Obtaining Calibrated Probabilities from Boosting

Alexandru Niculescu-Mizil
Department of Computer Science
Cornell University, Ithaca, NY 14853
alexn@cs.cornell.edu

Rich Caruana
Department of Computer Science
Cornell University, Ithaca, NY 14853
caruana@cs.cornell.edu

Abstract

Boosted decision trees typically yield good accuracy, precision, and ROC area. However, because the outputs from boosting are not well calibrated posterior probabilities, boosting yields poor squared error and cross-entropy. We empirically demonstrate why boosting predicts distorted probabilities. We examine three calibration methods for correcting this distortion: Platt Scaling, Isotonic Regression, and Logistic Correction. Experiments show that calibration significantly improves the probabilities predicted by boosting. After calibration, boosted decision trees predict better probabilities than other learning methods such as SVMs, neural nets, bagged decision trees, and k-nearest neighbor, even after these other methods are calibrated.

1 Introduction

In a recent evaluation of learning algorithms [1], boosted decision trees outperformed other learning methods on metrics such as accuracy, lift, area under the ROC curve, average precision, and precision/recall break even point. However, boosted decision trees had poor squared error and cross-entropy, suggesting that AdaBoost does not produce good probability estimates.

Friedman, Hastie, and Tibshirani [2] give one explanation for why boosting makes poorly calibrated predictions. They show that boosting can be viewed as an additive logistic regression model. A consequence of this is that the predictions made by boosting are trying to fit a logit of the true probabilities, as opposed to the true probabilities themselves. To obtain well calibrated probabilities, one must invert the logit transformation.

Intuitively, it is easy to see why AdaBoost [3] might predict poorly calibrated probabilities: at each round, AdaBoost puts more weight on cases that are hard to predict correctly, and less weight on cases predicted correctly. Eventually, the weights favor training base-level models that make correct predictions on the hard cases at the expense of making errors on the easy cases. Because the final prediction is a weighted combination of the base-level models, errors made on easy cases by a few models push the values predicted for them further away from 0 and 1, hurting calibration. This shifting is a by-product of boosting trading off reduced margin for easier high-margin cases for a larger margin for the more difficult cases nearer the decision boundary [4]. This shifting is also consistent with Breiman's interpretation of boosting as an *equalizer* (see Breiman's discussion in [2]. In Section 2 we demonstrate this probability shifting on real data.

To correct for boosting's poor calibration, we experiment with three methods for calibrating probabilities:

Logistic Correction: a method based on Friedman et al.'s analysis of boosting as an additive model

Platt Scaling: the method used by Platt to transform SVM outputs from $[-\infty, +\infty]$ to posterior probabilities [5]

Isotonic Regression: the method used by Elkan and Zadrozny to calibrate predictions from boosted naive bayes, SVM, and decision tree models [6, 7]

Logistic Correction and Platt Scaling convert predictions to probabilities by transforming them with a sigmoid. With Logistic Correction, the sigmoid parameters are derived from Friedman et al.'s analysis. With Platt Scaling, the parameters are fitted to the data using gradient descent. Isotonic Regression is a general-purpose non-parametric calibration method that assumes probabilities are a monotonic transformation (not just sigmoid) of the predictions.

In Section 2 we empirically show how boosting distorts probabilities. This analysis motivates the use of a sigmoid function to transform predictions to well-calibrated probabilities. Section 3 describes the three calibration methods. Section 4 presents an empirical comparison of the three calibration methods for boosting. Section 5 compares the performance of boosted decision trees and stumps to other learning methods such as SVMs and neural nets both be-

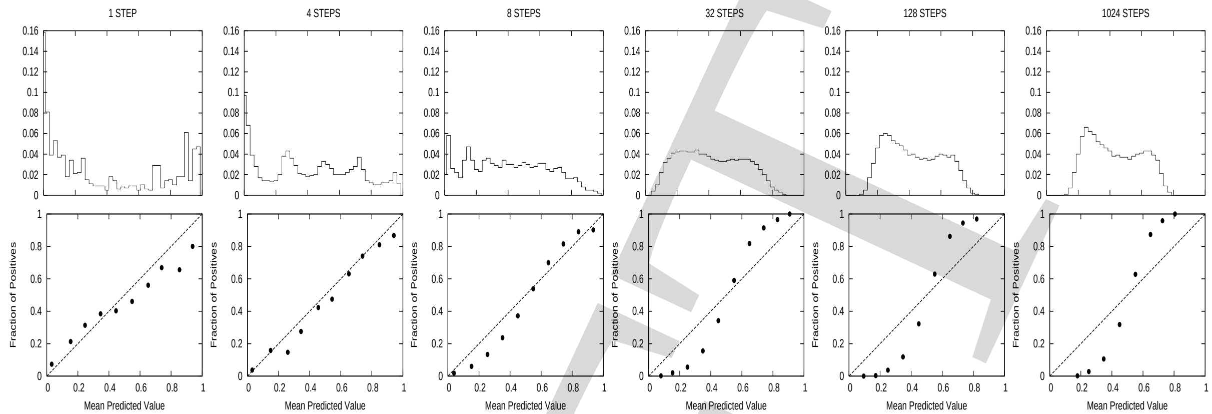

Figure 1: Effect of boosting on the predicted values. Histograms of the predicted values (top) and reliability diagrams (bottom) on the test set for boosted trees at different steps of boosting on the COV_TYPE problem.

fore and after calibration. After calibration, boosted trees predict better probabilities than the other learning methods.

