Pre-publication draft of a paper which appeared in the Proceedings of PAKDD'99 pp 123-132 Publisher: Springer-Verlag

Stochastic Attribute Selection Committees with Multiple Boosting: Learning More Accurate and More Stable Classifier Committees

Zijian Zheng and Geoffrey I. Webb

School of Computing and Mathematics
Deakin University, Geelong
Victoria 3217, Australia

{zijian,webb}@deakin.edu.au

Technical Report (TR C98/13)
May, 1998

ABSTRACT: Approaches to learning classifier committees, including Boosting, Bagging, SASC, and SASCB, have demonstrated great success in increasing the prediction accuracy of decision trees. This type of technique generates several classifiers to form a committee by repeated application of a single base learning algorithm. The committee members vote to decide the final classification. Boosting and Bagging create different classifiers by modifying the distribution of the training set. SASC (Stochastic Attribute Selection Committees) adopts a different method. It generates committees by stochastic manipulation of the set of attributes considered at each node during tree induction, but keeping the distribution of the training set unchanged. SASCB, a combination of Boosting and SASC, has been shown to be able to further increase, on average, the prediction accuracy of decision trees. It has been found that the performance of SASCB and Boosting is more variable than that of SASC, although SASCB is more accurate than the others on average. In this paper, we present a novel method to reduce variability of SASCB and Boosting, and further increase their average accuracy. It generates multiple committees by incorporating Bagging into SASCB. As well as improving stability and average accuracy, the resulting method is amenable to parallel or distributed processing, while Boosting and SASCB are not. This is an important characteristic for datamining in large datasets.

Keywords: committee learning, boosting, decision tree learning, classification learning, data mining, machine learning.

1 Introduction

For classifier learning, a key technique in KDD, prediction accuracy and computational requirements are two primary concerns. To increase the prediction accuracy of classifiers, classifier committee[1] learning techniques have been developed with great success (Freund 1996; Freund and Schapire 1996a; 1996b; Quinlan 1996; Breiman 1996a; 1996b; Dietterich and Kong 1995; Ali 1996; Chan, Stolfo, and Wolpert 1996; Schapire, Freund, Bartlett, and Lee 1997; Domingos 1997; Bauer and Kohavi 1998), especially Boosting[2] (Freund and Schapire 1996b; Quinlan 1996; Bauer and Kohavi 1998). This type of technique generates several classifiers to form a committee by using a single base learning algorithm. At the classification stage, the committee members vote to make the final decision.

Given a training set described using a set of attributes, conventional classifier learning algorithms such as decision tree learning algorithms (Breiman, Friedman, Olshen, and Stone 1984; Quinlan 1993) build one classifier. Usually, the classifier is correct for most parts of the instance space, but incorrect for some small parts of the instance space. If classifiers in a committee partition the instance space differently, and most points in the instance space are correctly covered by the majority of the committee, then the committee has a lower error rate than the individual classifiers.

Bagging (Breiman 1996a) and Boosting (Schapire 1990; Freund and Schapire 1996a; 1996b; Freund 1996; Schapire *et al.* 1997), as two representative methods of this type, can significantly decrease the error rate of decision tree learning (Quinlan 1996; Freund and Schapire 1996b; Bauer and Kohavi 1998). They repeatedly build different classifiers using a base learning algorithm, such as a decision tree generator, by changing the distribution of the training set. Bagging generates different classifiers using different bootstrap samples. Boosting builds different classifiers sequentially. The weights of training examples used for creating each classifier are modified based on the performance of the previous classifiers. The objective is to make the generation of the next classifier concentrate on the training examples that are misclassified by the previous classifiers. The main difference between Bagging and Boosting is that the latter adaptively changes the distribution of the training set based on the performance of previously created classifiers and uses a function of the performance of a classifier as the weight for voting, while the former stochastically changes the distribution of the training set and uses equal weight voting. Although Boosting is generally more accurate than Bagging, the performance of Boosting is more variable than that of Bagging (Quinlan 1996; Bauer and Kohavi 1998).

As an alternative approach to generating different classifiers to form a committee, SASC (Stochastic Attribute Selection Committees) (Zheng and Webb 1998) builds different classifiers by modifying the set of attributes considered at each node, while the distribution of the training set is kept unchanged. Each attribute set is selected stochastically. Experiments show that SASC, like Boosting, can also significantly reduce the error rate

[1] Committees are also referred to as ensembles (Dietterich 1997).

[2] Breiman (1996b) refers to Boosting as the arcing (adaptively resample and combine) method.

of decision tree learning, although the two techniques use quite different mechanisms (Zheng and Webb 1998). In addition, SASC is more stable than Boosting (Zheng and Webb 1998).

