A revised version of this paper appeared in the Proceedings of ICML'06.

An Empirical Comparison of Supervised Learning Algorithms Using Different Performance Metrics

Rich Caruana and Alexandru Niculescu-Mizil

Computer Science, Cornell University, Ithaca NY 14850, USA

Abstract. We present results from a large-scale empirical comparison between ten learning methods: SVMs, neural nets, logistic regression, naive bayes, memory-based learning, random forests, decision trees, bagged trees, boosted trees, and boosted stumps. We evaluate the methods on binary classification problems using nine performance criteria: accuracy, squared error, cross-entropy, ROC Area, F-score, precision/recall break-even point, average precision, lift, and calibration. Because some models (e.g. SVMs and boosted trees) do not predict well-calibrated probabilities, we compare the performance of the algorithms both before and after calibrating their predictions with Platt Scaling and Isotonic Regression. Before scaling, the models with the best overall performance are neural nets, bagged trees, and random forests. After scaling, the best models are boosted trees, random forests, and *unscaled* neural nets.

1 Introduction

There are few comprehensive empirical studies comparing learning algorithms. STATLOG is perhaps the best known study [1]. STATLOG was very comprehensive, but since it was performed new learning algorithms have emerged (e.g., bagging, boosting, SVMs, random forests) that have excellent performance. Also, learning algorithms are now evaluated on a broader set of performance metrics. For example, the IR and NLP communities use Precision/Recall metrics, ROC is used in medical informatics and has become popular in machine learning, etc. It is important to evaluate learning algorithms on a variety of performances metrics because different learning algorithms are designed to to optimize different criteria (e.g. SVMs and boosting optimize accuracy while neural nets typically optimize squared error) and it is not uncommon for an algorithm to have optimal performance on one performance metric and be suboptimal on another.

This paper presents the results of a large-scale empirical comparison between ten supervised learning algorithms using nine performance criteria. The ten algorithms are: SVMs, neural nets, logistic regression, naive bayes, memory-based learning, random forests, decision trees, bagged trees, boosted trees, and boosted stumps. We evaluate performance with accuracy, F-score, Lift, ROC Area, average precision, precision/recall break-even point, squared error, cross-entropy, and probability calibration. We test many variations and parameter settings for

each algorithm and select for each metric the variation/settings that perform best. For example, we test SVMs with many kernels and kernel parameters and report the performance of the kernel that has the best accuracy, the kernel that has best AUC, etc. About 2000 models are tested on each problem.

Some of the results are surprising. To preview: neural networks give the best average performance across all nine metrics and test problems. They are closely followed by bagged trees and random forests. Boosted trees have the best performance on accuracy, ROC area, average precision and break-even point, but perform poorly on squared error, cross-entropy and calibration. SVMs also have poor squared error, cross-entropy and calibration, but do well on other metrics. Boosting stumps yields significantly worse performance than boosting full decision trees on most problems. Memory-based learning performs better than single decision trees and boosted stumps, but is not competitive with the best methods. Because of their limited expressiveness, logistic regression and naive bayes have the worst overall performance.

The poor squared error, cross-entropy and calibration of algorithms such as boosted trees and SVMs is easy to explain. These three metrics interpret predictions as posterior probabilities, but SVMs and boosted trees are not designed to predict probabilities. Because of this, we compare the performance of each algorithm both before and after calibrating its predictions with Platt Scaling and Isotonic Regression. Calibration dramatically improves the performance of boosted trees, boosted stumps, SVMs and naive bayes on the probability metrics. After calibration, boosted trees have the best performance across the nine metrics, now outperforming neural nets. Calibration also improves the performance of random forests moving them to second place overall. Neural nets, bagged trees, memory-based learning and logistic regression, however, receive little or no benefit from calibration.

2 Methodology

2.1 Learning Algorithms

We explore the parameter space and variations of each learning algorithm as thoroughly as is computationally feasible. This section summarizes the parameters used for each learning algorithms and may be skipped.

SVMs: we use the following kernels in SVMLight [2]: 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 each kernel.

