C4.5, Class Imbalance, and Cost Sensitivity: Why Under-Sampling beats Over-Sampling

Chris Drummond CHRIS.DRUMMOND@NRC-CNRC.GC.CA
Institute for Information Technology, National Research Council Canada, Ottawa, Ontario, Canada, K1A 0R6

Robert C. Holte HOLTE@CS.UALBERTA.CA
Department of Computing Science, University of Alberta, Edmonton, Alberta, Canada, T6G 2E8

Abstract

This paper takes a new look at two sampling schemes commonly used to adapt machine algorithms to imbalanced classes and misclassification costs. It uses a performance analysis technique called cost curves to explore the interaction of over and under-sampling with the decision tree learner C4.5. C4.5 was chosen as, when combined with one of the sampling schemes, it is quickly becoming the community standard when evaluating new cost sensitive learning algorithms. This paper shows that using C4.5 with under-sampling establishes a reasonable standard for algorithmic comparison. But it is recommended that the least cost classifier be part of that standard as it can be better than under-sampling for relatively modest costs. Over-sampling, however, shows little sensitivity, there is often little difference in performance when misclassification costs are changed.

1. Introduction

In this paper, we experimentally study the two most common sampling schemes which are used to adapt machine algorithms to imbalanced classes and misclassification costs. We look at under-sampling and over-sampling that decrease and increase respectively the frequency of one class in the training set to reflect the desired misclassification costs. These schemes are attractive as the only change required is to the training data rather than to the algorithm itself. It is hard to justify a more sophisticated algorithm if it cannot outperform existing learners using one of these simple sampling schemes. Here, we study the sampling schemes and how they affect the decision tree learner C4.5 (Quinlan, 1993). We chose C4.5 not only because it is one of the most commonly used algorithms in the machine learning and data mining communities but also because it has become a de facto community standard against which every new algorithm is judged. More recently, as research into cost sensitivity and class imbalance have become more prevalent, C4.5 combined with under-sampling or over-sampling is quickly becoming the accepted baseline to beat (Domingos, 1999; Pazzani et al., 1994).

Using our own performance analysis technique, called cost curves (Drummond & Holte, 2000a), discussed briefly in the next section, we show that under-sampling produces a reasonable sensitivity to changes in misclassification costs and class distribution. On the other hand, over-sampling is surprisingly ineffective, often producing little or no change in performance as the training set distribution is changed. We go on to explore which aspects of C4.5 result in under-sampling being so effective and why they fail to be useful when over-sampling. We have previously shown that the splitting criterion has relatively little effect (Drummond & Holte, 2000b) on cost sensitivity. Breiman et al. (1984, p.94) observed that costs and class distribution primarily affect pruning. Still, we did not find that this was the main cause of the difference between the two sampling schemes. Over-sampling tends to reduce the amount of pruning that occurs. Under-sampling often renders pruning unnecessary. By removing instances from the training set, it stunts the growth of many branches before pruning can take effect. We find that over-sampling can be made cost-sensitive if the pruning and early stopping parameters are set in proportion to the amount of over-sampling that is done. But the extra computational cost of using over-sampling is unwarranted as the performance achieved is, at best, the same as under-sampling.

2. Cost Curves

In this section, we discuss cost curves at an intuitive level, hopefully sufficient for the reader to understand the experimental results. We refer the interested reader to our paper (Drummond & Holte, 2000a) for a more in-depth discussion of these curves. The bold continuous line in Figure 1 is an example of a cost curve produced when under-sampling the training set. Ignoring the outer (parenthetical) labels on the axes at present, this represents the error rate of C4.5 for the full range of class distributions. The x-value of each point on the curve indicated by a black circle is the fraction of the training set that is positive, $P(+)$. The y-value is the expected error rate of the decision tree grown on each of these training sets. To estimate error rates for other class distributions, for intermediate x-values, linear interpolation between points is used.

If a confusion matrix is generated for each training set distribution, we can determine the error rate on the positive and negative instances separately. Knowing these error rates allows the performance for each decision tree to be assessed across all class distributions. The performance of each individual tree is represented in Figure 1 by a dotted straight line. The performance of the tree at the training set distribution is indicated by a black circle. But this in just one point on the line, other points give the classifier's performance for quite different distributions. From this perspective a learning algorithm's cost sensitivity has two components: (1) producing a good range of classifiers suitable for different class distributions; (2) selecting the right classifier for the particular distribution. The dotted lines in Figure 1 have a wide range of gradients, indicating a wide variety of different trade-offs in the number of positive and negative examples correctly classified. So under-sampling has produced a good range of classifiers. We can decide whether it has chosen the right classifier by seeing if the line with the minimum error rate at any particular probability has been chosen. [1]