There are some other classifier committee learning approaches such as generating multiple trees by manually changing learning parameters (Kwok and Carter 1990), error-correcting output codes (Dietterich and Bakiri 1995), and generating different classifiers by randomizing the base learning process (Dietterich and Kong 1995; Ali 1996) which is similar to SASC (Zheng and Webb 1998). Reviews of related methods are provided in Dietterich (1997) and Ali (1996). A collection of recent research in this area can be found from Chan *et al.* (1996).

Base on the observation that both Boosting and SASC can significantly increase the prediction accuracy of decision trees but through different mechanisms, we developed another technique to further improve the accuracy of decision trees (Zheng, Webb, and Ting 1998). The new approach is called **SASCB** (**S**tochastic **A**ttribute **S**election **C**ommittees with **B**oosting), a combination of the Boosting and SASC techniques. SASCB has been shown to be able to outperform, on average, either SASC or Boosting alone in terms of lower error rate. However, as Boosting, SASCB is more variable than SASC, due to that the Boosting component is the driving mechanism in the SASCB procedure.

As far as the computational requirements are concerned, each of SASC, SASCB, Boost, and Bagging needs approximately T times as long as their base learning algorithm does for learning a single classifier, where T is the size of the committee. However, SASC and Bagging have an advantage over SASCB and Boosting. That is, the former are amenable to parallel and distributed processing while the latter are not, since the generation of each committee member, a classifier, is independent for the former while it must occur sequentially for the latter. This makes SASC and Bagging faster than SASCB and Boosting when multiple processors or computers are available.

In summary, we have the following three observations from the previous studies on committee learning. SASCB, as a combination of Boosting and SASC, can generally outperform Boosting and SASC. SASCB is more variable than SASC. Bagging is less variable than Boosting. In the light of these findings, in this paper, we present a novel approach, namely **SASCMB** (**S**tochastic **A**ttribute **S**election **C**ommittees with **M**ultiple **B**oosting), to improving the stability and average accuracy of SASCB and Boosting. It generates multiple committees by incorporating Bagging into SASCB using the multi-boosting technique (Webb 1998). We expect that splitting one committee into multiple committees, with each committee being created from a bootstrap sample of the training set, can reduce the variability of Boosting and SASCB, since the Boosting process is broken down into several small processes. In addition, we expect that introducing Bagging can further improve the accuracy, since it increases the diversity and independence of committee members. At the same time, the new algorithm is amenable to parallel and distributed processing.

In the following section, we briefly describe the Boosting, SASC, and SASCB techniques for decision tree learning. Section 2.3 presents the method of incorporating Bagging into

SascB. It is, then, empirically evaluated using a representative collection of natural domains. Finally, we summarize our conclusions.

2 Boosting, Sasc, and SascB

Since the SascMB technique is a combination of Bagging and SascB which is, in turn, a combination of Boosting and Sasc, we briefly discuss Boosting, Sasc, and SascB in this section. The classification process of them is presented in Section 2.4, since it is the same for all of them. For details about Boosting, see Schapire (1990), Quinlan (1996), Schapire *et al.* (1997), and Bauer and Kohavi (1998); for Sasc, see Zheng and Webb (1998); for SascB, see Zheng *et al.* (1998).

2.1 Boosting

Boosting is a general framework for improving base learning algorithms, such as decision tree learning, rule learning, and neural networks. The key idea of Boosting was presented in Section 1. Here, we describe our implementation of the Boosting algorithm with decision tree learning, called Boost. It follows the Boosted C4.5 algorithm (AdaBoost.M1) (Quinlan 1996) but uses a new Boosting equation as shown in Equation 1, derived from Schapire *et al.* (1997).

Given a training set D consisting of m instances and an integer T, the number of trials, Boost builds T pruned trees over T trials by repeatedly invoking C4.5 (Quinlan 1993). Let $w_t(x)$ denote the weight of instance x in D at trial t. At the first trial, each instance has weight 1; that is, $w_1(x) = 1$ for each x. At trial t, decision tree H_t is built using D under the distribution w_t. The error ϵ_t of H_t is, then, calculated by summing up the weights of the instances that H_t misclassifies and divided by m. If ϵ_t is greater than 0.5 or equal to 0, $w_t(x)$ is re-initialized using bootstrap sampling, and then the Boosting process continues. Note that the tree with $\epsilon_t > 0.5$ is discarded,[3] while the tree with $\epsilon_t = 0$ is accepted by the committee. Otherwise, the weight $w_{t+1}(x)$ of each instance x for the next trial is computed using Equation 1. These weights are, then, renormalized so that they sum to m.

$$w_{(t+1)}(x) = w_t(x) exp((-1)^{d(x)} \alpha_t), \tag{1}$$

where $\alpha_t = \frac{1}{2} ln((1-\epsilon_t)/\epsilon_t)$; $d(x) = 1$ if H_t correctly classifies x and $d(x) = 0$ otherwise.