ANN we train neural nets with gradient descent backprop and vary the number of hidden units {1,2,4,8,32,128} and the momentum {0,0.2,0.5,0.9}. We don't use validation sets for weight decay or early stopping. Instead, we stop the nets at many different epochs so that some nets underfit or overfit.

Logistic Regression (LOGREG): we train unregularized and regularized models, varying the regularization parameter by factors of 10 from 10^{-8} to 10^{4}.

Naive Bayes (NB): we use the Weka implementation. We handle continuous attributes three ways: model them as a single normal, model them with kernel estimation, or discretize them using supervised discretization.

KNN: we use 26 values of K ranging from $K = 1$ to $K = |trainset|$. We use KNN with Euclidean distance and Euclidean distance weighted by gain ratio. We also use distance weighted KNN, and locally weighted averaging. The kernel widths for locally weighted averaging vary from 2^0 to 2^{10} times the minimum distance between any two points in the train set.
Random Forests (RF): we use the Weka implementation. The forests have 1024 trees, and the size of the feature set to consider at each split is 1,2,4,6 or 8.
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), C44BS (C44 with Bayesian smoothing), and MMLLS (MML with Laplacian smoothing). See [3] for a description of C44LS.
Bagged trees (BAG-DT): we bag 100 trees of each type described above. With **boosted trees (BST-DT)** we boost each tree type as well. Boosting can overfit, so we consider boosted trees after 2,4,8,16,32,64,128,256,512,1024 and 2048 steps of boosting. With **boosted stumps (BST-STMP)** we boost single level decision trees generated with 5 different splitting criteria, each boosted for 2,4,8,16,32,64,128,256,512,1024,2048,4096,8192 steps.

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

2.2 Performance Metrics

We divide the nine performance metrics into three groups: threshold metrics, ordering/rank metrics and probability metrics.

For the threshold metrics, accuracy (ACC), F-score (FSC) and lift (LFT), it is not important how close a prediction is to a threshold, only if it is above or below threshold. Usually ACC and FSC have a fixed threshold (we use 0.5). For lift (see [4] for a description of Lift), often a fixed percent, p, of cases are predicted as positive and the rest as negative (we use $p = 25\%$).

The ordering/rank metrics depend only on the ordering of the cases, not the actual predicted values. As long as ordering is preserved, it makes no difference if predicted values fall between 0 and 1 or 0.89 and 0.90. These metrics measure how well the positive cases are ordered before negative cases and can be viewed as a summary of model performance across all possible thresholds. The rank metrics we use are area under the ROC curve (ROC), average precision (APR), and precision/recall break even point (BEP). See [5] for a discussion of ROC from a machine learning perspective.

The probability metrics are minimized (in expectation) when the predicted value for each case coincides with the true conditional probability of that case being positive class. The probability metrics are squared error (RMS), cross-entropy (MXE) and calibration (CAL). CAL measures the calibration of a model: if a model predicts values near 0.85 for a number of cases, then about 85% of those cases should prove to be positive class if the model is well calibrated at

$p = 0.85$. 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. This approximates 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.

2.3 Comparing Across Performance Metrics

Performance metrics such as accuracy or squared error have range $[0, 1]$, while others (lift, cross entropy) range from 0 to p where p depends on the data set. For some metrics lower values indicate better performance. For others higher values are better. Metrics such as ROC area have baseline rates that are independent of the data, while others such as accuracy have baseline rates that depend on the data. If baseline accuracy is 0.98, an accuracy of 0.981 probably is not good performance, but on another problem the Bayes optimal rate might be 0.60 and achieving an accuracy of 0.59 might be excellent performance.