2 Boosting and Calibration

In this section we empirically examine the relationship between boosting's predictions and posterior probabilities. First, we compare the predictions from boosting to the true conditional probabilities on a simple 1-D problem. The data is generated by two overlapping gaussians with variance 1.0. Negative examples are generated by a gaussian centered at -1.5, and positive examples by a gaussian centered at +1.5. For this problem we can compute the probability a case is positive. A well calibrated model would make predictions that follow this true probability.

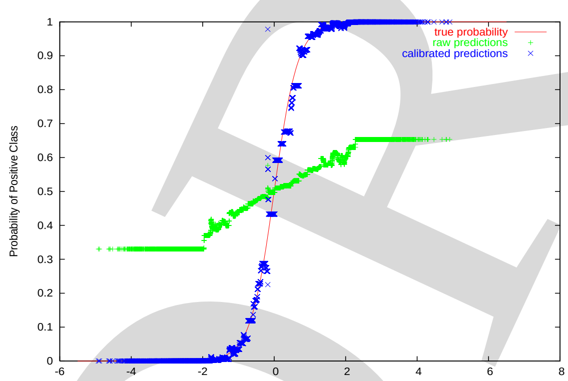

Figure 2: Raw and calibrated predictions from boosting.

We train boosted decision trees using AdaBoost to discriminate between positive and negative cases. On this problem, all trees have depth one since there is only one attribute (i.e. the trees are stumps). Figure 2 shows the values boosting predicts for a large test set after 1024 stages of boosting. Consistent with the behavior observed by Schapire and Freund [4], in regions of high purity where the probability of being positive is near 1 or 0, boosting predicts values near 0.67 and 0.33 instead of 1 or 0, respectively. Note that while most predicted values are far from the true probabilities, the ordering predicted by boosting is good: points to the left generally have lower predicted values. Figure 2 also shows that most of the discrepancy between the true probabilities and the values predicted by boosting can be corrected by passing the boosted predictions through a sigmoid. This calibration is a monotonic transformation of the predicted values and thus does not reorder cases.

For data sets for which we do not know the true probabilities, model calibration can be visualized through reliability diagrams [8]. To construct a reliability diagram, the prediction space is discretized into ten bins. Cases with predicted value between 0 and 0.1 are put in the first bin, between 0.1 and 0.2 in the second bin, etc. For each bin, the mean predicted value is plotted against the true fraction of positive cases in the bin. If the model is well calibrated the points will fall near the diagonal line.

The bottom row of Figure 1 shows reliability plots for a two-class version of the UCI Cover Type problem [9] after 1, 4, 8, 32, 128, and 1024 stages of boosting smoothed decision trees [10]. The top of the figure shows histograms of the predicted values for the same models. From the histograms we see that as the number of steps of boosting increases predicted values are pushed away from 0 and 1 and tend to collect on either side of the decision surface. The reliability plots show that this shifting of probability mass away from 0 and 1 hurts calibration and yields sigmoidal shaped reliability plots.

Figure 3 shows histograms and reliability diagrams for boosted decision trees after 1024 steps of boosting on seven test problems. The seven test problems are described in Table 1. (See Section 4 for more detail about these problems.) The figures present results measured on large independent test sets not used for training or validation. For six of the seven data sets the predicted values after boosting