[3] To make the algorithm efficient, this step is limited to $10 \times T$ times.

2.2 SASC

During the growth of a decision tree, at each decision node, a decision tree learning algorithm searches for the best attribute to form a test based on some test selection functions (Quinlan 1993). The key idea of SASC is to vary the members of a decision tree committee by stochastic manipulation of the set of attributes available for selection at decision nodes. This creates decision trees that each partition the instance space differently. In order to have a good quality tree in the sense that it can correctly cover most parts of the instance space, the tests used at decision nodes should be as good as possible with respect to the test selection function employed.

We use C4.5 (Quinlan 1993) with the modifications described below as the base classifier learning algorithm in SASC. When building a decision node, by default C4.5 uses the information gain ratio to search for the best attribute to form a test (Quinlan 1993). To force C4.5 to generate different trees using the same training set, we modified C4.5 by stochastically restricting the set of attributes available for selection at a decision node. This is implemented by using a probability parameter P.[4] At each decision node, an attribute subset is randomly selected with each available attribute having the probability P of being selected. The available attributes refer to those attributes that have non-negative gain values. For nominal attributes, they must not have been used in the path from the root to the current node. Numeric attributes are always available for selection. This stochastic attribute subset selection process will be repeated, if no attribute was selected and there are some attributes available at this node. The objective is to make sure that at least one available attribute is included in the subset if possible. After attribute subset selection, the algorithm chooses the attribute with the highest gain ratio to form a test for the decision node from the subset. A different random subset is selected at each node of the tree. The modified version of C4.5 is called **C4.5SAS** (**C4.5** **S**tochastic **A**ttribute **S**election).

The only difference between C4.5SAS and C4.5 is that when growing a tree, at a decision node, C4.5SAS stochastically creates an attribute subset and uses the best attribute in it to form a test as described above. All other parts are identical for these two algorithms. With $P = 1$, C4.5SAS generates the same tree as C4.5.

Having C4.5SAS, the design of SASC is very simple. C4.5SAS is invoked T times to generate T different decision trees to form a committee. As in Boosting, the first tree produced by SASC is the same as the tree generated by C4.5. The detailed description of C4.5SAS and SASC can be found in Zheng and Webb (1998).

2.3 SASCB

The objective of combining Boosting and SASC to form SASCB is to combine the advantages of both Boosting and SASC when generating different decision trees that partition the instance space differently. The combination strategy adopted in SASCB employs,

[4] The value of P does not change during induction.

when generating decision trees, both the stochastic selection of attribute subsets of SASC and the adaptive modification of the distribution of the training set of Boosting.

SASCB uses the same Boosting procedure as BOOST except that C4.5SAS, described in the previous subsection, is used instead of C4.5 as the base tree generator. Another difference is that when the weighted error rate of a tree is greater than 0.5 or equal to 0, SASCB does not change instance weights. Another tree will be built using the same distribution of the training set. Since the stochastic attribute subset selection process is involved, the tree generated this time should be different from the one created previously even though the same distribution of the training set is used. As in BOOST, the tree with an error rate greater than 0.5 is discarded,[5] while the tree with no errors on the training set is kept.

SASCB can be considered as introducing the stochastic attribute selection process into the generation of each tree in the Boosting process. It can also be thought of as adaptively modifying the distribution of the training set after the generation of each decision tree in the SASC process.

2.4 Decision Making in BOOST, SASC, and SASCB

At the classification stage, for a given example, all of BOOST, SASC, and SASCB make the final prediction through committee voting. In this paper, a voting method that uses the probabilistic predictions produced by all committee members without voting weights is adopted. With this method, each decision tree returns a distribution over classes that the example belongs to. This is performed by tracing the example down to a leaf of the tree. The class distribution for the example is estimated using the proportion of the training examples of each class at the leaf, if the leaf is not empty. This is the same as in C4.5 (Quinlan 1993). When the leaf contains no training examples, C4.5 produces a class distribution with the labeled class of the leaf having the probability 1, and all other classes having the probability 0. In this case, the three committee learning algorithms are different from C4.5. They estimate the class distribution using the training examples at the parent node of the empty leaf. The decision tree committee members vote by summing up the class distributions provided by all trees. The class with the highest score (sum of probabilities) wins the voting, and serves as the predicted class of BOOST, SASC, and SASCB for this example. There is no voting weight when summing up class distributions. Class distribution provides more detailed information than is obtained when each committee member votes for a single class, and this information is meaningful for committee voting.

