Prepublication draft of paper published in Machine Learning, 40, 159–39 (2000)

MultiBoosting: A Technique for Combining Boosting and Wagging

GEOFFREY I. WEBB webb@deakin.edu.au
School of Computing and Mathematics
Deakin University, Geelong, Vic, 3217, Australia.

Editor: Robert Schapire

Abstract. MultiBoosting is an extension to the highly successful AdaBoost technique for forming decision committees. MultiBoosting can be viewed as combining AdaBoost with wagging. It is able to harness both AdaBoost's high bias and variance reduction with wagging's superior variance reduction. Using C4.5 as the base learning algorithm, Multi-boosting is demonstrated to produce decision committees with lower error than either AdaBoost or wagging significantly more often than the reverse over a large representative cross-section of UCI data sets. It offers the further advantage over AdaBoost of suiting parallel execution.

Keywords: Boosting, bagging, wagging, aggregation, decision committee, decision tree

1. Introduction

Decision committee learning has demonstrated spectacular success in reducing classification error from learned classifiers. These techniques develop a classifier in the form of a committee of subsidiary classifiers. The committee members are applied to a classification task and their individual outputs combined to create a single classification from the committee as a whole. This combination of outputs is often performed by majority vote. Examples of these techniques include classification ensembles formed by stochastic search (Ali, Brunk, & Pazzani, 1994), bagging (Breiman, 1996a), AdaBoost (Freund & Schapire, 1997), Nock and Gascuel's (1995) decision committees, averaged decision trees (Oliver & Hand, 1995), and stacked generalization (Wolpert, 1992).

Two decision committee learning approaches, AdaBoost and bagging, have received extensive attention. There is a large body of evidence that AdaBoost, applied to standard decision tree induction systems, provides a powerful off-the-shelf general-purpose learning algorithm. Indeed, to date there is good reason to identify AdaBoost with standard decision tree learners as the off-the-shelf method-of-choice for a learning task where there is no prior opportunity to evaluate the relative effectiveness of alternative approaches, there is no a priori knowledge available about the domain, and the primary goal of learning is to develop a classifier with the lowest possible error (see, for example Bauer & Kohavi, in press; Breiman, 1996b; Freund & Schapire, 1997, 1996; Quinlan, 1996; Schapire, Freund, Bartlett, & Lee, 1997).

Both AdaBoost and bagging are generic techniques that can be employed with any base classification technique. They operate by selectively resampling from the

training data to generate derived training sets to which the base learner is applied. Bagging, has also demonstrated remarkable consistency in its ability to reduce error (Breiman, 1996a; Quinlan, 1996).

A number of studies that compare AdaBoost and bagging (Bauer & Kohavi, in press; Breiman, 1996b; Quinlan, 1996) suggest that AdaBoost and bagging have quite different operational profiles. In general, it appears that bagging is more consistent, increasing the error of the base learner less frequently than does AdaBoost. However, AdaBoost appears to have greater average effect, leading to substantially larger error reductions than bagging on average. AdaBoost tends to reduce both the bias and variance terms of error while bagging tends to reduce the variance term only (Bauer & Kohavi, in press; Breiman, 1996b). Another notable feature of both AdaBoost and bagging is that, on average, error keeps reducing as committee size is increased, but that the marginal error reduction associated with each additional committee member tends to decrease. Each additional member, on average, has less impact on a committee's prediction error than any one of its predecessors (Schapire et al., 1997). If we assume that both these algorithms are effective through different mechanisms, it is plausible that a combination of the two will produce even greater effect. It has been noted that increasing the disagreement in the predictions of committee members without affecting their individual error rates can increase the accuracy of an ensemble[1] (Krogh & Vedelsby, 1995). Using multiple approaches to generating committee members should increase diversity in the committee membership which might be expected to increase disagreement between predictions. If this can be done without greatly increasing the error in the individual predictions it can be expected to decrease error in the resulting committee's predictions.

As the earlier members of each type of committee have greatest effect, there may be value in sacrificing the latter members of one type of committee for members with the effectiveness of the initial members of the other type. MultiBoosting is a decision committee technique that combines AdaBoost with wagging, a variant of bagging that is better suited to the task than direct bagging. Experimental results suggest that it is in general more effective at error reduction than either of its constituent methods.

This paper describes AdaBoost, bagging, wagging, and the new MultiBoost algorithm. The effects of these four algorithms on the error, bias and variance of learned classifiers are explored in a number of experiments. These experiments confirm that AdaBoost reduces both bias and variance while bagging and wagging have little effect on bias and greater effect on variance. MultiBoost is shown to achieve most of AdaBoost's superior bias reduction coupled with most of bagging's superior variance reduction. This results, in general, in lower prediction error than any of AdaBoost, bagging, or wagging when applied to the C4.5 learning algorithm on a wide cross-section of typical machine learning tasks.

2. AdaBoost and bagging

Before describing the new MultiBoost algorithm, it is desirable to outline its antecedent algorithms, AdaBoost, bagging, and wagging. All of these algorithms

use a base learning algorithm that forms a single classifier from a training set of examples. This base classifier is provided with a sequence of training sets that the committee learning algorithm synthesizes from the original training set. The resulting classifiers become the constituent members of the decision committee.

Bagging (Breiman, 1996a) takes as input a base classification learning algorithm $\mathcal{L}$ and training set T, and returns a committee of classifiers C^*. T is a vector of n class–description pairs. Each pair (y_i, x_i) associates class $y_i \in Y$ with description $x_i \in X$. $\mathcal{L}(T) \rightarrow (C(X) \rightarrow Y)$ is a function from training sets to classifiers, which are in turn functions from descriptions to classes. A committee C^* of t classifiers is formed by applying $\mathcal{L}$ to each element of a vector of derived training sets, $T'_1, T'_2 \ldots T'_t$. Each resulting classifier C_i is added to C^*. Each T'_i is a bootstrap sample from T. For a training set T containing n cases, a bootstrap sample is formed by uniform probability random selection from T with replacement, n times. This will create a training set with the same number of cases as the original, but some cases from the original may be represented more than once while others may not be represented at all. The expected frequency with which cases from T are represented in a single bootstrap sample T' is described by the discrete Poisson distribution.

To classify a new case with description x, each classifier C_i from C^* is applied to x: resulting in classifications $C_1(x), C_2(x) ... C_t(x)$, where each classifier returns a classification $y_i \in Y$. The committee's classification $C^*(x)$ is the class that obtains the most votes from the committee members when applied to x

$$C^*(x) = \underset{y \in Y}{argmax} \sum_{i=1}^{t} 1(C_i(x) = y) \tag{1}$$

where $1(\cdot)$ is the indicator function.

Wagging (Bauer & Kohavi, in press) is variant of bagging, that requires a base learning algorithm that can utilize training cases with differing weights. Rather than using random bootstrap samples to form the successive training sets, wagging assigns random weights to the cases in each training set. Bauer and Kohavi's (in press) original formulation of wagging used Gaussian noise to vary the instance weights. However, this can lead to some instance weights being reduced to zero, effectively removing them from the training set. Instead, following a suggestion from J. R. Quinlan (personal communication, May 1998) the new technique uses the continuous Poisson distribution[2] to assign random instance weights. As the assignment of instance weights by bagging can be modeled by the discrete Poisson distribution, use of the continuous Poisson distribution can be viewed as assigning instance weights using an equivalent distribution to bagging, but over a continuous rather than discrete space.

Individual random instance weights (approximately) conforming to the continuous Poisson distribution are calculated by the following formula:

$$Poisson() = -log\left(\frac{Random(1 \ldots 999)}{1000}\right) \tag{2}$$

where $Random(min \ldots max)$ returns a random integer value between min and max inclusive. After values are assigned using this random process, the vector of

weights is always standardized to sum to n, for consistency with the implementation of AdaBoost that is used.

AdaBoost, like bagging, forms a committee of classifiers by applying a single base learning algorithm to successive derived training sets formed by sampling from the base training set. However, AdaBoost differs from bagging by associating a weight with each example in the training set. When forming the first derived training set T_1, each weight is initialized to $\frac{1}{n}$. In general, the weight $w_{i,j}$ for the i^{th} item on the j^{th} training set, T_j is a function of the performance of previous classifiers $(C_1 \ldots C_{j-1})$ on the previous derived training sets $(T_1 \ldots T_{j-1})$. The AdaBoost algorithm may use this weight during sampling, forming a bootstrap sample from a distribution over the training data defined by the weights. Alternatively, all training examples may be used to learn every committee member, with the weights passed into the base learning algorithm, causing it to pay differential attention to different objects. Quinlan (1996) obtained better results using this reweighting technique than Freund and Schapire (1996) obtained using resampling in another set of experiments, suggesting that reweighting is more effective than resampling. In consequence, reweighting has been used in the current research.

A variant of Freund and Schapire's (1997, 1996) AdaBoost algorithm is presented in Table 1. The algorithm presented is Bauer and Kohavi's (in press) variant of the original AdaBoost.M1. This variant is chosen because—

- It uses a one step weight update process that is less subject to numeric underflow than the original two step process (step 14).
- It prevents numeric underflow (step 15).
- It continues producing more committee members beyond the point where $\epsilon \geq 0.5$ (step 5). This measure is claimed to improve predictive accuracy (Breiman, 1996b; Bauer & Kohavi, in press).

Note, however, a minor change. Whereas Bauer and Kohavi (in press) halt induction and treat C_t as having infinite weight if $\epsilon_t = 0$, the current algorithm sets β_t to a very small value (10^{-10}) in this circumstance and resamples from the training data to restart the boosting process. This allows the possibility that more than one committee member with zero error will be derived, enabling these to vote against each other and giving other committee members a casting vote in case of a draw between committee members with zero resubstitution error. This modification is made to ensure that in the experiments that are performed, the new MultiBoost algorithm does not obtain an advantage through always employing its full quota of constituent classifiers while AdaBoost occasionally forgoes some members due to reaching zero resubstitution error. On most of the data sets used in this research this modification to the algorithm does not affect the predictive accuracy of the resulting committee. For the few data sets on which this does result in a difference in accuracy, the change improves performance.

Note also, whereas the instance weights in AdaBoost are normally treated as a distribution, summing to 1, for use with the base learning algorithm, C4.5, in this implementation the weights sum to n.

Table 1. The AdaBoost algorithm

AdaBoost
inputs:

S, a sequence of m examples $\langle (x_1, y_1), \ldots, (x_m, y_m) \rangle$ with labels $y_i \in Y = \{1, ..., k\}$.
base learning algorithm $BaseLearn$.
integer T specifying the number of iterations.

1. $S' = S$ with all instance weights set to 1.
2. For $t = 1$ to T {
3. $C_t = BaseLearn(S')$.
4. $\epsilon_t = \dfrac{\sum_{x_j \in S': C_t(x_j) \neq y_j} weight(x_j)}{m}$ *[the weighted error on the training set]*
5. if $\epsilon_t > 0.5$ then
6. set S' to a bootstrap sample from S with weight 1 for each instance
7. goto step 3.
8. if $\epsilon_t = 0$ then
9. set β_t to 10^{-10}
10. set S' to a bootstrap sample from S with weight 1 for each instance.
11. otherwise,
12. $\beta_t = \dfrac{\epsilon_t}{(1 - \epsilon_t)}$.
13. For each $x_j \in S'$,
14. divide $weight(x_j)$ by $2\epsilon_t$ if $C_t(x_j) \neq y_j$ and $2(1 - \epsilon_t)$ otherwise.
15. if $weight(x_j) < 10^{-8}$, set $weight(x_j)$ to 10^{-8}.
16. }

Output the final classifier:

$$C^*(x) = \underset{y \in Y}{argmax} \sum_{t: C_t(x) = y} log \frac{1}{\beta_t}.$$

3. Bias and variance

A number of recent investigations of decision committees have analyzed error performance in terms of *bias* and *variance*. The decomposition of a learner's error into bias and variance terms originates from analyses of learning models with numeric outputs (Geman, Bienenstock, & Doursat, 1992). The squared *bias* is a measure of the contribution to error of the central tendency or most frequent classification of the learner when trained on different training data. The *variance* is a measure of the contribution to error of deviations from the central tendency. Bias and variance are evaluated with respect to a distribution of training sets $\mathcal{T}$, such as a distribution containing all possible training sets of a specified size for a specified domain.

