When Efficient Model Averaging Out-Performs Boosting and Bagging

Ian Davidson[1] and Wei Fan[2]

[1] State University of New York, Albany, NY 12222
davidson@cs.albany.edu.
[2] IBM T.J. Watson, 9 Skyline Drive, NY 10532
weifan@us.ibm.com

Abstract. The Bayes optimal classifier (BOC) is an ensemble technique used extensively in the statistics literature. However, compared to other ensemble techniques such as bagging and boosting, BOC is less known and rarely used in data mining. This is partly due to BOC being perceived as being inefficient and because bagging and boosting consistently outperforms a single model, which raises the question: "Do we even need BOC in datamining?". We show that the answer to this question is "yes" by illustrating several recent *efficient* model averaging approximations to BOC *can* significantly outperform bagging and boosting in realistic situations such as extensive class label noise, sample selection bias and many-class problems. That model averaging techniques outperform bagging and boosting in these situations has not been published in the machine learning, mining or statistical communities to our knowledge.

1 Introduction and Motivation

The typical aim of classification data mining is to build the most accurate model. Research activity in the machine learning and data mining fields has produced a large number of ensemble techniques such as boosting and bagging that can combine multiple base models to increase accuracy beyond the single best model. However, the use of the classic Bayesian statistical approach to multiple models, Bayesian model averaging, is rarely reported in data mining and even when it is, yields indifferent results [1,5]. The Bayesian approach of using multiple models is to weight each model's belief in a prediction ($P(y_i|\mathbf{x},\theta)$) by the model's posterior probability ($P(\theta|D)$) and is known as Bayesian model averaging in the statistical literature [11] and the Bayes optimal classifier (BOC) in the machine learning and data mining literature [14]. The techniques we shall use in this paper are approximations to the BOC since we do not average over all models. Given a test set instance, $\mathbf{x}$, to be classified into one of k classes ($y_1 ... y_k$), a training dataset D and model space Θ the BOC chooses the class that maximizes equation (1).

$$argmax_{y_i} : P(y_i) = \int_{\theta \in \Theta} P(y_i|\mathbf{x},\theta) P(\theta|D) d\theta \quad (1)$$

The BOC is claimed to be optimal for a number of reasons under specific conditions: Kohavi claims it reduces bias and variance [12] while Buntine claims

J. Fürnkranz, T. Scheffer, and M. Spiliopoulou (Eds.): PKDD 2006, LNAI 4213, pp. 478–486, 2006.

it prevents overfitting [1] and Domingos [5] states "no other method can consistently achieve lower error rates". In addition there is significant experimental evidence supporting its effectiveness in the statistical literature [11]. Yet there seems to be little interest in using model averaging in data mining. Instead the two most common ensemble model techniques are bagging and boosting [3]. These techniques are rightfully extensively used as they are relatively easy to implement, have been shown to work for a variety of learners and a variety of data sets. In contrast model averaging is typically more difficult to implement, particularly since BOC in its rudimentary form requires performing an integration over the entire model space which is computationally prohibitive for the *complex* model spaces such as trees and large datasets used in the data mining. Compared to model averaging, ensemble techniques such as bagging and boosting **combine** not average and the base unit is a **vote** not a probability. Minka [15] succinctly notes the difference that BOC is a method of "soft model selection" using multiple models when there is *uncertainty* to which model is the best model. For example, if all posterior probabilities are the same then model uncertainty is maximum. In contrast techniques such as bagging and boosting are methods to *combine* multiple models to create a new (and potentially more appropriate) model space than the base model space [15].

2 When Averaging Will Outperform Combining

In this section we compare and contrast model averaging and model combination and conclude the section by describing data set conditions where model averaging should perform better than model combination. In later sections we empirically test these claims. It may appear that model averaging and techniques such as boosting and bagging are similar but how the multiple models are used are quite different. Many ensemble techniques explicitly or implicitly combine multiple models. For example, the serial nature of boosting to focus on misclassified instances *effectively* means that the original tree built from the unweighted training set has subsequent additional trees grafted onto the leaf nodes. In this way the model space boosting is searching is a combination of the base model class (trees). Similarly, it has been noted that bagging models can create a more express model space than the base model space [15]. In contrast model averaging never explicitly combines models. Rather, each model's predictions are weighted by the belief (posterior) that it is the true model. Furthermore, in the presence of excessive number of training instances equation 1 will simply have most of the posterior mass on a single model and perform no better than a single tree. As the statistical literature [11] and Minka [15] note model averaging should perform well when there is substantial model uncertainty where best-model uncertainty can be quantified as being one minus the posterior probability of the most probable model. We note that there are other general measures of model uncertainty such as the entropy in the posterior [2]. Formally:

Definition 1. ***Best-Model Uncertainty*** *is the degree of belief that the most probable model (θ^*) is* ***not*** *the best model in the model class (Θ). That is:* $ModelUncertainty(\Theta, D) = argmin_{\theta \in \Theta}\ (1 - P(\theta|D))$

When model uncertainty exists, it is because there is insufficient data to conclusively state that one model is the best and hence building combinations of models can be perceived as building an overly complex model given the amount of data. We see from definition 1 that model uncertainty is likely to exist if there is no highly probable model. By a simple expansion of $P(\theta|D) = \frac{P(\theta)P(D|\theta)}{P(D)}$ using Bayes theorem we see that this can occur if the numerator, particularly the likelihood is small. In decision tree learning this can occur for a number of reasons. Since each tree path forms a distinct part of a model we can say that $P(D|\theta) = \Pi_{1\ldots n} P(D_i|Leaf(D_i))$ where $Leaf(D_i)$ is a function returning the leaf the i^{th} instance is assigned to. The term $P(D_i|Leaf(D_i)) = \frac{Count(Class(D_i), Leaf(D_i))}{Number(Leaf(D_i))}$ where $Count(Class(D_i), Leaf(D_i))$ returns the number of training set instances at the leaf having the same label as D_i and $Number(Leaf(D_i))$ the total number of instances at the leaf node. The leaf nodes of the tree may be poorly or incorrectly populated for any number of reasons. The (non-exhaustive) set of data set conditions we believe where this will occur and shall explore in this paper are: 1) Training sets with excessive class label errors (i.e. security/fraud applications where labeling is difficult), 2) High dimensional data but a relatively few number of instances (i.e. bioinformatics problems) and 3) Multi-class problems involving a large number of classes (i.e. fault diagnosis).

In addition, even if model uncertainty is not particularly great, it may be worth removing by averaging. In this paper we shall explore a situation discussed in our recent work, sample selection bias [7]. Since the training and test data sets are drawn from different distributions, even a highly probable model for the training set may produce poor predictions on the test set.

3 Efficient Model Averaging Techniques

Random Decision Trees (RDT). RDT were originally proposed in [8]. Rather than using purity functions (i.e. information gain) like traditional decision tree algorithms, RDT chooses a feature/column to split on *randomly.* A discrete feature is chosen only once in each decision path. A continuous feature can be chosen multiple times in the same decision path with a different decision threshold each time. Since both feature and decision thresholds for continuous features are chosen randomly, each RDT is likely to be different. The tree stops growing if either the number of instances at the current node is zero or the depth of the tree exceeds some predefined limit. In our experiments, the depth of the tree is limited to be up to the number of features in order to give each feature equal opportunity to be chosen in any decision path. During classification, each random tree computes a posterior probability at the leaf node. That is, if there are n examples at a leaf node and q_i belong to class label y_i, the posterior probability $P(y_i|\mathbf{x}, \theta) = \frac{q_i}{n}$. For a given test instance, the posterior probability

outputs from multiple decision trees are **averaged** to produce a final probability estimate with the most probable class for an instance being the prediction. As found in [8], typically 30 random trees give satisfactory results and as little as 10 random trees produce results better than using a single traditionally grown decision tree [8]. RDT have been shown to have accuracy comparable to or higher than bagging and random forest but at a fraction of the training cost for a variety of data sets including those with many irrelevant attributes [8,9]. Though with RDT each *tree structure* is created randomly the leaf probabilities are estimated from the training data set. Therefore, in the limit as the number of trees approaches infinity, RDT effectively computes the following: $P(y_i|\mathbf{x}, \Theta) = \sum_{\theta \in \Theta} P(\theta|D).\frac{q_{i,\theta}}{n_\theta}$ RDT though superficially similar to random forest of trees (RFT) are different in a number of ways (see [9] for details). Furthermore, RDT are different to the random trees referred to by Dietterich [3] as in that work the splits are randomly chosen from the twenty most informative splits.