There are three other approaches to voting. One is using the categorical predictions provided by all trees, without voting weights. In this case, each tree produces a single predicted class for an example. Then, the committee members vote by predicting the most frequent class returned by all trees.

[5] This step is also limited to $10 \times T$ times, where T is the number of Boosting trials.

SascMB(Att, D, P, S, N)
INPUT: Att: a set of attributes,
D: a training set represented using Att and classes,
P: a probability value,
S: the size of each subcommittee,
N: the number of subcommittees.
OUTPUT: a committee, H, consisting of N subcommittees with each containing S trees.
Set T, the number of trials, $= S \times N$
Set instance weight $w_1(x) = 1$ for each x in D
H_1 := **C4.5Sas**(Att, D, w_1, 1)
t := 2
WHILE ($t <= T$)
{ D' := training cases in D that are misclassified by $H_{(t-1)}$
$\epsilon_{(t-1)} = \frac{1}{|D|} \sum_{x \in D'} w_{(t-1)}(x)$
IF ($\epsilon_{(t-1)} > 1/2$)
$t := t - 1$
ELSE IF (t modulus $S = 1$)
Reset $w_t(x)$ using bootstrap sampling, i.e., $w_t(x)$ is set at 0 and incremented 1 unit every time instance x is selected during uniformly sampling $|D|$ instances from D with replacement
ELSE IF ($\epsilon_{(t-1)} \neq 0$)
Calculate $w_t(x)$, the weight of each x in D, from $w_{(t-1)}(x)$ using Equation 1 and renormalize these weights so that they sum to $|D|$
H_t := **C4.5Sas**(Att, D, w_t, P)
$t := t + 1$
}
RETURN H

Figure 1: The SascMB learning algorithm

The other two methods are the same as the two mentioned above but each tree is given a weight α_t for voting, which is a function of the performance of the decision tree on the training set, and is defined in Equation 1. The last of these alternatives, weighted voting of categorical predictions, corresponds to the original AdaBoost.M1. These three voting methods perform either worse than or similarly to the method that we use here (Zheng and Webb 1998; Zheng *et al.* 1998).

3 SascMB: Incorporating Bagging into SascB

Figure 1 presents the details of the SascMB algorithm. It is resulted from incorporating Bagging into SascB. SascMB generates N subcommittees. This process can be parallelized. Each subcommittee contains S decision trees built using the SascB procedure described in the previous section. The generation of the first subcommittee (or one of the

subcommittees if using parallel or distributed processing) starts from the initial training set D with each training instance having the weight 1. The first tree in this subcommittee is the same one as that built by C4.5 using the entire training set. The generation of every other subcommittee starts from a bootstrap sample of D. A bootstrap sample is created by uniformly sampling $|D|$ instances from D with replacement.

At the classification stage, all the members of all the subcommittees generated by SascMB vote to predict a class for a given instance. SascMB uses the same default voting method as Boost, Sasc, and SascB, since it generally performs better than the other three voting approaches (Zheng and Webb 1998; Zheng *et al.* 1998) as mentioned in Section 2.4.

4 Experiments

In this section, we empirically evaluate SascMB to examine whether incorporating Bagging into SascB can increase stability and average accuracy of learned committees. It is compared with other committee learning algorithms: SascB, Boost, and Sasc. In addition, a multiple Boosting algorithm MB is also included in the comparison. C4.5, the base decision tree learning algorithm of all these committee learning algorithms, is used as the base line for the comparison.

MB is the same as SascMB except that it does not include the stochastic attribute selection component. In other words, MB uses the same procedure as SascMB for generating multiple decision tree committees, but the former uses C4.5 instead of C4.5Sas. Another minor difference between them is that the instance weights are reset using bootstrap sampling after building a tree with a weighted error rate equal to 0 or greater than 0.5 for MB, since without the stochastic attribute selection component, MB creates the same tree on the same distribution of the training set. Note that Boosting cannot change instance weights under this condition. MB differs from Webb's (1998) MultiBoost by using bootstrap sampling in place of stochastic weighting at the start of the generation of each subcommittee. It is interesting to compare SascMB with MB.