To permit averaging across metrics and problems, performances must be placed on a comparable scale. One way to do this is to normalize the performance for each problem and metric from 0 to 1, where 0 is baseline performance and 1 is Bayes optimal. Because we cannot estimate the Bayes optimal rate on real problem we use the best observed performance as a proxy. We use the following baseline model: predict p for every case, where p is the percent of positives in the test set. If a model performs worse than baseline, its normalized score will be negative. CAL, the metric used to measure probability calibration, is unusual in that the baseline model has excellent calibration.[1] This creates a problem when normalizing CAL scores because the baseline model and Bayes optimal model have similar CAL scores. Unlike the other measures, CAL is scaled so that the mean observed CAL score is 0.0 and the best observed CAL score is 1.0.

One disadvantage of normalized scores is that recovering a raw performance requires knowing what performances define the top and bottom of the scale, and as new best models are found the top of the scale may change. We will make the performances that define the top and bottom of the scales for each problem and metric available on the web so that others may compare to our results.

2.4 Calibration Methods

Some of the learning algorithms we examine are not designed to predict probabilities. For example the outputs of an SVM are just normalized distances to the decision boundary. And naive bayes models are known to predict poorly calibrated probabilities because of the unrealistic independence assumption.

[1] Because of this, CAL typically is not used alone, but is used in conjunction with other measures such as ROC or RMS to insure that models have both good discrimination *and* good calibration. This does not mean CAL is a poor metric – it is effective at distinguishing poorly calibrated models from well calibrated models.

Table 1. Description of problems

PROBLEM	#ATTR	TRAIN SIZE	TEST SIZE	%POZ
ADULT	14/104	4000	35222	25%
COVT	54	4000	25000	36%
LTR.P1	16	4000	14000	3%
LTR.P2	16	4000	14000	53%
MEDIS	63	4000	8199	11%
MG	124	4000	12807	17%
SLAC	59	4000	25000	50%
HS	200	4000	4366	24%

A number of methods have been proposed for mapping predictions to posterior probabilities. Platt [6] proposed transforming SVM predictions to posterior probabilities by passing them through a sigmoid. Platt's method also works well for boosted trees and boosted stumps. A sigmoid, however, might not be the correct transformation for all learning algorithms.

Zadrozny and Elkan[7, 8] used a more general method based on Isotonic Regression [9] to calibrate predictions from SVMs, naive bayes, boosted naive bayes, and decision trees. Isotonic Regression is more general in that the only restriction it makes is that the mapping function be isotonic (monotonically increasing). A standard algorithm for Isotonic Regression that finds a piecewise constant solution is the pool-adjacent violators (PAV) algorithm [10]. To calibrate models, we use the same 1000 points validation set that will be used for model selection.

2.5 Data Sets

We compare the algorithms on 8 binary classification problems. ADULT, COVT and LETTER are from UCI Machine Learning Repository [11]. ADULT is the only problem that has nominal attributes. For ANNs, SVMs and KNNs we transform nominal attributes to boolean. Each DT, BAG-DT, BST-DT, BST-STMP, and NB model is trained twice, once with the transformed attributes and once with the original attributes. COVT has been converted to a binary problem by treating the largest class as the positive and the rest as negative. We converted LETTER to boolean in two ways. LTR.p1 treats the letter "O" as positive and the remaining 25 letters as negative, yielding a very unbalanced binary problem. LTR.p2 uses letters A-M as positives and the rest as negatives, yielding a difficult, but well balanced, problem. HS is the IndianPine92 data set [12] where the difficult class Soybean-mintill is the positive class. SLAC is a problem from the Stanford Linear Accelerator. MEDIS and MG are medical data sets. The characteristics of these data sets are summarized in Table 1.

3 Performances by Metric

For each test problem we randomly select 5000 cases for training and use the rest of the cases as a large final test set. We use 5-fold cross validation on the 5000

Table 2. Normalized scores for each learning method by metric (avg. over 8 problems)