Parametric Bootstrap Model Averaging. The bootstrap model averaging approach [2] is a frequentist approach to model averaging. The philosophy of model averaging is to remove model uncertainty that arises due to **a single finite data set**. The typical view of mining is that single training set of size n is available from the underlying distribution that generated the data F. This masks the underlying uncertainty that the training data is one of many that could have been chosen/generated. Building a model for each possible data set would create a distribution which is the frequentist analog to the posterior distribution but does not use a prior. However, typically we do not have the luxury of many different data sets drawn independently of the same size so we can not compute the uncertainty over the model space from them. To approximate this distribution using a *single data set*, Efron [6] created the non-parametric and parametric bootstrapping approaches. Non-parametric bootstrapping which has been used extensively by the machine learning community is an example of attempting to simulate draws from F when no knowledge of its form is known. Parametric bootstrapping which has received little attention is used when some underlying knowledge on the form of F is known. In this paper we make use of a simple generative parametric model of F which assumes that the independent variables are conditionally independent given the dependent variable value. The parameters for this model are estimated from the data and virtual training examples are drawn/bootstrapped from it to build models. More complex models of F could be used and remains an active research area. Formally the bootstrap model averaging approach is: $P(y_i|\mathbf{x}, \Theta) = \sum_{D', \theta_i = L(D')} P(y_i|\mathbf{x}, \theta_i) P(D'|D)$ In this paper, the learner, L, used with PBMA is J48 (Weka's equivalent to C4.5) to produce trees built with the default parameters.

4 Experiments

We now illustrate the performance of model averaging techniques and other ensemble techniques when considering model uncertainty is beneficial (see section

2) and find that bagging and boosting do not perform well. We use the Weka software to perform bagging and AdaBoost using the J48 decision tree base learner (equivalent to C4.5). This requires changing Weka slightly as the default implementation of bagging in Weka aggregates conditional probabilities and *not* votes [10]. In all experiments, regardless of the ensemble technique (PBMA, Boosting, Bagging), the default parameters were used for J48. We randomly divide the data into 80% for training and the remaining 20% for testing unless otherwise stated.

Model Uncertainty Due to Class Label Noise

If there exists a deterministic relationship between the independent variables and class label, and the model space is suitable, it is likely any reasonable learner will find a highly probable model that can correctly map feature vectors of independent variables to their correct class labels. Hence the model uncertainty would be relatively small. However, the addition of class label noise, i.e., the correct class label being flipped to another class, would almost certainly reduce the probability of the "correct model" otherwise trained from dataset without label noise, hence increasing model uncertainty. We took several UCI data sets and introduced 20% class label noise in the training sets only by randomly flipping 20% of class labels. We then try the model averaging approaches and compare them against the other ensemble techniques. To illustrate the efficiency of PBMA and RDT we use only 30 models but use 100 models for bagging and boosting. Table 1 (left) shows that the model averaging techniques outperform the other ensemble techniques and single models in each situation. A pair-wise student t-test using the same training and test divisions, shows that the poorer performing model average technique, PBMA, outperforms both bagging and boosting at the 99% confidence interval for all data sets. As expected boosting performs quite poorly as it apparently continues to build models to fit the added noise while bagging can only occasionally significantly outperform a single model.

Biased Training Sets

Recent work by Zadrozny and ourselves [17,7] has classified many learners including decision trees and naive Bayes as global learners that can be affected by sample bias. In particular the sample bias investigated is that the probability of the instance being selected in the training data (denoted by the event, $s = 1$) is conditionally independent of the instance label (y) given the instance's description (x), formally: $P(s = 1|x, y) = P(s = 1|x)$. This type of sample bias is effectively when instances are not chosen at random to be in the training set but instead depending on their description but not their label. The test set is drawn randomly. This sample bias occurs prevalently in applications where the occurrence of particular instances' change but not the concept (i.e. relationship between x and y). For the rest of this paper when we refer to **sample bias**, we refer to this type of bias. Decision trees are known to be unstable, and is categorized as "global" classifier in [7]. The structure of decision tree

is sensitive to sample selection bias, which makes it unlikely to find that most probable decision tree otherwise trained from dataset without sample selection bias.