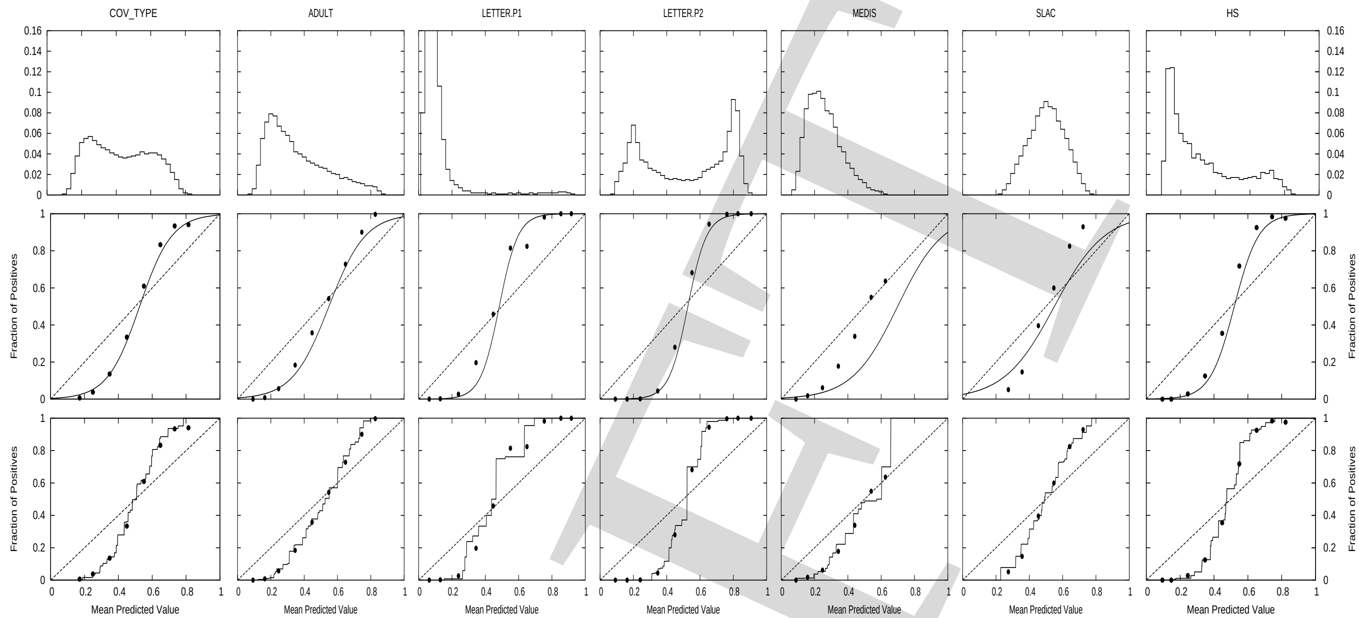

Figure 3: Histograms of predicted values and reliability diagrams for boosted decision trees.

do not approach 0 or 1. The one exception is LETTER.P1, a highly skewed data set that has only 3% positive class. On this problem some of the predicted values do approach 0, though careful examination of the histogram shows that even on this problem there is a sharp drop in the number of cases predicted to have probability near 0.

Table 1: Description of the test problems

PROBLEM	#ATTR	TRAIN SIZE	TEST SIZE	%POZ
ADULT	14/104	4000	35222	25%
COV_TYPE	54	4000	25000	36%
LETTER.P1	16	4000	14000	3%
LETTER.P2	16	4000	14000	53%
MEDIS	63	4000	8199	11%
SLAC	59	4000	25000	50%
HS	200	4000	4366	24%

The reliability plots in Figure 3 display roughly sigmoid-shaped reliability diagrams, motivating the use of a sigmoid to transform predictions into calibrated probabilities. The reliability plots in the middle row of the figure also show sigmoids fitted using Platt's method. The reliability plots in the bottom of the figure show the function fitted with Isotonic Regression. Isotonic Regression and Platt's Method are described in the next section.

3 Calibration

In this section we describe three methods for calibrating model predictions, Logistic Correction, Platt Scaling, and Isotonic Regression.

3.1 Logistic Correction

Before describing Logistic Correction, it is useful to briefly review AdaBoost. Start with each example in the train set (x_i, y_i) having equal weight. At each step i a weak learner h_i is trained on the weighted train set. The error of h_i determines the model weight α_i and the future weight of each training example (give more weight to misclassified cases and less weight to correctly classified cases). There are two equivalent formulations. The first formulation, also used by Friedman, Hastie, and Tibshirani [2] assumes $y_i \in \{-1, 1\}$ and $h_i \in \{-1, 1\}$. The output of the boosted model is:

$$F(x) = \sum_{i=1}^{T} \alpha_i h_i(x) \qquad (1)$$

Friedman et al. show that AdaBoost builds an additive logistic regression model for minimizing $E(exp(-yF(x)))$. They show that $E(exp(-yF(x)))$ is minimized by

$$F(x) = \frac{1}{2} log \frac{P(y=1|x)}{P(y=-1|x)} \qquad (2)$$

This suggests applying a logistic correction in order to get back the conditional probability:

$$P(y=1|x) = \frac{1}{1 + exp(-2F(x))} \qquad (3)$$

As we will see in section 4, this logistic correction works well when boosting simple base learners such as decision stumps. However, if the base learners are powerful enough that the training data becomes fully separable (in the span of the base learners), the probabilities after correction will be only 0's and 1's [11] and thus have poor calibration.

3.2 Platt Calibration

An alternate formulation of AdaBoost assumes $y_i \in \{0, 1\}$ and $h_i \in \{0, 1\}$. The output of the boosted model is:

$$f(x) = \frac{\sum_{i=1}^{T} \alpha_i h_i(x)}{\sum_{i=1}^{T} \alpha_i} \quad (4)$$

We will use this model for Platt Calibration and Isotonic Regression, and treat $f(x)$ as the raw (uncalibrated) output of boosting. Note that:

$$F(x) = (\sum_{i=1}^{T} \alpha_i)(2f(x) - 1) \quad (5)$$

Rosset et al. show that when the data is separable in the span of the base learner, AdaBoost approximates an l_1 margin-maximizing hyperplane in the space of the base learners [11]. In this respect boosting is similar to SVMs, which find l_2 margin-maximizing hyperplanes.

