Matching Model Versus Single Model: A Study of the Requirement to Match Class Distribution Using Decision Trees

Kai Ming Ting

Gippsland School of Computing and Information Technology,
Monash University, Australia
kaiming.ting@infotech.monash.edu.au

Abstract. A tacit assumption in classifier induction is that the class distribution of the training set must match the class distribution of the test set. A direct implementation is to retrain a model using a data set with matching class distribution every time the operating condition changes (i.e., the matching model). The alternative is to modify the decision rule of a previous trained model to the new operating condition. The latter is the single model approach commonly used and recommended by many researchers. In this paper, we argue with empirical support using decision trees that learning using the matching class distribution is desirable. We also make explicit the differences and limitations of the two methods for the single model approach: rescaling and thresholding.

1 Introduction

In order to produce the best classifier for the test condition, conventional wisdom requires the class distribution of the training data to match that of the testing data. Strict compliance to this principle requires one to retrain a new model using a matching (class) distribution whenever the test class distribution changes. However, many opt to modify the decision output of a previously trained model to adjust to the new condition. We call the first approach *the matching model approach* and the second the *single model approach*. The second approach has a practical advantage because it uses a single model only for all operating conditions. Thus, many researchers (e.g., [2, 4, 7, 14]) use the single model approach by default or recommend others to use it.

However, despite its common application, it is unclear if the performance of the single model approach will match that of the matching model approach. This paper aims to answer this open question.

Both approaches have multiple variations. For the matching model approach, one may collect a separate training set with matching class distribution; or retrain using the same training set by re-weighting the instances according to new class distribution; or use over-sampling or under-sampling to reproduce a new training set from the original one. For the single model approach, one may *rescale* the output probability by a ratio of the changed prior probability [2, 4, 7]; or use *thresholding* in which one determines the appropriate threshold of the previously trained model using ROC or cost curves [9, 12].

Previous analyses on rescaling have not taken into account the type of model employed. There are two types of models, depending on whether the learning algorithm is modeling posterior probability or class-conditional probability density function (or class density model, hereafter). Some suggest that one should rescale a model regardless of its type. We show that different model types can be treated differently when rescaling. Examples of posterior probability models are decision trees, logistic regression and neural networks; and class density models are Naive Bayesian classifiers and linear discriminant functions.

In addition, we show that model sensitivity is a necessary condition when considering rescaling a model. An algorithm that is insensitive to prior probability produces models with the same structure when trained using different class distributions. We reveal that a previous analysis on rescaling [7] is based on an implicit assumption of model insensitivity.

Because of differing and sometimes conflicting factors as described above, it is not always clear which is the most appropriate approach to produce the best model in accordance to the requirement of matching distribution. It is thus important to have a clear understanding of the relationship between different factors and provide a guide as to what to apply under different scenarios. This paper contributes toward this end.

Specifically, we investigate the following issues:

- Does the type of model influence what we do when rescaling?
- The limitations of rescaling.
- Does the single model perform comparably to the matching model?

We provide analyses on the first two issues and a comprehensive experiment comparing the two approaches.

Despite the now known fact that optimal learning can be achieved by using unmatched class distribution, it is still important to know which of the two matched class distribution approaches perform better because most of the current research is comparing optimal learning against the commonly used single model approach rather than the matching model approach. We show in this paper that the latter can be significantly better than the former.

The issue of optimal learning using unmatched class distribution has been explored elsewhere [12, 14]. This issue is not discussed in this paper. Another related issue is: what does one do when there is insufficient training data for reasonable estimation in the modeling process? For example, given a fixed training set with class distribution 1:99 while the test distribution is known to be 99:1. In this paper, we assume the training set has sufficient data to make reasonable estimation and focus on the stated issues, and leave the issue of insufficient data to be discussed elsewhere.

Bayes decision rules form the basis of this paper and are described in Section 2. We show that rescaling a class density model is simpler than rescaling a posterior probability model. Section 3 discusses the limitations of rescaling and its implicit assumption in a previous analysis. Section 4 describes thresholding and representing the performance of a model using cost curves. Section 5 shows the experimental result comparing the single model approach with the matching model approach. We provide a discussion of the related issues in Section 6 and conclude in the last section.

2 Bayes Decision Rules and Rescaling

Let $x =< x_1, x_2, ..., x_n >$ be a vector of attribute-values representing an example in a classification task; and $C \in \{1, 2\}$ is the class label of the task.

