1.6A PREDICTING GOOD PROBABILITIES WITH SUPERVISED LEARNING

Rich Caruana and Alexandru Niculescu-Mizil

Computer Science, Cornell University, Ithaca, New York

1. INTRODUCTION

This paper presents the results of an empirical evaluation of the probabilities predicted by seven supervised learning algorithms. The algorithms are SVMs, neural nets, decision trees, memory-based learning, bagged trees, boosted trees, and boosted stumps. For each algorithm we test many different variants and parameter settings: we compare ten styles of decision trees, neural nets of many sizes, SVMs using different kernels, etc. A total of 2000 models are tested on each problem.

Experiments with seven classi£cation problems suggest that neural nets and bagged decision trees are the best learning methods for predicting well-calibrated probabilities. However, while SVMs and boosted trees are not well calibrated, they have excellent performance on other metrics such as accuracy and area under the ROC curve (AUC). We analyze the predictions made by these models and show that they are distorted in a speci£c and consistent way. To correct for this distortion, we experiment with two methods for calibrating probabilities:

Platt Scaling: a method for transforming SVM outputs from $[-\infty, +\infty]$ to posterior probabilities (Platt, 1999)

Isotonic Regression: the method used by Elkan and Zadrozny to calibrate predictions from boosted naive bayes, SVM, and decision tree models (Zadrozny & Elkan, 2002; Zadrozny & Elkan, 2001)

Comparing the performance of the learning algorithms before and after calibration, we see that calibration signi£cantly improves the performance of boosted trees and SVMs. After calibration, these two learning methods outperform neural nets and bagged decision trees and become the best learning methods for predicting calibrated posterior probabilities. Boosted stumps also bene£t significantly from calibration, but their performance overall is not competitive. Not surprisingly, the two model types that were well calibrated to start with, neural nets and bagged trees, do not bene£t from calibration.

2. METHODOLOGY

2.1. Learning Algorithms

This section summarizes the parameters used with each learning algorithm.

KNN: we use 26 values of K ranging from $K = 1$ to $K = |trainset|$. We use KNN with Euclidean distance and distance weighted by gain ratio. We also use distance weighted KNN, and locally weighted averaging.

ANN we train neural nets with backprop varying the number of hidden units $\{1,2,4,8,32,128\}$ and momentum $\{0,0.2,0.5,0.9\}$. We don't use validation sets to do weight decay or early stopping. Instead, we stop the nets at many different epochs so that some nets under£t or over£t.

Decision trees (DT): we vary the splitting criterion, pruning options, and smoothing (Laplacian or Bayesian smoothing). We use all of the tree models in Buntine's IND package: BAYES, ID3, CART, CART0, C4, MML, and SMML. We also generate trees of type C44LS (C4 with no pruning and Laplacian smoothing)(Provost & Domingos, 2003), C44BS (C44 with Bayesian smoothing), and MMLLS (MML with Laplacian smoothing).

Bagged trees (BAG-DT): we bag 25-100 trees of each tree type. **Boosted trees (BST-DT):** we boost each tree type. Boosting can over£t, so we use 2,4,8,16,32,64,128,256,512,1024 and 2048 steps of boosting. **Boosted stumps (BST-STMP):** we use stumps (single level decision trees) generated with 5 different splitting criteria boosted for 2,4,8, 16,32,64,128,256,512,1024,2048,4096,8192 steps.

SVMs: we use the following kernels in SVM-Light(Joachims, 1999): linear, polynomial degree 2 & 3, radial with width $\{0.001,0.005,0.01,0.05,0.1,0.5,1,2\}$ and vary the regularization parameter by factors of ten from 10^{-7} to 10^{3}.

With ANN's, SVM's and KNN's we scale attributes to 0 mean 1 std. With DT, BAG-DT, BST-DT and BST-STMP we don't scale the data. In total, we train about 2000 different models on each test problem.

