Obtaining calibrated probability estimates from decision trees and naive Bayesian classifiers

Bianca Zadrozny ZADROZNY@CS.UCSD.EDU
Charles Elkan ELKAN@CS.UCSD.EDU
Department of Computer Science and Engineering, University of California, San Diego, La Jolla, California 92093-0114

Abstract

Accurate, well-calibrated estimates of class membership probabilities are needed in many supervised learning applications, in particular when a cost-sensitive decision must be made about examples with example-dependent costs. This paper presents simple but successful methods for obtaining calibrated probability estimates from decision tree and naive Bayesian classifiers. Using the large and challenging KDD'98 contest dataset as a testbed, we report the results of a detailed experimental comparison of ten methods, according to four evaluation measures. We conclude that binning succeeds in significantly improving naive Bayesian probability estimates, while for improving decision tree probability estimates, we recommend smoothing by m-estimation and a new variant of pruning that we call curtailment.

1. Introduction

In many supervised learning applications, it is not enough simply to predict for each test example which class is most likely for the example. What is needed instead is, for a given class, a ranking of all the examples in the test set from the most probable member to the least probable member. Any score that is an increasing or decreasing function of class membership probability is sufficient to induce a correct ranking of examples. However in some applications, even a complete ranking of examples is not enough. What is needed in addition is an accurate, correctly calibrated estimate of the true probability that each test example is a member of the class of interest.

Calibrated probability estimates are needed in particular when a cost-sensitive decision must be made about each example, and the costs associated with different examples are different. For example, suppose that the cost of requesting a charitable donation from an individual x is \$0.68, and that the best estimate of the amount that x will donate, if he or she does make a donation, is $y(x)$. In this case the optimal decision-making policy is to solicit x if and only if $\hat{P}(\text{donate}|x)y(x) > 0.68$, where $\hat{P}(\text{donate}|x)$ is the estimated probability that x will donate, i.e. that x is a member of the class of donors.

For each example x, we need to know how $\hat{P}(\text{donate}|x)$ compares to the threshold $0.68/y(x)$. This decision-making threshold is different for each x, so it is vital to estimate the true probability $P(\text{donate}|x)$ as accurately as possible for each x. Simply knowing that one person is more likely to donate than another is not enough, because the second person may donate a larger amount than the first, if he or she does make a donation. Knowing a numerical score $s(x) = f(P(\text{donate}|x))$ without knowing the transformation function f is not enough, because $0.68/y(x)$ is the correct threshold for making decisions only if f is the identity function.

This paper presents simple but successful methods for obtaining calibrated probability estimates from decision tree and naive Bayes classifiers. Instead of doing an experimental comparison of the methods on many datasets that are relatively small and easy to achieve good performance on, we prefer to do a detailed investigation using multiple performance metrics on one large dataset that is known to be challenging. This dataset is the one first used in the data mining contest associated with the 1998 KDD conference. With associated documentation, it is now available in the UCI KDD repository (Bay, 2000). It is the only publicly available dataset known to us for which real-world misclassification cost information is available. It is also the only public dataset for which example-specific cost information is available.

The KDD'98 dataset contains information about persons who have made donations in the past to a certain charity. The decision-making task is the one outlined above, to choose which past donors to request a new donation from, This task is completely analogous to typical one-to-one marketing tasks for many other organizations, both non-profit and for-profit (Mozer *et al.*, 2000; Piatetsky-Shapiro & Masand, 1999), so our conclusions are directly useful for real-world applications.

2. Methods

This section explains our methods for obtaining calibrated probability estimates from decision tree and naive Bayesian classifiers. We first explain the deficiencies that cause standard decision tree methods not to give accurate probability estimates, and then we explain how to overcome these limitations. A final subsection presents a simple method for obtaining calibrated probabilities from a naive Bayesian classifier.

2.1 Deficiencies of decision tree methods

Throughout this paper, C4.5 (Quinlan, 1993) is the representative decision tree learning method used, but most of our analyses and suggestions apply equally to other decision tree methods such as CART (Breiman *et al.*, 1984).

When classifying a test example, C4.5 and other decision tree methods assign by default the raw training frequency $p = k/n$ as the score of any example that is assigned to a leaf that contains k positive training examples and n total training examples. These training frequencies are not accurate conditional probability estimates for at least two reasons:

1. High bias: Decision tree growing methods try to make leaves homogeneous, so observed frequencies are systematically shifted towards zero and one.
2. High variance: When the number of training examples associated with a leaf is small, observed frequencies are not statistically reliable.