Given a model that produces the posterior probability $P(C|x)$ for a test example x, the Bayes decision rule used to make a final prediction is as follows:

$$\begin{gathered}\text{Predict class 1 if } P(1|x) > P(2|x), \\ \text{otherwise predict class 2.}\end{gathered} \tag{1}$$

Alternatively, if it is the class density function $P(X|C)$ of the data that is being modeled, then the rule can be similarly described as:

$$\begin{gathered}\text{Predict class 1 if } P(x|1)P(1) > P(x|2)P(2), \\ \text{otherwise predict class 2.}\end{gathered} \tag{2}$$

$P(C)$ is the prior probability for class C.

The second equation is derived from the first by applying the Bayes rule

$$P(C|x) = \frac{P(x|C)P(C)}{P(x)}. \tag{3}$$

$P(x)$ can be ignored in the Bayes decision rules because it is a constant independent of class.

Let M be the factor of the changed ratio of $P(C)$.

$$M = \frac{P'(1)}{P'(2)} : \frac{P(1)}{P(2)}, \tag{4}$$

where $P(C)$ and $P'(C)$ are the prior probabilities for class C in the training set and testing set, respectively.

Rescaling as suggested by some authors [4, 14] can be done by simply rescaling $P(C|x)$ of the previously trained model by the ratio of the changed $P(C)$. Following this suggestion, Equation (1) can be rescaled to the test condition as follows.

$$\begin{aligned} P(1|x)\frac{P'(1)}{P(1)} &> P(2|x)\frac{P'(2)}{P(2)}, \\ \frac{P(1|x)}{P(2|x)} &> \frac{P(1)}{P(2)}\frac{P'(2)}{P'(1)}, \\ \frac{P(1|x)}{P(2|x)} &> \frac{1}{M}. \end{aligned} \tag{5}$$

The same rescaling can be applied to both posterior probability models and class density models. The following rescaling for the class density model shows that it produces the same result as in the case for posterior probability model. Rescaling Equation (2) gives:

$$\begin{aligned} P(x|1)P(1)\frac{P'(1)}{P(1)} &> P(x|2)P(2)\frac{P'(2)}{P(2)} \\ \frac{P(x|1)P(1)}{P(x|2)P(2)} &> \frac{P(1)}{P(2)}\frac{P'(2)}{P'(1)}, \\ &> \frac{1}{M} \end{aligned} \tag{6}$$

However, Equation (6) can be re-written and simplified as follows.

$$\frac{P(x|1)}{P(x|2)} > \frac{P'(2)}{P'(1)} \tag{7}$$

This means that *rescaling a class density model can be done without knowing the training prior probability.*

A caveat is in order here. The analysis thus far assumes that the algorithm, whether it is producing class density models or posterior probability models, is totally insensitive to the training prior probability. If this assumption does not hold, then the rescaled model cannot guarantee to produce the same output as that from the retrained matched model. We will have further discussion on this issue in the next section.

3 The Limitations of Rescaling

The above analyses do not take practical considerations into account which limit its applicability in practice. Here we list two practical constraints and reveal a limiting implicit assumption in a previous analysis on rescaling.

(a) Rescaling does not adapt to the output range of the model. For example, if the output range for $P(1|x)$ is between 0.3 and 0.7 (which is the typical range of values obtained using the Laplace estimate in decision trees in most of the experiments we conducted), then rescaling for $M > 2$ will always predict class 1 using (5)! An example is shown in Figure 1 in which rescaling a model trained with natural distribution performs poorly with respect to the models trained from the matching distribution using the default threshold = 0.5 in each operating condition $1 \leq M \leq 10$.

(b) Rescaling relies on accurate probability estimation. Many classifiers are poor probability estimators [10]. As a result, rescaling becomes an unreliable method to provide accurate prediction.

(c) The following analysis on rescaling has an implicit assumption that the learning algorithm is totally insensitive to prior probability.