MODEL	C	ACC	FSC	LFT	ROC	APR	BEP	RMS	MXE	CAL	MEAN	OPT
BST-DT	P	.860*	.854	.956*	**.977**	**.958**	**.952**	**.929**	**.932**	.808*	**.914**	**.941**
BST-DT	I	.812*	.921*	.948	.965	.937	.942	.896	.859	.798*	.897	.939*
RF	P	**.866**	.871	**.958**	.977*	.957*	.948*	.892	.898	.702	.897	.906
RF	I	.847*	.915	.948	.966	.937	.940*	.881	.828	.791	.895	.910
ANN	-	.817*	.875	.947*	.963	.926	.929	.872	.878	**.826**	.892	.932
SVM	P	.823	.851	.928	.961	.931	.929	.882	.880	.769	.884	.909
BAG-DT	I	.820	.886	.947	.961	.930	.925	.859	.822	.771	.880	.905
BAG-DT	-	.836	.849	.953	.972	.950*	.928	.875	.901	.637	.878	.899
RF	-	.844*	.845	**.958**	.977*	.957*	.948*	.882	.899	.567	.875	.882
BAG-DT	P	.822	.843	.953	.972	.950*	.929	.863	.874	.666	.875	.893
ANN	I	.816*	**.934**	.943	.949	.906	.920	.835	.782	.767	.873	.927
SVM	I	.806	.914	.924	.946	.910	.928	.859	.799	.759*	.872	.912
ANN	P	.833	.863	.947*	.963	.926	.929	.842	.839	.651	.866	.903
KNN	P	.759	.820	.914	.937	.893	.898	.786	.805	.706	.835	.869
KNN	I	.753	.866	.905	.926	.873	.893	.782	.756	.746	.834	.883
KNN	-	.759	.839	.914	.937	.893	.898	.783	.769	.684	.831	.858
BST-STMP	P	.698	.760	.898	.926	.871	.854	.740	.783	.678	.801	.834
BST-STMP	I	.677	.821	.892	.916	.850	.852	.708	.674	.679	.785	.832
BST-DT	-	.861*	.885	.956*	**.977**	**.958**	**.952**	.596	.598	.045	.758	.795
DT	I	.631	.787	.848	.864	.778	.807	.617	.627	.645	.734	.814
DT	P	.611	.771	.856	.871	.789	.808	.586	.625	.688	.734	.794
DT	-	.612	.789	.856	.871	.789	.808	.583	.638	.512	.717	.782
LOGREG	-	.602	.623	.829	.849	.732	.714	.614	.620	.678	.696	.704
LOGREG	I	.590	.640	.827	.848	.721	.726	.607	.594	.636	.688	.699
NB	I	.537	.616	.786	.830	.721	.731	.582	.576	.635	.668	.682
BST-STMP	-	.701	.782	.898	.926	.871	.854	.355	.339	.123	.650	.676
SVM	-	.810	.891	.928	.961	.931	.929	.484	.447	-.546	.648	.712
LOGREG	P	.597	.606	.832	.855	.733	.717	.584	.591	.302	.646	.653
NB	P	.536	.615	.786	.833	.733	.730	.539	.565	.161	.611	.625
NB	-	.414	.637	.746	.767	.698	.689	.271	-.980	-.918	.258	.293

cases to obtain five trials. For each trial we use 4000 cases to train the different models, 1000 cases to calibrate the models and select the best parameters, and then report performance on the large final test set. We would like to run more trials, but this is a *very* expensive set of experiments. Fortunately, even with only five trials we are able to discern interesting differences between methods.

Table 2 shows the normalized score for each algorithm on each of the nine metrics. For each problem and metric we find the best parameter settings for each algorithm using the 1k validation sets set aside by cross-validation, then report that model's normalized score on the final test set. Each entry in the table averages these scores across the five trials and eight test problems. The second column tells if model predictions were calibrated after training. A "-" means

the model predictions were not calibrated – they are the raw model predictions. (The one exception is SVMs, where we linearly scale distances to the separating hyperplane to the range [0,1] before computing the three probability metrics.) A "P" or "I" in the second column indicates that the model predictions were scaled after the model was trained using Platt Scaling (PLT) or Isotonic Regression (ISO), respectively. These scaling methods were discussed in Section 2.4. In the table, higher scores always indicate better performance.