The bias/variance analysis is useful in focusing attention on two significant factors that govern the accuracy of classifiers learned by a learning system. If a learning

system when provided different training data develops classifiers that differ in their predictions, then the extent of such variations provides a lower limit on the average error of those classifiers when applied to any subsequent set of test data. If there is only one correct answer and the answers provided by different classifiers differ then not all can be correct!

However, preventing such variance between the classifiers will not guarantee the elimination of prediction error. This error is also governed by both the degree to which the correct answer for an object can differ from that for other objects with identical descriptions (irreducible error) and the accuracy of the learning bias—if the predictions of all classifiers are identical but wrong, then error will also result.

Unfortunately, however, the definitions of these terms that are appropriate for numeric regression, where a prediction is not just either right or wrong but there are varying degrees of error, do not readily translate in an appropriate manner to contexts where the value to be predicted is categorical. As a result, a number of different formulations of bias and variance have been proposed in the field of classification learning (Breiman, 1996b; Kohavi & Wolpert, 1996; Kong & Dietterich, 1995; Friedman, 1997). Each of these definitions is able to offer valuable insight into different aspects of a learner's performance. Five bias/variance related metrics are used in this research. Rather than distracting from the central theme of this paper by presenting the detail of these metrics here, the formal definitions and details of their estimation have been placed in the Appendices. Brief informal descriptions, are provided below. Appendix A provides detailed definitions of these metrics and discusses how they relate to other bias/variance formulations. Appendix B details how they are estimated in the current research.

Values for the metrics are estimated by ten runs of three-fold cross-validation. In each three-fold cross-validation, the data are randomly divided into three sub-sets. Each case in each subset is classified once by a classifier learned by the algorithm under evaluation from the other two subsets. Thus, each case is classified once for each three-fold cross-validation, and hence ten times in all, each time by a classifier formed by the learner from a different random sample of data. Table 2 presents hypothetical classifications for three cases across ten such three-fold cross-validation trials by the classifiers learned by a hypothetical learner $\mathcal{L}$. For each case, the resulting estimate for each of the bias and variance measures is presented. It is assumed that irreducible error is zero in this domain. That is, any two objects that share a common description also share the same class.

The central tendency is the most frequent classification for an object. Error is the proportion of classifications that are incorrect.

Contribution of bias to error is that portion of the total error across the distribution of test sets that is due to errors committed by the central tendency of the learning algorithm. This is the proportion of classifications that are both incorrect and equal to the central tendency. *Contribution of variance to error* is that portion of the total error across the distribution of test sets that is due to errors that are deviations from the central tendency of the learning algorithm. This is the proportion of classifications that are both incorrect and not equal to the central tendency. These two terms sum to total error. They are used to evaluate the extent

Table 2. Example of estimation of bias/variance measures for three cases from ten n-fold cross-validation trials

	Case 1	Case 2	Case 3
Correct class	a	b	c
Prediction 1	a	a	a
Prediction 2	a	a	b
Prediction 3	a	a	c
Prediction 4	a	a	c
Prediction 5	b	a	c
Prediction 6	b	a	c
Prediction 7	b	b	c
Prediction 8	c	b	c
Prediction 9	c	b	c
Prediction 10	c	b	c
Central tendency	a	a	c
Error	0.6	0.6	0.2
Contribution of bias to error	0.0	0.6	0.0
Contribution of variance to error	0.6	0.0	0.2
$bias_{KW}{}^2_x$	0.27	0.36	0.03
$variance_{KW\,x}$	0.33	0.24	0.17
$bias_{KD}$	0.0	1.0	0.0

to which variations in the classifiers formed from one training set to another affect the error of the learning algorithm. High contribution of bias to error indicates high error resulting from the learning bias whereas high contribution of variance to error indicates high error resulting from the algorithm's responsiveness to variations between training sets. These terms equate to Breiman's (1996b) bias and variance definitions except that irreducible error is aggregated into the two terms. This is desirable in the current context for two reasons. First, it is difficult to usefully estimate irreducible error in 'real-world' learning tasks for which the true underlying class distribution is not known, because there are usually too few cases at any given point in the instance space to reliably estimate the class distribution at that point. Second, irreducible error is invariant across learning algorithms for a single task and hence not a significant factor in comparative evaluation.

The bias and variance measures $bias_{KW}{}^2_x$ and $variance_{KW\,x}$, developed by Kohavi and Wolpert (1996) are used in this research as representative of definitions derived from those for numeric regression. Following Kohavi and Wolpert (1996), irreducible error is aggregated into $bias_{KW}{}^2_x$. Like contributions of bias and variance to error, these two terms sum to *error*.

Kong and Dietterich's (1995) bias measure, $bias_{KD}$, is also used. It represents the probability of error for the central classification tendency of the learning algorithm, without consideration of the probability of that bias being established. This can be viewed as a measure of the quality of the central tendency, without consideration of the frequency or strength of that central tendency, and is valuable when comparing the learning biases of different algorithms.

4. Previous bias/variance analyses of decision committee performance

Breiman (1996b) shows that bagging can be expected to reduce the variance of a classifier. This is because bagging can be viewed as a method for developing a classifier that classifies using an estimate of the central tendency for the learner. While no satisfactory account has been offered of why such a reduction in variance should not be accompanied by a corresponding increase in error due to bias, a number of studies using both artificial and 'real-world' data, a number of different base learners, and a variety of different definitions of bias and variance have all shown that, on the whole, bagging does tend to decrease variance without unduly affecting bias (Breiman, 1996b; Schapire et al., 1997; Bauer & Kohavi, in press).

In contrast, (Friedman, Hastie, & Tibshirani, to appear) show that AdaBoost can be viewed as a form of additive logistic regression. They state that boosting by reweighting "appears to be a purely 'bias' reduction procedure, intended to increase the flexibility of stable (highly biased) weak learners."

However, in empirical studies AdaBoost appears to reduce both bias and variance (Breiman, 1996b; Schapire et al., 1997; Bauer & Kohavi, in press). No clear account of why it should have this effect has yet gained wide acceptance.

Both Breiman (1996b) and Schapire et al. (1997) present results with artificial data that suggest that AdaBoost is more effective than bagging at reducing both bias and variance. However, Bauer and Kohavi (in press) cast some doubt upon this result. In their studies with a cross-section of data sets from the UCI repository (Blake, Keogh, & Merz, 1999), AdaBoost is on-the-whole more effective at reducing bias than is bagging, but bagging is more effective than AdaBoost at reducing variance.

5. Alternative accounts of decision committee performance

A number of alternative explanations have been advanced for the performance of bagging and AdaBoost. Breiman (1996a) argues that bagging can transform order correct learners for an object x (those that have greater probability of forming a classifier that selects the most probable class for x than any other class) into optimal classifiers for that object. However, he does not present any evidence that the base learners that are normally employed with bagging will usually be order correct. He also makes the important observation that instability (responsiveness to changes in the training data) is a prerequisite for bagging to be effective. A committee of classifiers that all agree in all circumstances will give identical performance to any of its members in isolation. A variance reduction process will have no effect if there is no variance.

Freund and Schapire (1997) present a proof that AdaBoost can rapidly reduce *resubstitution error* (the error of the learned classifier on the training data). However, reduction of resubstitution error need not imply reduction of error outside the training data. They argue that the application of structural risk minimization can produce such a link, but have shown subsequently (Schapire et al., 1997) that the empirical evidence does not support this theoretical supposition.

Instead, Schapire et al. (1997) argue that AdaBoost is effective because it boosts the margins of the committee's weighted classifications. The margin for a committee voting to classify an object x is the difference between the sum of weighted votes for the correct class for x and of the weighted votes for the class that obtains the greatest weighted vote of the remaining classes. However, Breiman (1997) argues that this account is flawed, demonstrating algorithms that are more effective than AdaBoost at increasing margins but less effective at reducing error.

Breiman (1996b) argues that *adaptive resampling* is the key mechanism by which AdaBoost reduces error. An adaptive resampling algorithm learns a committee by repeated sampling from a training set. The probability of a given training case being included in a given sample is increased if it has been misclassified by committee members learned from previous samples. The success of an alternative adaptive resampling algorithm, arc-x4, provides some support for this argument. However, Bauer and Kohavi (in press) show that while AdaBoost is equally effective at reducing error using either reweighting or resampling, arc-x4 is much less effective using reweighting than using resampling. This leaves room for speculation that adaptive resampling does not account for all of AdaBoost's performance.

It has also been observed that for both bagging and AdaBoost, an increase in committee size usually leads to a decrease in prediction error, but the relative impact of each successive addition to a committee is ever diminishing. Most of the effect of each technique is obtained by the first few committee members (Breiman, 1996a; Freund & Schapire, 1996; Breiman, 1996b; Bauer & Kohavi, in press).

6. MultiBoosting

The observations that bagging and AdaBoost appear to operate by different mechanisms, have different effects, and both have greatest effect obtained from the first few committee members, suggest that it might be possible to obtain benefit by combining the two. As the mechanisms differ, their combination may out-perform either in isolation. Given that

- bagging mainly reduces variance, while AdaBoost reduces both bias and variance; and
- there is evidence that bagging is more effective than AdaBoost at reducing variance (Bauer & Kohavi, in press)

their combination may be able to retain AdaBoost's bias reduction while adding bagging's variance reduction to that already obtained by AdaBoost. As most of the effect of each approach is obtained by the first few committee members, it is possible that even quite small committees formed by a combination of the approaches might obtain most of the benefit of each approach in isolation.

One method for combining the two might be to simply develop two sub-committees, one containing members formed by boosting and the other containing members formed by bagging. However, this raises the non-trivial question of how the votes of the two sub-committees should be combined. AdaBoost weights the votes of its

Table 3. Determining sub-committee termination indexes

SCTerm
input:
integer T specifying the number of iterations.
output:
vector of integers I_i specifying the iteration at which each subcommittee $i \geq 1$ should terminate.

1. Set $n = \lfloor \sqrt{T} \rfloor$.
2. For $i = 1 \ldots n-1$, $I_i = \lceil i \times T/n \rceil$.
3. For $i = n \ldots \infty$, $I_i = T$.

committee members while bagging does not, making the votes of members of each committee incommensurable. Rather than pursuing this approach and tackling this question, this research explores bagging a set of sub-committees each formed by application of AdaBoost. Instead of using bagging per se, which would reduce the number of training examples available to form each sub-committee, wagging (Bauer & Kohavi, in press) was employed. Maintaining all examples in the training set was perceived as important, as Quinlan (1996) argues that AdaBoost using reweighting obtains benefit from access to all training examples during induction of every committee member.

To summarize, MultiBoosting can be considered as wagging (which is in turn a variant of bagging) committees formed by AdaBoost[3]. A decision has to be made as to how many sub-committees should be formed for a single run, and the size of those sub-committees. In the absence of an a-priori reason for selecting any specific values for these factors, the current implementation of MultiBoosting, MultiBoost, takes as an argument a single committee size T, from which it by default sets the number of sub-committees and the size of those sub-committees to $\sqrt{T}$. As both these values must be whole numbers, it is necessary round off the result. The precise method for deriving the values is presented in Table 3. For ease of implementation, this is achieved by setting a target final sub-committee member index, where each member of the final committee is given an index, starting from one. This allows the premature termination of boosting one sub-committee, due to too great or too low error, to lead to an increase in the size of the next sub-committee. If the last sub-committee is prematurely terminated, an additional sub-committee is added with a target of completing the full complement of committee members. Should this additional sub-committee also fail to reach this target, this process is repeated, adding further sub-committees until the target total committee size is achieved. The resulting MultiBoost algorithm is presented in Table 4.

In addition to the bias and variance reduction properties that this algorithm may inherit from each of its constituent committee learning algorithms, MultiBoost has the potential computational advantage over AdaBoost that the sub-committees may be learned in parallel, although this would require a change to the handling of early termination of learning a sub-committee. The AdaBoost process is inherently

Table 4. The MultiBoost algorithm

MultiBoost
input:

S, a sequence of m labeled examples $\langle (x_1, y_1), \ldots, (x_m, y_m) \rangle$ with labels $y_i \in Y$.

base learning algorithm **BaseLearn**.

integer T specifying the number of iterations.

vector of integers I_i specifying the iteration at which each subcommittee $i \geq 1$ should terminate.