"Theorem 1: To make a target probability threshold p' correspond to a given probability threshold p, the number of negative examples in the training set should be multiplied by $\frac{p'}{1-p'}\frac{1-p}{p}$." [7]

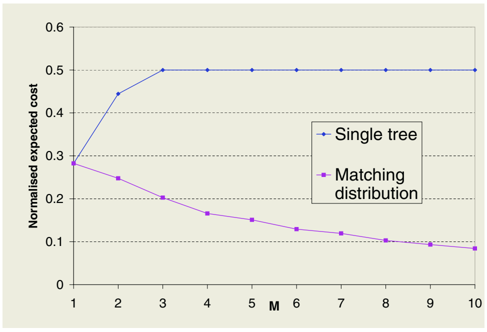

Fig. 1. An example of rescaling a single tree compared to trees trained from matching distribution (using threshold = 0.5) in the coding data set. Single tree is trained from the natural distribution ($M = 1$) and then rescaled to different operating conditions, $M > 1$.

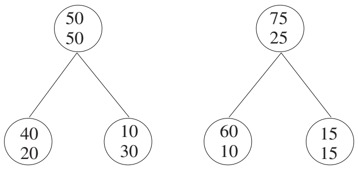

Fig. 2. Retrain a model using a different class distribution: an example.

Two models with different thresholds will produce the same decision, if and only if the learning algorithm produces models with the same structure when trained using different class distributions. Figure 2 shows a simple example: a model is initially trained using 50:50 distribution; and then a second model is trained with the *same split* using a 75:25 distribution. Assuming p and p' are the corresponding thresholds in these models that produce the same prediction. Using Elkan's formulation, the relationship can be computed as $p' = 1.5p$. This is interpreted as: for the first model, predict class 1 if the output probability is more than $p = 0.5$ otherwise predict class 2; the second model can then use $p' = 0.75$ to produce the same prediction.

This is only true if the algorithm is totally insensitive to prior probability, or in the case of decision trees, the splitting criterion is totally insensitive to prior probability. The model insensitivity assumption is not true with decision tree learning algorithms, in general, as shown in the experimental results in [12, 14], for example.

It is important to point out that rescaling does not provide the kind of fine adjustment as in thresholding because thresholding adapts to the output range of the model, and can be determined for a wider range of M value. Thresholding using exactly the same model comes close to that of the matching distribution; thus, it is much better then rescaling as shown in Figure 1 (the plot for thresholding is eliminated to increase readability.) In addition, thresholding does not rely on accurate probability estimation, but only requires the class ranking to be correct (see [10] for a discussion of probability estimation trees.) We will describe thresholding in the next section.

4 Thresholding and Cost Curves

The key difference between thresholding and rescaling is that each threshold is determined empirically rather than an adjustment through a decision rule (based on Bayes rule), as described in the last two sections.

A decision threshold is the cut-off level used to decide the final prediction of a classification model. In a two-class problem, the final prediction is class positive if the model's posterior probability of a test example is above the threshold; otherwise it is class negative. When the threshold is changed, the model's performance also changes. The best threshold is usually determined empirically for each testing condition using a holdout set.

The procedure used to determine the best threshold can be further aided by either a cost curve [5] or a ROC curve [9], which represents the performance of a model. We use cost curves in this paper and describe the pertinent detail below.

The normalised expected cost of a classifier can be expressed in terms of true positive rate (TP), false positive rate (FP) and probability-cost function (PCF); it is defined by [5] as

$$\begin{aligned} NEC &= (1 - TP)PCF + FP(1 - PCF) \qquad (8) \\ &= (1 - TP - FP)PCF + FP, \\ \text{where } PCF &= \frac{P'(1)C'(2|1)}{P'(1)C'(2|1) + P'(2)C'(1|2)}. \end{aligned}$$

For the purpose of discussion in this paper, $PCF = P'(1)$ since we assume $C'(2|1) = C'(1|2)$, where $C(a|b)$ is the cost of misclassifying an example of class b as belonging to class a. This assumption is solely for ease of discussion and the result is by no means restricted to the cost-insensitive case only since doubling $P'(1)/P'(2)$ has the same effect of doubling $C'(2|1)/C'(1|2)$.

The performance of a classification model that uses a fixed decision threshold is represented by a pair $\{TP, FP\}$. Given the pair, it can be represented as a line in the cost space, which consists of the normalised expected cost in the y-axis and PCF in the x-axis, indicated by the second linear equation. Because both are normalised, they range from 0 to 1. Different cost lines are obtained by varying the decision threshold of the same model, as shown in Figure 3a. The cost curve representing the performance of the single model that uses varying decision thresholds is the lowest envelop of all cost lines produced by the model. Examples of cost curves produced from a single tree and multiple trees, using thresholding, are given in Figure 3b.