The second to last column, MEAN, is the mean normalized score over the nine metrics, eight problems, and five trials. The models in the table are sorted by the mean normalized score in this column. For now, ignore the last column, OPT. This column will be discussed later in this section.

In the table, the algorithm with the best performance on each metric is **boldfaced**. Other algorithm's whose performance is statistically indistinguishable from the best algorithm at $p = 0.05$ using paired t-tests on the 5 trials are *****'ed.[2] Entries in the table that are neither bold nor starred indicate performance that is significantly lower than the best models at $p = 0.05$.[3]

Averaging across all nine metrics, the strongest models in the table are calibrated boosted trees, calibrated random forests, uncalibrated neural nets, PLT-calibrated SVMs, and calibrated or uncalibrated bagged trees. If calibration is not used, the best models overall are neural nets, bagged trees, and random forests. With or without calibration, the poorest performing models are naive bayes, logistic regression, and decision trees. Memory-based methods (e.g. KNN) are remarkably unaffected by calibration, but exhibit mediocre overall performance. Boosted stumps, even after calibration, also have mediocre performance, and do not perform nearly as well as boosted full trees.

Looking at individual metrics, we see that boosted trees, which have poor squared error, cross-entropy, and probability calibration prior to calibration, dominate the other algorithms on these metrics after calibration. Other methods that also predict good probabilities are calibrated random forests and uncalibrated neural nets. Interestingly, calibrating neural nets with either PLT or ISO hurts their calibration. If neural nets are trained well to begin with it is better not to adjust their predictions.

[2] Performing this many independent t-tests, each at $p = 0.05$, is problematic. Some differences that are labeled significant in the table probably are not truly significant at $p = 0.05$. We considered applying a more stringent experiment-wise p-value that takes into account the number of tests performed, but the strong correlations between performances on different metrics, and on calibrated and uncalibrated models, makes this problematic as well, so we decided to keep it simple. Most of the differences in the table are significant well beyond $p = 0.05$. Doing the t-tests at $p = 0.01$ adds few additional stars to table.

[3] Note that it is possible for the difference between the scores 0.90 and 0.89 to be statistically significant, and yet for the same 0.90 score to be statistically indistinguishable from a poorer score of 0.88 if the variance of the 0.88 score is higher than the variance of the 0.89 score.

Boosted trees also have top performance on the ordering metrics: area under the ROC, average precision, and the precision/recall break-even point.[4] Random forests have virtually the same performance as boosted trees on these metrics. The neural nets and SVMs also order cases extremely well.

On metrics that compare predictions to a threshold, accuracy, F-score, and Lift, the best models are calibrated random forests, followed by calibrated boosted trees, and neural nets. We do not yet understand why ISO-calibration improves neural net F-Score – on all other metrics calibration hurts neural net performance. If one does not need to treat model predictions as probabilities, uncalibrated boosted trees and random forests have excellent performance on both the threshold and ordering metrics.

The last column, OPT, is the mean normalized score for the nine metrics when model selection is done by cheating by looking at the final test sets. The means in this column represent the best performance that could be achieved with each learning method if model selection were done optimally. We present these numbers because parameter optimization is more critical (and more difficult) with some algorithms than with others. For example, bagging works well with most decision tree types and requires little tuning, but neural nets and SVMs require careful parameter selection. As expected, the mean normalized scores in the "cheating" column (OPT) tend to be higher than the mean normalized scores when selection is done using 1k validation sets because model selection using the validation sets does not always select the model with the best performance on the final test set.

Comparing the MEAN and OPT columns, selection using 1k validation sets yields on average about 0.03 decrease in normalized score compared to optimal selection. As expected, high variance models have the biggest drop in performance when selection is sub-optimal. For some of these models the loss is enough that it affects their position in the table. For example, ANNs have enough variance that model selection using 1k validation sets clearly is sub-optimal, causing ANNs to drop several positions in the table when selection is done this way. Random Forests, however, as expected have small variance, and thus lose very little performance when selection is done using 1k validation sets. SVM's have variance between RF and ANNs, and thus lose more than RF, but less than ANN. Boosted trees also have relatively high variance, but their overall performance after PLT or ISO calibration is so strong that they remain the best model overall even when selection is done using 1-k validation sets.