4.1 Experimental Domains and Methods

Forty natural domains from the UCI machine learning repository (Merz and Murphy 1997) are used. They include all the domains used by Quinlan (1996) for studying Boosting. Table 1 summarizes the the characteristics of these domains, including dataset size, the number of classes, the number of numeric attributes, and the number of discrete attributes. This test suite covers a wide variety of different domains with respect to dataset size, the number of classes, the number of attributes, and types of attributes.

In every domain, two stratified 10-fold cross-validations (Kohavi 1995) were carried out for each algorithm. The result reported for each algorithm in each domain is an average value over 20 trials. All the algorithms are run on the same training and test set

Table 1: Description of learning tasks

Domain	Size	No. of Classes	No. of Att.	
			Numeric	Discr
Annealing	898	6	6	32
Audiology	226	24	0	69
Automobile	205	7	15	10
Breast cancer (W)	699	2	9	0
Chess (KR-KP)	3169	2	0	36
Chess (KR-KN)	551	2	0	39
Credit (Aust)	690	2	6	9
Credit (Ger)	1000	2	7	13
Echocardiogram	131	2	6	1
Glass	214	6	9	0
Heart (C)	303	2	13	0
Heart (H)	294	2	13	0
Hepatitis	155	2	6	13
Horse colic	368	2	7	15
House votes 84	435	2	0	16
Hypo	3772	5	7	22
Hypothyroid	3163	2	7	18
Image	2310	7	19	0
Iris	150	3	4	0
Labor	57	2	8	8
LED 24	200	10	0	24
Letter	20000	26	16	0
Liver disorders	345	2	6	0
Lung cancer	32	3	0	56
Lymphography	148	4	0	18
NetTalk(Letter)	5438	163	0	7
NetTalk(Phoneme)	5438	52	0	7
NetTalk(Stress)	5438	5	0	7
Pima	768	2	8	0
Postoperative	90	3	1	7
Primary tumor	339	22	0	17
Promoters	106	2	0	57
Sick	3772	2	7	22
Solar flare	1389	2	0	10
Sonar	208	2	60	0
Soybean	683	19	0	35
Splice junction	3177	3	0	60
Vehicle	846	4	18	0
Waveform-21	300	3	21	0
Wine	178	3	13	0

partitions with their default option settings. Pruned trees are used for all the algorithms. All BOOST, SASC, SASCB, MB, and SASCMB use probabilistic predictions (without voting weights) for voting to decide the final classification. Schapire *et al.* (1997) show that the test accuracy of Boosting increases as T increases even after the training error reaches zero. It is interesting to see the performance improvement that can be achieved with two orders of magnitude increase in computation. Therefore, the number of trials (the parameter T) is set at 100 in the experiments for BOOST, SASC, and SASCB. The subcommittee size and the number of subcommittees are set at 5 and 20 respectively, resulting in 100 trees in total for MB and SASCMB. The probability of each attribute being selected into the subset (the parameter P) is set at the default, 33%, for SASC, SASCB, and SASCMB.

4.2 Results

Table 2 shows the error rates of the six algorithms. To facilitate pairwise comparisons among the six algorithms, error ratios are derived from Table 2 and presented in Table 3. An error ratio, for example for BOOST vs C4.5, presents a result for BOOST divided by the corresponding result for C4.5 – a value less than 1 indicates an improvement due to BOOST. To compare the error rates of two algorithms in a domain, a two-tailed pairwise t-test on the error rates of the 20 trials is carried out. The difference is considered as significant, if the significance level of the t-test is better than 0.05. In Table 3, **boldface** (*italic*) font, for example for BOOST vs C4.5, indicates that BOOST is significantly more (less) accurate than C4.5. The last two rows in Table 3 present the numbers of wins, ties, and losses between the error rates of the corresponding two algorithms in the 40 domains, and the significance levels of a one-tailed pairwise sign-test on these win/tie/loss records.

From Tables 2 and 3, we have the following observations.

(1) Incorporating Bagging into SASCB can reduce variability of learned committees in terms of decreasing the frequency of producing significantly higher error rate than the base decision tree learning algorithm.

While both BOOST and SASCB obtain significantly higher error rates than C4.5 in five out of the 40 domains, SASCMB only has significantly higher error rates than C4.5 in two domains. The highest relative error increase of BOOST and SASCB over C4.5 is 61% and 38% respectively. It is 34% for SASCMB, the smallest one among the three algorithms. Note that SASC and MB are more stable than SASCMB, but they are less accurate than SASCMB on average (see below, for the discussion).

(2) Incorporating Bagging into SASCB can also reduce average error rate of learned committees. SASCMB outperforms BOOST, SASC, SASCB, and MB in terms of lower error rate.