Artificial Sample Selection Bias. We can artificially introduce sample bias into the previously mentioned UCI data sets by first dividing the data into training and test sets. We then sort only the training set on the attributes and remove the first 10% of all instances. Note this introduces a training sample bias but does not change the relationship between the independent variables and class-labels as our previous experiments did. Table 1 (right) shows that model averaging performs better than other ensemble techniques even though only 30 models are used and 100 models are used for bagging and boosting. Unlike the previous situation, boosting performs better than bagging on average **but both perform worse than a single tree**.

Sample Selection Bias in Newsgroup Classification. We now focus on the newsgroup data where the training and test data sets are drawn from similar but not identical distributions. We perform experiments on the 20 Newsgroups [16] datasets using the standard bydate version division into training (60%) and test (40%) based on posting date. The division creates a temporal bias. For example, in the GUNS newsgroup the word "Waco" occurs extensively in news articles in the training set but not in the test set as interest in the topic fades. However, since the proportion of each class label is the **same** in the training and test data sets there is no class label bias. We used the tool Rainbow [13] to extract features from these news articles. The feature vector for a document consists of the frequencies of top ten words by selecting words with highest average mutual information with the class variable. To better understand the performance of model averaging we treat the problem as binary classification between a variety of newsgroups. Table 2 illustrates the improvement that model averaging provides. Again, the better performance is obtained using only 30 models for the model averaging approaches as opposed to 100 models for bagging and boosting. In this situation boosting performs better than bagging but the model averaging approaches outperform both and the single best model.

Table 1. Average error of various techniques on standard UCI data sets. Left: With class label noise in training set only over one hundred repetitions. Right: With biased training sets over one hundred repetitions. UP=Unpruned, P=Pruned.

	Breast	**Vote**	**Pima**	**Credit**	**Wine**	**Breast**	**Vote**	**Pima**	**Credit**	**Wine**
UP. Tree	10.6	5.2	31.8	36.5	37.6	1.5	3.7	26.5	3.1	35.3
P. Tree	4.5	12.6	31.8	37.4	37.0	2.1	3.7	27.6	12.2	35.3
RDT	**0.5**	3.7	**30.0**	**30.6**	**26.4**	**0.5**	**2.7**	26.1	**1.0**	**27.8**
PBMA	2.5	**2.2**	30.9	30.7	26.7	1.0	3.1	**25.7**	1.3	28.0
Bagging	4.0	4.2	33.9	34.3	34.3	4.04	3.7	28.0	5.61	38.2
Boosting	15.7	19.3	34.0	42.4	38.2	2.5	4.4	28.4	5.02	38.1

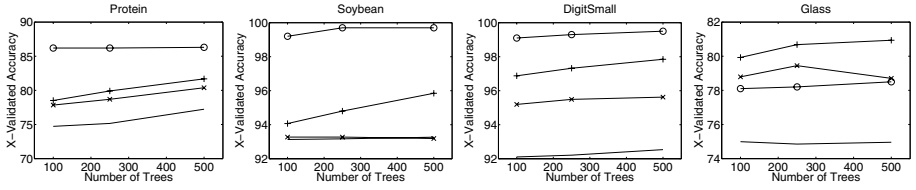

Fig. 1. Accuracy versus No. trees: Bagging(-), Boosting(-x-), RDT(-o-) & PBMA(-+-)

Table 2. Error of various techniques on newsgroup data. Training set is first 60% of the year's postings and the test set the last 40%.

	BaseBall Hockey	**Christian Sale**	**M.Cycle Guns**	**MidEast Guns**	**MidEast Electrical**	**Mac. Religion**
Unpruned Tree	15.7	7.9	10.5	20.3	14.4	18.4
Pruned Tree	15.7	7.4	10.2	20.3	14.4	18.7
RDT	**11.9**	6.6	**8.5**	10.5	**7.2**	12.7
PBMA	12.0	**6.0**	8.6	**9.8**	7.4	**11.9**
Bagging	14.8	7.9	10.5	20.3	14.4	18.4
Boosting	12.6	7.7	9.2	10.7	11.7	13.2