Generally this has happened, but we do notice that there is a better classifier from a probability of 0 to 0.2 and another one from a probability of 0.8 to 1. The first is the classifier that always predicts a negative, so has zero error rate when the probability of positive is zero (all instances are negative), and an error rate of one when the probability of positive is one (all instances are positive). It is represented by the diagonal continuous line going from (0,0) to (1,1). The second is the classifier that always predicts positive and forms the opposite diagonal. The triangle underneath these two lines is the majority classifier that chooses the most common class. We feel it is important to take particular note of how the learning algorithm performs with respect to this classifier. Using an algorithm with a performance that is worse than such a trivial classifier is of doubtful merit. For this data set, C4.5 is only useful when one class is less than four times more frequent than the other class.

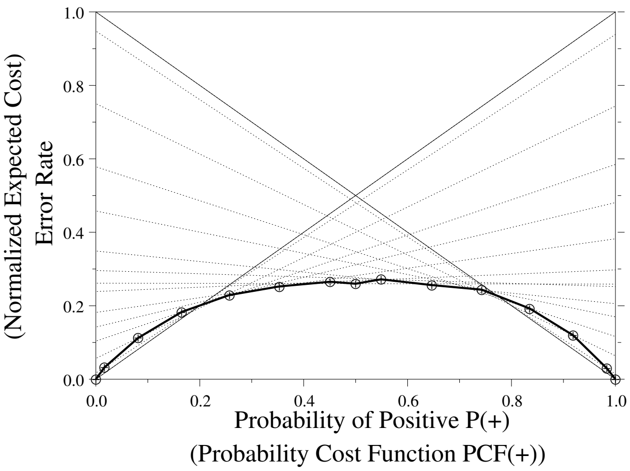

Figure 1. An Example Cost Curve

It is now commonplace to deal with class imbalances much more extreme than 4:1. But it is not class distribution alone which must be considered it is also the cost of misclassification. By applying a different interpretation to the x- and y-axes in the cost curve plot, it can be seen to represent the expected cost of a classifier across all possible choices of misclassification costs and class distributions. We begin by relabeling the y-axis in Figure 1 so that instead of representing error rate it represents expected cost. But like error rate it is normalized so that the best performance is zero and the worst performance is one. This we call the normalized expected cost. The x-axis is also relabeled to include misclassification costs. We multiply the original value, $P(+)$, by the cost of misclassifying a positive instance as negative. and normalize so that x ranges from 0 to 1. We call the normalized version of $C(-|+) * P(+)$ the "probability cost function", $PCF(+)$.

There are two simplifications worthy of note. When misclassification costs are equal, $C(-|+) = C(+|-)$, this definition of x reduces to $P(+)$ our original measure of class distribution. When the class frequencies are equal, $P(+) = P(-)$, this definition reduces to $C(-|+)/(C(-|+) + C(+|-))$. Here the misclassifica-

[1] as emphasized in Drummond and Holte (2000a) it is not necessarily optimal to have the class distribution in the training set identical to the class distribution that will be used in testing. Although this paper ignores this important issue, the conclusions drawn apply in the more general case.

tion costs are normalized so that, like the probabilities, they sum to one. In the more general case, we must consider variation in both costs and class frequencies at the same time. So in Figure 1 even if the class distribution is worse than 4:1, if the misclassification cost for the minority class is greater than that for the majority class this will tend to pull us back towards the center of the diagram. If the misclassification costs exactly balance the class distribution, we would be at the center and have potentially the maximum advantage over the majority, or least cost, classifier.

To explore the differences between the two sampling schemes we generate curves based on decision trees grown on training sets with the class distribution changed by adding or deleting instances. For under-sampling a fraction of the instances of one class are randomly deleted to form the new training set. Under-sampling is done on each training set produced by 10-fold cross-validation, which is repeated 50 times to reduce sampling variance. For over-sampling, instances of one of the classes are duplicated up to the floor of the desired ratio. The remaining fraction is chosen randomly without replacement from the training data. Thus if we had a ratio of 4.35 to 1, the instances of one class would be duplicated 4 times and then a random sample would make up the remaining 0.35. For over-sampling, 10-fold cross-validation is repeated only 10 times as the sampling variance is smaller.

3. Comparing the Sampling Schemes

In this section, we compare the performance of under and over-sampling on 4 data sets, three from the UCI Irvine collection (Blake & Merz, 1998) and one from earlier work by one of the authors (Kubat et al., 1997). We chose these data sets as they produced cost curves that captured all the qualitative features we observed in a larger set of experiments (including other UCI data sets: vote, hepatitis, labor, letter-k and glass2). For these data sets, under-sampling combined with C4.5 is a useful baseline to evaluate other algorithms. Over-sampling, on the other hand, is not to be recommended when used with C4.5. Even with large changes to the training set distribution it often produced classifiers little different in performance to the one trained on the original distribution. Thus it largely fails to achieve its very purpose.