All the five committee learning algorithms achieve significant error rate reduction over C4.5 at a level better than 0.0001 using a one-tailed pairwise sign-test on the error rates of these algorithms in the 40 domains. Among them, SASCMB obtains the lowest

Table 2: Error rates (%)

Domain	C4.5	Boost	Sasc	SascB	MB	SascMB
Annealing	7.40	4.90	5.85	4.12	4.67	5.06
Audiology	21.39	15.41	18.73	15.19	15.88	15.43
Automobile	16.31	13.42	14.35	15.88	16.10	16.82
Breast (W)	5.08	3.22	3.44	3.08	3.08	3.15
Chess (KR-KP)	0.72	0.36	0.67	0.36	0.39	0.39
Chess (KR-KN)	8.89	3.54	9.26	4.09	5.27	5.63
Credit (Aust)	14.49	13.91	14.71	14.20	12.82	12.61
Credit (Ger)	29.40	25.45	25.10	25.15	23.90	23.50
Echocardiogram	37.80	36.24	37.01	39.20	31.68	30.47
Glass	33.62	21.09	25.27	21.99	24.33	21.31
Heart (C)	22.07	18.80	16.65	18.63	18.13	18.29
Heart (H)	21.09	21.25	18.88	21.09	19.20	18.53
Hepatitis	20.63	17.67	18.40	15.79	17.12	17.12
Horse colic	15.76	19.84	17.39	19.43	15.90	16.04
House votes 84	5.62	4.82	4.59	4.25	3.90	4.25
Hypo	0.46	0.32	0.46	0.36	0.33	0.40
Hypothyroid	0.71	1.14	0.76	0.98	0.82	0.95
Image	2.97	1.58	2.06	1.58	1.77	1.93
Iris	4.33	5.67	5.00	5.67	5.00	5.00
Labor	23.67	10.83	18.83	9.83	12.33	10.50
LED 24	36.50	32.75	29.00	32.50	32.00	30.50
Letter	12.16	2.95	3.74	2.76	3.45	3.32
Liver disorders	35.36	28.88	29.90	29.47	26.73	27.29
Lung cancer	57.50	53.75	45.83	53.75	47.08	49.17
Lymphography	21.88	16.86	18.48	16.50	16.86	14.76
NetTalk(Letter)	25.88	22.14	21.98	19.91	21.37	20.12
NetTalk(Ph)	18.97	16.01	18.03	14.60	15.22	14.73
NetTalk(Stress)	17.25	11.91	12.44	11.30	12.26	10.54
Pima	23.97	26.57	23.76	26.43	23.31	23.18
Postoperative	29.44	38.89	28.89	38.89	32.22	34.44
Primary tumor	59.59	55.75	54.72	55.02	55.02	55.30
Promoters	17.50	4.68	7.09	4.73	5.64	5.64
Sick	1.30	0.92	1.42	1.04	1.10	1.33
Solar flare	15.62	17.57	15.70	17.57	16.31	15.95
Sonar	26.43	14.64	16.32	13.93	19.68	17.79
Soybean	8.49	6.22	5.42	5.64	6.66	5.49
Splice junction	5.81	4.80	4.50	3.65	4.23	3.81
Vehicle	28.50	22.40	25.12	22.40	24.00	23.52
Waveform-21	23.83	18.33	19.83	17.50	18.00	17.67
Wine	8.96	3.35	4.48	1.96	3.07	1.68
average	19.18	15.97	16.10	15.76	15.42	15.09