Pruning methods as surveyed by Esposito *et al.* (1997) can in principle alleviate the second problem by removing leaves that contain too few examples. However, standard pruning methods are not suitable for unbalanced datasets, because they are based on accuracy maximization. On the KDD'98 dataset C4.5 produces a pruned tree that is a single leaf. Since the base rate of positive examples $P(\text{donate}) = P(\text{class } j = 1)$ is about 5%, this tree has accuracy 95%, but it is useless for estimating example-specific conditional probabilities $P(j = 1|x)$. In general, trees pruned with the objective of maximizing accuracy are not useful for ranking test examples, or for estimating class membership probabilities.

The standard C4.5 pruning method is not alone in being incompatible with accurate probability estimation. Quinlan's latest decision tree learning method, C5.0, and CART also do pruning based on accuracy maximization. C4.5 and C5.0 have rule set generators as an alternative to pruning, but because these methods are based on accuracy maximization, they are also unsuitable for probability estimation. C5.0 can do loss minimization pruning with a fixed cost matrix, but this is not adequate when costs are different for different examples.

In this paper we attempt to improve directly the accuracy of decision tree probability estimates. Our experiments use C4.5 without pruning and without collapsing to obtain raw scores that are correlated with accurate class membership probabilities. The choice to do no pruning is supported by the results of Bradford *et al.* (1998), who find that performing no pruning and variants of pruning adapted to loss minimization both lead to similar performance. Not using pruning is also suggested by Bauer and Kohavi (1999) in their Section 7.3.

Our methods all transform the leaf scores of a standard decision tree. Completely different methods have been suggested, but they have major drawbacks. Smyth *et al.* (1995) use kernel density estimators at the leaves of a decision tree. However their algorithms are based on C4.5 and CART with pruning, so they are unsuitable for highly unbalanced datasets. Their experiments use only synthetic data where both classes are equiprobable, or the less probable class has base rate 1/3. Our experiments use an unbalanced real-world dataset where the less probable class has base rate about 1/20. Estimating probabilities using bagging has been suggested by Domingos (1999), but as pointed out recently by Margineantu (2000), bagging gives voting estimates that measure the uncertainty of the base classification method concerning an example, not the actual class-conditional probability of the example.

2.2 Improving probability estimates by smoothing

As discussed by Provost and Domingos (2000) and others, one way of improving the probability estimates given by a decision tree is to make these estimates smoother, i.e. to adjust them to be less extreme. Provost and Domingos suggest using the Laplace correction method. For a two-class problem, this method replaces the conditional probability estimate $p = \frac{k}{n}$ by $\frac{k+1}{n+2}$ where k is the number of positive training examples associated with a leaf and n is the total number of training examples associated with the leaf.

The Laplace correction method adjusts probability estimates to be closer to 1/2, which is not reasonable when the two classes are far from equiprobable, as is the case in many real-world applications. In general, one should consider the overall average probability of the positive class, i.e. the base rate, when smoothing probability estimates. From a Bayesian perspective, a conditional probability estimate should be smoothed towards the corresponding unconditional probability.

We replace the probability estimate $p = \frac{k}{n}$ by $p' = \frac{k+b \cdot m}{n+m}$ where b is the base rate and m is a parameter that controls how much scores are shifted towards the base rate. This smoothing method is called m-estimation (Cestnik, 1990). Previous papers have suggested choosing m by cross-validation (Cussens, 1993). Given a base rate b, we suggest using m such that $bm = 10$ approximately. This heuristic ensures that raw probability estimates that

are likely to have high variance, those with $k \leq 10$, are given low credence. Experiments show that the effect of smoothing by m-estimation is qualitatively similar for a wide range of values of m, so, as is highly desirable, the precise value chosen for m is unimportant.

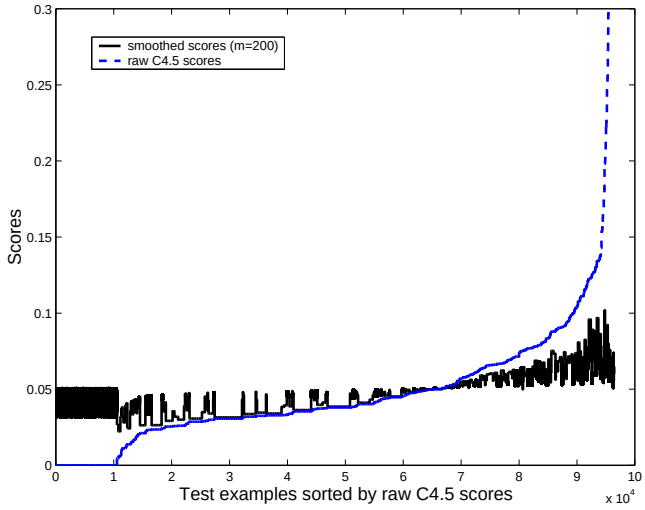

Figure 1. Smoothed scores and raw scores from C4.5 without pruning for test examples sorted by raw score.