Platt [5] uses a sigmoid to map SVM outputs on $[-\infty, +\infty]$ to posterior probabilities on $[0, 1]$. The sigmoidal shape of the reliability diagrams for boosting in Figure 3 and the similarity between boosting and SVMs suggest that Platt's calibration method for SVMs should be effective at calibrating boosting predictions. In this section we closely follow the description of Platt's method in [5].

Let the output of boosting be $f(x)$ given by equation 4. To get calibrated probabilities, pass the output of boosting through a sigmoid:

$$P(y = 1|f) = \frac{1}{1 + exp(Af + B)} \quad (6)$$

where the parameters A and B are fitted using maximum likelihood estimation from a fitting training set (f_i, y_i). Gradient descent is used to find A and B such that they are the solution to:

$$\underset{A,B}{argmin}\{-\sum_i y_i log(p_i) + (1 - y_i) log(1 - p_i)\}, \quad (7)$$

where

$$p_i = \frac{1}{1 + exp(Af_i + B)} \quad (8)$$

Two questions arise: 1) where does the sigmoid training set (f_i, y_i) come from? 2) how to avoid overfitting to this training set?

One possible answer to question 1 is to use the same training set used for boosting: for each example (x_i, y_i) in the boosting training set, use $(f(x_i), y_i)$ as a training example for the sigmoid. Unfortunately, if the base learner can learn complex models, this introduces unwanted bias in the sigmoid training set that can lead to poor results [5].

An alternate solution is to split the training data into a boosting training set and a calibration validation set. After boosting is trained on the first set, the predictions on the validation set are used to fit the sigmoid. Cross validation can be used to allow both boosting and the sigmoid to be trained on the full data set. The training data is split into C parts. Boosting is performed using C-1 parts, while the C-th part is held aside for use as a calibration validation set. From each of the C validation sets we obtain a sigmoid training set that does not overlap with the boosting training set. The union of these C validation sets is used to fit the sigmoid parameters. Following Platt, all experiments in this paper use 3-fold cross-validation to estimate the sigmoid parameters

As for the second question, an out-of-sample model is used to avoid overfitting to the sigmoid train set . If there are N_+ positive examples and N_- negative examples in the train set, for each training example Platt Calibration uses target values y_+ and y_- (instead of 1 and 0, respectively), where

$$y_+ = \frac{N_+ + 1}{N_+ + 2};\ y_- = \frac{1}{N_- + 2} \quad (9)$$

For a more detailed treatment, and a justification of these particular target values see [5]. The middle row of Figure 3 shows sigmoids fitted with Platt Scaling on the seven test problems using 3-fold CV.

3.3 Isotonic Regression

An alternative to Platt Calibration is to use Isotonic Regression [12]. Zadrozny and Elkan successfully used Isotonic Regression to calibrate predictions made by SVMs, Naive Bayes, boosted Naive Bayes, and decision trees [6, 7].

The basic assumption in Isotonic Regression is:

$$y_i = m(f_i) + \epsilon_i \quad (10)$$

where m is an isotonic (monotonically increasing) function. Then, given a training set (f_i, y_i), the Isotonic Regression problem is finding the isotonic function $\hat{m}$ such that

$$\hat{m} = argmin_z \sum (y_i - z(f_i))^2 \quad (11)$$

One algorithm for Isotonic Regression is pair-adjacent violators (PAV) [13] presented in Table 3. PAV finds a stepwise constant solution for the Isotonic Regression problem.

As in the case of Platt calibration, if we use the boosting training set (x_i, y_i) to get the training set $(f(x_i), y_i)$ for Isotonic Regression, we introduce unwanted bias – in the fully separable case, boosting will order all the negative examples before the positive examples, and Isotonic Regression will output just a 0,1 function. The same methods discussed in Section 3.2 can be used to get an unbiased training set. For the experiments with Isotonic Regression we again use the 3-fold CV methodology used with Platt Scaling. The bottom row of Figure 3 shows functions fitted with Isotonic Regression for the seven test problems.