Table 3: Error rate ratios

Domain	Boost	Sasc	SascB	MB	SascMB	SascMB vs			
	vs C4.5					Boost	Sasc	SascB	MB
Annealing	**.66**	**.79**	**.56**	**.63**	**.68**	1.03	.86	*1.23*	1.08
Audiology	**.72**	**.88**	**.71**	**.74**	**.72**	1.00	**.82**	1.02	.97
Automobile	**.82**	.88	.97	.99	1.03	*1.25*	1.17	1.06	1.04
Breast (W)	**.63**	**.68**	**.61**	**.61**	**.62**	.98	.92	1.02	1.02
Chess (KR-KP)	**.50**	.93	**.50**	**.54**	**.54**	1.08	**.58**	1.08	1.00
Chess (KR-KN)	**.40**	1.04	**.46**	**.59**	**.63**	*1.59*	**.61**	*1.38*	1.07
Credit (Aust)	.96	1.02	.98	.88	**.87**	**.91**	**.86**	**.89**	.98
Credit (Ger)	**.87**	**.85**	**.86**	**.81**	**.80**	**.92**	**.94**	**.93**	.98
Echocardiogram	.96	.98	1.04	**.84**	**.81**	**.84**	**.82**	**.78**	.96
Glass	**.63**	**.75**	**.65**	**.72**	**.63**	1.01	**.84**	.97	**.88**
Heart (C)	**.85**	**.75**	**.84**	**.82**	**.83**	.97	1.10	.98	1.01
Heart (H)	1.01	.90	1.00	.91	.88	**.87**	.98	**.88**	.97
Hepatitis	.86	**.89**	**.77**	**.83**	**.83**	.97	.93	1.08	1.00
Horse colic	*1.26*	1.10	*1.23*	1.01	1.02	**.81**	.92	**.83**	1.01
House votes 84	.86	.82	**.76**	**.69**	**.76**	.88	.93	1.00	1.09
Hypo	**.70**	1.00	.78	**.72**	.87	1.25	.87	1.11	1.21
Hypothyroid	*1.61*	1.07	*1.38*	*1.15*	*1.34*	.83	*1.25*	.97	*1.16*
Image	**.53**	**.69**	**.53**	**.60**	**.65**	*1.22*	.94	*1.22*	1.09
Iris	1.31	1.15	1.31	1.15	1.15	.88	1.00	.88	1.00
Labor	**.46**	.80	**.42**	**.52**	**.44**	.97	**.56**	1.07	.85
LED 24	.90	**.79**	.89	**.88**	**.84**	.93	1.05	.94	.95
Letter	**.24**	**.31**	**.23**	**.28**	**.27**	*1.13*	**.89**	*1.20*	.96
Liver disorders	**.82**	**.85**	**.83**	**.76**	**.77**	.94	**.91**	.93	1.02
Lung cancer	.93	**.80**	.93	.82	.86	.91	1.07	.91	1.04
Lymphography	**.77**	.84	**.75**	**.77**	**.67**	.88	**.80**	.89	.88
NetTalk(Letter)	**.86**	**.85**	**.77**	**.83**	**.78**	**.91**	**.92**	1.01	**.94**
NetTalk(Ph)	**.84**	**.95**	**.77**	**.80**	**.78**	**.92**	**.82**	1.01	.97
NetTalk(Stress)	**.69**	**.72**	**.66**	**.71**	**.61**	**.88**	**.85**	**.93**	**.86**
Pima	*1.11*	.99	*1.10*	.97	.97	**.87**	.98	**.88**	.99
Postoperative	*1.32*	.98	*1.32*	1.09	*1.17*	**.89**	*1.19*	.89	1.07
Primary tumor	**.94**	**.92**	**.92**	**.92**	**.93**	.99	1.01	1.01	1.01
Promoters	**.27**	**.41**	**.27**	**.32**	**.32**	1.21	.80	1.19	1.00
Sick	**.71**	1.09	**.80**	.85	1.02	*1.45*	.94	*1.28*	*1.21*
Solar flare	*1.12*	1.01	*1.12*	1.04	1.02	**.91**	1.02	**.91**	.98
Sonar	**.55**	**.62**	**.53**	**.74**	**.67**	1.22	1.09	*1.28*	.90
Soybean	**.73**	**.64**	**.66**	**.78**	**.65**	.88	1.01	.97	.82
Splice junction	**.83**	**.77**	**.63**	**.73**	**.66**	**.79**	**.85**	1.04	**.90**
Vehicle	**.79**	**.88**	**.79**	**.84**	**.83**	1.05	.94	1.05	.98
Waveform-21	**.77**	**.83**	**.73**	**.76**	**.74**	.96	.89	1.01	.98
Wine	**.37**	**.50**	**.22**	**.34**	**.19**	**.50**	**.37**	.86	.55
average	.80	.84	.78	.78	.77	.99	.91	1.01	.98
w/t/l	33/0/7	32/1/7	32/1/7	35/0/5	33/0/7	27/0/13	29/1/10	19/1/20	21/4/15
p. of wtl	$< .0001$	$< .0001$	$< .0001$	$< .0001$	$< .0001$	.0192	.0017	.5000	.2025

average error rate 15.09%. The average error rate is 19.18%, 15.97%, 16.10%, 15.76%, and 15.42% for C4.5, BOOST, SASC, SASCB, and MB respectively. SASCMB also achieves the greatest average relative error reduction (23%) over C4.5 among these five committee learning algorithms.