Figure 1 shows the smoothed scores with $m = 200$ of the KDD'98 test set examples sorted by their raw scores given by C4.5 without pruning. As expected, smoothing shifts all scores towards the base rate of approximately 0.05, which is desirable given that C4.5 scores tend to be overestimates or underestimates. While raw C4.5 scores range from 0 to 1, smoothed scores range from 0.0224 to 0.1018.

2.3 Curtailment

As discussed above, without pruning decision tree learning methods tend to overfit training data and to create leaves in which the number of examples is too small to induce conditional probability estimates that are statistically reliable. Smoothing attempts to correct these estimates by shifting them towards the overall average probability, i.e. the base rate b. However, if the parent of a small leaf, i.e. a leaf with few training examples, contains enough examples to induce a statistically reliable probability estimate, then assigning this estimate to a test example associated with the small leaf may be more accurate then assigning it a combination of the base rate and the observed leaf frequency, as done by smoothing. If the parent of a small leaf still contains too few examples, we can use the score of the grandparent of the leaf, and so on until the root of the tree is reached. At the root, of course, the observed frequency is the training set base rate.

We call this method of improving conditional probability estimates curtailment because when classifying an example, we stop searching the decision tree as soon as we reach a node that has less than v examples, where v is a parameter of the method. The score of the parent of this node is then assigned to the example in question. As for smoothing, v can be chosen by cross-validation, or using a heuristic such as making $bv = 10$. We choose $v = 200$ for all our experiments. Informal experiments show that values of v between 100 and 400 give similar results, so the exact setting of v is not critical.

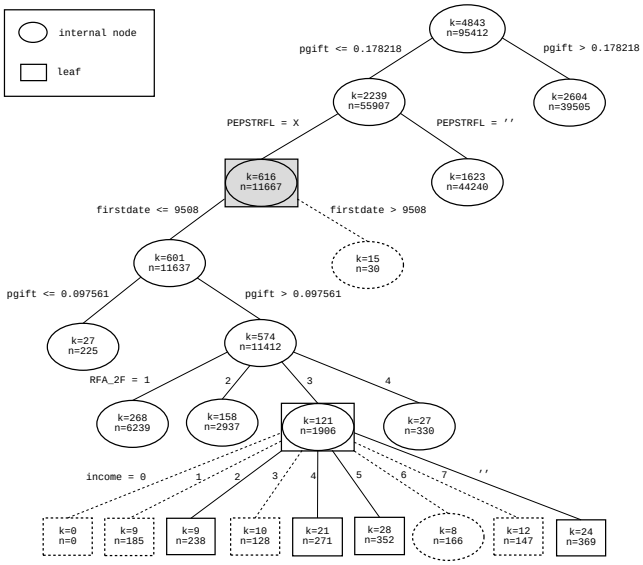

Figure 2. Part of the decision tree obtained by curtailment with $v = 200$. The dotted nodes are present in the original C4.5 tree, but are effectively eliminated from the curtailment tree because $n < v$. The node in grey is an example of a node that can serve both as an internal node and as a leaf, because one of its branches has been eliminated from the tree, but not all.

By eliminating nodes that have few training examples, given the KDD'98 training set curtailment effectively creates the decision tree shown in part in Figure 2. The distinction between internal nodes and leaves is blurred in this tree, because a node may serve as an internal node for some examples and as a leaf for others, depending on the attribute values of the examples. Curtailment is not equivalent to any type of pruning, nor to traditional early stopping during the growing of a tree, because those methods eliminate all the children of a node simultaneously. In contrast, curtailment may eliminate some children and keep others, depending on the number of training examples associated with each child. Intuitively, curtailment is preferable to pruning for probability estimation because nodes are removed from a decision tree only if they are likely to give unreliable probability estimates.

Curtailment is reminiscent of methods proposed by Bahl *et al.* (1989) and Buntine (1993) to smooth decision tree probabilities. To obtain smoothed probability estimates, these methods calculate a weighted average of training fre-

quencies at nodes on the path from the root to each leaf. A drawback of these approaches is that weights are determined through cross-validation. Since many weights must be set, overfitting is a danger.

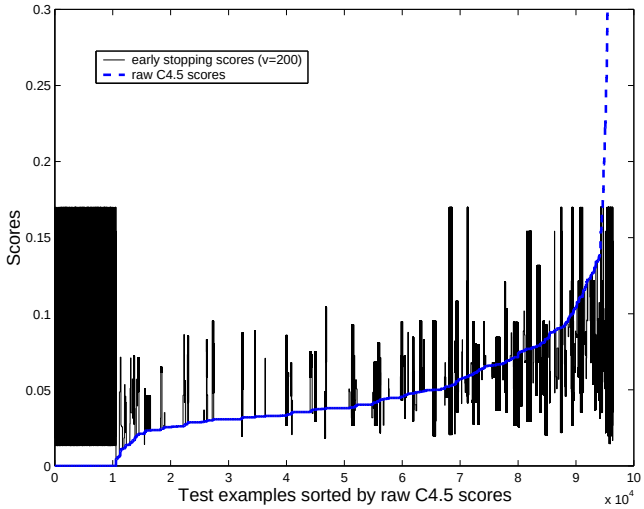

Figure 3. Curtailment scores and raw scores from C4.5 without pruning for test examples sorted by raw score (v=200).