We compare the single model approach with the matching model approach, both using thresholding, in the following section.

5 Single Tree vs Trees Trained from Matching Distribution

5.1 Experimental Settings

The experiment is aimed to compared the performance of a single tree to that of trees trained from a matching distribution for different testing conditions. A cost curve is produced from each case; the latter uses 100 trees, one for each of the testing conditions (denoted by an integral value of M for 1 to 100), whereas the former is using one tree for all conditions.

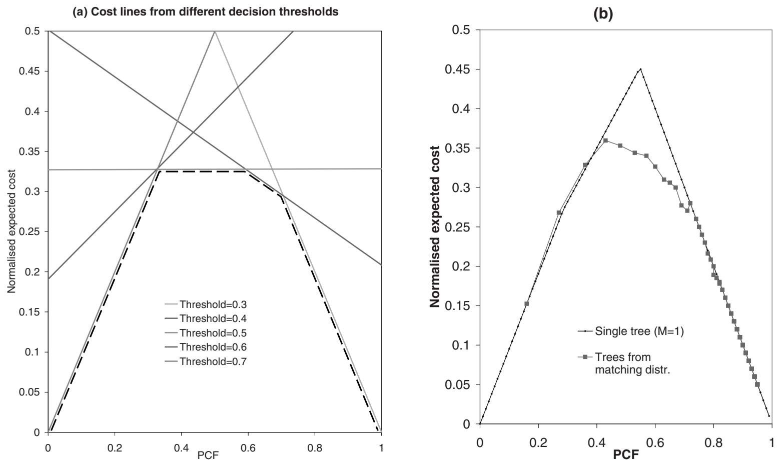

Fig. 3. (a) Cost lines and curves: each cost line is produced using a fixed decision threshold. The *cost curve* for the model is the lowest envelop of all the cost lines for the model, indicated as the dotted line. (b) Cost curves for a single tree trained from $M = 1$ and trees trained from matching distribution in the flare data set. Note that the cost curve for trees trained from matching distribution spans over the range corresponding to $1 \leq M \leq 100$, and not the entire range.

We use the decision tree induction algorithm C4.5 [11] and its default setting in all experiments, while taking the following modifications into consideration. The algorithm is modified to take M into consideration in the training process. For example, with M=2, every minority class training instance will be weighted twice as high as every majority class instance. Cost-sensitive pruning is done in accordance with the modified weights, though the default pruning method of C4.5 is used unaltered. We also added the DKM splitting criterion [3]. We report the results of pruned trees using the gain ratio criterion unless stated otherwise. Note that the probability estimates of decision trees are smoothed using the Laplace-estimate (as used by [10, 12]).

We use thirteen data sets obtained from the UCI repository [1]. The minority prior spans from 50% to about 1%, that is from balanced to highly skewed distributions; and the data sizes range from 1000 to about 49000. There are seven data sets which have more than two classes and they are converted to two classes (marked with $'$). A stratified 10-fold cross-validation is conducted for each data set.

To compute cost curves, we use an algorithm provided by [9] to obtain all pairs of $\{TP, FP\}$ in one pass through a test set for all the different thresholds of a model. Note that although the test set we used to produce the cost curves is derived from the natural distribution, we can still compute the expected cost for different testing or operating conditions (denoted by PCF or its equivalent M) using Equation (8) since the resultant cost curve, like the ROC curve, is independent of prior probability and cost.

Table 1. Average AUCC for a single tree and matching trees trained for the testing condition $1 \leq M \leq 100$. Bold face indicates the lowest value in each data set.

Data set		Gain Ratio			DKM		
	%Minority	Single Tree		Matching	Single Tree		Matching
	prior	$M = 1$	$M = M_b$	Trees	$M = 1$	$M = M_b$	Trees
Coding	50.0	**3.39**	**3.39**	3.69	**3.45**	**3.45**	3.71
Kr-vs-kp	47.8	**0.26**	**0.26**	0.42	**0.26**	**0.26**	0.33
Abalone′	31.7	7.22	6.96	**6.58**	7.30	6.73	**6.62**
German	30.0	7.37	**5.77**	6.60	7.45	**5.50**	6.50
Adult	24.1	7.22	7.34	**6.01**	7.32	7.09	**6.40**
Splice′	24.0	2.16	2.65	**1.49**	2.28	2.46	**1.53**
Solar Flare	15.7	13.68	12.81	**12.62**	13.91	12.64	**12.44**
Satellite′	9.7	14.96	13.68	**13.25**	14.44	**13.71**	13.72
Pendigits′	9.6	2.65	1.93	**1.57**	2.37	2.39	**2.08**
Hypothyroid	4.8	4.94	4.03	**3.44**	3.76	4.14	**3.38**
Letter-a′	3.9	2.92	**2.28**	2.32	3.19	2.32	**2.15**
Nursery′	2.6	1.91	**1.27**	1.39	1.91	**1.27**	1.39
Nettalk-stress′	1.1	32.75	**21.94**	23.65	32.75	**20.19**	20.24
Win:loss ratio							
	wrt matching trees	2:11	6:7		2:11	6:7	
	wrt M_b tree	2:9			3:8		
	wrt DKM tree	7:3	5:6	6:6			