2.2. Performance Metrics

Finding models that predict the true underlying probability for each test case would be optimal. Unfortunately, we usually do not know how to train models to predict true underlying probabilities. Either the correct parametric model type is not known, or the training sample is too

small for model parameters to be estimated accurately, or there is noise in the data. Typically, all of these problems occur to varying degrees. Moreover, usually we don't have access to the true underlying probabilities. We only know if a case is positive or not, making it dif£cult to detect when a model predicts the true underlying probabilities.

Some performance metrics are minimized (in expectation) when the predicted value for each case is the true underlying probability of that case being positive. We call these probability metrics. The probability metrics we use are squared error (RMS), cross-entropy (MXE) and calibration (CAL). CAL measures the calibration of a model: if the model predicts 0.85 for a number of cases, it is well calibrated if 85% of cases are positive. CAL is calculated as follows: Order all cases by their predictions and put cases 1-100 in the same bin. Calculate the percentage of these cases that are true positives to estimate the true probability that these cases are positive. Then calculate the mean prediction for these cases. The absolute value of the difference between the observed frequency and the mean prediction is the calibration error for these 100 cases. Now take cases 2-101, 3-102, ... and compute the errors in the same way. CAL is the mean of all these binned calibration errors.

Other metrics don't treat predicted values as probabilities, but still give insight into model quality. Two commonly used metrics are accuracy (ACC) and area under ROC curve (AUC). Accuracy measures how well the model discriminates between classes. AUC is a measure of how good a model is at ordering the cases, i.e. predicting higher values for instances that have a higher probability of being positive. See (Provost & Fawcett, 1997) for a discussion of ROC from a machine learning perspective. AUC depends only on the ordering of the predictions, not the actual predicted values. If the ordering is preserved it makes no difference if the predicted values are between 0 and 1 or between 0.49 and 0.51.

2.3. Data Sets

We compare the algorithms on 7 binary classi£cation problems. The data sets are summarized in Table 1.*

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%

3. Calibration Methods

3.1. Platt Calibration

Let the output of a learning method be $f(x)$. To get calibrated probabilities, pass the output through a sigmoid:

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

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

$$\underset{A,B}{argmin}\{-\sum_i y_i log(p_i) + (1-y_i) log(1-p_i)\}, \quad (2)$$

where

$$p_i = \frac{1}{1+exp(Af_i+B)} \quad (3)$$

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

One possible answer to question 1 is to use the same training set used for training the model: for each example (x_i, y_i) in the training set, use $(f(x_i), y_i)$ as a training example for the sigmoid. Unfortunately, if the learning algorithm can learn complex models it will introduces unwanted bias in the sigmoid training set that can lead to poor results (Platt, 1999).

An alternate solution is to split the training data into a model training set and a calibration validation set. After the model is trained on the £rst set, the predictions on the validation set are used to £t the sigmoid. Cross validation can be used to allow both the model and the sigmoid to be trained on the full data set. The training data is split into C parts. The model is learned 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 model training set. The union of these C validation sets is used to £t 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 over£tting 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, respec-

*Unfortunately, none of these are meteorology data.

Table 2. Performance of learning algorithms prior to calibration

MODEL	ACC	AUC	RMS	MXE	CAL
ANN	0.8720	0.9033	0.2805	0.4143	0.0233
BAG-DT	0.8728	0.9089	0.2818	0.4050	0.0314
KNN	0.8688	0.8970	0.2861	0.4367	0.0270
DT	0.8433	0.8671	0.3211	0.5019	0.0346
SVM	0.8745	0.9067	0.3390	0.5767	0.0765
BST-STMP	0.8470	0.8866	0.3659	0.6241	0.0502
BST-DT	0.8828	0.9138	0.3050	0.4810	0.0542

tively), where

$$y_+ = \frac{N_+ + 1}{N_+ + 2};\ y_- = \frac{1}{N_- + 2} \tag{4}$$

For a more detailed treatment, and a justi£cation of these particular target values see (Platt, 1999). The middle row of Figure 1 shows sigmoids £tted with Platt Scaling on the seven test problems using 3-fold CV.