Figure 3 shows curtailment scores for the KDD'98 test set examples sorted by their raw C4.5 scores. The jagged lines in the chart show that many scores are changed significantly by curtailment. Overall, the range of scores is reduced as with smoothing, but not as much. The minimum curtailment score is 0.0045 while the maximum is 0.1699.

Curtailment effectively eliminates from a decision tree nodes that yield probability estimates that are not statistically reliable because they are based on a small sample of examples. But since C4.5 tries to make decision tree nodes homogeneous, even probability estimates that are based on many training examples tend to be too high or too low. Smoothing can compensate for this bias by shifting estimates towards the overall average probability. Therefore we investigate the combination of smoothing and curtailment. Specifically, we apply m-estimation to the observed frequencies found at the nodes chosen by curtailment.

2.4 The Kearns-Mansour splitting criterion

The previous subsections present methods for obtaining more accurate probability estimates from a decision tree. A different splitting criterion could conceivably also improve the accuracy of probability estimates. When a decision tree is constructed, the training set is recursively partitioned into smaller subsets. The choice at each node of the attribute to be used for partitioning is based on a metric applied to each attribute, called a splitting criterion. The metric measures the degree of homogeneity of the subsets that are induced if the set of examples at this node is partitioned based on the values of this attribute.

Consider a discrete attribute A that has values $A = a_1$ through $A = a_m$ for some $m \geq 2$. In the two-class case, standard splitting criteria have the form

$$I(A) = \sum_{k=1}^{m} P(A = a_k) h(p_k, 1 - p_k)$$

where $p_k = P(j = 1 | A = a_k)$ and all probabilities are frequencies in the training set to be split based on A. The function h measures the impurity or heterogeneity of each subset of training examples. All such functions are qualitatively similar. Whatever the form of h, its unique maximum is $h(p, 1-p) = 1$ when $p = 0.5$, and its only minima are $h(p, 1-p) = 0$ when $p = 0$ or $p = 1$.

C4.5 uses the average amount of information needed to identify the class of an example in each subset as the function h. In the two class case this quantity, called entropy, is $h(p, 1-p) = -p \log_2(p) - (1-p) \log_2(1-p)$. Experimental results of Drummond and Holte (2000) and Dietterich *et al.* (1996) show that the impurity function $h(p, 1-p) = 2\sqrt{p(1-p)}$ suggested by Kearns and Mansour (1996) typically leads to somewhat smaller unpruned decision trees. Smaller trees are preferable for probability estimation since the number of examples at each leaf is potentially larger, which results in scores that are more statistically reliable. Therefore, we compare below the accuracy of probability estimates obtained using the C4.5 entropy function and using the Kearns-Mansour function, in both cases with no pruning, as discussed in Section 2.1.

2.5 Calibrating naive Bayes classifier scores

Naive Bayesian classifiers are based on the assumption that within each class, the values of the attributes of examples are independent. It is well-known that these classifiers tend to give inaccurate probability estimates (Domingos & Pazzani, 1996). Given an example x, suppose that a naive Bayesian classifier computes the score $n(x)$. Because attributes tend to be positively correlated, these scores are typically too extreme: for most x, either $n(x)$ is near 0 and then $n(x) < P(j = 1|x)$ or $n(x)$ is near 1 and then $n(x) > P(j = 1|x)$. However, naive Bayesian classifiers tend to rank examples well: if $n(x) < n(y)$ then $P(j = 1|x) < P(j = 1|y)$.

We use a histogram method to obtain calibrated probability estimates from a naive Bayesian classifier. We sort the training examples according to their scores and divide the sorted set into b subsets of equal size, called bins. For each bin we compute lower and upper boundary $n(\cdot)$ scores. Given a test example x, we place it in a bin according to its score $n(x)$. We then estimate the corrected probability that x belongs to class j as the fraction of training examples in the bin that actually belong to j.

The number of different probability estimates that binning

can yield is limited by the number of alternative bins. This number, $b = 10$ in our experiments, must be small in order to reduce the variance of the binned probability estimates, by increasing the number of examples whose 0/1 memberships are averaged inside each bin. Binning reduces the resolution, i.e. the degree of detail, of conditional probability estimates, while improving the accuracy of these estimates by reducing both variance and bias compared to uncalibrated estimates.