In the following figures, we have scaled the y-axis differently for each data set to improve clarity and do not show the performance of individual classifiers for under-sampling only for over-sampling. We also include a vertical dashed line at the x-value corresponding to the fraction of positive examples in the data set. The bold dashed line in Figure 2 shows the performance of C4.5 using under-sampling on the sonar data set. Sonar has 208 instances, 111 mines and 97 rocks, with 60 real valued attributes. Under-sampling produces a cost curve that is reasonably cost sensitive, it is quite smooth and largely within the lower triangle that represents the majority, or least cost, classifier. The bold continuous curve represents over-sampling. What is most striking about it is that it is almost straight. The performance varies little from that at data set's original frequency, indicated by the vertical dotted line. If we look at the dotted lines, representing the performance of the trees generated using over-sampled training sets, we see remarkably little difference between them.

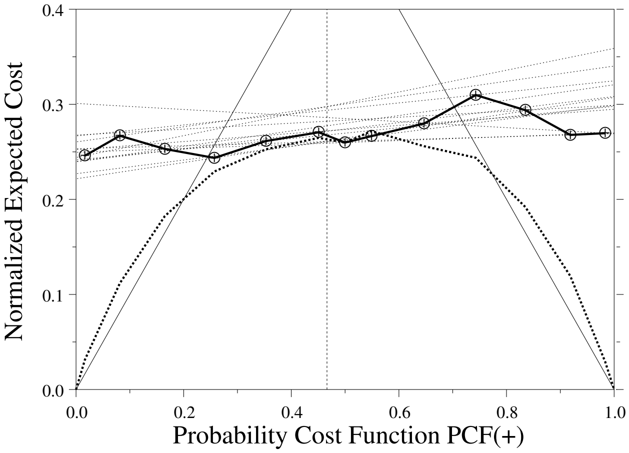

Figure 2. Sonar: Comparing Sampling Schemes

Figure 3 shows the different sampling schemes for the Japanese credit data set. It has 690 instances, 307 positive and 383 negative, with 15 attributes, 6 real and 9 nominal. For under-sampling, again the curve is reasonably smooth and this time remains completely within the triangle representing the majority, or least cost, classifier. It is still noticeable, however, for a PCF(+) value of 0 to 0.1 and 0.9 to 1.0 that it is only marginally better. So for class or cost ratios of greater than 9:1 there is little to be gained over using a trivial classifier. Here the bold curve for over-sampling shows some sensitivity to costs. The dotted lines, representing individual trees, show a reasonable diversity. But overall the expected cost, particularly when misclassification costs or class frequencies are severely imbalanced, is a lot worse than when under-sampling.

Figure 4 shows the under-sampling schemes for the breast cancer data set from the Institute of Oncology, Ljubljana. It has 286 instances, 201 non-recurrences and 85 recurrences, with 9 nominal attributes. For this data set, C4.5 only just outperforms the least cost clas-

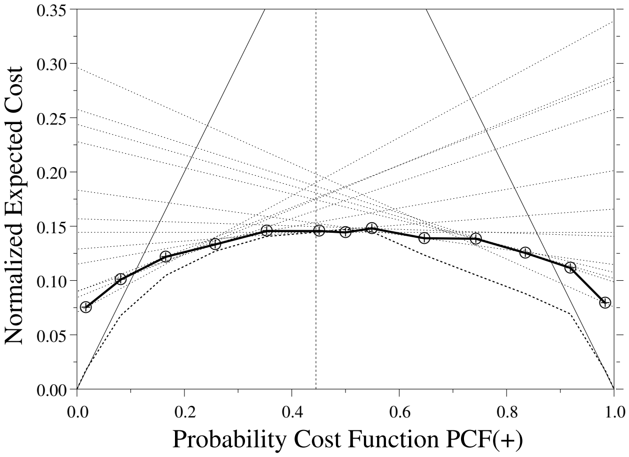

Figure 3. Credit: Comparing Sampling Schemes

sifier at the original frequency of the data set. There is a reasonable diversity in the individual trees, but this produces little real advantage in using C4.5, except perhaps if the misclassification costs balance the difference in class frequencies. Still under-sampling does not misbehave badly, the curve mainly stays within the triangle for the least cost classifier.