3.2. Isotonic Regression

An alternative to Platt Calibration is Isotonic Regression (Robertson et al., 1988). Zadrozny and Elkan used Isotonic Regression to calibrate predictions made by SVMs, Naive Bayes, boosted Naive Bayes, and decision trees (Zadrozny & Elkan, 2002; Zadrozny & Elkan, 2001).

The basic assumption in Isotonic Regression is:

$$y_i = m(f_i) + \epsilon_i \tag{5}$$

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

$$\hat{m} = argmin_z \sum (y_i - z(f_i))^2 \tag{6}$$

One algorithm for Isotonic Regression is pair-adjacent violators (PAV) (Ayer et al., 1955) presented in Table 3. PAV £nds a stepwise constant solution for the Isotonic Regression problem.

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 $\hat{m}$

As in the case of Platt calibration, if we use the model training set (x_i, y_i) to get the training set $(f(x_i), y_i)$ for Isotonic Regression, we introduce unwanted bias. The same methods discussed in Section 3.1 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 1 shows functions £tted with Isotonic Regression for the seven test problems.

4. EMPIRICAL RESULTS

Table 2 shows the average performance of the learning algorithms on the seven test problems. For each problem, we select the best model trained with each learning algorithm using a 1K validation set and report it's performance on large £nal test sets. The learning methods with best performance on the probability metrics (RMS, MXE, and CAL) are neural nets and bagged decision trees. The learning methods with the poorest performance are SVMs [†], boosted stumps, and boosted decision trees. Interestingly, although SVMs and the boosted models predict poor probabilities, they outperform neural nets and bagged trees on accuracy and AUC. This suggests that SVMs and the boosted models are learning good models, but their predictions are distorted and thus have poor calibration.

Model calibration can be visualized through reliability diagrams (DeGroot & Fienberg, 1982). To construct a reliability diagram, the prediction space is discretized into ten bins. Cases with predicted value between 0 and 0.1 fall in the £rst 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. If the model is well calibrated the points will fall near the diagonal line.

Figure 1 shows histograms and reliability diagrams for boosted trees after 1024 steps of boosting on seven test problems. The results are for large test sets not used for training or validation. For six of the seven data sets the predicted values after boosting 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

[†] SVM predictions are scaled to [0,1] by $(x - min)/(max - min)$.