1. $S' = S$ with instance weights assigned to be 1.
2. set $k = 1$.
3. For $t = 1$ to T {
4. If $I_k = t$ then
5. reset S' to random weights drawn from the continuous Poisson distribution.
6. standardize S' to sum to n.
7. increment k.
8. $C_t = BaseLearn(S')$.
9. $\epsilon_t = \dfrac{\sum_{x_j \in S': C_t(x_j) \neq y_j} weight(x_j)}{m}$ *[the weighted error on the training set]*
10. if $\epsilon_t > 0.5$ then
11. reset S' to random weights drawn from the continuous Poisson distribution.
12. standardize S' to sum to n.
13. increment k.
14. go to Step 8.
15. otherwise if $\epsilon_t = 0$ then
16. set β_t to 10^{-10}
17. reset S' to random weights drawn from the continuous Poisson distribution.
18. standardize S' to sum to n.
19. increment k.
20. otherwise,
21. $\beta_t = \dfrac{\epsilon_t}{(1 - \epsilon_t)}$.
22. For each $x_j \in S'$,
23. divide $weight(x_j)$ by $2\epsilon_t$ if $C_t(x_j) \neq y_j$ and $2(1 - \epsilon_t)$ otherwise.
24. if $weight(x_j) < 10^{-8}$, set $weight(x_j)$ to 10^{-8}.
25. }

Output the final classifier:

$$C^*(x) = \underset{y \in Y}{argmax} \sum_{t: C_t(x) = y} log \frac{1}{\beta_t}.$$

sequential, minimizing the potential for parallel computation. However, each classifier learned with wagging is independent of the rest, allowing parallel computation, a property that MultiBoost inherits at the sub-committee level.

7. Evaluation

To evaluate the relative efficacy of AdaBoost, bagging and MultiBoost, they were applied to a wide cross-section of data sets from the UCI repository (Blake et al., 1999). Many different committee sizes have been studied in previous research. Relatively small committee sizes are attractive from the perspective of exploring the benefits that can be obtained with relatively modest increases in computation. Further, as the most benefit is obtained from the first few committee members, subsequent computation, while improving accuracy, provides relatively poor returns for investment of computational resources. These reasons, and the relative low computational demands involved, support the study of committees of size ten, and consequently each technique was evaluated with this committee size. However, it is interesting to compare the performance of MultiBoost, which aggregates a number of constituent boosted sub-committees, against boosted committees of the size of those constituents. Also, it is credible that different committee learning techniques may demonstrate different performance profiles at differing committee sizes. For these reasons, committees of size 100 were also included in the study. A MultiBoost committee of size 100 is composed of ten sub-committees each comprising ten boosted committee members.

As MultiBoost uses wagging, rather than bagging, committees formed by wagging using the continuous Poisson distribution were also included. This is equivalent to MultiBoost with sub-committees of size 1 and each committee member receiving an equal vote.

The base learning algorithm, C4.5, was also included for comparison.

7.1. Data sets

Thirty-six representative data sets from the UCI repository were used in the experiments. These data sets were restricted to 'real world' data with the exception of the waveform data that was included due to its use in numerous previous committee learning experiments. Table 5 presents the number of cases, the number of attributes by which each case is described, and the number of classes for each data set. All data sets on which the algorithms have been evaluated are presented herein.

7.2. Statistics employed

One motivation for determining comparative performance across a set of different learning tasks is to assess whether one algorithm demonstrates a general advantage over another. It is well known that no algorithm can hold a general advantage

Table 5. Description of data sets

Name	Cases	Class	Atts	Name	Cases	Class	Atts
adult	48842	2	14	hypo	3772	4	29
anneal	898	6	38	iris	150	3	4
audio	226	24	69	kr-vs-kp	3196	2	36
autos	205	7	24	labor-neg	57	2	16
balance-scale	625	3	4	lenses	24	3	4
breast cancer Slov.	286	2	9	letter-recognition	20000	26	16
breast cancer Wisc.	699	2	9	lymphography	148	4	18
Cleveland	303	2	13	new-thyroid	215	3	5
credit (Aust.)	690	2	15	phoneme	5438	52	7
discordant	3772	2	29	Pima diabetes	768	2	8
echocardiogram	74	2	6	promoters	106	2	57
German	1000	2	20	primary tumor	339	22	17
glass	214	7	9	soybean large	683	19	35
heart	270	2	13	segment	2310	7	19
hepatitis	155	2	19	sick	3772	2	29
horse-colic	368	2	21	splice junction	3177	3	60
house-votes-84	435	2	16	tic-tac-toe	958	2	9
Hungarian	294	2	13	waveform	300	3	21

in terms of off-training-set error over another across all possible learning tasks (Schaffer, 1994; Wolpert, 1995). However, it is credible that 'real-world' learning tasks are not uniformly distributed across the set of all possible tasks, and hence that a general advantage is possible for the set of 'real-world' tasks, or at least a specific sub-set thereof (Rao, Gordon, & Spears, 1995). If we take the thirty-six data sets used in the experimentation as broadly representative of those in the UCI repository (as I believe them to be) and hence at the very least as representative of those commonly used in experimental machine learning research, it should be possible to use statistical analysis to draw conclusions relating to the expected general relative performance of the algorithms for at least this restricted class of learning tasks.

This leads, however, to the vexatious issue of what statistical analyses to use. The practice of using t-tests to compare the performance of two algorithms on sequences of training and tests sampled from a single data set has come under recent scrutiny (Dietterich, 1998; Salzberg, 1997). While such a test can provide an accurate evaluation of the probability of obtaining the observed outcomes by chance, it has limited ability to predict relative performance even on further training/test set samples from the same domain but from outside the initial data set, let alone on other domains.

There is a further problem that with large numbers of algorithms and data sets, the probability of type one error occurring climbs dramatically if individual hypothesis tests are performed on many algorithm-pair-dataset-tuples.

To avoid these problems I do not perform significance tests on individual data sets. Rather, a single significance test is performed for every pair of algorithms that are compared, a sign test on the win/draw/loss record across all data sets. If the probability of obtaining the observed outcomes by chance is sufficiently low (I use the common 0.05 critical level), I conclude that the observed performance is indicative of a general underlying advantage to one algorithm or the other with respect to type of learning task studied in this research.

In addition to this sign test, the following aggregate descriptive statistics are employed.

Mean error. This is the mean of error across all datasets. This provides a very gross indication of relative performance. It is debatable whether error rates in different domains are commensurable, and hence whether averaging error rates across domains is very meaningful. Nonetheless, a low average error rate is indicative of a tendency toward low error rates for individual domains.

Geometric mean error ratio. A number of recent papers have used the mean ratio of error rates across domains as a measure of relative performance (Quinlan, 1996; Bauer & Kohavi, in press). The ratio of error rates on a single domain for algorithms x and y, where $E(x)$ and $E(y)$ denote the respective mean error of each algorithm, is $E(x)/E(y)$. On the face of it, this measure allows for the relative difficulty of error reduction in different domains. For example, consider the case where two algorithms x and y achieve error rates of 0.10 and 0.20 for domain A and 0.35 and 0.20 for domain B, respectively. The ratios will be 0.50 and 1.75, giving a mean ratio of 1.125, indicating that, on average, algorithm x has error 12.5% greater than algorithm y. This may seem like a credible summary of their relative performance. While x out-performed y on the 'easier' task, it was out-performed on the more difficult. However, if we consider the ratio of y to x, we get $0.20/0.10 = 2.0$ and $0.20/0.35 = 0.571$ giving a mean ratio of 1.286, indicating that, on average algorithm y has error 28.6% greater than algorithm x! Clearly it is not desirable to conclude that two algorithms out-perform each other. One solution is to use the geometric mean rather than the arithmetic mean to aggregate the result over multiple domains. The geometric mean of a set of values $v_1, v_2, \ldots v_n$ is

$$e^{\frac{\sum_{i=1}^{n} log(v_i)}{n}} = \sqrt[n]{\prod_{i=1}^{n} v_i}. \tag{3}$$

The geometric mean of a set of ratio values $a_1/b_1, a_2/b_2, \ldots a_n/b_n$ has the desirable property that if it is greater than one then the geometric mean of $b_1/a_1, b_2/a_2, \ldots b_n/a_n$ will be less than one, and vice versa. For the example of algorithms x and y on domains A and B, above, the geometric mean of x_i/y_i is 0.935 while the geometric mean of y_i/x_i is 1.069, suggesting that it is x that enjoys the true advantage in terms of error reduction over the two domains.

However, even this measure should be treated with considerable caution. For example, increasing average accuracy should be as desirable as reducing average error. We should be just as interested in algorithms that can, on average, provide relative increases in accuracy, as we are in algorithms that can, on average, produce

relative reductions in error. The accuracy of an algorithm will equal $1 - error$. Hence, to continue the previous example, algorithm x and y's accuracy for A is 0.90 and 0.80 and for B is 0.65 and 0.80. The accuracy ratios for x over y are therefore 1.125 and 0.813 and for y over x, 0.889 and 1.231. The geometric means of these two ratios are 0.956 and 1.046, respectively, exhibiting the desirable property that one is greater than 1.0 while the other is below 1.0. However, it is x over y for which the value is below 1.0. If we take the geometric mean of accuracy/error to indicate a general tendency to increase or decrease accuracy/error, then we have the undesirable outcome, when we take this result together the earlier geometric mean of error ratios, that algorithm x, in general, reduces *both* error and accuracy with respect to y. Such an outcome is possible because error ratio favors an algorithm that performs well when error is low, whereas accuracy ratio favors an algorithm that performs well when error is high.

Despite these problems, the geometric mean of error ratios is provided as a summary statistic. Like mean error, however, this statistic should be interpreted with caution. It should be noted that while this statistic cannot be calculated if either algorithm achieves zero error for any data set, this rarely occurs in machine learning research and did not occur in the studies reported herein.

Geometric mean bias/variance ratio. Geometric means of ratios between two algorithms' bias or variance outcomes are also provided. It should be recalled that these statistics favor strong performance at low values of the measure being averaged over strong performance at high values. Like geometric mean error ratios, these statistics should be interpreted with caution. These statistics cannot be calculated if an algorithm achieves a mean of zero for the relevant metric on any single data set, but this condition did not arise in the current research.

Win/Draw/Loss record. This is the major comparative descriptive statistic employed in this paper. The three values are, respectively, the number of datasets for which algorithm a_1 obtained better, equal, or worse performance outcomes than algorithm a_2 on a given measure. Sign tests can be applied to these summaries. These tests will indicate the probability of obtaining the observed record of wins to losses, or more extreme, by chance. If the sign test result is significantly low then it is reasonable to conclude that it is unlikely that the outcome was obtained by chance and hence that the record of wins to losses represents a systematic underlying advantage to one of the algorithms with respect to the type of domains on which they have been tested.

7.3. Error rates

Tables 6 and 7 provide comparative summaries of the error of each algorithm across all data sets. These summary tables have the following format. In the following descriptions, *row* indicates the mean error on a data set for the algorithm with which a row is labeled, while *col* indicates the mean error for the algorithm with which the column is labeled. The first row of each summary table presents the mean error across all data sets. Rows labeled ṙ present the geometric mean of the error ratio col/row. Rows labeled s present the win/draw/loss statistic, where the

Table 6. Comparison of error for $t = 10$

Algorithm		C4.5	Bagging	Wagging	AdaBoost	MultiBoost
Mean, all data sets		0.177	0.159	0.164	0.161	0.156
C4.5	ṙ		0.889	0.930	0.845	0.826
	s		30/3/3	28/4/4	25/1/10	29/1/6
	p		< 0.001	< 0.001	0.017	< 0.001
Bagging	ṙ			1.046	0.950	0.929
	s			10/1/25	16/2/18	21/2/13
	p			0.017	0.864	0.229
Wagging	ṙ				0.908	0.888
	s				20/2/14	23/2/11
	p				0.392	0.058
AdaBoost	ṙ					0.977
	s					21/4/11
	p					0.110

Table 7. Comparison of error for $t = 100$

Algorithm		C4.5	Bagging	Wagging	AdaBoost	MultiBoost
Mean, all data sets		0.177	0.153	0.157	0.149	0.145
C4.5	ṙ		0.856	0.879	0.742	0.730
	s		29/5/2	29/3/4	28/1/7	30/2/4
	p		< 0.001	< 0.001	0.001	< 0.001
Bagging	ṙ			1.027	0.867	0.853
	s			4/6/26	20/2/14	27/2/7
	p			< 0.001	0.392	0.001
Wagging	ṙ				0.844	0.830
	s				21/4/11	27/1/8
	p				0.110	0.002
AdaBoost	ṙ					0.983
	s					24/2/10
	p					0.024

first value is the number of data sets for which $col < row$, the second is the number for which $col = row$ and the last is the number for which $col > row$. Rows labeled p present the result of a two-tailed sign test on the win-loss record. This is the probability of obtaining the observed record of wins to losses, or more extreme, if wins and losses were equi-probable random events. Detailed breakdowns of error by data set are provided in Appendix C.