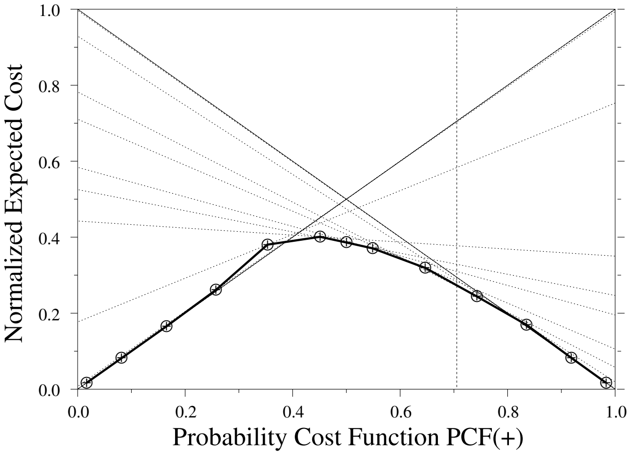

Figure 4. Breast Cancer: Under-sampling

Figure 5 compares the costs curves for under- and over-sampling on this data set. The bold over-sampling curve tracks that for under-sampling when $PCF(+)$ exceeds the frequency in the data set. However, when $PCF(+)$ is less than the original frequency the curve quickly flattens out and the dotted lines representing individual trees become bunched.

Figure 6 shows the different sampling schemes for the sleep data set. It has 840 instances, 700 1's and 140 2's, with 15 real valued attributes. Here, under-sampling produces a reasonably cost sensitive curve, that is smooth and stays within the triangle of the least cost classifier. But there is little diversity in the individual trees produced when over-sampling and the resulting cost curve is quite straight. Interestingly though when the costs or class frequencies become extremely imbalanced we suddenly start to see some improvement in expected cost.

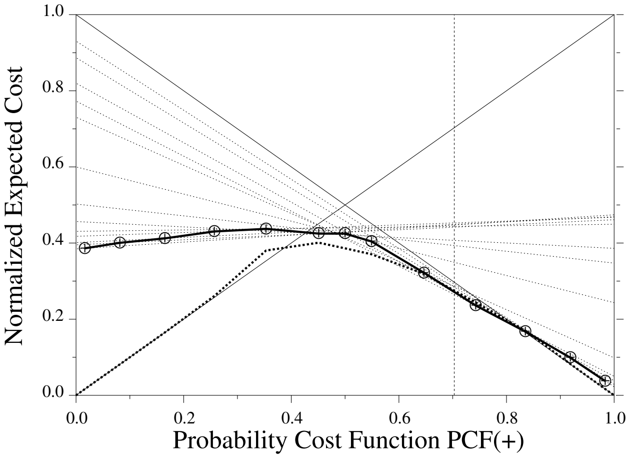

Figure 5. Breast Cancer: Comparing Sampling Schemes

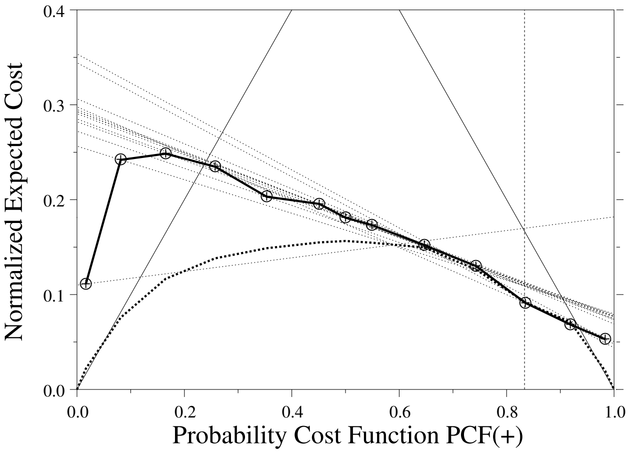

Figure 6. Sleep: Comparing Sampling Schemes

4. Investigating Over-sampling Curves

In the previous section, it was apparent that the cost sensitivity for over-sampling was much less than that for under-sampling. At least with hindsight this is not surprising. We might expect over-sampling to limit the amount of pruning. Increasing the number of instances of the already larger class on a particular branch should make pessimistic pruning reluctant to remove that branch. For the credit data set, though, over-sampling did show some degree of cost sensitivity, pruning and the early stopping criterion were still having some effect. Figure 7 shows the cost curve and lines for individual trees once pruning is disabled. There is still some variability in the performance of individual trees but it is much reduced, resulting in a much straighter cost curve. The variability is further reduced when the stopping criterion is decreased to 1, as shown in Figure 8. The cost curve is now almost straight except at the far right hand side where the expected cost decreases appreciably.