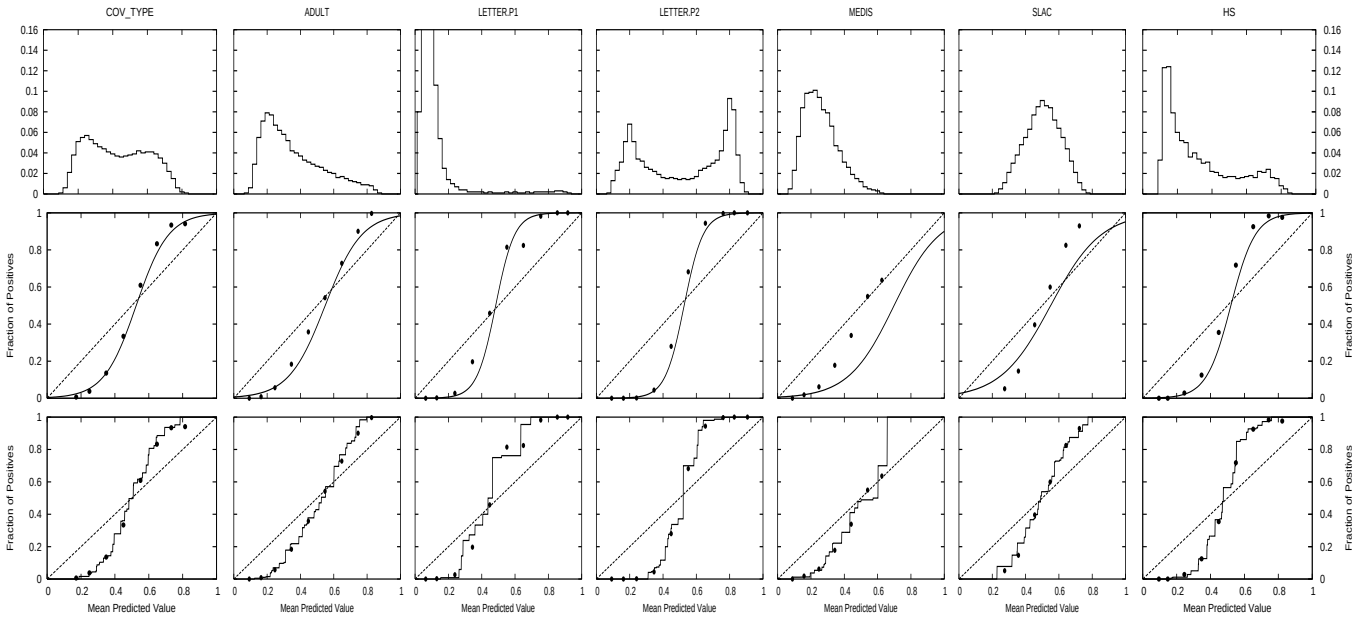

Figure 1. Histograms of predicted values and reliability diagrams for boosted decision trees.

Table 4. Squared error and cross-entropy performance of learning algorithms

	SQUARED ERROR			CROSS-ENTROPY		
ALGORITHM	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

to have probability near 0.

The reliability plots in Figure 1 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 £gure also show sigmoids £tted using Platt's method. The reliability plots in the bottom of the £gure show the function £tted with Isotonic Regression.

To show how calibration transforms the predictions, we plot histograms and reliability diagrams for the seven problem for boosted trees after 1024 steps of boosting, after Platt Calibration (Figure 2) and after Isotonic Regression (Figure 3). The reliability diagrams for Isotonic Regression are very similar to the ones for Platt Scaling, so we omit them in the interest of space. The £gures 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 signi£cant 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 £gures. 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.

Table 4 compares the RMS and MXE performance of the learning methods before and after calibration. Figure 4 shows the squared error results from Table 4 graphically.

After calibration with Platt Scaling or Isotonic Regression, boosted decision trees have better squared error and cross-entropy than the other learning methods. The next best methods are SVMs, bagged decision trees and neural nets. While Platt Scaling and Isotonic Regression signi£cantly improve the performance of the SVM models, they have little or no effect on the performance of bagged

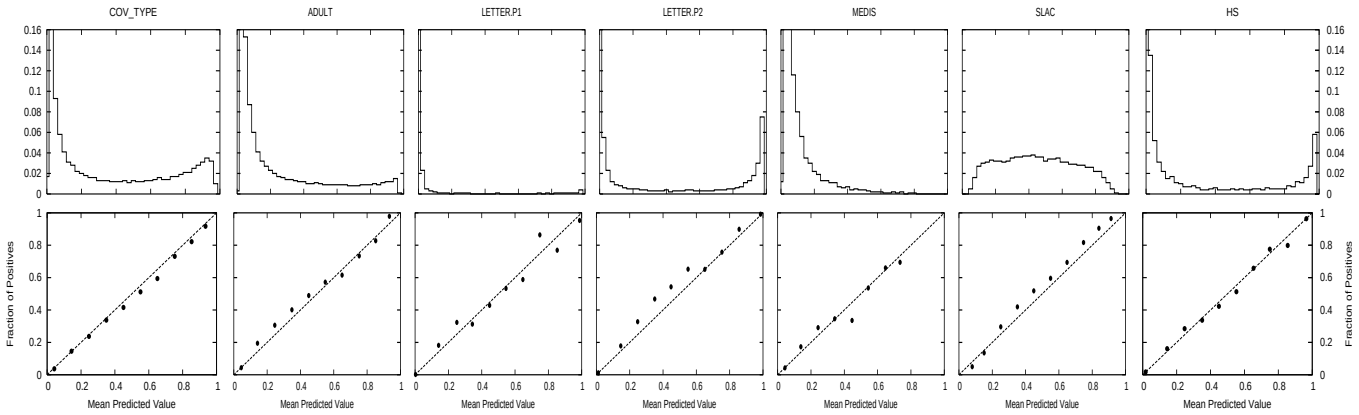

Figure 2. Histograms of predicted values and reliability diagrams for boosted trees calibrated with Platt's method.