The area under cost curve (AUCC) is used as a generic measure to compare the performance of different algorithms under all operating conditions, like that in the ROC curve [12, 14]. Here we limit ourselves to operating conditions $1 \leq M \leq 100$ for models trained from naturally skewed class distribution data sets (where class 1 in Equation (4) represents the minority class.) $M > 1$ allows us to concentrate on situations that bias the minority class which often occurs in practice. This is equivalent to assigning higher cost to the minority class in cost-sensitive learning.

All AUCC figures reported refer to a partial area under the cost curve. An algorithm which has smaller AUCC is performing better in general under those operating conditions. AUCC is computed by integrating over M rather than PCF because a precise value of performance is available for each M.

5.2 Experimental Results

Table 1 shows the average result for both the gain ratio and DKM splitting criteria. In addition to a single tree trained from the natural distribution, the results of a single tree trained from the balanced distribution (i.e., the M_b tree) are also provided. The last three rows show the ratios of the number of data sets in which one approach wins and loses with respect to (wrt) the other.

The gain ratio results, in the '$M = 1$' column, clearly show that a single tree trained from the natural distribution performs worse than trees trained from a matching distribution with a win:lose ratio of 2:11.

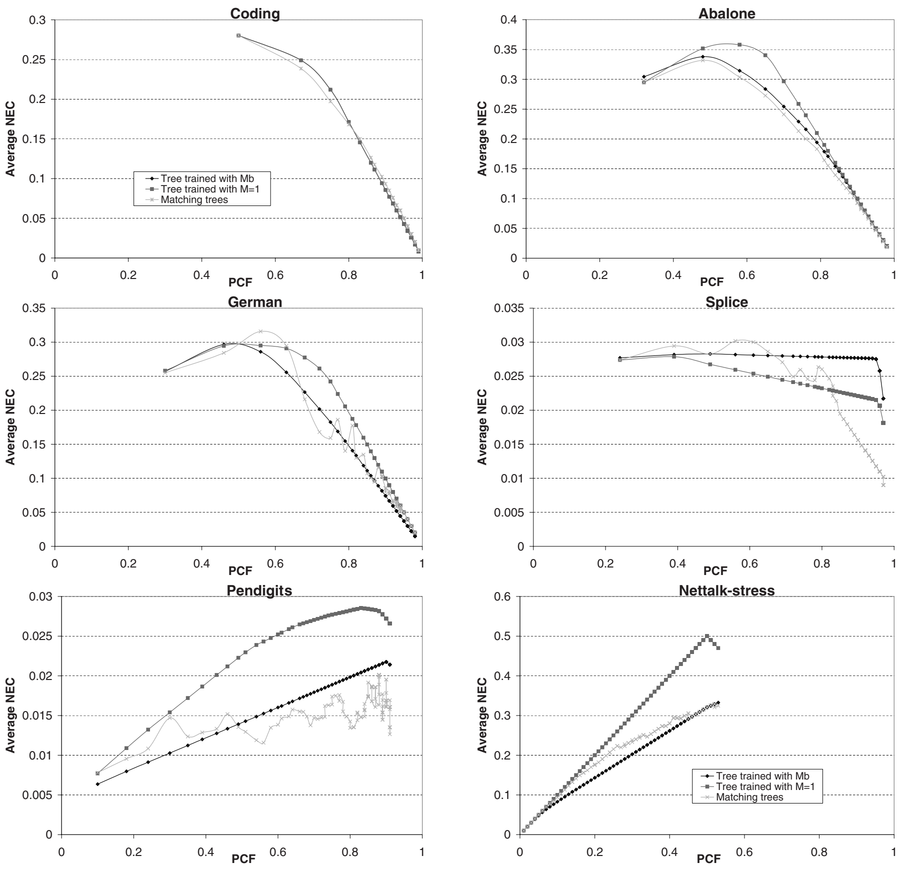

Fig. 4. Single tree versus matching trees: the detailed results for pruned trees using partial cost curves constructed from $M = 1$ (the left-most point) to $M = 100$ (the right-most point). Each point is an average over a 10-fold cross-validation. Coding and kr-vs-kp have identical cost curves for single trees trained with $M = 1$ and $M = M_b$ since they are naturally balanced. Not all results are presented because of space limitation.

It is interesting to note that there are two data sets in which a single tree performs better and the only data characteristic that stands out from the other data sets is that they have balanced distribution. If that is an important criterion, then a single tree trained from the balanced distribution will perform better than a single tree trained from the natural distribution. The results in the '$M = M_b$' column of Table 1 indeed demonstrates that with a win:lose ratio of 2:9. This result agrees with that obtained by Weiss & Provost [14] using the AUC measure for ROC curves though they are using sampling and we are using re-weighting to change the training priors.

Comparing to trees trained from the matching distribution, a single tree trained from the balanced distribution is a better contender than that trained from the natural distribution but it is still slightly in favour of matching trees, with the win:lose ratios of 6:7. This result shows that it is still advisable to train a tree that matches the operating condition.

We repeat the experiments using the DKM criterion. The results in Table 1 show similar trends for all the pair-wise comparisons as those for gain ratio.

Figure 4 shows the detailed results for gain ratio trees. It is interesting to note that the M_b tree almost always performs better than the $M = 1$ tree in most operating conditions in most data sets, and in many cases there is only a minor difference between the two in the natural distribution operating condition. Interestingly, M_b tree performs better than $M = 1$ tree in some cases even in the natural distribution operating condition (at the left-most point)! An example is in the pendigits data set. The only exception to the above overwhelming one-sided result is in the splice data set.