At both committee sizes MultiBoost obtains the lowest mean error. At $t = 10$ bagging comes next, at $t = 100$, AdaBoost obtains the second lowest mean error. Wagging has the highest mean error of the four committee learning techniques at both committee sizes. All of the comparative metrics rank the algorithms in the same order, with the minor exception, that the geometric mean of error ratios favors AdaBoost over bagging at $t = 10$.

Considering specifically the relative performance of MultiBoost, it achieves a lower geometric mean ratio of error over that of C4.5 than any other algorithm[4]. The geometric mean error ratios of MultiBoost over each other algorithm all favor MultiBoost. In comparison to every other algorithm, MultiBoost achieves lower error for more data sets than the other algorithm achieves lower error. The frequency of this advantage is not statistically significant at $t = 10$, but is significant against all algorithms at $t = 100$.

MultiBoosting frequently beats each of AdaBoost and bagging. Does it, however, usually obtain a mid-point between the two, or is it outperforming the better of the two? At $t = 10$ the error of MultiBoost is lower than the minimum of the error of AdaBoost and bagging for 11 datasets and higher for 21. At $t = 100$ MultiBoost obtains lower error for 16 datasets and higher error for 15 datasets. At both committee sizes MultiBoosting frequently obtains lower error than either of AdaBoost or bagging.

It is notable that while AdaBoost usually greatly reduces the error of the base learner, it occasionally increases the error, sometimes substantially. Of the datasets examined, at $t = 100$ AdaBoost obtains higher error than C4.5 on seven datasets. Of these seven, MultiBoost reduces C4.5's error on two (balance-scale and echocardiogram), maintains it on one (iris), increases error but to a lesser degree than AdaBoost on three (adult, breast cancer Slovenia and horse-colic) and increases the error of AdaBoost on one (lenses). These results suggest that MultiBoosting moderates this aspect of AdaBoost's operational profile.

7.4. Wagging and bagging restarts in AdaBoost

A variant of Bauer and Kohavi's (in press) variant of the original AdaBoost is used. It differs from their variant in using wagging to reset weights when committee learning halts due to too high or too low resubstitution error, rather than bagging. This change was made for consistency with the MultiBoosting implementation used in the experiments. To evaluate whether the use of wagging rather than bagging disadvantaged AdaBoost, a variant that differed only by using bagging in these circumstances was also included in the cross-validation trials. At $t = 10$, the two variants obtained different results for only four data sets. For one of these bagging resulted in lower error, while for three wagging obtained lower error. The mean error across all domains did not differ when measured to three decimal places. The geometric mean error ratio of bagged over wagged was 1.002, slightly favoring the wagged variant. At $t = 100$ the two variants differed on twelve data sets. For four data sets, bagging resulted in lower error than wagging and for the remaining eight wagging obtained the lower error. The mean error rates across all domains were identical measured to three decimal places and the geometric mean error ratio was 1.000, suggesting that neither approach enjoyed a measurable general advantage over the other. Substituting the bagging variant for the wagging variant in the above analyses does not produce any appreciable difference in the outcomes.

Table 8. Comparison of contribution of bias to error for $t = 10$

Algorithm		C4.5	Bagging	Wagging	AdaBoost	MultiBoost
Mean, all data sets		0.114	0.110	0.113	0.099	0.100
	ṙ		0.947	0.999	0.773	0.792
C4.5	s		21/3/12	15/4/17	31/2/3	33/0/3
	p		0.163	0.860	< 0.001	< 0.001
	ṙ			1.055	0.817	0.836
Bagging	s			6/4/26	31/1/4	27/1/8
	p			0.001	< 0.001	0.002
	ṙ				0.774	0.793
Wagging	s				31/2/3	30/1/5
	p				< 0.001	< 0.001
	ṙ					1.024
AdaBoost	s					12/2/22
	p					0.121

8. Bias/variance analysis of performance

One of the reasons for developing the MultiBoost approach was the evidence that bagging was more effective than AdaBoost at reducing variance while the reverse was true with respect to reducing bias. Given that each algorithm obtained the greatest effect from the first few committee members, combining the two might be expected to achieve most of the variance reduction of bagging as well as most of the bias reduction of AdaBoost. To gain insight into the extent to which this analysis can explain MultiBoost's performance, the five measures of these two factors discussed in Section 3 were also evaluated for all of the experiments. Detailed breakdowns of the mean values for contribution of bias and variance to error for committee sizes 10 and 100 are presented in Appendix C.

Tables 8 and 9 summarize the relative performance of the algorithms with respect to contribution of bias to error at committee sizes 10 and 100, respectively. At $t = 10$, the algorithms are ordered from lowest to highest on mean value across all data sets, AdaBoost, MultiBoost, bagging, wagging, then C4.5. This order is repeated at $t = 100$ except that wagging and C4.5 exchange places. All other statistics conform to the same ordering, except that wagging loses to C4.5 more often than it wins at $t = 10$ as well as at $t = 100$. Both AdaBoost and MultiBoost achieve statistically significant win/draw/loss records over bagging, wagging, and C4.5. While AdaBoost achieves lower error than MultiBoost substantially more often than the reverse at $t = 10$, this advantage is not significant at the 0.05 level, and is eroded to a narrow margin at $t = 100$.

Tables 10 and 11 summarize the relative performance of the algorithms with respect to contribution of variance to error at committee sizes 10 and 100, respectively. At $t = 10$, the order of the algorithms with respect to the mean across all data sets is bagging, wagging, MultiBoost, AdaBoost, then C4.5. At $t = 100$,

Table 9. Comparison of contribution of bias to error for $t = 100$

Algorithm		C4.5	Bagging	Wagging	AdaBoost	MultiBoost
Mean, all data sets		0.114	0.114	0.116	0.104	0.107
	ṙ		0.983	1.008	0.796	0.808
C4.5	s		18/1/17	13/4/19	24/1/11	26/1/9
	p		> 0.999	0.377	0.041	0.006
	ṙ			1.026	0.810	0.822
Bagging	s			10/6/20	27/2/7	26/3/7
	p			0.099	0.001	0.001
	ṙ				0.790	0.801
Wagging	s				28/0/8	29/0/7
	p				0.001	< 0.001
	ṙ					1.015
AdaBoost	s					14/5/17
	p					0.720

Table 10. Comparison of contribution of variance to error for $t = 10$

Algorithm		C4.5	Bagging	Wagging	AdaBoost	MultiBoost
Mean, all data sets		0.063	0.050	0.051	0.062	0.057
	ṙ		0.803	0.813	1.037	0.934
C4.5	s		27/4/5	27/2/7	17/0/19	18/3/15
	p		< 0.001	0.001	0.868	0.728
	ṙ			1.012	1.292	1.163
Bagging	s			18/5/13	8/3/25	11/6/19
	p			0.473	0.005	0.200
	ṙ				1.276	1.149
Wagging	s				9/2/25	12/6/18
	p				0.009	0.362
	ṙ					0.900
AdaBoost	s					23/2/11
	p					0.058

however, the margins have narrowed greatly, and MultiBoost achieves the greatest reduction by a very narrow margin, the order being otherwise unchanged.

Bagging outperforms all of the other algorithms on both geometric mean ratio and win/draw/loss record, although the results on the latter are only significant at the 0.05 level with respect to C4.5 and AdaBoost at $t = 10$. On both metrics, wagging outperforms C4.5 and AdaBoost at both committee sizes and MultiBoost at $t = 10$, but is outperformed on geometric mean error ratio by MultiBoost at $t = 100$. MultiBoost outperforms AdaBoost on both measures at both committee sizes, the win/draw/loss record being significant at the 0.05 level for $t = 100$.

Table 11. Comparison of contribution of variance to error for $t = 100$

Algorithm		C4.5	Bagging	Wagging	AdaBoost	MultiBoost
Mean, all data sets		0.063	0.039	0.041	0.045	0.038
	ṙ		0.615	0.641	0.690	0.616
C4.5	s		30/3/3	29/1/6	26/2/8	28/2/6
	p		< 0.001	< 0.001	0.003	< 0.001
	ṙ			1.043	1.123	1.002
Bagging	s			10/5/21	11/4/21	14/5/17
	p			0.071	0.110	0.720
	ṙ				1.077	0.962
Wagging	s				13/4/19	15/3/18
	p				0.377	0.728
	ṙ					0.892
AdaBoost	s					24/7/5
	p					0.001

It seems that at a committee size of 100, MultiBoost achieves most of AdaBoost's mean reduction in error due to bias and actually improves upon wagging's mean reduction in error due to variance.

Comparing the outcomes at $t = 10$ to those at $t = 100$ a surprising trend emerges. For all committee algorithms, the average contribution of bias to error increases as the committee size increases. As all of the committee approaches reduced average contribution of bias to error at $t = 10$, the average effect of the additional 90 committee members is to undo some of the bias reduction effect of the smaller committees. In contrast, all committee algorithms demonstrate greater average reductions in contribution of variance to error as the committee size increases. For all algorithms this reduction in variance outweighs the increase in bias, resulting in an overall reduction in error.

These results are surprising. AdaBoost, for example, appears to perform bias reduction, and appears to achieve greater benefit the greater the committee size. Why then should larger committees achieve less bias reduction than smaller committees? A number of possible explanations present themselves. The first is that this unexpected result is simply an artifact of the use of contribution of bias to error rather than one of the alternative bias definitions. However, Kohavi and Wolpert's (1996) bias and variance measures also follow the same pattern as the contribution of bias and variance to error, as shown by Tables 12 and 13 that present the mean values of these terms across all data sets along with the geometric mean and win/draw/loss records over C4.5 for each of the committee methods.

So it appears that the unexpected behaviour of contribution of bias to error as committee size increases is not related to the new definitions of bias and variance employed. Another possibility is that this is a chance artifact of the two committee sizes. It is, after all, very dangerous to infer a trend from two points. To test this possibility, further cross-validation trials were performed for AdaBoost at commit-

Table 12. Summary, mean $bias_{KW}{}_x^2$

$t = 10$		**C4.5**	**Bagging**	**Wagging**	**AdaBoost**	**MultiBoost**
Mean, all data sets		0.117	0.111	0.115	0.101	0.102
	ṙ		0.934	0.987	0.782	0.791
Compare **C4.5**	s		21/7/8	20/1/15	31/1/4	32/0/4
	p		0.024	0.500	< 0.001	< 0.001
$t = 100$		**C4.5**	**Bagging**	**Wagging**	**AdaBoost**	**MultiBoost**
Mean, all data sets		0.117	0.115	0.117	0.104	0.105
	ṙ		0.963	0.983	0.772	0.784
Compare **C4.5**	s		19/2/15	16/4/16	25/2/9	28/2/6
	p		0.608	> 0.999	0.009	< 0.001

Table 13. Summary mean $variance_{KW}{}_x$

$t = 10$		**C4.5**	**Bagging**	**Wagging**	**AdaBoost**	**MultiBoost**
Mean, all data sets		0.060	0.048	0.049	0.059	0.055
	ṙ		0.831	0.842	1.023	0.940
Compare **C4.5**	s		26/4/6	25/5/6	16/1/19	17/6/13
	p		0.001	0.001	0.736	0.585
$t = 100$		**C4.5**	**Bagging**	**Wagging**	**AdaBoost**	**MultiBoost**
Mean, all data sets		0.060	0.039	0.040	0.045	0.040
	ṙ		0.649	0.652	0.716	0.657
Compare **C4.5**	s		33/2/1	32/3/1	25/2/9	27/4/5
	p		< 0.001	< 0.001	0.009	< 0.001

tee sizes 50 and 200. The same sequence of cross-validation folds as employed for the previous experiments were used in these subsequent experiments. Tables 14, 15, and 16 summarize, respectively, the relative error, contribution of bias to error, and mean $bias_{KW}{}_x^2$ of AdaBoost at each of these committee sizes along with the committee sizes previously examined.