With most learning methods, in order to obtain binned estimates that do not overfit the training data, we should partition the training set into two subsets. One subset would be used to learn the classifier that yields uncalibrated scores, while the other subset would be used for the binning process. More training examples would be assigned to the first subset because learning a classifier involves setting many more parameters than setting the binned probabilities. For naive Bayesian classifiers, however, separate subsets are not necessary. We use the entire training set both for learning the classifier and for binning.

3. Experimental design

The KDD'98 dataset mentioned in the introduction has already been divided in a fixed, standard way into a training set and a test set. The training set consists of 95412 records for which it is known whether or not the person made a donation (class $j = 1$ or $j = 0$) and how much the person donated, if a donation was made. The test set consists of 96367 records for which similar donation information was not published until after the KDD'98 competition. In order to make our experimental results directly comparable with those of previous work, we use the standard training set/test set division. All examples in the training set are used as training data for the learning algorithms, with seven fields as attributes. This paper does not address the issue of feature selection, so our choice of attributes is fixed and based informally on the KDD'99 winning submission of Georges and Milley (1999).

Since there is no standard metric for comparing the accuracy of probability estimates, we use four different metrics to compare the probability estimates given by each method explained in Section 2. One contribution of this paper is an investigation of the extent to which these evaluation metrics agree and disagree.

The first metric is squared error, defined for one example x as $\sum_j (t(j|x) - p(j|x))^2$ where $p(j|x)$ is the probability estimated by the method for example x and class j, and $t(j|x)$ is the true probability of class j for x. For test sets where true labels are known, but not probabilities, $t(j|x)$ is defined to be 1 if the label of x is j and 0 otherwise.

The second metric is log-loss or cross entropy, defined as $-\sum_j t(j|x) \log_2 \frac{p(j|x)}{t(j|x)}$. We calculate the squared error and log-loss for each x in the training and test sets to obtain the mean squared error (MSE) and the average log-loss for each set. MSE is also called the Brier score (Brier, 1950).

Lift charts are the third method we use to compare probability estimates (Piatetsky-Shapiro & Masand, 1999). To obtain a lift chart, we first sort the examples in descending order according to their scores. Given $0 \leq a \leq 1$, let the target set $T(a)$ be the fraction a of examples with the highest scores. The lift at a is defined as $l(a) = r(a)/P(j = 1)$ where $r(a)$ is the proportion of positive examples in $T(a)$. By definition $l(1) = 1$ and we expect l to be a decreasing function. A lift chart is a plot of $l(a)$ as a function of a. The greater the area below a lift chart, the more useful scores are. Lift charts are isomorphic to ROC curves (Swets *et al.*, 2000), but more widely used in commercial applications of probability estimation.

Finally, our fourth evaluation metric for a given set of probability estimates is the profit obtained when we use these estimates to choose individuals for solicitation according to the policy "choose x if and only if $P(j = 1|x)y(x) > 0.68$" as discussed in the introduction. This paper is not concerned with donation amount estimation, so we use fixed values for $y(x)$ obtained using a simple linear regression described by Zadrozny and Elkan (2001).

A metric that we choose not to use is the area under the ROC curve (Hanley & McNeil, 1982), the metric used by Provost and Domingos (2000). As they point out, because it is based on the ROC curve this metric fails to distinguish between scores that rank examples correctly, and scores that satisfy the stronger property of being well-calibrated probability estimates. Also, the area under the ROC curve is less useful for comparing methods when their ROC curves intercept, which is the case for many of the methods investigated here.

4. Experimental results

In this section we present an experimental comparison of the methods presented in Section 2. For comparison, we also report the results of using a constant probability estimate for all examples. We use three different constant probability estimates: zero, one, and the base rate of positive examples. In addition, we report the results of applying bagging (Breiman, 1996) to decision trees with Laplace smoothing and with curtailment, by averaging the probability estimates given by trees learned from 100 independent resamples. Because all regularized probability estimates are under 0.5, combining zero/one predictions would give all zero estimates. Except for Kearns-Mansour, all decision tree methods use the C4.5 entropy splitting criterion.

Table 1 shows MSE on the training and test sets for each of the probability estimation methods. Since MSE is a measure of error, smaller is better. Both the Kearns-Mansour splitting criterion and the C4.5 entropy criterion result in an MSE for the test set that is much bigger than the MSE for

Method	Training set	Test set
Bagged Laplace	0.07011	0.09812
Kearns-Mansour	0.08643	0.10424
C4.5 without pruning	0.08789	0.10211
Laplace (C4.4)	0.08950	0.10090
Bagged curtailment	0.09455	**0.09515**
Curtailment	0.09508	0.09535
Smoothing	0.09510	0.09557
Smoothed curtailment	0.09516	0.09525
Binned naive Bayes	0.09546	0.09528
All base rate	0.09636	0.09602
Naive Bayes	0.10089	0.10111
All zero (C4.5 with pruning)	0.10151	0.10113
All one	1.89848	1.89886

Table 1. MSE for the KDD'98 training and test sets. Methods are ordered by training set MSE. The best test set MSE is highlighted.