The matching model is always better than the single model in abalone and adult. Compared to the single model trained from the balanced distribution, the matching model is significantly better in a substantial part of the testing conditions in splice, pendigits and hypothyroid. The reverse is true in kr-vs-kp, german, nursery and nettalk-stress.

The cost curves for matching trees are not "smooth" because there are two variables: a different tree and a different threshold for each point in the curve. Single tree has only one variable, that is the threshold.

Note that the above discussion on the detailed results are for the gain ratio trees only. The relative results between the two approaches correspond to the summarised results shown in Table 1, whether it is for the gain ratio trees or the DKM trees. However, one can expect some variation in local operating points as shown in Figure 4 and unsmooth curves for matching trees in all cases.

6 Discussion

We begin this investigation assuming that the single model approach will perform comparable to the matching model approach at its best, but will never perform better. The assumption turns out to be incorrect, as shown in the experimental results in Section 5.2 in which the single model can sometimes outperform the matching model when the training class distribution is balanced. This unintuitive result is a direct result of the now known fact that the best classifier can be induced from an unmatching class distribution [5, 12, 14].

Quite a few researchers (e.g., [8, 14]) have shown that models trained from the balanced distribution perform better than models trained from the natural distribution, using either rescaling or thresholding. However, none of them has compared them with matching models. Our result shows that matching models cannot be ignored by assuming that single models will perform comparably to matching models.

The re-weighting method is best suited for scenarios in which a skewed distribution data set is readily available and misclassifying a minority class is more costly than misclassifying a majority class. The main advantage of the re-weighting method over the sampling methods is that for any class distribution we obtain a training set with the same data size, using all the examples.

This is achieved through weight normalization in which the total weight is always equal to the total size of the original data [13]. This effectively uses a mixture of up-weighting for one class and down-weighting for another in each class distribution modification.

If the test condition is expressed as misclassification cost, one common method is to apply the minimum expected cost criterion to a previously trained model and make the final decision with a class that has the minimum expect cost. It can be easily shown that this is equivalent to rescaling.

7 Conclusions and Future Work

This paper's first contribution is establishing the relationship among different methods for the single model approach and their limitations. Its second contribution is answering the open question of whether the single model approach is sufficient to implement the matching distribution assumption, in comparison to the matching model approach using decision trees.