The error shows a steady decrease as the committee size increases, but both measures of bias show a slight but steady increase (although the geometric mean favors $t = 200$ over both $t = 100$ and $t = 50$, and $t = 100$ over $t = 50$). While the mean across all data sets does not increase on either measure from $t = 100$ to $t = 200$, the smaller committee size achieves a lower measure for more data sets than the larger committee size. The results suggest that at small committee sizes there is a rapid decrease in the contribution of bias to error as committee size increases (at $t = 10$ there is a substantial decrease over the base learner) but at some point this trend reverses. It seems likely that the exact curve will vary substantially from domain to domain, and there does not appear to be much point in attempting to map it more precisely.

Another possible explanation is that as the committee size increases the central tendency changes for the worse. One manner in which to evaluate this explana-

Table 14. Error for AdaBoost at $t = 10, 50, 100$, and 200

		$t = 10$	$t = 50$	$t = 100$	$t = 200$
Mean, all data sets		0.161	0.150	0.149	0.148
	ṙ		0.900	0.878	0.859
$t = 10$	s		28/3/5	27/2/7	28/2/6
	p		< 0.001	0.001	< 0.001
	ṙ			0.976	0.955
$t = 50$	s			22/6/8	24/4/8
	p			0.016	0.007
	ṙ				0.978
$t = 100$	s				22/4/10
	p				0.050

Table 15. Contribution of bias to error for AdaBoost at $t = 10, 50, 100$, and 200

		$t = 10$	$t = 50$	$t = 100$	$t = 200$
Mean, all data sets		0.099	0.102	0.104	0.104
	ṙ		1.032	1.030	1.016
$t = 10$	s		11/3/22	8/4/24	9/5/22
	p		0.080	0.007	0.029
	ṙ			0.997	0.984
$t = 50$	s			9/10/17	12/5/19
	p			0.169	0.281
	ṙ				0.987
$t = 100$	s				12/7/17
	p				0.458

Table 16. Mean $bias_{KW}{}_x^2$ for AdaBoost at $t = 10, 50, 100$, and 200

		$t = 10$	$t = 50$	$t = 100$	$t = 200$
Mean, all data sets		0.101	0.102	0.104	0.104
	ṙ		0.987	0.987	0.985
$t = 10$	s		12/5/19	11/6/19	11/7/18
	p		0.281	0.200	0.265
	ṙ			1.000	0.998
$t = 50$	s			10/11/15	11/9/16
	p			0.424	0.442
	ṙ				0.998
$t = 100$	s				8/13/15
	p				0.210

Table 17. Summary of $bias_{KD}$

$t = 10$		**C4.5**	**Bagging**	**Wagging**	**AdaBoost**	**MultiBoost**
Mean, all data sets		0.149	0.139	0.143	0.133	0.132
	ṙ		0.926	0.975	0.800	0.803
Compare **C4.5**	*s*		22/5/9	20/4/12	27/1/8	26/3/7
	p		0.029	0.215	0.002	0.001
$t = 100$		**C4.5**	**Bagging**	**Wagging**	**AdaBoost**	**MultiBoost**
Mean, all data sets		0.149	0.140	0.143	0.131	0.134
	ṙ		0.929	0.955	0.760	0.773
Compare **C4.5**	*s*	-	23/5/8	22/4/10	27/1/8	26/0/10
	p		0.011	0.050	0.002	0.011

tion is to consider Kong and Dietterich's (1995) definition of bias. As explained in Section 3, $bias_{KD}$ can be considered as a measure of the quality of the central tendency, without consideration of the frequency or strength of that central tendency. A summary of the algorithms' performance on this measure is presented in Table 17. As can be seen, the average quality of the central tendency improves markedly from the base learner to committees of size 10. AdaBoost demonstrates further improvement as the committee size increases, both in the mean across all data sets and in the geometric mean over C4.5. This result supports the proposition that the increase in AdaBoost's contribution of bias to error as committee size increases is due to a decrease in variance rather than a deterioration in the quality of the central tendency. It appears that AdaBoost transforms the bias of the base learner in a manner that has positive average effect, but that this transformation is achieved less frequently at lower committee sizes than higher, resulting in greater variance for the smaller committees.

The effect of the increase in committee size on the quality of the central tendency of the other committee algorithms is not so clear cut, however. For bagging, the mean across all data sets increases minutely, as does the geometric mean over C4.5, but the win/draw/loss record against C4.5 improves marginally. Wagging achieves the same mean across all data sets at both committee sizes, but achieves a slight improvement in the two comparative statistics against C4.5. MultiBoost experiences a small increase in the mean across all data sets, a deterioration in the win/draw/loss record against C4.5, but an improvement in the geometric mean against C4.5.

So, at least for AdaBoost, the effect appears to be real, and not an artifact of the measure of bias that is employed or of the arbitrary choice of two specific committee sizes. A possible explanation of this effect is that the committee induction algorithms perform a transformation of bias from the base learning algorithm to the committee algorithm, but that at smaller committee sizes there is considerable fluctuation in the bias that is established. This fluctuation might be between the base algorithm's bias and the bias of a large committee, or between different biases that tend to be associated with different committee sizes. As can be seen from

Table 18. Comparison of MultiBoost $t = 100$ against AdaBoost $t = 10$

	error	contrib. bias	contrib. variance	$bias_{KW_x}^2$	$variance_{KW_x}$	$bias_{KD}$
ṙ	0.864	1.045	0.594	1.002 -	0.642	0.966
s	33/2/1	8/2/26	35/1/0	12/4/20	35/1/0	22/4/10
p	< 0.001	0.003	< 0.001	0.215	< 0.001	0.050

the contribution of bias to error at $t = 200$, when this new bias is established, on average, it contributes less to error than the bias of the base algorithm. However, at smaller committee sizes, different runs establish different biases, leading to greater variance. The bias is less consistently achieved, and hence contributes less to error.

8.1. Comparison of MultiBoost $\boldsymbol{t = 100}$ *against AdaBoost* $\boldsymbol{t = 10}$

One of the motivations for studying the two committee sizes 10 and 100, was that this would allow a comparison of the performance of MultiBoost against that of AdaBoost with the latter producing committees of the size of the sub-committees employed within the former. This enables direct comparison of the difference between the elements being wagged within MultiBoost against the result of that wagging. Table 18 presents a summary of the comparative performance of the two algorithms at these respective committee sizes. The win/draw/loss records denote MultiBoost$^{t=100}$'s wins, draws and losses over AdaBoost$^{t=10}$. The geometric means relate to the ratios of MultiBoost's performance over AdaBoost's. It is clear that the wagging of sub-committees within MultiBoost is very effective at reducing the error of the sub-committees formed by AdaBoost. However, both the contribution of bias to error and mean $bias_{KW_x}^2$ suggest that MultiBoost$^{t=100}$ slightly degrades the performance of AdaBoost$^{t=10}$ in this respect. It should be recalled, though, that there is reason to believe that the score on these measures is reduced by an increase in variance for AdaBoost at this committee size. Certainly, on $bias_{KD}$, which measures the error of the central tendency irrespective of the frequency with which that tendency is established, MultiBoost$^{t=100}$ appears to be improving performance over that of AdaBoost$^{t=10}$ with reasonable consistency (although it is interesting to note that the mean for AdaBoost$^{t=10}$ is 0.001 less than that for MultiBoost$^{t=100}$). MultiBoost is remarkably consistent at reducing the effect of variance, both as measured by the contribution of variance to error and mean $variance_{KW_x}$.

8.2. On the optimal number of sub-committees

At the outset of this research, in view of the lack of theoretical grounds for favoring the effect of either AdaBoost or wagging within MultiBoost, it was decided that the sub-committee size should be selected so as to equally balance the two. It is interesting, however, to consider other possible trade-off between the two factors. The more sub-committees there are, the more wagging is performed. The greater

Table 19. Summary of error for alternative sub-committee sizes

		MultiBoost	**MB50×2**	**MB2×50**
Mean, all data sets		0.145	0.143	0.147
Wagging	ṙ	0.830	0.875	0.824
	s	27/1/8	29/1/6	22/0/14
	p	0.002	< 0.001	0.243
AdaBoost	ṙ	0.983	1.037	0.976
	s	24/2/10	18/3/15	20/6/10
	p	0.024	0.728	0.099
MultiBoost	ṙ		1.055	0.992
	s		17/2/17	11/2/23
	p		> 0.999	0.058
MB50×2	ṙ			0.941
	s			16/2/18
	p			0.864

Table 20. Summary of contribution of bias to error for alternative sub-committee sizes

		MultiBoost	**MB50×2**	**MB2×50**
Mean, all data sets		0.107	0.105	0.105
Wagging	ṙ	0.801	0.870	0.780
	s	29/0/7	29/2/5	28/1/7
	p	< 0.001	< 0.001	0.001
AdaBoost	ṙ	1.015	1.101	0.987
	s	14/5/17	10/2/24	17/6/13
	p	0.720	0.024	0.585
MultiBoost	ṙ		1.086	0.973
	s		14/1/21	22/2/12
	p		0.311	0.121
MB50×2	ṙ			0.897
	s			19/3/14
	p			0.487

the sub-committee size, the greater the effect of boosting. The two extreme options were selected, to give some insight into the trade-offs—two sub-committees each containing 50 members (MB2×50) and 50 sub-committees each containing two members (MB50×2). Tables 19, 20, and 21 present, respectively, comparisons of the error, contribution of bias to error, and contribution of variance to error of both of these MultiBoost variants, against each other, the default MultiBoost, wagging, and AdaBoost with committees of size 100.

MB50×2 achieves the lowest mean error across all data sets. However, the win/draw/loss records show that MB50×2 achieves lower error for only two more data sets than MB2×50 achieves the lower error. When the geometric mean ratios against each other are compared, MB2×50 comes out ahead. These mixed results

Table 21. Summary of contribution of variance to error for alternative sub-committee sizes

		MultiBoost	MB50×2	MB2×50
Mean, all data sets		0.038	0.037	0.042
Wagging	ṙ	0.962	0.875	1.042
	s	15/3/18	27/1/8	15/4/17
	p	0.728	0.002	0.860
AdaBoost	ṙ	0.892	0.813	0.967
	s	24/7/5	25/3/8	17/7/12
	p	0.001	0.005	0.458
MultiBoost	ṙ		0.910	1.083
	s		22/4/10	7/6/23
	p		0.050	0.005
MB50×2	ṙ			1.190
	s			6/2/28
	p			< 0.001

suggest that neither approach enjoys a clear general advantage over the other with respect to error reduction.

Turning our attention to the contribution of bias and variance to error, there is some evidence that MB2×50 improves modestly upon AdaBoost on both these measures (although the win/draw/loss records are very far from statistical significance). MB50×2 appears to enjoy a clear advantage over wagging with respect to both bias and variance reduction. MB50×2 also appears to enjoy a clear advantage over MB2×50 with respect to variance. Further, while the result is not statistically significant, there is a suggestion of a small advantage to MB2×50 with respect to bias reduction. These results suggest that favoring AdaBoost is appropriate when the contribution bias to error of the base learner is high, and favoring bagging is appropriate when variance is the dominant contributor to error.

8.3. The application of bagging and wagging in MultiBoost

The original formulation of MultiBoost applied wagging rather than bagging to form the sub-committees, even though the inspiration for the technique was derived from bagging. Wagging was employed because AdaBoost appears to derive advantage from being able to consider all of the training data and wagging enables the data to be perturbed without completely removing any datum from consideration. However, the investigations reported above have shown that bagging enjoys a clear advantage over wagging, a result supported by Bauer and Kohavi (in press). This suggests that MultiBoost might benefit from modification to employ bagging rather than wagging of sub-committees, as was performed in a variant of MultiBoost described by Zheng and Webb (1998).