Table 2: Performance of boosted decision stumps before and after calibration

	SQUARED ERROR				CROSS-ENTROPY			
PROBLEM	RAW	PLATT	ISOTONIC	LOGISTIC	RAW	PLATT	ISOTONIC	LOGISTIC
COV_TYPE	0.4189	**0.3933**	0.3937	0.3961	0.7571	**0.6705**	0.6714	0.6785
ADULT	0.3625	0.3195	**0.3176**	0.3177	0.5891	**0.4570**	0.4632	0.4585
LETTER.P1	0.2147	0.1221	0.1261	**0.1207**	0.2825	**0.0797**	0.0837	0.0820
LETTER.P2	0.4340	0.3448	**0.3446**	0.3450	0.8018	**0.5437**	0.5485	0.5452
MEDIS	0.2974	0.2792	**0.2776**	0.2820	0.4573	0.3829	**0.3754**	0.3836
SLAC	0.4564	0.4428	0.4404	**0.4367**	0.8698	0.8304	0.8351	**0.8114**
HS	0.3771	**0.2667**	0.2674	0.2760	0.6110	**0.3351**	0.3365	0.3824
MEAN	0.3659	0.3098	**0.3096**	0.3106	0.6241	**0.4713**	0.4734	0.4774

Table 4: Performance of boosted decision trees before and after calibration

	SQUARED ERROR				CROSS-ENTROPY			
PROBLEM	RAW	PLATT	ISOTONIC	LOGISTIC	RAW	PLATT	ISOTONIC	LOGISTIC
COV_TYPE	0.3705	**0.3326**	0.3340	0.3921	0.6241	**0.5113**	0.5179	0.7787
ADULT	0.3433	**0.3309**	0.3340	0.3417	0.5439	**0.4957**	0.5072	0.5330
LETTER.P1	0.1145	0.0835	**0.0813**	0.0961	0.0975	**0.0375**	0.0378	0.1172
LETTER.P2	0.2695	0.1734	**0.1724**	0.1986	0.3901	0.1451	**0.1412**	0.5255
MEDIS	0.3081	**0.2803**	0.2809	0.3216	0.4757	**0.3877**	0.3885	0.5596
SLAC	0.4411	0.4303	**0.4271**	0.4510	0.8270	0.7888	**0.7786**	0.8626
HS	0.2882	**0.2237**	0.2266	0.2615	0.4084	**0.2429**	0.2501	0.5988
MEAN	0.3050	**0.2650**	0.2652	0.2947	0.4810	**0.3727**	0.3745	0.5679

Table 3: PAV Algorithm

Algorithm 1. PAV algorithm for estimating posterior probabilities from uncalibrated model predictions.

1 Input: training set (f_i, y_i) sorted according to f_i
2 Initialize $m_{i,i} = y_i$, $w_{i,i}$ = 1
3 While $\exists\ i\ s.t.\ \hat{m}_{k,i-1} \geq \hat{m}_{i,l}$
  Set $w_{k,l} = w_{k,i-1} + w_{i,l}$
  Set $\hat{m}_{k,l} = (w_{k,i-1}\hat{m}_{k,i-1} + w_{i,l}\hat{m}_{i,l})/w_{k,l}$
  Replace $\hat{m}_{k,i-1}$ and $\hat{m}_{i,l}$ with $\hat{m}_{k,l}$
4 Output the stepwise const. function generated by $\hat{m}$

4 Empirical Results

In this section we apply the three scaling methods to predictions from boosted trees and boosted stumps on seven classification problems. The ADULT, COV_TYPE and LETTER problems are from the UCI Machine Learning Repository [9]. COV_TYPE was converted to a binary problem by treating the largest class as positives and the rest as negatives. LETTER was converted to boolean two ways. LETTER.p1 treats the confusable letter "O" as positive and the remaining 25 letters as negative, yielding a very unbalanced problem. LETTER.p2 uses letters A-M as positives and N-Z as negatives, yielding a well balanced problem. HS is the IndianPine92 data set [14] where the difficult class Soybean-mintill is the positive class. SLAC is a particle physics problem from collaborators at the Stanford Linear Accelerator, and MEDIS is a medical data set [15]. Characteristics of these data sets are summarized in Table 1.

We first examine the case when the data is not separable in the span of the base learners. To do this we use boosted stumps. We boost five different types of stumps by using all of the splitting criteria in Buntine's IND package [16]. Because boosting can overfit [11, 2], and because many iterations of boosting can make calibration worse (see Figure 1), we consider boosted stumps after 2,4,8,16,32,64,128,256,512,1024,2048,4096 and 8192 steps of boosting. Table 2 shows the squared-error and cross-entropy[1] on large final test sets for the best boosted stump models before and after calibration. The best models are selected using 1K validation sets. The results clearly show that the performance of boosted stumps is dramatically improved by calibration. The average squared error is reduced about 15% by calibration, and cross-entropy is reduced about 24%. All three scaling methods work equally well. Logistic Correction has the advantage that no extra time is required to fit the calibration model and no cross-validation is needed to create independent test sets.