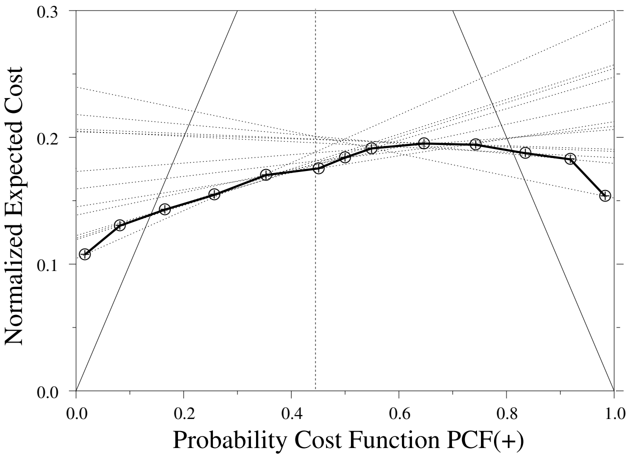

Figure 7. Credit: Disabling Pruning

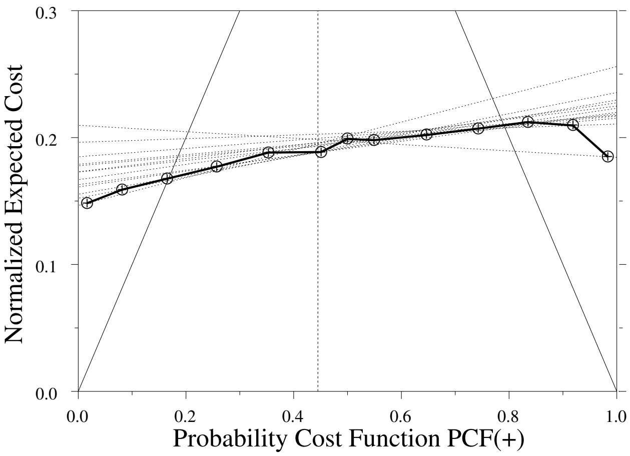

Figure 8. Credit: Disabling Stopping Criterion

The over-sampling cost curve for the sleep data set also exhibited this same tendency, being largely straight except when $PCF(+)$ is much less than the original frequency of the data set. This curvature was traced to the special stopping criterion for continuous attributes included in C4.5. The code for this criterion is shown in Figure 9. As the number of known items increases, the minimum number of instances allowed on each side of the split increases from the normal stopping criterion to a maximum of 25. As we are looking at two class problems (MaxClass = 1) and we have set the stopping criterion to 1 (MINOBJS = 1) this has no effect until the number of known items reached 20. But as more and more samples are added through oversampling this minimum number increases, thus preventing the growth of some branches of the tree. If this feature is disabled, making the minimum number 1, we remove the down turn of the cost curve, as shown for the sleep data set in Figure 10.

```
MinSplit = 0.10 * KnownItems / (MaxClass + 1);
if ( MinSplit <= MINOBJS ) MinSplit = MINOBJS;
else if ( MinSplit > 25 ) MinSplit = 25;
```

Figure 9. Special Stopping Criterion

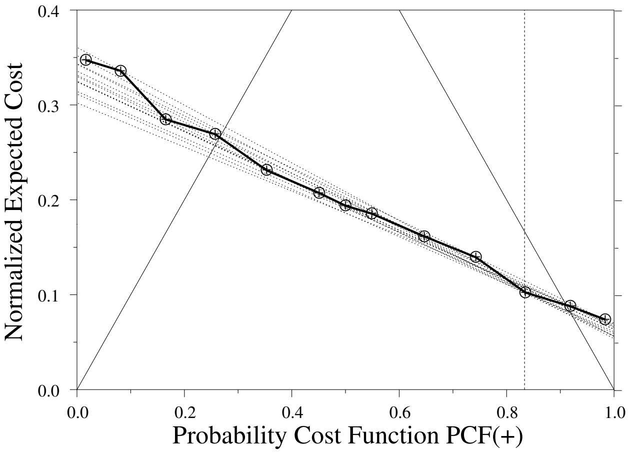

Figure 10. Sleep: Disabling Special Stopping Criterion

Generally, for data sets where there was appreciable pruning on the tree at the original frequency oversampling produced some overall cost sensitivity. This was true for the both the credit and breast cancer data sets. Turning off pruning and the standard stopping criterion resulted in almost flat costs curves. For data sets with continuous attributes (all bar the breast cancer data set) disabling the special stopping criterion also removed the small additional sensitivity shown at the ends of the curves. Surprisingly though for the sonar data set, where the cost curve was initially flat, removing this special stopping criterion actually caused the curve to turn up at the end. In this case, with many extra instances added by over-sampling, the trees appeared to be much more complex but somewhat less accurate across the whole distribution. Why this occurred is the subject of future work.