4 Performances by Problem

Table 3 shows the normalized score for each algorithm on each of the eight test problems. Each entry is an average over the nine performance metrics and five trials when selection is done using 1k validation sets.

[4] PLT calibration does not change the ordering predicted by models, so it does not affect these metrics. ISO calibration, however, can introduce ties in the predicted values that may affect performance on the ordering metrics.

As the No Free Lunch Theorem suggests, there is no universally best learning algorithm. Even the best models (calibrated boosted trees, calibrated random forests, and uncalibrated neural nets) perform poorly on some problems, and models that have poor average performance perform well on a few problems or metrics. For example, the best models on the ADULT problem are bagged trees and calibrated boosted stumps. Boosted trees and random forests perform much worse. Bagged trees also perform very well on MG and SLAC. On MEDIS, the best model is logistic regression. The only models that never exhibit excellent performance on any of these problems are naive bayes, un-bagged un-boosted decision trees, and memory-based learning. Surprisingly, although SVMs have excellent overall performance, they never yield the best performance on any problem or metric. Currently we are adding a few text problems to the test suite to see if SVMs will perform best on data sets with very high dimension.

Boosting full decision trees yields better performance than boosting stumps on five of the eight problems. Occasionally boosted stumps perform very well, but when they do not, they can perform so badly that their average performance is poor. On ADULT, when boosting decision trees, the first iteration of boosting hurts the performance of *all* tree types, and never recovers in subsequent rounds. When this happens even single decision trees outperform their boosted counterparts. Bagged trees, however, consistently outperform un-bagged trees on all eight problems. Bagging is "safer" than boosting, even on the six metrics for which boosting yields the best overall performance. Interestingly, although neural nets rarely are the top performers in Table 2 or Table 3, they never perform poorly on any problem or metric, and thus have excellent average performance across all of the problems and metrics. Given that neural nets also don't need to be calibrated, they are remarkably robust models.

5 Relative Computational Cost

With neural nets there are many parameters one could adjust: net size, architecture, backprop method, update interval, learning rate, momentum, etc. Because each parameter can affect performance, both singly and in combinations, many different nets must be trained to adequately explore the parameter space. As expected, ANNs were one of the most expensive algorithms we trained.

SVMs also require adjusting many parameters. The SVM parameters that have a big impact on the performance include the kernel, kernel parameters, and regularization parameter. While most SVMs were fast to train, a few of them took much longer. It is the cost of these few SVMs that makes SVMs expensive.

Although we experimented with a variety of MBL methods, distance measures, and control parameters, MBL proved less expensive than neural nets and SVMs because our training sets contain only 4k points. On larger train sets MBL would be slower. As expected, LOGREG and NB were very fast to train.

With boosted trees and stumps the parameters to set are the base tree/stump type, and how many iterations to boost. What makes boosting expensive in practice is that it is not easy to parallelize. Training one tree or stump is fast, but

Table 3. Normalized scores of each learning algorithm by problem (avg. over 9 metrics)