When using very weak base learners such as 1-level stumps, the Logistic Correction suggested by theory is effective. Unfortunately, because the base learners are so simple, boosted decision stumps are not able to capture all of the structure in most real data-sets. For example, boosted 1-level stumps can't learn models that require parity-like interaction of 2 or more variables. In general, boosted 1-level stumps are optimal when the true model is additive in the features, but suboptimal when the true model requires

[1]To protect against an infinite cross-entropy we prevent the models from predicting exactly 0 or 1

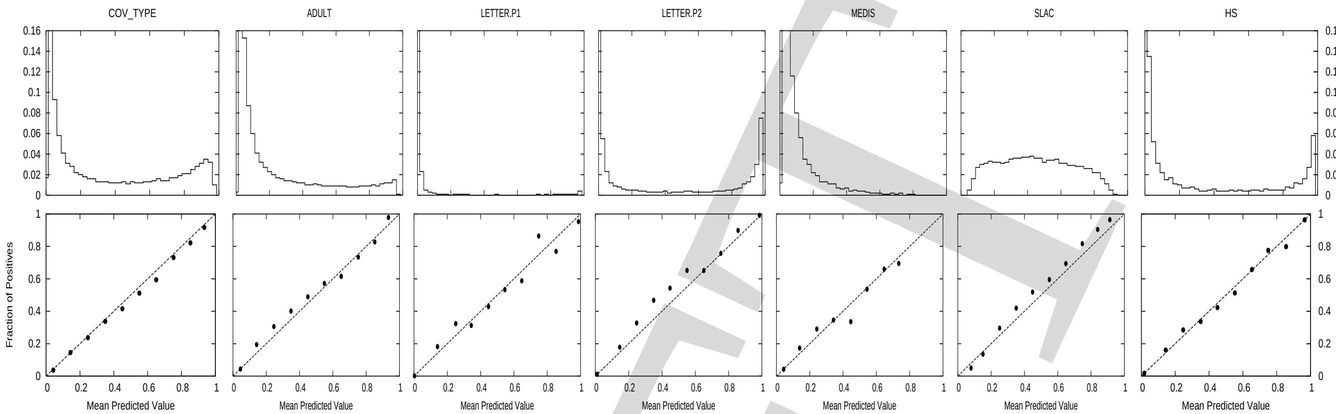

Figure 4: Histograms of predicted values and reliability diagrams for boosted trees calibrated with Platt's method.

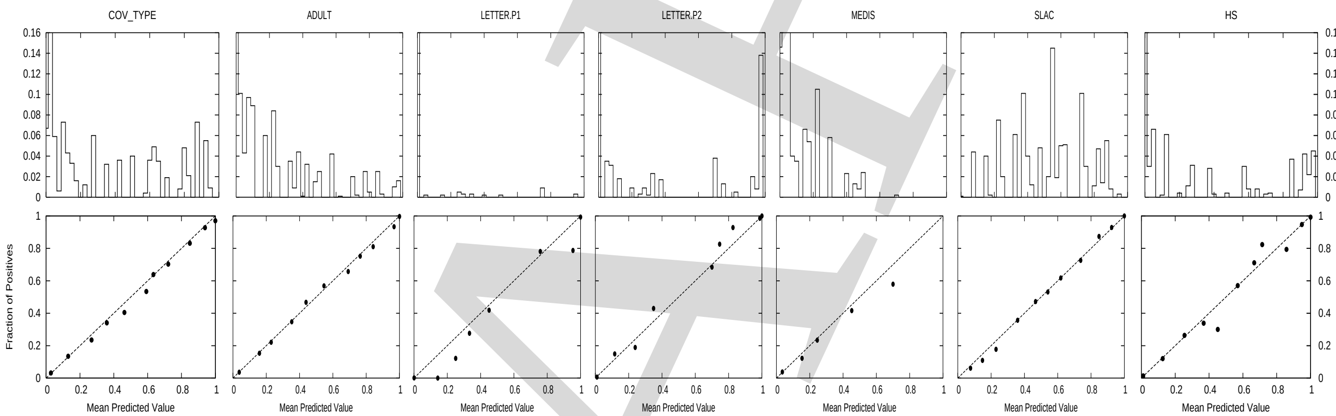

Figure 5: Histograms of predicted values and reliability diagrams for boosted trees calibrated with Isotonic Regression.

non-additive interaction between 2 or more features [2]. As we will see in Table 4, boosting full decision trees yields significantly better performance on 5 of the 7 test problems.

Unlike 1-level stumps, decision trees are complex enough base models to make many data-sets separable (in their span). This means that boosted decision trees are expressive enough to capture the full complexity of most data-sets. Unfortunately this means they are expressive enough to perfectly classify the training set, pushing the probabilities predicted by Logistic Correction to 0 or 1. Although Logistic Correction is no longer effective, Figure 3 shows that good posterior estimates can still be found by fitting a sigmoid or an isotonic function on an independent test set.

Table 4 presents the performance of the best boosted decision trees before and after calibration. To prevent the results from depending on one specific tree style, we boost ten different styles of trees. We use all the tree types in Buntine's IND package [16] (ID3, CART, CART0, C4.5, MML, SMML, BAYES). We also boost three new tree types that should predict better probabilities: C4.5 with no pruning and Laplacian smoothing [17]; C4.5 with no pruning and Bayesian smoothing; and MML trees with Laplacian smoothing. We consider the boosted models after 2,4, 8,16,32,64,128,256,512,1024 and 2048 steps of boosting.