Method	Training set	Test set
Bagged Laplace	0.18841	∞
Kearns-Mansour	0.24326	∞
C4.5 without pruning	0.25422	∞
Laplace (C4.4)	0.26477	0.29705
Bagged curtailment	0.27691	**0.28300**
Smoothing	0.28037	0.28529
Curtailment	0.28090	0.28439
Smoothed curtailment	0.28146	0.28352
Binned naive Bayes	0.28336	0.28365
All base rate	0.28961	0.28880
Naive Bayes	0.30198	0.30400
All zero (C4.5 with pruning)	∞	∞
All one	∞	∞

Table 2. Average log-loss on the KDD'98 training set and test set. Methods are ordered by training set log-loss.

the training set. Adjusting the C4.5 probability estimates using Laplace smoothing, as recommended by Provost and Domingos (2000) (they call this method C4.4), makes only small changes in most leaf scores, so the MSE for the test set is only slightly improved. Furthermore, bagging Laplace decision trees extremely overfits the training data. Although bagged curtailment generates the best probability estimates for the test set according to the MSE metric, the improvement over curtailment is slight, especially considering the high computational cost. Smoothed curtailment generates the second best probability estimates for the test set according to the MSE metric, but curtailment and smoothing separately and naive Bayesian binning are only slightly worse.

Table 2 shows the average log-loss on the training and test sets for each method. As for MSE, a smaller average log-loss is better. The ranking of the methods by log-loss is the same as the ranking by MSE, except for smoothing and curtailment on the training set. An important disadvantage of the log-loss metric is that if any method estimates $\hat{P}(j = 1|x) = 0$ for any example whose actual class is $j = 1$, the average log-loss of the method is infinite, which is uninformative about the accuracy of the probability estimates for other examples.

The profit on the training and test sets achieved by each method is given in Table 3. For the training and test sets, the ranking of the methods by profit is almost the same as by MSE or by average log-loss. One difference is that, as measured by profit achieved, smoothed curtailment is slightly better on the test set than bagged curtailment. However, the differences between methods in profit achieved are much more accentuated than the differences in MSE or average log-loss. Empirically, it is very difficult to do well on the KDD'98 test set. The best profit we achieve, \$14741, is better than that obtained by the winner of the KDD'98 contest. In the contest, the median profit achieved was \$10720. Predicting $j = 0$ for all test examples gives profit \$0 while predicting $j = 1$ for all test examples gives profit \$10560.

Method	Training set	Test set
Bagged Laplace	\$48639	\$10190
Kearns-Mansour	\$30448	\$8873
C4.5 without pruning	\$24212	\$10538
Bagged curtailment	\$22109	\$14502
Laplace (C4.4)	\$21240	\$12601
Smoothing	\$19248	\$13897
Smoothed curtailment	\$17081	**\$14741**
Curtailment	\$16823	\$14190
Binned naive Bayes	\$14897	\$14642
All base rate	\$11923	\$12252
All one	\$10790	\$10560
Naive Bayes	\$10083	\$9531
All zero (C4.5 with pruning)	\$0	\$0

Table 3. Profit attained on the KDD'98 training and test sets. Methods are ordered by profit on the training set.

The lift charts for each method are shown in Figures 4 and 5, for the training and test set respectively. Since the lift charts intercept, we do not report the area under the lift charts. The ranking of the methods according to the lift charts is similar to the ranking using the other metrics. Again, we see that C4.5 without pruning, with or without the Kearns-Mansour splitting criterion and Laplace adjustment, overfits the training set. The lift charts for these methods are considerably better for the training set than for the test set.

The naive Bayesian classifier gives excellent lift charts on both the training and test sets. As mentioned above, naive Bayesian scores tend to be correlated well with true probabilities, even though they are not calibrated well. Lift charts depend only on the ranking of scores, and not on their magnitude. The binned naive Bayes scores and the original naive Bayes scores are monotonically related, so both methods perform equally well as measured by lift, at

target set sizes a that are multiples of the bin size. Because binning assigns the same probability estimate to all examples in each bin, the ranking of examples inside bins is lost. For this reason, the lift chart for binned naive Bayes is approximately constant for $a < 1/b = 0.1$.