To test this hypothesis, MultiBoost was modified appropriately. Table 22 summarizes the relative error of the two MultiBoost variants. At $t = 10$, wagging

Table 22. Relative error with wagging and bagging of sub-committees

		$t = 10$		$t = 100$	
		wagging subcmtys	**bagging subcmtys**	**wagging subcmtys**	**bagging subcmtys**
Mean, all data sets		0.156	0.161	0.145	0.145
	ṙ		1.059		1.032
wagging subcmtys	s		9/4/23		14/2/20
	p		0.020		0.392

Table 23. Relative contribution of bias to error with wagging and bagging of sub-committees

		$t = 10$		$t = 100$	
		wagging subcmtys	**bagging subcmtys**	**wagging subcmtys**	**bagging submctys**
Mean, all data sets		0.100	0.099	0.107	0.105
	ṙ		1.014		1.026
wagging subcmtys	s		17/2/17		13/2/21
	p		> 0.999		0.229

Table 24. Relative contribution of variance to error with wagging and bagging of sub-committees

		$t = 10$		$t = 100$	
		wagging subcmtys	**bagging subcmtys**	**wagging subcmtys**	**bagging submctys**
Mean, all data sets		0.057	0.062	0.038	0.041
	ṙ		1.122		0.993
wagging subcmtys	s		7/1/28		15/8/13
	p		0.001		0.851

sub-committees enjoys a clear advantage on all measures. However, for the larger sub-committees, this advantage is eroded: the mean error across all domains for the two variants is identical, but the geometric mean and win/draw/loss records both still favor wagging, although not significantly.

Tables 23 and 24 present the relative performance of the variants with respect to contribution of bias and variance to error. On the former measure, the mean across all data sets favors bagging, but the geometric mean ratios favor wagging at both committee sizes. Neither enjoys a significant win/draw/loss record advantage over the other at either committee size, suggesting that neither enjoys a general advantage with respect to bias. However, for variance, wagging enjoys a clear advantage at $t = 10$, although this appears to be totally eroded at the larger committee size.

In all, these results suggest that there may be a slight advantage to the use of wagging within MultiBoost

9. Summary

MultiBoost combines wagging, which is a variance reduction technique, with AdaBoost, which performs both bias and variance reduction. This is achieved by wagging a set of sub-committees of classifiers, each sub-committee formed by AdaBoost. When forming decision committees using C4.5 as the base learning algorithm, MultiBoost is demonstrated to produce committees with lower error than AdaBoost significantly more often than the reverse across a wide cross-section of UCI data sets. This improved predictive accuracy is obtained with negligible increase in computational overheads. Indeed, MultiBoost offers a potential computational advantage over AdaBoost in that it is amenable to parallel execution. Each sub-committee may be learned independently of the others.

To summarize the experimental results on learning committees of decision trees—

- MultiBoost achieves greater mean error reductions than any of AdaBoost, wagging, or bagging decision trees at both committee sizes that were investigated. Although the win/draw/loss records do not reach statistical significance for committees of size 10, at size 100, MultiBoost achieves lower error significantly more often than higher error in comparison to all three of the alternative committee algorithms.

- AdaBoost is shown to, in general, reduce both error due to bias and variance, while bagging and wagging primarily reduce error due to variance. This result is consistent with Bauer and Kohavi (in press), Breiman (1996b) and Schapire et al.'s (1997) previous findings.

- Wagging using the continuous Poisson distribution is not as effective at variance reduction as is bagging. This result is consistent with Bauer and Kohavi's (in press) findings comparing wagging using Gaussian noise against bagging.

- Bagging and wagging tend to be more effective at variance reduction than AdaBoost. This is consistent with Bauer and Kohavi's (in press) results using 'real-world' data but contrary to Breiman (1996b) and Schapire et al.'s (1997) results with artificial data.

- MultiBoost achieves most of the bias reduction of AdaBoost together with most of the variance reduction of wagging.

- MultiBoosting using wagging of AdaBoosted sub-committees is more effective than MultiBoosting using bagging of AdaBoosted sub-committees, at least at small committee sizes.

- All committee algorithms have lower mean contributions of bias to error at $t = 10$ than at $t = 100$. At least in the case of AdaBoost, there is evidence that this occurs despite an improvement in the quality of the central tendency.

A possible explanation for this effect is that at $t = 10$ variance is increased, decreasing the frequency with which the central tendency is established and hence decreasing the opportunity for the central tendency to contribute to error.

MultiBoost has been demonstrated to, in general, achieve lower error than either AdaBoost of wagging when applied to a base learning algorithm and learning tasks for which there is sufficient scope for both bias and variance reduction. The benefits of MultiBoosting over either of its constituents might be expected to be significantly lower in an application for which there is little scope for both bias and variance reduction, such as decision stump or naive Bayesian learning, where variance is low. For conventional decision tree learning, however, MultiBoosting appears to offer significant gains over the standard AdaBoost algorithm.

Acknowledgments

I am grateful to Zijian Zheng and Ross Quinlan for fruitful discussions that have helped to stimulate this research. I am also grateful to John Slaney for suggesting the use of geometric means for aggregate statistics over ratio values. The Breast Cancer, Lymphography and Primary Tumor data sets were provided by the Ljubljana Oncology Institute, Slovenia. Thanks to the UCI Repository's maintainers and donors, for providing access to the data sets used herein.

Appendix A

Bias and variance measures for classification learning

This Appendix provides definitions of the five measures of bias and variance used in the current research, accompanied by a discussion of how these measures relate to previous bias/variance measures for classification learning.

It is important to note that a learner is applied to a finite vector of objects drawn from a distribution of training sets. However, its performance is evaluated with respect to test data drawn from a distribution of objects that may not directly relate to the distribution of training sets. Following Breiman (1996b), the misclassification error of $\mathcal{L}(T)$ is defined as the probability that it will misclassify a (y, x) pair drawn from the test instance distribution Y, X:

$$error(\mathcal{L}(T)) = P_{Y,X}(\mathcal{L}(T)(X) \neq Y) \tag{A.1}$$

The *intrinsic noise* or *irreducible prediction error* for an instance distribution Y, X is the lowest expected error that classifier (learned or otherwise) may obtain. It reflects the degree of randomness of the output variable and equals the prediction error of the optimal Bayes classifier

$$C^{Bayes}_{Y,X}(x) = \underset{y}{argmax}\, P(y \mid x) \tag{A.2}$$

Note that both the optimal Bayes classifier and the irreducible prediction error can only be calculated from the test data distribution. Neither can be calculated directly from a training set.

The central tendency $C^{\circ}_{\mathcal{L},\mathcal{T}}(x)$ for learner $\mathcal{L}$ over the distribution of training data sets $\mathcal{T}$ is the class with the greatest probability of selection for description x by classifiers learned by $\mathcal{L}$ from training sets drawn from $\mathcal{T}$.

$$C^{\circ}_{\mathcal{L},\mathcal{T}}(x) = \underset{y}{argmax}\, P_{\mathcal{T}}(\mathcal{L}(\mathcal{T})(x) = y) \tag{A.3}$$

Most bias/variance analyses for classification learning decompose classification error into three terms, bias, variance, and irreducible prediction error (Breiman, Friedman, Olshen, & Stone, 1984; Kohavi & Wolpert, 1996; Friedman, 1997). While it is useful to take account of irreducible prediction error for some analyses of classification algorithms, for the analysis of classification committees, interest centers on the manner in which bias and variance are affected. Further, it is difficult to estimate irreducible prediction error from the limited samples of data usually employed in classification learning experiments. In view of these two considerations, this research follows the lead of Kong and Dietterich (1995) in decomposing error into only the bias and variance terms.

Of previous bias/variance definitions for classification learning, the closest to the definitions for numeric regression are Friedman's (1997). However, these definitions only apply to the two class case, a limitation that mitigates against their use in the current context. Also closely capturing the definitions for numeric regression are Kohavi and Wolpert's (1996) definitions. They define three terms, ${bias_{KW}}_x^2$, $variance_{KW\,x}$, and σ_x with relation to description x.

$${bias_{KW}}_x^2 = \frac{1}{2}\sum_{y \in Y}[P_{Y,X}(Y = y|X = x) - P_{\mathcal{T}}(\mathcal{L}(\mathcal{T})(x) = y)]^2 \tag{A.4}$$

$$variance_{KW\,x} = \frac{1}{2}\left(1 - \sum_{y \in Y} P_{\mathcal{T}}(\mathcal{L}(\mathcal{T})(x) = y)^2\right) \tag{A.5}$$

$$\sigma_x = \frac{1}{2}\left(1 - \sum_{y \in Y} P_{Y,X}(Y = y|X = x)^2\right) \tag{A.6}$$

These definitions have the advantage of decomposing error into three terms that sum to error and measure the strength of the central tendency, the degree of variation between inferred classifiers, and the degree of variation in the output variable (classification) for a single input. However, these measures of bias and variance do not measure the extent to which each of these underlying quantities contributes to error. For example, consider an input x for a three class (a, b, and c) classification task for which $P_{\mathcal{T}}(\mathcal{L}(\mathcal{T})(x) = a) = 0.6$, $P_{\mathcal{T}}(\mathcal{L}(\mathcal{T})(x) = b) = 0.2$, and

$P_{\mathcal{T}}(\mathcal{L}(\mathcal{T})(x) = c) = 0.2$. Suppose also that $P_{Y,X}(Y = a|X = x) = 1.0$ (there is no intrinsic noise). $bias_{KW\,x}^2 = 0.12$. $variance_{KW\,x} = 0.28$. Now consider the scenario where $P_{\mathcal{T}}(\mathcal{L}(\mathcal{T})(x) = a) = 0.6$, $P_{\mathcal{T}}(\mathcal{L}(\mathcal{T})(x) = b) = 0.4$, $P_{\mathcal{T}}(\mathcal{L}(\mathcal{T})(x) = c) = 0.0$, and $P_{Y,X}(Y = a|X = x) = 1.0$. This is the same as the previous scenario except that whereas the errors in the former were equally divided between predictions of classes b and c, in the new scenario all errors involve predictions of class b. Now, $bias_{KW\,x}^2 = 0.16$ and $variance_{KW\,x} = 0.24$. Bias has increased. However, the probability of a classification deviating from the central tendency has not changed, nor has the strength or nature of the central tendency. All that has changed is the distribution between other classes of the deviations from the central tendency, in a manner that does not affect classification performance.

In contrast, Kong and Dietterich's (1995) definition of bias directly measures the error of the central tendency over the distributions of training sets ($\mathcal{T}$) and test cases (Y, X).

$$bias_{KD} = P_{(Y,X),\mathcal{T}}(C^{\circ}_{\mathcal{L},\mathcal{T}}(X) \neq Y) \tag{A.7}$$

$$variance_{KD} = error(\mathcal{L}(T)) - bias_{KD} \tag{A.8}$$

However, (A.8) does not adequately measure the error due to deviations from the central tendency. For example, consider a description x for which the central tendency differs from the optimal Bayes classifier. This will occur if the class most commonly selected for x by the learner is not the most common class for x in the test distribution. In this situation $bias_{KD} > 0.5$ and $error < bias_{KD}$. See, for example, Case 2 of Table 2. As a consequence, $variance_{KD}$ will be negative. If the negative values for $variance_{KD}$ on some descriptions are perfectly matched by a set of corresponding positive values on other descriptions, the total $variance_{KD}$ for a domain may be zero, irrespective of the extent to which the classifiers learned from different training sets differ from one another. To capture the usual intention for the variance term, it should increase as deviations from the central tendency increase. Further, while (A.7) measures the error of the central tendency, it can be argued that this differs from the error *due* to the central tendency. As a greater proportion of the learned classifiers differ from the central tendency, the contribution of the central tendency to the total error will decrease, and consequently, it could be argued, a measure of bias should also decrease. For these reasons, $variance_{KD}$ does not appear very useful for the comparisons of alternative learning algorithms to be performed herein, and has not been adopted in this work. There is value for this research in $bias_{KD}$, however, which captures the quality of the central tendency without regard for the consistency with which that central tendency is established.