MODEL	C	COVT	ADULT	LTR.P1	LTR.P2	MEDIS	SLAC	HS	MG	MEAN
BST-DT	P	**.971**	.875	**.965**	**.982**	.765	.882	**.965**	.908	**.914**
BST-DT	I	.954*	.871	.910*	.975*	.727	.903	.949	.892	.897
RF	P	.919	.840	.914	.948	.816*	.915*	.924	.897	.897
RF	I	.912	.882	.878	.944	.787	**.947**	.911	.899	.895
ANN	-	.808	.908	.919	.901	.826*	.910	.947	.921	.893
SVM	P	.805	.898	.941	.957	.774	.890	.937*	.869	.884
BAG-DT	I	.900	.936*	.847	.900	.725	.920	.877	.938*	.880
BAG-DT	-	.918	**.948**	.877	.804	.765	.934*	.829	**.949**	.878
RF	-	.895	.918	.902	.827	.794	.926*	.831	.909	.875
BAG-DT	P	.904	.902	.863	.899	.736	.897	.879	.917	.875
ANN	I	.806	.890	.837	.890	.805*	.921	.941*	.890	.873
SVM	I	.793	.888	.911*	.957	.725*	.906	.925	.868	.872
ANN	P	.799	.868	.895	.853	.802*	.893	.928	.890	.866
KNN	P	.847	.809	.918	.924	.679	.808	.815	.882	.835
KNN	I	.847	.802	.889	.931	.681	.819	.820	.879	.834
KNN	-	.844	.799	.911	.925	.612	.838	.820	.896	.831
BST-STMP	P	.711	.943*	.775	.677	.759*	.813	.822	.909	.801
BST-STMP	I	.690	.945*	.705	.658	.726	.825	.825	.909	.785
BST-DT	-	.857	.768	.850	.850	.401	.830	.825	.688	.758
DT	I	.725	.882	.731	.748	.474	.802	.657	.851	.734
DT	P	.729	.880	.724	.755	.458	.804	.646	.874	.734
DT	-	.682	.893	.708	.729	.467	.792	.609	.858	.717
LOGREG	-	.701	.855	.203	.448	**.837**	.885	.720	.916	.696
LOGREG	I	.679	.891	.254	.433	.800	.857	.695	.892	.688
NB	I	.646	.922	.678	.534	.744	.757	.328	.736	.668
BST-STMP	-	.669	.838	.334	.569	.580	.766	.642	.803	.650
SVM	-	.615	.684	.495	.769	.552	.849	.555	.668	.648
LOGREG	P	.667	.849	.133	.425	.725	.854	.678	.840	.646
NB	P	.605	.801	.632	.457	.671	.734	.309	.679	.611
NB	-	.518	.617	.462	.398	-.180	.662	-1.03	.618	.258

training thousands takes time. Like boosting, bagging requires varying the base tree type. Unlike boosting, however, bagging is easy to parallelize and usually only 100 iterations are needed. This makes bagged trees cheap, and rather attractive given their all around good performance. Random forests are less expensive than boosted trees, but more expensive than bagged trees because random forests require more trees than bagging before performance converges. Also the fraction of features used in each tree in a random forest is a tunable parameter.

6 Related Work

STATLOG is perhaps the best known study [1]. STATLOG was a very comprehensive study when it was performed, but since then important new learning

algorithms have been introduced such as bagging, boosting, SVMs, and random forests. LeCun et al.[13] present a study that compares several learning algorithms (including SVMs) on a handwriting recognition problem using three performance criteria: accuracy, rejection rate, and computational cost. Cooper et al. [14] present results from a study that evaluates nearly a dozen learning methods on a real medical data set using both accuracy and an ROC-like metric. Lim et al. [15] perform an empirical comparison of decision trees and other classification methods using accuracy as the main criterion. Bauer and Kohavi [16] present an impressive empirical analysis of ensemble methods such as bagging and boosting. Perlich et al. [17] conducts an empirical comparison between decision trees and logistic regression. Provost and Domingos [3] examine the issue of predicting probabilities with decision trees, including smoothed and bagged trees. Provost and Fawcett [5] discuss the importance of evaluating learning algorithms on metrics other than accuracy such as ROC.

7 Conclusions

This paper presents, to the best of our knowledge, the first comprehensive empirical evaluation of the performance of learning algorithms on a large variety of performance metrics. An interesting result is that good performance on threshold or ordering metrics does not guarantee good performance on probability metrics. The most important example of this are uncalibrated boosted decision trees, which have the best performance on the ordering metrics, but below average performance on the probability metrics.

Calibration with either Platt's method or Isotonic Regression is remarkably effective at obtaining excellent performance on the probability metrics from learning algorithms that performed well on the ordering metrics. Calibration dramatically improves the performance of boosted trees, SVMs, boosted stumps, and Naive Bayes, and provides a small, but noticeable improvement for random forests. Neural nets, bagged trees, memory based methods, and logistic regression are not improved by calibration.