Throughout this paper we have not shown end points for the cost curves for over-sampling, values when PCF(+) is one or zero. For under-sampling, the limit at either end is to have a single class. In this case C4.5 will simply choose that class and therefore have a normalized expected cost of zero. For over-sampling, the minority class never disappears entirely, the ratio just gets larger and larger. Experiments increasing the degree of over-sampling were stopped due to the excessive size of the training set. Using very large internal weights (c. 100,000,000) does produce just the majority classifier. All the instances appear to be of one class, an error made by trying to add two numbers with vastly different exponents but both represented as floats.

5. Investigating Under-sampling Curves

If disabling pruning and the early stopping criterion resulted in straighter curves for over-sampling, it would be tempting to think it might have the same effect when under-sampling. But this turns out not to be the case.

Figure 11 shows cost curves for the sonar data set using under-sampling as the different features of C4.5 are disabled. Surprisingly, we see no real change when we disable pruning, nor do we see a change when the early stopping criterion is reduced to one. All the curves are indistinguishable from the original curve discussed in Section 3. The apparently bold line, the lowest curve in Figure 11, is these three curves overlapping. We do get a change by disabling the special stopping criterion, the dashed line, but it is very small. We get a reasonably large change when we disable the threshold for continuous attributes (log(# distinct instances)/# instances) added in release 8 (Quinlan, 1996). But the major change is observed within the triangle defined by the least cost classifier. Although the curves stray further outside this triangle, they still maintain roughly the same shape and certainly do not become as straight as those produced when over-sampling.

Pruning has a much larger effect on classifier performance for the Japanese credit data set. When pruning is turned off, we get the dashed line shown in Figure 12. Setting the stopping criterion to one also has a sizeable effect. But again both effects show up mainly while the curve is inside the least cost classifier triangle and the basic shape of the curves is maintained. For this data set, disabling the special stopping criterion and the release 8 threshold produce no appreciable effect. So disabling these features did not produce the same performance for under-sampling as over-sampling. But if we represent misclassification costs and class frequencies by means of internal weights within C4.5, disabling these features does seem to make the difference. Up-weighting, analogous to over-sampling, increases the weight of one of the classes keeping the weight of the other class at one. Down-weighting, analogous to under-sampling, decreases the weight of one of the classes keeping the weight of the other class at one. Figure 13 compares the performance of the sampling schemes (the continuous lines) to the weighting (the dashed lines) for the sonar data set. The curve for up-weighting is very close to that for over-sampling, perhaps not surprising as in over-sampling we duplicate instances. The curve for down-weighting is close but sometimes better than that for under-sampling. This was also true for the other data sets.

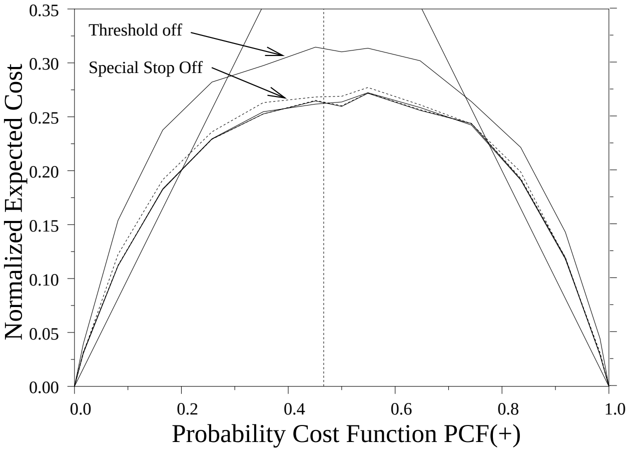

Figure 11. Sonar: Under-sampling

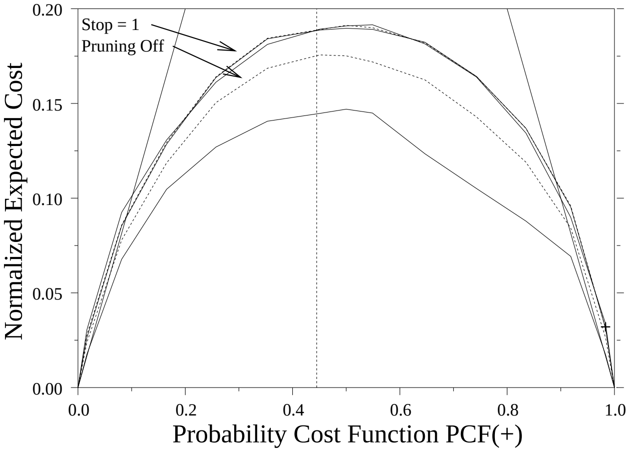

Figure 12. Credit: Under-sampling

Figure 14 shows the effect of disabling various factors using the down-weighting scheme. To produce this set of curves we first disabled the threshold added in release 8 as this had a strong interaction with down-weighting. Now we can see that for the sonar data set turning off pruning and then the stopping criterion does produce a curve that is very straight. Although,