Breiman's (1996b) formulation provides a simple and intuitive operationalization of the general bias/variance concept in terms of the contribution of the central tendency and deviations from the central tendency to error. Breiman first decomposes error into irreducible and reducible error. A misclassification $C(x)$ will contribute to irreducible error if $C(x) = C_{X,Y}^{Bayes}(x)$. Any misclassification that does not contribute to irreducible error instead contributes to reducible error. Bias equates to

misclassifications where $C(x)$ contributes to reducible error and $C(x) = C^{\circ}_{\mathcal{L},\mathcal{T}}(x)$. Any reducible error that does not contribute to bias instead contributes to variance.

$$bias_B = P_{(Y,X),\mathcal{T}}\left(\mathcal{L}(\mathcal{T})(X) \neq Y \wedge \mathcal{L}(\mathcal{T})(X) \neq C^{Bayes}_{Y,X}(X) \wedge \mathcal{L}(\mathcal{T})(X) = C^{\circ}_{\mathcal{L},\mathcal{T}}(X)\right) \tag{A.9}$$

$$variance_B = P_{(Y,X),\mathcal{T}}\left(\mathcal{L}(\mathcal{T})(X) \neq Y \wedge \mathcal{L}(\mathcal{T})(X) \neq C^{Bayes}_{Y,X}(X) \wedge \mathcal{L}(\mathcal{T})(X) \neq C^{\circ}_{\mathcal{L},\mathcal{T}}(X)\right) \tag{A.10}$$

This decomposition cleanly captures the concept of bias measuring error due to central tendency and variance measuring error due to deviations from the central tendency. This differs substantially from Kohavi and Wolpert (1996) and Friedman's (1997) definitions of variance, which measure the frequency with which predictions of individual learned classifiers differ from the central tendency, irrespective of whether those deviations result in correct or erroneous classifications. It is accepted, however, that this latter approach to variance is more consistent with definitions of bias and variance for numeric regression.

Nonetheless, there remains the practical obstacle to the application of Breiman's definitions (other than to artificial domains for which this factor can be controlled) that its evaluation requires knowledge, or at least estimation, of the irreducible error. To circumvent this difficulty, the current research uses a less complex decomposition that distributes the irreducible error across the bias and variance terms. This is similar in principle to Bauer and Kohavi's (in press) practice of aggregating irreducible error with the bias term, but is more natural in the context of Breiman's (1996b) formulation. This results in the following definitions.

$$bias = P_{(Y,X),\mathcal{T}}\left(\mathcal{L}(\mathcal{T})(X) \neq Y \wedge \mathcal{L}(\mathcal{T})(X) = C^{\circ}_{\mathcal{L},\mathcal{T}}(X)\right) \tag{A.11}$$

$$variance = P_{(Y,X),\mathcal{T}}\left(\mathcal{L}(\mathcal{T})(X) \neq Y \wedge \mathcal{L}(\mathcal{T})(X) \neq C^{\circ}_{\mathcal{L},\mathcal{T}}(X)\right) \tag{A.12}$$

These definitions have the desirable properties that

- bias is a direct measure of the contribution of the central tendency to total error;
- variance is a direct measure of the contribution to error of deviations from the central tendency; and
- $error = bias + variance$.

It does, however, differ markedly from the form of bias/variance analysis used for numeric regression. In recognition of this shift, these bias and variance terms will be referred to as *contribution of bias to error* and *contribution of variance to error*.

These two terms fail to directly measure the degree of variance irrespective of the target distribution, which is of interest because it sets a lower limit on the error

that may be achieved by a learner that is independent of the target function. Nor does it capture the accuracy of the central tendency. As (A.4) and (A.5) provide better measures for the former purpose and (A.7) provides a better measure for the latter purpose, they will be used in conjunction with (A.11) and (A.12) to provide a detailed profile of the bias/variance behavior of the algorithms. Note that due to the difficulty of estimating intrinsic noise, following the practice of Kohavi and Wolpert (1996), the value reported for $bias_{KW}$ is the sum of (A.6) and (A.4).

Our primary interest will be in the impact on bias and variance of different committee learning algorithms. As the irreducible error component of (A.11) and (A.12) will be constant across all learners, its aggregation into the bias and variance terms will not affect our conclusions. Such aggregation does, however, greatly simplify and increase the accuracy of the estimation of these terms during experimentation.

Appendix B

A bias/variance estimation method

The definitions of bias and variance all refer to distributions about which the only knowledge we have in a standard machine learning experimental situation is a small sample from the distribution. In consequence, it is necessary to estimate these terms.

Bauer and Kohavi (in press) estimate bias and variance using a two stage process. First, the available data D are randomly divided into a training pool TP_1 (2/3) and a test set TS_1 (the remaining 1/3). Then ten training sets $T_1^1 \ldots T_1^{10}$ are formed by random selection, each containing 1/2 of the objects from TP_1. This process is repeated two more times, with different random TP_2, TS_2 and TP_3, TS_3 selections, to produce training sets $T_2^1 \ldots T_2^{10}$ and $T_3^1 \ldots T_3^{10}$. The bias and variance terms are then estimated by evaluation of the predictions of $\mathcal{L}(T_i^1) \ldots \mathcal{L}(T_i^{10})$ when applied to TS_i, for $i \in \{1, 2, 3\}$. Of particular relevance to estimation of (A.11) and (A.12), the central tendency $C^{\circ}_{\mathcal{L},\mathcal{T}}(x)$ for an object $x \in TS_i$ is estimated by

$$\underset{y}{argmax} \left(\sum_{j=1}^{10} 1 \left[\mathcal{L}(T_i^j)(x) = y \right] \right). \tag{B.1}$$

A similar approach has been employed in the current research. However, instead of the use of training sets selected from a pool of training objects, which requires the use of relatively few (in the method described above, only 1/3) of the available objects for each learning session, three-fold cross-validation has been used. By this technique, the available data D are divided into three folds, f_1, f_2, and f_3. This process is repeated ten times producing thirty folds, $f_1^1, f_2^1, f_3^1, f_1^2, \ldots f_3^{10}$. Each of the folds from a triple is used as a test set once for a classifier learned from the other two folds in the triple. For convenience we will use the abbreviations $T_1^i = f_2^i \cup f_3^i$, $T_2^i = f_1^i \cup f_3^i$, and $T_3^i = f_1^i \cup f_2^i$. Using this approach, $C^{\circ}_{\mathcal{L},\mathcal{T}}(x)$ for an object $x \in D$ is estimated by

$$\underset{y}{argmax} \left(\sum_{i=1}^{10} \sum_{j=1}^{3} 1 \left[x \in f^i \wedge \mathcal{L}(T_j^i)(x) = y \right] \right). \tag{B.2}$$

The use of cross-validation in this manner has the advantage that every available example is used the same number of times, both for training and for testing. In contrast, Bauer and Kohavi (in press), by the use of random selection of cases, may use different cases different numbers of times in a given role. This random process may dramatically effect the results obtained. Cross validation is performed ten times in order to average out the effect of another single random event that may substantially affect the results—the division of cases into folds. The variance between individual cross-validation trials can be very high. Averaging over a number of cross-validation trials gives a more accurate estimation of the average case performance of an algorithm for learning from training sets of the specified size drawn from the available data.

Once the cross-validation trials have been completed, the relevant measures are estimated directly from the observed distributions of results.

Appendix C

Detailed results

Tables C.1 to C.3 present the mean error, contribution of bias to error, and contribution of variance to error for each algorithm on each data set. Tables of values are presented rather than graphical representations of relative performance as these allow more precise analysis of comparative performance. Note that due to rounding of values to three decimal places, in some cases the reported values for bias and variance do not sum precisely to the reported value for error.

Notes

1. Krogh and Vedelsby's (1995) work actually relates to ensembles for numeric regression, but the principle also applies to committees for classification.
2. The continuous Poisson distribution is more commonly known as the exponential distribution.
3. The current implementation differs from strict wagging of boosted sub-committees by using equal instance weights for the first sub-committee. This was done in the belief that boosting with the training set unmodified would out-perform boosting from a wagged distribution, and hence it would be beneficial to include this stronger sub-committee in the final committee.
4. As arithmetic mean error ratios are usually reported rather than geometric mean error ratios, it is interesting to consider whether the use of this non-standard statistic is affecting the results. At $t = 10$ the arithmetic means of error ratio for MultiBoost against AdaBoost, bagging, and wagging are 0.982, 0.941, and 0.903. At $t = 100$ they are 0.990, 0.884, and 0.864, respectively. As can be seen, MultiBoost enjoys an advantage irrespective of whether arithmetic or geometric means are considered.

Table C.1. Mean error for each algorithm

		$t = 10$				$t = 100$			
Data Set	**C4.5**	**Bag**	**Wag**	**AdaB**	**MultB**	**Bag**	**Wag**	**AdaB**	**MultB**
adult	0.140	0.141	0.140	0.163	0.153	0.139	0.137	0.151	0.148
anneal	0.096	0.072	0.074	0.059	0.050	0.074	0.076	0.052	0.051
audio	0.255	0.229	0.254	0.196	0.204	0.232	0.247	0.200	0.194
autos	0.232	0.224	0.228	0.182	0.185	0.206	0.220	0.168	0.180
balance-scale	0.216	0.180	0.177	0.210	0.189	0.162	0.163	0.225	0.199
br. cancer Slov.	0.284	0.273	0.280	0.337	0.324	0.274	0.279	0.348	0.315
br. cancer Wisc.	0.056	0.042	0.043	0.038	0.041	0.038	0.038	0.034	0.035
Cleveland	0.242	0.213	0.209	0.226	0.204	0.195	0.197	0.207	0.192
credit (Aust.)	0.149	0.137	0.141	0.162	0.149	0.141	0.142	0.140	0.136
credit (German)	0.283	0.266	0.263	0.288	0.280	0.255	0.254	0.256	0.252
discordant	0.012	0.012	0.012	0.012	0.010	0.012	0.012	0.012	0.012
echocardiogram	0.276	0.270	0.273	0.291	0.297	0.264	0.268	0.280	0.262
glass	0.350	0.279	0.290	0.283	0.281	0.268	0.284	0.253	0.249
heart	0.233	0.212	0.204	0.218	0.210	0.188	0.196	0.203	0.197
hepatitis	0.203	0.194	0.202	0.181	0.175	0.180	0.181	0.162	0.169
horse-colic	0.164	0.165	0.162	0.195	0.168	0.171	0.168	0.199	0.169
house-votes-84	0.052	0.047	0.050	0.051	0.047	0.047	0.048	0.048	0.045
Hungarian	0.216	0.194	0.203	0.213	0.193	0.189	0.193	0.208	0.201
hypo	0.005	0.004	0.005	0.004	0.004	0.005	0.005	0.004	0.003
iris	0.057	0.056	0.063	0.063	0.063	0.057	0.058	0.058	0.057
kr-vs-kp	0.008	0.008	0.009	0.006	0.005	0.008	0.009	0.005	0.005
labor-neg	0.179	0.151	0.165	0.160	0.172	0.147	0.154	0.135	0.144
lenses	0.192	0.246	0.300	0.258	0.275	0.287	0.287	0.287	0.300
letter-recognition	0.137	0.090	0.091	0.056	0.062	0.071	0.075	0.035	0.037
lymphography	0.218	0.208	0.207	0.214	0.207	0.205	0.206	0.184	0.174
new-thyroid	0.085	0.066	0.085	0.055	0.059	0.059	0.069	0.060	0.051
phoneme	0.210	0.203	0.205	0.179	0.180	0.201	0.203	0.167	0.166
Pima diabetes	0.264	0.251	0.243	0.271	0.263	0.241	0.234	0.262	0.244
primary tumor	0.612	0.588	0.585	0.618	0.599	0.582	0.582	0.587	0.581
promoters	0.258	0.175	0.186	0.157	0.122	0.153	0.161	0.090	0.086
segment	0.040	0.033	0.035	0.023	0.026	0.031	0.033	0.018	0.020
sick	0.014	0.014	0.015	0.012	0.012	0.014	0.014	0.011	0.012
soybean large	0.111	0.085	0.093	0.078	0.077	0.084	0.085	0.079	0.070
splice junction	0.065	0.062	0.060	0.057	0.057	0.056	0.057	0.047	0.045
tic-tac-toe	0.165	0.118	0.128	0.051	0.070	0.093	0.100	0.013	0.020
waveform	0.280	0.231	0.232	0.219	0.216	0.196	0.207	0.178	0.193