With excellent performance on all nine metrics, calibrated boosted decision trees were the best learning algorithm overall. They are followed by calibrated random forests, uncalibrated neural nets, calibrated SVMs, and bagged trees. The models that performed the poorest were naive bayes, logistic regression, decision trees, and boosted stumps. Although some of the methods clearly perform better or worse than others *on average*, there is significant variability across the problems and metrics. Even the best models sometimes perform poorly, and models with poor average performance occasionally perform exceptionally well.

The learning methods developed in the last decade — boosting, random forests, SVMs, and bagging — have excellent overall performance if their predictions are calibrated after training. Neural nets, however, are still very competitive and have the advantage of not requiring this extra calibration step.

References

1. King, R., Feng, C., Shutherland, A.: Statlog: comparison of classification algorithms on large real-world problems. Applied Artificial Intelligence **9** (1995)
2. Joachims, T.: Making large-scale svm learning practical. In: Advances in Kernel Methods. (1999)
3. Provost, F., Domingos, P.: Tree induction for probability-based rankings. Machine Learning (2003)
4. Giudici, P.: Applied Data Mining. John Wiley and Sons, New York (2003)
5. Provost, F.J., Fawcett, T.: Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. In: Knowledge Discovery and Data Mining. (1997) 43–48
6. Platt, J.: Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. In: Adv. in Large Margin Classifiers. (1999)
7. Zadrozny, B., Elkan, C.: Transforming classifier scores into accurate multiclass probability estimates. In: KDD. (2002)
8. Zadrozny, B., Elkan, C.: Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In: ICML. (2001)
9. Robertson, T., Wright, F., Dykstra, R.: Order Restricted Statistical Inference. John Wiley and Sons, New York (1988)
10. Ayer, M., Brunk, H., Ewing, G., Reid, W., Silverman, E.: An empirical distribution function for sampling with incomplete information. Annals of Mathematical Statistics **5** (1955) 641–647
11. Blake, C., Merz, C.: UCI repository of machine learning databases (1998)
12. Gualtieri, A., Chettri, S.R., Cromp, R., Johnson, L.: Support vector machine classifiers as applied to aviris data. In: Proc. Eighth JPL Airborne Geoscience Workshop. (1999)
13. LeCun, Y., Jackel, L.D., Bottou, L., Brunot, A., Cortes, C., Denker, J.S., Drucker, H., Guyon, I., Muller, U.A., Sackinger, E., Simard, P., Vapnik, V.: Comparison of learning algorithms for handwritten digit recognition. In Fogelman, F., Gallinari, P., eds.: International Conference on Artificial Neural Networks, Paris, EC2 & Cie (1995) 53–60
14. Cooper, G.F., Aliferis, C.F., Ambrosino, R., Aronis, J., Buchanan, B.G., Caruana, R., Fine, M.J., Glymour, C., Gordon, G., Hanusa, B.H., Janosky, J.E., Meek, C., Mitchell, T., Richardson, T., Spirtes, P.: An evaluation of machine learning methods for predicting pneumonia mortality. Artificial Intelligence in Medicine **9** (1997)
15. Lim, T.S., Loh, W.Y., Shih, Y.S.: An empirical comparison of decision trees and other classification methods. Technical Report 979, Madison, WI (1997)
16. Bauer, E., Kohavi, R.: An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. Machine Learning **36** (1999)
17. Perlich, C., Provost, F., Simonoff, J.S.: Tree induction vs. logistic regression: a learning-curve analysis. J. Mach. Learn. Res. **4** (2003) 211–255
18. Vapnik, V.: Statistical Learning Theory. John Wiley and Sons, New York (1998)
19. Schapire, R.: The boosting approach to machine learning: An overview. In: In MSRI Workshop on Nonlinear Estimation and Classification. (2001)
20. Breiman, L.: Bagging predictors. Machine Learning **24** (1996) 123–140