Specifically for the first contribution, we show that

- Rescaling for a class density model is easier than rescaling a posterior probability model because no knowledge of the training prior probability is required. It is also simpler than rescaling by a ratio of changed probability. This result suggests that when using class density models such as Naive Bayes, prior probability needs not be estimated during training; and then apply Equation (7) which uses the appropriate prior probability for classification.
- Rescaling is not a recommended approach because it relies on accurate probability estimation and does not adapt to the output range of the model. We also reveal that a previous analysis advocating the use of rescaling has an implicit strong assumption, that is, the learning algorithm must be totally insensitive to prior probability. We argue that the rescaled model cannot guarantee to produce the same output as that from the matching model, as claimed by the analysis when this assumption is violated; which is the case for decision trees, in general.
- Thresholding is the preferred method for the single model approach because it determines the threshold empirically that avoids the limitations of rescaling.

For the second contribution, we show using decision trees that there is no foolproof substitute for the matching model approach that recommends training a model using data whose class distribution matches the testing condition. Using matching model is usually better than a single model trained from the natural distribution. One possible substitute for the matching model is to train a single model using a balanced distribution (either natural or derived). The single model trained from the derived balanced distribution is only likely to work better than the matching model when the original distribution is highly skewed.

We show that model sensitivity, as exhibited by using two different splitting criteria with different degrees of sensitivity to priors in decision trees, does not play a role in determining whether the single model approach or the matching model approach should be used.

Our experiments have been limited to decision trees. It is possible that other learning algorithms which are designed for probability estimation might behave quite differently from the results presented. Also, we have limited this study to the case of changed class distribution that happens uniformly across the sample space. It is possible that class distribution changes non-uniformly; in which case re-weighting instances uniformly as we have done for the matching model here is an inappropriate method. The analysis is restricted to the two-class case because of the current limitation of the cost space or ROC analysis that we employed. We intend to explore these issues in the near future.

Acknowledgement

Discussion with Phil Rayment, Geoff Webb and David Albrecht has helped to develop the material presented in this paper. Ross Quinlan provides C4.5. The anonymous reviewers have helped to improve the presentation of this paper.

References

1. Blake, C. & Merz, C.J.: UCI Repository of machine learning databases. [www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California (1998)
2. Breiman, L., Friedman, J.H., Olshen, R.A. & Stone, C.J.: Classification And Regression Trees. Chapman & Hall. (1993)
3. Dietterich, T.G., Kearns, M., & Mansour, Y.: Applying the weak learning framework to understand and improve C4.5. Proceedings of Thirteenth International Conference on Machine Learning (1996) 96–104. San Francisco: Morgan Kaufmann
4. Duda, O.R., Hart, P.E. & Stork, D.G.: Pattern Classification. John Wiley (2001)
5. Drummond, C. & Holte, R.C.: Explicitly Representing Expected Cost: An Alternative to ROC Representation. Proceedings of the Sixth International Conference on Knowledge Discovery and Data Mining (2000) 198–207
6. Drummond, C. & Holte, R.C.: Exploiting the Cost (In)sensitivity of Decision Tree Splitting Criteria. Proceedings of The Seventeenth International Conference on Machine Learning (2000) 239–246. San Francisco: Morgan Kaufmann
7. Elkan, C.: The Foundations of Cost-Sensitive Learning. Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (2001) 973–978
8. Hooper, P.M.: Reference Point Logistic Regression and the Identification of DNA Functional Sites. Journal of Classification. 18 (2001) 81–107
9. Provost, F. & Fawcett, T.: Robust Classification for Imprecise Environments. Machine Learning. 42 (2001) 203–231
10. Provost, F. & Domingos, P.: Tree-Induction for Probability-based Ranking. Machine Learning. 52 (2003) 199–215
11. Quinlan, J.R.: C4.5: Program for Machine Learning. Morgan Kaufmann (1993)
12. Ting, K.M.: Issues in Classifier Evaluation using Optimal Cost Curves. Proceedings of The Nineteenth International Conference on Machine Learning (2002) 642–649
13. Ting, K.M.: An Instance-Weighting Method to Induce Cost-Sensitive Trees. IEEE Transactions on Knowledge and Data Engineering. Vol.14 No.3 (2002) 659–665
14. Weiss, G. & Provost, F.: Learning when Training Data are Costly: The Effect of Class Distribution on Tree Induction. Journal of Artificial Intelligence Research. 19 (2003) 315–354