Table C.2. Mean contribution of bias to error for each algorithm

		$t = 10$				$t = 100$			
Data Set	**C4.5**	**Bag**	**Wag**	**AdaB**	**MultB**	**Bag**	**Wag**	**AdaB**	**MultB**
adult	0.124	0.121	0.122	0.115	0.112	0.125	0.125	0.120	0.121
anneal	0.067	0.045	0.047	0.039	0.031	0.053	0.053	0.038	0.038
audio	0.154	0.155	0.160	0.116	0.118	0.173	0.180	0.123	0.122
autos	0.109	0.128	0.132	0.085	0.094	0.120	0.125	0.105	0.102
balance-scale	0.111	0.087	0.092	0.132	0.107	0.093	0.100	0.151	0.125
br. cancer Slov.	0.240	0.251	0.245	0.233	0.228	0.242	0.247	0.249	0.244
br. cancer Wisc.	0.034	0.032	0.032	0.027	0.029	0.032	0.030	0.029	0.026
Cleveland	0.157	0.156	0.143	0.143	0.133	0.148	0.139	0.158	0.145
credit (Aust.)	0.110	0.113	0.120	0.110	0.109	0.123	0.120	0.111	0.111
credit (German)	0.202	0.201	0.211	0.182	0.185	0.212	0.209	0.197	0.201
discordant	0.009	0.009	0.010	0.008	0.007	0.010	0.009	0.010	0.010
echocardiogram	0.220	0.184	0.177	0.177	0.211	0.208	0.197	0.178	0.182
glass	0.191	0.179	0.192	0.164	0.184	0.194	0.204	0.170	0.166
heart	0.146	0.136	0.139	0.145	0.145	0.144	0.147	0.154	0.158
hepatitis	0.140	0.147	0.157	0.121	0.123	0.155	0.155	0.117	0.130
horse-colic	0.139	0.144	0.147	0.132	0.134	0.155	0.151	0.140	0.138
house-votes-84	0.044	0.039	0.040	0.039	0.033	0.043	0.043	0.035	0.034
Hungarian	0.170	0.165	0.161	0.155	0.145	0.163	0.170	0.166	0.165
hypo	0.004	0.003	0.004	0.002	0.003	0.004	0.004	0.003	0.002
iris	0.045	0.046	0.046	0.045	0.049	0.047	0.044	0.049	0.048
kr-vs-kp	0.006	0.006	0.006	0.003	0.003	0.005	0.006	0.003	0.003
labor-neg	0.123	0.104	0.121	0.091	0.112	0.116	0.130	0.084	0.118
lenses	0.121	0.108	0.117	0.154	0.158	0.158	0.150	0.204	0.242
letter-recognition	0.040	0.042	0.044	0.021	0.024	0.043	0.046	0.021	0.022
lymphography	0.172	0.150	0.160	0.134	0.116	0.145	0.155	0.116	0.130
new-thyroid	0.052	0.034	0.054	0.037	0.039	0.042	0.053	0.034	0.033
phoneme	0.159	0.146	0.149	0.111	0.112	0.150	0.151	0.111	0.114
Pima diabetes	0.180	0.202	0.193	0.186	0.193	0.206	0.200	0.205	0.206
primary tumor	0.361	0.385	0.373	0.349	0.343	0.400	0.413	0.371	0.391
promoters	0.113	0.089	0.100	0.048	0.049	0.067	0.070	0.033	0.046
segment	0.017	0.021	0.022	0.012	0.013	0.020	0.022	0.012	0.013
sick	0.009	0.010	0.011	0.008	0.007	0.010	0.011	0.009	0.009
soybean large	0.061	0.055	0.058	0.047	0.049	0.057	0.059	0.057	0.048
splice junction	0.049	0.047	0.049	0.035	0.039	0.047	0.047	0.036	0.038
tic-tac-toe	0.067	0.067	0.067	0.011	0.015	0.054	0.060	0.006	0.007
waveform	0.150	0.142	0.162	0.130	0.134	0.147	0.154	0.132	0.147

Table C.3. Mean contribution of variance to error for each algorithm

		$t = 10$				$t = 100$			
Data Set	**C4.5**	**Bag**	**Wag**	**AdaB**	**MultB**	**Bag**	**Wag**	**AdaB**	**MultB**
adult	0.015	0.020	0.017	0.048	0.041	0.013	0.012	0.031	0.027
anneal	0.029	0.027	0.028	0.019	0.019	0.022	0.023	0.014	0.013
audio	0.101	0.073	0.094	0.080	0.086	0.059	0.067	0.076	0.072
autos	0.123	0.097	0.096	0.097	0.091	0.086	0.094	0.063	0.078
balance-scale	0.105	0.092	0.084	0.079	0.082	0.069	0.064	0.074	0.074
br. cancer Slov.	0.044	0.022	0.035	0.104	0.096	0.033	0.032	0.099	0.071
br. cancer Wisc.	0.023	0.010	0.010	0.011	0.012	0.005	0.007	0.005	0.009
Cleveland	0.084	0.057	0.066	0.083	0.071	0.048	0.058	0.049	0.048
credit (Aust.)	0.039	0.024	0.021	0.052	0.040	0.018	0.022	0.029	0.025
credit (German)	0.081	0.064	0.052	0.106	0.095	0.044	0.045	0.059	0.052
discordant	0.003	0.003	0.002	0.004	0.003	0.002	0.002	0.002	0.002
echocardiogram	0.055	0.055	0.070	0.101	0.080	0.086	0.096	0.114	0.086
glass	0.159	0.073	0.080	0.083	0.082	0.100	0.098	0.119	0.098
heart	0.087	0.044	0.049	0.049	0.039	0.076	0.065	0.073	0.065
hepatitis	0.063	0.047	0.045	0.060	0.053	0.025	0.026	0.045	0.039
horse-colic	0.025	0.021	0.015	0.062	0.034	0.016	0.017	0.059	0.031
house-votes-84	0.008	0.008	0.010	0.013	0.014	0.004	0.005	0.013	0.011
Hungarian	0.046	0.029	0.042	0.058	0.048	0.027	0.023	0.043	0.036
hypo	0.001	0.001	0.001	0.002	0.001	0.001	0.001	0.001	0.001
iris	0.012	0.010	0.017	0.017	0.013	0.010	0.014	0.009	0.009
kr-vs-kp	0.002	0.002	0.002	0.003	0.002	0.003	0.003	0.002	0.002
labor-neg	0.056	0.047	0.044	0.068	0.060	0.032	0.025	0.051	0.026
lenses	0.071	0.138	0.183	0.104	0.117	0.129	0.138	0.083	0.058
letter-recognition	0.097	0.049	0.047	0.035	0.038	0.028	0.029	0.014	0.015
lymphography	0.046	0.058	0.047	0.080	0.091	0.060	0.051	0.069	0.043
new-thyroid	0.033	0.032	0.031	0.018	0.020	0.017	0.015	0.025	0.018
phoneme	0.051	0.057	0.056	0.067	0.068	0.051	0.052	0.056	0.053
Pima diabetes	0.084	0.049	0.050	0.085	0.070	0.035	0.034	0.057	0.039
primary tumor	0.251	0.203	0.212	0.269	0.256	0.182	0.169	0.216	0.190
promoters	0.145	0.087	0.086	0.108	0.073	0.086	0.092	0.057	0.040
segment	0.022	0.013	0.013	0.010	0.013	0.011	0.011	0.006	0.007
sick	0.005	0.004	0.004	0.004	0.004	0.003	0.003	0.003	0.003
soybean large	0.051	0.030	0.035	0.031	0.028	0.027	0.026	0.023	0.021
splice junction	0.016	0.015	0.012	0.022	0.019	0.008	0.009	0.011	0.008
tic-tac-toe	0.099	0.051	0.062	0.040	0.055	0.038	0.039	0.007	0.012
waveform	0.130	0.089	0.071	0.089	0.082	0.049	0.053	0.046	0.046

References

Ali, K., Brunk, C., & Pazzani, M. (1994). On learning multiple descriptions of a concept. *Proceedings of Tools with Artificial Intelligence* (pp. 476–483) New Orleans, LA.

Bauer, E., & Kohavi, R. (in press). An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*.

Blake, C., Keogh, E., & Merz, C. J. (1999). UCI repository of machine learning databases. [Machine-readable data repository]. University of California, Department of Information and Computer Science, Irvine, CA.

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

Breiman, L. (1996b). Bias, variance, and arcing classifiers. (Technical report 460). Berkeley, CA: University of California: Department of Statistics.

Breiman, L. (1997). Arcing the edge. (Technical report 486). Berkeley, CA: University of California: Department of Statistics.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and Regression Trees*. Belmont, CA: Wadsworth International.

Dietterich, T. G. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, *10*(7), 1895–1923.

Freund, Y., & Schapire, R. E. (1996). Experiments with a new boosting algorithm. *Proceedings of the Thirteenth International Conference on Machine Learning* (148–156) Bari, Italy: Morgan Kaufmann.

Freund, Y., & Schapire, R. E. (1995). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, *55*, 119–139.

Friedman, J., Hastie, T., & Tibshirani, R. (to appear). Additive logistic regression: A statistical view of boosting. *Annals of Statistics*.

Friedman, J. H. (1997). On bias, variance, 0/1-loss, and the curse-of-dimensionality. *Data Mining and Knowledge Discovery*, *1*, 55–77.

Geman, S., Bienenstock, E., & Doursat, R. (1992). Neural networks and the bias/variance dilemma. *Neural Computation*, *4*, 1–48.

Kohavi, R., & Wolpert, D. (1996). Bias plus variance decomposition for zero-one loss functions. *Proceedings of the 13th International Conference on Machine Learning* (275–283) Bari, Italy: Morgan Kaufmann.

Kong, E. B., & Dietterich, T. G. (1995). Error-correcting output coding corrects bias and variance. *Proceedings of the Twelfth International Conference on Machine Learning* (pp. 313–321). Tahoe City, CA: Morgan Kaufmann.

Krogh, A., & Vedelsby, J. (1995). Neural network ensembles, cross validation, and active learning. G. Tesauro, D. Touretzky, & T. Leen (Eds.), *Advances in Neural Information Processing Systems*, Vol. 7. Boston, MA: MIT Press.

Nock, R., & Gascuel, O. (1995). On learning decision committees. *Proceedings of the Twelfth International Conference on Machine Learning* (pp. 413–420). Taho City, CA: Morgan Kaufmann.

Oliver, J. J., & Hand, D. J. (1995). On pruning and averaging decision trees. *Proceedings of the Twelfth International Conference on Machine Learning* (pp. 430–437). Taho City, CA: Morgan Kaufmann.

Quinlan, J. R. (1996). Bagging, boosting, and C4.5. *Proceedings of the Thirteenth National Conference on Artificial Intelligence* (pp. 725–730). AAAI/MIT Press.

Rao, R. B., Gordon, D., & Spears, W. (1995). For every generalization action is there really an equal and opposite reaction? Analysis of the conservation law for generalization performance. *Proceedings of the Twelfth International Conference on Machine Learning* (pp. 471–479). Taho City, CA: Morgan Kaufmann.

Salzberg, S. L. (1997). On comparing classifiers: Pitfalls to avoid and a recommended approach. *Data Mining and Knowledge Discovery*, *1*, 317–327.

Schaffer, C. (1994). A conservation law for generalization performance. *Proceedings of the 1994 International Conference on Machine Learning* Morgan Kaufmann.

Schapire, R. E., Freund, Y., Bartlett, P., & Lee, W. S. (1998). Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics*, *26*, 1651-1686.

Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, *5*, 241–259.

Wolpert, D. H. (1995). *Off-training set error and a priori distinctions between learning algorithms.* (Technical Report SFI TR 95-01-003). Santa Fe, NM: The Santa Fe Institute.

Zheng, Z., & Webb, G. I. (1998). Multiple boosting: A combination of boosting and bagging. *Proceedings of the 1998 International Conference on Parallel and Distributed Processing Techniques and Applications* (pp. 1133–1140). CSREA Press.

Received September 15, 1998
Accepted March 17, 1999
Final manuscript June 4, 1999