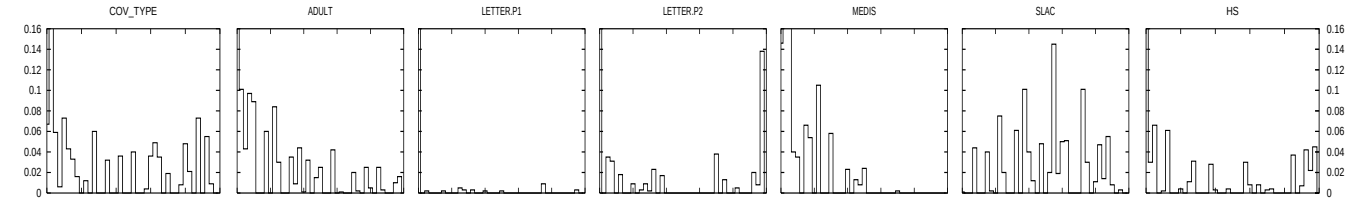

Figure 3. Histograms of predicted values for boosted trees calibrated with Isotonic Regression.

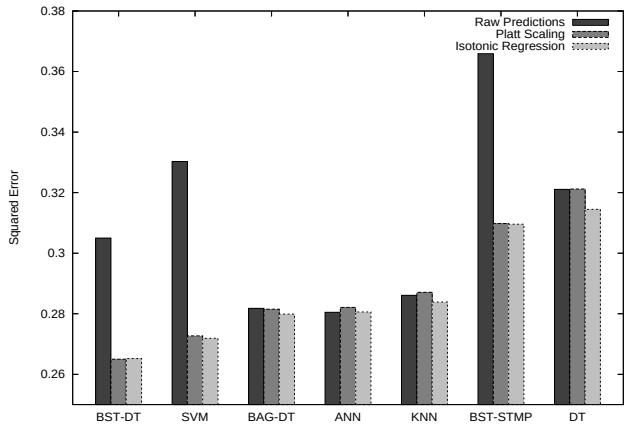

Figure 4. Squared error performance of learning algorithms

decision trees and neural nets. While neural nets and bagged trees yield better probabilities before calibration, Platt Scaling or Isotonic Regression improve the calibration of maximum margin methods enough for boosted trees and SVMs to become the best methods for predicting good probabilities once calibrated.

Acknowledgements

Thanks to B. Zadrozny and C. Elkan for the Isotonic Regression code, to C. Young at Stanford Linear Accelerator for the SLAC data, and to T. Gualtieri at Goddard Space Center for help with the Indian Pines Data. This work was supported by NSF Grant IIS-0412930.

References

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

DeGroot, M., & Fienberg, S. (1982). The comparison and evaluation of forecasters. *Statistician*, *32*, 12–22.

Joachims, T. (1999). Making large-scale svm learning practical. *Advances in Kernel Methods*.

Platt, J. (1999). Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. *Advances in Large Margin Classi£ers* (pp. 61–74).

Provost, F., & Domingos, P. (2003). Tree induction for probability-based rankings. *Machine Learning*, *52*.

Provost, F. J., & Fawcett, T. (1997). Analysis and visualization of classi£er performance: Comparison under imprecise class and cost distributions. *Knowledge Discovery and Data Mining* (pp. 43–48).

Robertson, T., Wright, F., & Dykstra, R. (1988). *Order restricted statistical inference*. New York: John Wiley and Sons.

Zadrozny, B., & Elkan, C. (2001). Obtaining calibrated probability estimates from decision trees and naive bayesian classi£ers. *ICML* (pp. 609–616).

Zadrozny, B., & Elkan, C. (2002). Transforming classi£er scores into accurate multiclass probability estimates. *KDD* (pp. 694–699).