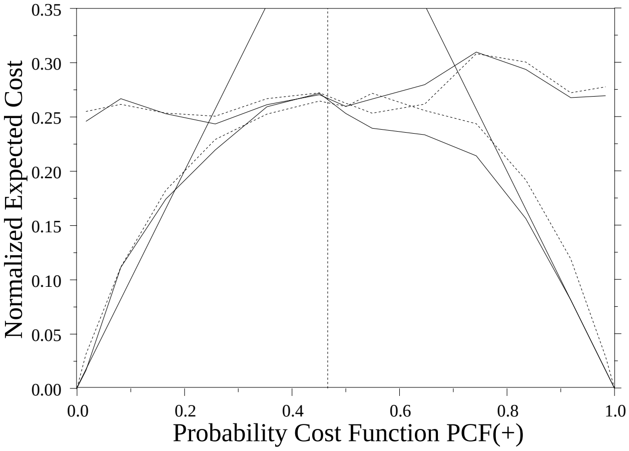

Figure 13. Sonar: Comparing Weighting and Sampling

as we noted before when over-sampling the sonar data set, if the special stopping criterion is removed the line curves up at the ends.

If the performance curves of under-sampling and down-weighting are similar and these internal factors have the required effect when down-weighting why do they seem not to have the same effect when under-sampling? The answer seems to be that much of the cost sensitivity when under-sampling comes from the actual removal of instances. When we turned off many of the factors when down weighting, the branch was still grown and the region still labeled. When the instances are removed from the training set this cannot happen.

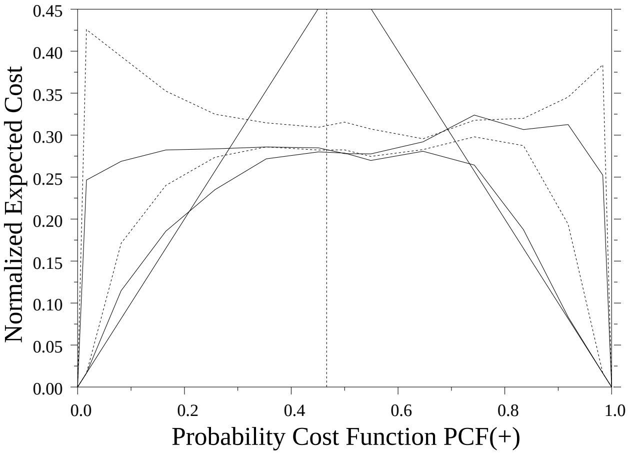

Figure 14. Sonar: Disabling Factors when Weighting

6. Improving Over-sampling

We have seen that pruning and the stopping criterion often have a large impact on cost sensitivity. But simply disabling these factors did not make under-sampling as ineffective as over-sampling. Here we show that increasing the amount of pruning and the influence of the early stopping criterion in relation to the number of duplicates in the training set when over-sampling does have a strong effect. Figure 15 illustrates the effect on the sonar data set.

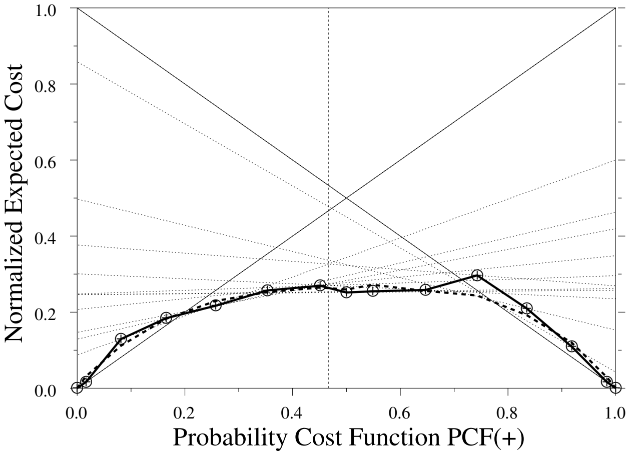

Figure 15. Sonar: Changing Defaults for Over-sampling

Our change to the default values are based on the ratio used when over-sampling the training set. The early stopping criterion (the -m switch in C4.5) is set to 2 times this ratio and the pruning confidence factor (the -c switch in C4.5) is set to 0.25 divided by the ratio. So, if one class is over-sampled by 2.5 times then the stopping criterion is 5 and the confidence factor is 0.1. This produces the bold continuous curve in Figure 15 for the sonar data. This is remarkably similar to the dashed curve for under-sampling, although there is some difference on the right hand side of the figure. On the other data sets the difference in much smaller. In fact, on the credit data set as shown in Figure 16 the difference in negligible.

