E. Pednault, B. Rosen and C. Apte. The importance of estimation errors in cost-sensitive learning. IBM Research Report RC-21757, May 30, 2000.

[14] J. R. Quinlan. C4.5: programs for machine learning. Morgan Kaufmann Publishers Inc., San Francisco, CA, 1993.

[15] J. R. Quinlan. Induction of decision trees. Machine Learning 1: 81-106, 1986.

[16] G. M. Weiss and H. Hirsh. A quantitative study of small disjuncts. Proceedings of the Seventeenth National Conference on Artificial Intelligence, 2000.

[17] G. M. Weiss and F. Provost. Learning when training data are costly: the effect of class distribution on tree induction. Journal of Artificial Intelligence Research, 2003.

[18] "Data Mining Tools See5 and C5.0. RuleQuest Research. <http://www.rulequest.com/see5-info.html>