Model Uncertainty Due to Large Numbers of Classes
We examine four data sets where the number of classes is large: Digit (10), Glass (6), Protein (6), Soybean (19). Furthermore, the number of columns is relatively high and instances relatively small. For each data set the number of columns and instances is: Digit (17, 469), Glass (10, 214), Protein (21, 116) and Soybean (36, 683) respectively. Assuming a uniform distribution of classes amongst the instances there are as little as 19 instances per class (not leaf) and hence model uncertainty is likely to be high. We calculated the ten-fold cross-validated accuracy ten times for bagging, boosting, RDT and PBMA on these four data sets building, 100, 250 and 500 trees. We found that the poorer performing model average technique with respect to performance over all data sets, PBMA, outperforms both bagging and boosting for all data sets at the 99% confidence interval. When comparing the model combination approaches bagging, boosting to the model averaging approaches RDT and PBMA we find that both averaging approaches work significantly better except for the Glass data set when only PBMA outperform both combination techniques (see Figure 1).

5 Conclusion and Future Work

Model averaging is a method of choice for statisticians to make use of multiple models. However, model averaging is rarely used in data mining and instead other ensemble techniques such as boosting and bagging are used extensively. We believe this is for two primary reasons: a) Model averaging is perceived as

being computationally inefficient and b) boosting and bagging have been shown to work for a variety of problems. A valid question is then: "Does the data mining practioner require model averaging as an ensemble technique?". The answer to this question is yes, there are situations where model averaging can significantly outperform both bagging and boosting using two recent efficient approaches to model averaging. This is because a) boosting and bagging are model *combination* not averaging approaches and b) model averaging works best when there is model uncertainty. The situations where averaging to remove model uncertainty is beneficial we explored in this paper include when the number of classes is large, excessive class label noise and training sample bias occurs. We believe that these are not academic situations but will occur in fields such as bioinformatics and security and demonstrated that one of the situations (training sample bias) occurs in the newsgroup data sets. In this work we identified situations where model averaging performs well, but we empirically saw that boosting and bagging performed badly. For some of these situations (i.e. class noise) and techniques (boosting) it is well known why the technique performs badly, it would be interesting to explore why bagging and boosting failed in these other situations.

Acknowledgements. We thank the reviewers for their helpful comments.

References

1. Buntine W. (1990). *A Theory of Learning Classification Rules*, Ph.D. Thesis, UTS.
2. Davidson I. (2004). An Ensemble Technique for Stable Learners with Performance Bounds, *AAAI 2004*.
3. Dietterich, T. G. (2000). An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 40 (2).
4. Domingos, P., (1997). Why Does Bagging Work? A Bayesian Account and its Implications. *KDD 1997*.
5. Domingos P. (2000). Bayesian Averaging of Classifiers and the Overfitting Problem, *AAAI 2000*
6. Efron, B. (1982). The jackknife, the bootstrap, and other resampling plans. SIAM Monograph 38.
7. Fan W., Davidson I., Zadrozny B. and Yu P., (2005) An Improved Categorization of Classifier's Sensitivity on Sample Selection Bias, *ICDM 2005*.
8. Fan W., Wang H., Yu P.S, Ma S., (2003). Is random model better? On its accuracy and Efficiency, *ICDM 2003*.
9. Fei Tony Liu, Kai Ming Ting, Wei Fan, (2005). Maximizing Tree Diversity by Building Complete-Random Decision Trees. *PAKDD 2005*.
10. Frank, Eibe, *Personal Communication*, 2004.
11. Hoeting J., Madigan D., Raftery A., and Volinsky C., (1999). "Bayesian Model Averaging: A Tutorial", *Statistical Science 14*.
12. Kohavi R., Wolpert D., (1996). Bias Plus Variance Decomposition for 0-1 Loss Functions, *ICML 1996*.
13. McCallum, A. (1996). Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. http://www.cs.cmu.edu/~mccallum/bow.

14. Mitchell T., (1997). *Machine Learning*, McGraw-Hill.
15. Minka, T.P., Bayesian model averaging is not model combination, MIT Media Lab note (7/6/00), http://research.microsoft.com/~minka/papers/bma.html
16. Rennie, J. (2003) *20 Newsgroups*. Technical Report, Dept C.S., MIT.
17. Zadrozny B., (2004). Learning and evaluating classifiers under sample selection bias, ICML 2004.