7. Discussion

Generally we found that using under-sampling established a reasonable baseline for algorithmic comparison. However, one problem with under-sampling is that it introduces non-determinism into what is otherwise a deterministic learning process. The values obtained from cross-validation give the expected performance of a classifier based on a random subsample of the data set. With a deterministic learning process any variance from the expected performance that occurs when deploying the classifier is largely due to testing on a limited sample of the true data. But for under-sampling there is also variance due to non-determinism

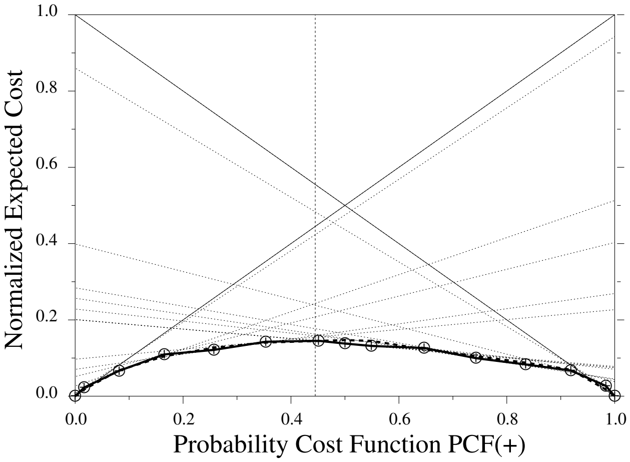

Figure 16. Credit: Changing Defaults for Over-sampling

of the under-sampling process. If our measure of success is purely the difference in the expectations then this not important. But the choice between two classifiers might also depend on the variance and then using under-sampling might less desirable.

8. Related Work

Most relevant to this paper are previous works comparing under-sampling and over-sampling. This survey will focus on the results for variants of C4.5 since they are most closely related to the present paper. Domingos (1999) reports that, on 2-class problems, C4.5-Rules produces lower cost (better) classifiers using under-sampling than it did using over-sampling. Ling and Li (1998) compare over-sampling and under-sampling for boosted C4.5 (with certainty factors added) on three different direct-marketing data sets and report that under-sampling produces a larger (better) lift index, although extreme over-sampling performs almost as well.

Japkowicz and Stephen (2002) compare random and systematic methods of over-sampling and under-sampling. In the artificial domains studied, under-sampling was ineffective at reducing error rate. Over-sampling was effective, but most effective was supplying an appropriate misclassification cost matrix. The reason for this study coming to the opposite conclusion of all other studies is not clear.

9. Conclusions

In this paper, we have used cost curves to explore the interaction of over and under-sampling with the decision tree learner C4.5. We have shown that under-sampling produces a reasonable sensitivity to changes in misclassification costs and class distribution. On the other hand, over-sampling is surprisingly ineffective, often producing little or no change in performance. But it is noteworthy that representing these changes internally by down-weighting gives the best performance overall.

References

Blake, C. L., & Merz, C. J. (1998). UCI repository of machine learning databases, University of California, Irvine, CA . www.ics.uci.edu/∼mlearn/MLRepository.html.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and regression trees*. Belmont, CA: Wadsworth.

Domingos, P. (1999). MetaCost: A general method for making classifiers cost-sensitive. *Proceedings of the Fifth International Conference on Knowledge Discovery and Data Mining* (pp. 155–164).

Drummond, C., & Holte, R. C. (2000a). Explicitly representing expected cost: An alternative to ROC representation. *Proceedings of the Sixth International Conference on Knowledge Discovery and Data Mining* (pp. 198–207).

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

Japkowicz, N., & Stephen, S. (2002). The class imbalance problem: A systematic study. *Intelligent Data Analysis*, *6*.

Kubat, M., Holte, R. C., & Matwin, S. (1997). Learning when negative examples abound: One-sided selection. *Proceedings of the Ninth European Conference on Machine Learning* (pp. 146–153).

Ling, C., & Li, C. (1998). Data mining for direct marketing: problems and solutions. *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining* (pp. 73–79).

Pazzani, M., Merz, C., Murphy, P., Ali, K., Hume, T., & Brunk, C. (1994). Reducing misclassification costs. *Proceedings of the Eleventh International Conference on Machine Learning* (pp. 217–225).

Quinlan, J. R. (1993). *C4.5 programs for machine learning*. San Mateo, CA: Morgan Kaufmann.

Quinlan, J. R. (1996). Improved use of continuous attributes in C4.5. *Journal of Artificial Intelligence Research*, *4*, 77–90.