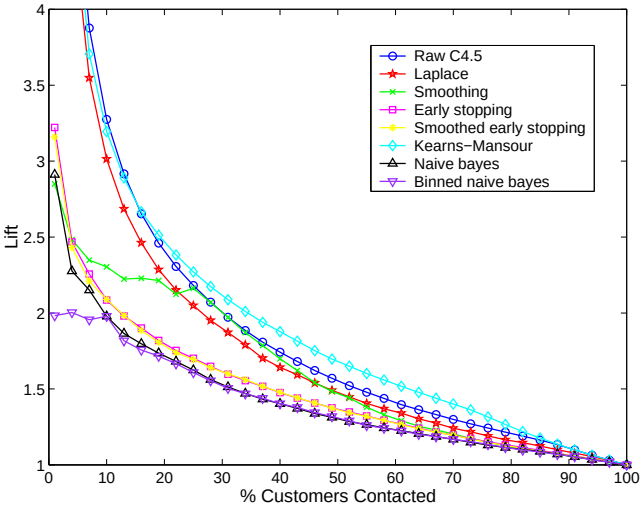

Figure 4. Lift charts for the KDD'98 training set.

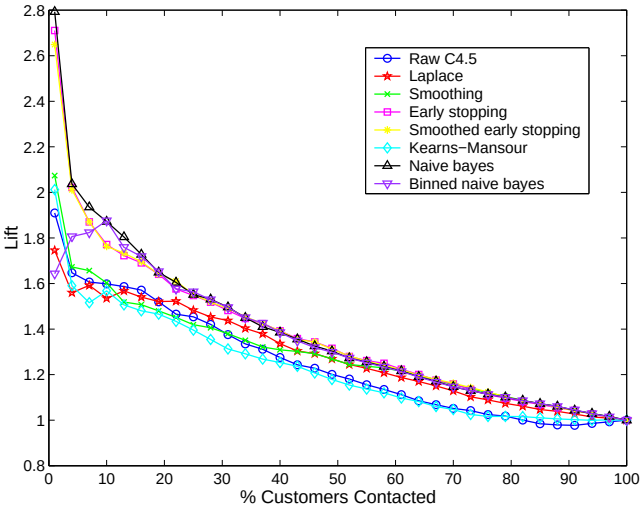

Figure 5. Lift charts for the KDD'98 test set.

Because the results above use one fixed training set and one fixed test set, it is difficult to say anything about the statistical significance of differences between methods. Therefore, to confirm that differences are genuine, we apply the methods to a completely different dataset, the CoIL challenge dataset, which is also available in the UCI KDD repository (Bay, 2000). The CoIL dataset also has a standard training set/test set division. Table 4 shows the MSE for each method on the training and test sets.

Method	Training set	Test set
C4.5 without pruning	0.09481	0.11420
Kearns-Mansour	0.09575	0.11431
Bagged Laplace	0.09621	0.11331
Laplace (C4.4)	0.09586	0.11548
Binned naive Bayes	0.10485	**0.10742**
Curtailment	0.10657	0.10839
Smoothed curtailment	0.10681	0.10844
Bagged curtailment	0.10686	0.10860
Smoothing	0.10723	0.11091
All base rate	0.11240	0.11192
All zero (C4.5 with pruning)	0.11955	0.11900
Naive Bayes	0.12896	0.13409
All one	1.88045	1.88100

Table 4. MSE for the CoIL training and test sets. Methods are ordered by training set MSE. The best test set MSE is highlighted. According to the heuristics $bm = 10$ and $bv = 10$, we use the fixed values $m = 170$ and $v = 170$ for the smoothing and curtailment parameters, since for this dataset $b \approx 0.06$. Additional experiments show that other values of m and v do not change the ranking of the methods.

The ranking of the methods is similar to the ranking for the KDD'98 dataset. However, on the CoIL dataset naive Bayesian binning gives slightly better probability estimates than smoothed curtailment. Bagged Laplace-smoothed decision trees do not overfit the training data as badly, but bagging actually reduces the performance of curtailment. The log-loss metric ranks the methods similarly, so for space reasons log-loss results are not shown here. Because cost information is not available for the CoIL dataset, profit results cannot be computed.

5. Conclusions

We have investigated carefully the relative performance of ten different methods for estimating class membership probabilities accurately. Eight methods use a single decision tree, two use an ensemble of decision trees, and two use a naive Bayesian classifier. Our experiments use four different evaluation methods. Three numerical metrics, namely MSE, average log-loss, and profit achieved in a decision-making task, are sensitive to whether probability estimates are well-calibrated. Lift charts are the fourth evaluation method. Like ROC curves, they are sensitive to accuracy in ranking test examples, but not to whether probability estimates are numerically accurate.