As expected, Logistic Correction is not competitive when boosting full decision trees. The other two calibration methods provide a significant improvement in the quality of the predicted probabilities. Both methods reduce squared error about 13%, and reduce cross-entropy about 22%.

Comparing the results in Tables 2 and 4 we see that boosted trees significantly outperform boosted stumps on five of the seven problems. On average over the seven test problems, boosted trees yield about 14% lower squared error and 21% lower cross-entropy than boosted stumps. Boosted stumps, however, do win by a small margin on two of the problems, and have the nice property that the theoretically suggested Logistic Correction works well.[2]

To show how calibration transforms the predictions, we plot histograms and reliability diagrams for the seven prob-

[2] We also tried boosting 2-level decision trees. Boosted 2-level trees outperformed boosted 1-level stumps, but did not perform as well as boosting full trees. Moreover, 2-level trees are complex enough base-level models that Logistic Correction is no longer as effective as Platt Scaling or Isotonic Regression.

lem for boosted trees after 1024 steps of boosting, after Platt Calibration (Figure 4) and after Isotonic Regression (Figure 5). The figures show that calibration undoes the shift in probability mass caused by boosting: after calibration many more cases have predicted probabilities near 0 and 1. The reliability diagrams are closer to the diagonal, and the S shape characteristic of boosting's predictions is gone. On each problem, transforming the predictions using either Platt Scaling or Isotonic Regression yields a significant improvement in the quality of the predicted probabilities, leading to much lower squared error and cross-entropy. The main difference between Isotonic Regression and Platt Scaling for boosting can be seen when comparing the histograms in the two figures. Because Isotonic Regression generates a piecewise constant function, the histograms are quite coarse, while the histograms generated by Platt Scaling are smooth and easier to interpret.

5 Boosted Trees vs. Other Methods

Boosted trees have such poor initial calibration that it is not surprising that Platt Calibration and Isotonic Regression significantly improves their squared error and cross-entropy. But this does not necessarily mean that calibrated boosted trees yield competitive probabilistic predictions. Table 5 compares the performance of boosted trees and stumps with five other learning methods before and after calibration. Figure 6 depicts the same squared error results graphically. In the figure and table, the performance is shown for boosted decision trees (BST-DT), SVMs, bagged decision trees (BAG-DT), neural nets (ANN), memory based learning methods (KNN), boosted stumps (BST-STMP) and vanilla decision trees (DT). For each learning method, models were trained using many different parameter settings.[3] To obtain the performance for pre-calibrated SVMs, we scaled SVM predictions to [0,1] by $(x - min)/(max - min)$. For each problem and each calibration method, we select the best model trained with each learning algorithm using a 1K validation set and report it's performance on the final test set. The entries in the table represent the average performance over the seven problems.

After calibration with Platt Scaling or Isotonic Regression, boosted decision trees have better squared error and cross-entropy than the other learning methods. Boosting can yield state-of-the-art probabilistic prediction if boosting's predictions are calibrated.

After boosted decision trees, the next best methods are SVMs, bagged decision trees and neural nets. While Platt Scaling and Isotonic Regression significantly improve the performance of the SVM models, they have little or no effect on the performance of bagged decision trees and neural nets. Both bagged decision trees and neural nets yield

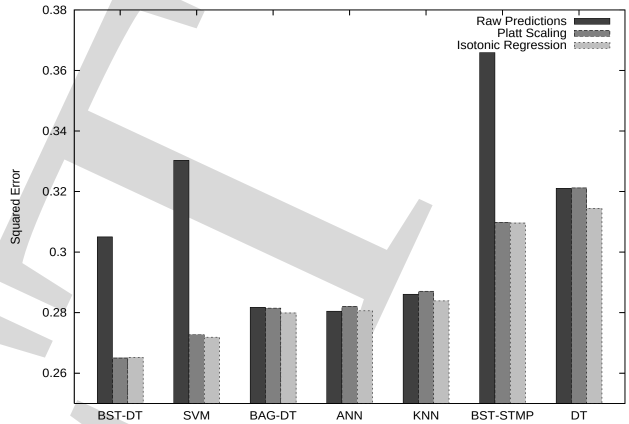

Figure 6: Squared error performance of learning algorithms

well calibrated predictions that do not benefit from post-calibration. It is interesting to note that neural networks with a single sigmoid output unit can be viewed as a linear classifier (in the span of it's hidden units) with a sigmoid at the output that calibrates the predictions. In this respect neural nets are similar to SVMs and boosted decision trees that have been calibrated using Platt's method.

Platt Scaling is less effective than Isotonic Regression with bagged trees, neural nets, memory based learning, and vanilla decision trees. This probably is because a sigmoid is not the right function for mapping the raw predictions generated by these learning algorithms to posterior probabilities. In these cases, Isotonic Regression yields somewhat better squared error and cross-entropy than Platt Scaling, but still not enough to make these learning methods perform as well as calibrated boosted decision trees.