The ranking of the ten probability estimation methods is almost the same according to all four evaluation methods. The differences between methods seem small with the metrics MSE and average log-loss. However the lift chart differences are large, as are the differences in profit achieved. Therefore we recommend these latter two evaluation methods for future research.

In general, a good probability estimation method must make major changes in the scores produced by a standard decision tree or naive Bayesian classifier. We recommend not using any pruning methods based on error minimization, and instead applying a regularization method such as m-estimation, binning, or our curtailment method. Simply using Laplace smoothing, or changing the splitting criterion used to grow a decision tree, has little benefit.

Specifically, for decision trees the method we recommend is a combination of curtailment with smoothing. A major advantage of this method is that it produces relatively small and hence understandable decision trees, while still giving high-resolution, well-calibrated probability estimates.

Although previous research (Provost & Domingos, 2000; Bauer & Kohavi, 1999) has suggested that bagging can substantially improve probability estimates produced by decision trees, this finding is not confirmed by our experiments. Identifying the reasons for this phenomenon is a topic for further research. Our work focuses on obtaining calibrated probability estimates for large unbalanced datasets. Further work is also needed to assess the performance of our methods on balanced and/or small datasets.

References

Bahl, L. R., Brown, P. F., de Souza, P. V., & Mercer, R. L. (1989). A tree-based statistical language model for natural language speech recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing, 37*, 1001–1008.

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

Bay, S. D. (2000). UCI KDD archive. Department of Information and Computer Sciences, University of California, Irvine. `http://kdd.ics.uci.edu/`.

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

Breiman, L. (1996). Bagging predictors. *Machine Learning, 24*, 123–140.

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

Brier, G. W. (1950). Verification of forecasts expressed in terms of probability. *Monthly Weather Review, 78*, 1–3.

Buntine, W. (1993). Learning classification trees. *Artificial Intelligence frontiers in statistics* (pp. 182–201). Chapman & Hall.

Cestnik, B. (1990). Estimating probabilities: A crucial task in machine learning. *Proceedings of the Ninth European Conference on Artificial Intelligence* (pp. 147–149). Pitman.

Cussens, J. (1993). Bayes and pseudo-Bayes estimates of conditional probabilities and their reliability. *Proceedings of the European Conference on Machine Learning* (pp. 136–152). Springer Verlag.

Dietterich, T., Kearns, M., & Mansour, Y. (1996). Applying the weak learning framework to understand and improve C4.5. *Proceedings of the Thirteenth International Conference on Machine Learning* (pp. 96–104). Morgan Kaufmann.

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). ACM Press.

Domingos, P., & Pazzani, M. (1996). Beyond independence: Conditions for the optimality of the simple Bayesian classifier. *Proceedings of the Thirteenth International Conference on Machine Learning* (pp. 105–112). Morgan Kaufmann.

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

Esposito, F., Malerba, D., & Semeraro, G. (1997). A comparative analysis of methods for pruning decision trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 19*, 476–491.

Georges, J., & Milley, A. H. (1999). KDD'99 competition report. Available at `http://www-cse.ucsd.edu/users/elkan/kdresults.html`.

Hanley, J. A., & McNeil, B. J. (1982). The meaning and use of the area under receiver operating characteristic curve. *Radiology, 143*, 29–36.

Kearns, M., & Mansour, Y. (1996). On the boosting ability of top-down decision tree learning algorithms. *Proceedings of the Annual ACM Symposium on the Theory of Computing* (pp. 459–468).

Margineantu, D. (2000). On class probability estimates and cost-sensitive evaluation of classifiers. *Workshop Notes, Workshop on Cost-Sensitive Learning, International Conference on Machine Learning.*

Mozer, M. C., Wolniewicz, R., Grimes, D., Johnson, E., & Kaushanksy, H. (2000). Predicting subscriber dissatisfaction and improving retention in the wireless telecommunications industry. *IEEE Transactions on Neural Networks*, 690–696.

Piatetsky-Shapiro, G., & Masand, B. (1999). Estimating campaign benefits and modeling lift. *Proceedings of the Fifth International Conference on Knowledge Discovery and Data Mining* (pp. 185–193). ACM Press.

Provost, F., & Domingos, P. (2000). *Well-trained PETs: Improving probability estimation trees* (Technical Report CDER #00-04-IS). Stern School of Business, New York University.

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

Smyth, P., Gray, A., & Fayyad, U. (1995). Retrofitting decision tree classifiers using kernel density estimation. *Proceedings of the Twelfth International Conference on Machine Learning* (pp. 506–514).

Swets, J. A., Dawes, R. M., & Monahan, J. (2000). Better decisions through science. *Scientific American, 283*, 82–87.

Zadrozny, B., & Elkan, C. (2001). *Learning and making decisions when costs and probabilities are both unknown* (Technical Report CS2001-0664). University of California, San Diego.