6 Conclusions

In this paper we empirically demonstrated that boosting yields poorly calibrated predictions, limiting it's use when probabilistic predictions are needed. We experimented with three calibration methods: Logistic Correction justified by Friedman et al.'s analysis, Platt Scaling justified by the sigmoidal shape of the reliability plots, and Isotonic Regression, a general non-parametric calibration method.

Both Platt Scaling and Isotonic Regression yield better reliability diagrams, and significantly improve the squared-error and cross-entropy of boosted decision trees. Logistic Correction is effective when boosting very weak models such as 1-level decision stumps, but is not competitive when boosting more powerful models such as full decision trees (or even 2-level trees).

When compared to other learning algorithms, boosted decision trees that have been calibrated with Platt Scaling or Isotonic Regression yield the best average squared error and cross-entropy performance on seven test problems.

[3]See [1] for a description of all the models that were trained.

Table 5: Squared error and cross-entropy performance of learning algorithms

ALGORITHM	SQUARED ERROR			CROSS-ENTROPY		
	RAW	PLATT	ISOTONIC	RAW	PLATT	ISOTONIC
BST-DT	0.3050	**0.2650**	**0.2652**	0.4810	**0.3727**	**0.3745**
SVM	0.3303	0.2727	0.2719	0.5767	0.3988	0.3984
BAG-DT	0.2818	0.2815	0.2799	**0.4050**	0.4082	0.3996
ANN	**0.2805**	0.2821	0.2806	0.4143	0.4229	0.4120
KNN	0.2861	0.2871	0.2839	0.4367	0.4300	0.4186
BST-STMP	0.3659	0.3098	0.3096	0.6241	0.4713	0.4734
DT	0.3211	0.3212	0.3145	0.5019	0.5091	0.4865

This, combined with their excellent performance on other measures such as accuracy, precision, and area under the ROC curve [1], may make calibrated boosted decision trees the model of choice for many applications.

Acknowledgments

Thanks to Bianca Zadrozny and Charles Elkan for use of their Isotonic Regression code. Thanks to Charles Young et al. at SLAC (Stanford Linear Accelerator) for the SLAC data set. Thanks to Tony Gualtieri at Goddard Space Center for help with the Indian Pines Data set.

References

[1] R. Caruana and A. Niculescu-Mizil. An empirical comparison of supervised learning algorithms using different performance metrics. In *In preparation. Draft available at http://www.cs.cornell.edu/c̃aurana/*, 2004.

[2] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *The Annals of Statistics*, 38(2):337–374, 2000.

[3] R. Schapire. The boosting approach to machine learning: An overview. In *In MSRI Workshop on Nonlinear Estimation and Classification*, 2001.

[4] R. Schapire, Y. Freund, P. Bartlett, and W. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *Annals of Statistics*, 26(5):1651–1686, 1998.

[5] J. Platt. Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. In A.J. Smola, P. Bartlett, B. Schoelkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 61–74, 1999.

[6] B. Zadrozny and C. Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *KDD*, pages 694–699, 2002.

[7] B. Zadrozny and C. Elkan. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In *ICML*, pages 609–616, 2001.

[8] M.H. DeGroot and S.E. Fienberg. The comparison and evaluation of forecasters. *Statistician*, 32(1):12–22, 1982.

[9] C.L. Blake and C.J. Merz. UCI repository of machine learning databases, 1998.

[10] W. Buntine. Learning classification trees. *Statistics and Computing*, 2:63–73, 1992.

[11] Saharon Rosset, Ji Zhu, and Trevor Hastie. Boosting as a regularized path to a maximum margin classifier. *J. Mach. Learn. Res.*, 5:941–973, 2004.

[12] T. Robertson, F. Wright, and R. Dykstra. *Order Restricted Statistical Inference*. John Wiley and Sons, New York, 1988.

[13] M. Ayer, H. Brunk, G. Ewing, W. Reid, and E. Silverman. An empirical distribution function for sampling with incomplete information. *Annals of Mathematical Statistics*, 5(26):641–647, 1955.

[14] A. Gualtieri, S. R. Chettri, R.F. Cromp, and L.F. Johnson. Support vector machine classifiers as applied to aviris data. In *Proc. Eighth JPL Airborne Geoscience Workshop*, 1999.

[15] G. F. Cooper, C. F. Aliferis, R. Ambrosino, J. Aronis, B. G. Buchanan, R. Caruana, M. J. Fine, C. Glymour, G. Gordon, B. H. Hanusa, J. E. Janosky, C. Meek, T. Mitchell, T. Richardson, and P. Spirtes. An evaluation of machine learning methods for predicting pneumonia mortality. *Artificial Intelligence in Medicine*, 9, 1997.

[16] http://ic.arc.nasa.gov/projects/bayes-group/ind/ind-program.html.

[17] F. Provost and P. Domingos. Tree induction for probability-based rankings. *Machine Learning*, 52(3), 2003.