A direct comparison shows that the average relative error reduction of SASCMB over BOOST and SASC is 1% and 9% respectively. A one-tailed sign-test suggests that SASCMB has significantly lower error rate than BOOST and SASC ($p = 0.0192$ and 0.0017 respectively). The average relative error reduction of SASCMB over MB is 2%, but a one-tailed sign-test fails to show that this reduction is significant at a level of 0.05. The average error ratio of SASCMB over SASCB is 1.01, although the average error rate of SASCMB is lower than that of SASCB. This is because SASCB performs better than SASCMB in domains in which they have relatively low error rates, and vice versa in domains in which they have relatively high error rates. It might be thought a disadvantage of SASCMB that the average error ratio compared to SASCB is greater than 1. However, we argue that this is a statistical anomaly, due to SASCB's superior performance when C4.5 has lower error rates. Increasing accuracy is as important as decreasing error. The average *accuracy ratio*, a measure that favors better performance at large error rates, of SASCMB against SASCB (an accuracy for SASCMB divided by the corresponding accuracy for SASCB) is also 1.01. Note that the average error rate of SASCMB is 0.67 percentage points lower, a considerable reduction, than that of SASCB.

5 Conclusions

We have presented a new classifier committee learning method, SASCMB, for decision tree learning. It generates multiple committees through incorporating Bagging into SASCB. In the new algorithm, the Boosting process is broken down into several small processes with each creating one subcommittee. The Bagging component of SASCMB further increases the diversity and independence of committee members. Our aim is to improve the stability and average accuracy of learned committees. Another advantage of SASCMB over SASCB and Boosting is that SASCMB is amenable to parallel and distributed processing, which is important for datamining in large datasets.

The results of experiments with a representative collection of natural domains suggest that SASCMB is more stable than SASCB and Boosting. It achieves the lowest error rate among the five committee learning algorithms on average in the 40 domains under investigation. It also achieves the greatest average relative error reduction over the base decision tree learning algorithm among the five committee learning algorithms. The experiments show that SASCMB can significantly outperform SASC and Boosting on average in terms of lower error rate. At the very least, SASCMB is as accurate as SASCB and MB, while demonstrating greater stability and amenability to parallel and distributed processing.

Acknowledgments

The authors are grateful to J. Ross Quinlan for providing C4.5.

References

Ali, K.M. 1996. Learning Probabilistic Relational Concept Descriptions. Ph.D. diss., Dept of Info. and Computer Science, Univ. of California, Irvine.

Bauer, E. and Kohavi, R. 1998. An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants. Submitted to *Machine Learning* (available at http://reality.sgi.com/ronnyk/vote.ps.gz).

Breiman, L. 1996a. Bagging Predictors. *Machine Learning* 24: 123-140.

Breiman, L. 1996b. Arcing Classifiers. Technical Report (available at http://www.stat.Berkeley.EDU/users/breiman/). Dept of Statistics, Univ of California, Berkeley, CA.

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

Chan, P., Stolfo, S., and Wolpert, D. 1996. Working Notes of AAAI Workshop on Integrating Multiple Learned Models for Improving and Scaling Machine Learning Algorithms (available at http://www.cs.fit.edu/~imlm/papers.html), Portland, Oregon.

Dietterich, T.G. and Bakiri, G. 1995. Solving Multiclass Learning Problems via Error-correcting Output Codes. *Journal of Artificial Intelligence Research* 2: 263-286.

Dietterich, T.G. and Kong, E.B. 1995. Machine Learning Bias, Statistical Bias, and Statistical Variance of Decision Tree Algorithms. Technical Report, Dept of Computer Science, Oregon State University, Corvallis, Oregon (available at ftp://ftp.cs.orst.edu/pub/tgd/papers/tr-bias.ps.gz).

Dietterich, T.G. 1997. Machine Learning Research. *AI Magazine* 18: 97-136.

Domingos, P. 1997. Why does Bagging Work? a Bayesian Account and its Implications. In Proceedings of the Third International Conference on Knowledge Discovery and Data Mining, 155-158. AAAI Press.

Freund, Y. 1996. Boosting a Weak Learning Algorithm by Majority. *Information and Computation* 121(2): 256-285.

Freund, Y. and Schapire, R.E. 1996a. A Decision-theoretic Generalization of On-line Learning and an Application to Boosting. Unpublished manuscript (available at http://www.research.att.com/~yoav).

Freund, Y. and Schapire, R.E. 1996b. Experiments with a New Boosting Algorithm. In Proceedings of the Thirteenth International Conference on Machine Learning, 148-156. San Francisco, CA: Morgan Kaufmann.

Kohavi, R. 1995. A Study of Cross-validation and Bootstrap for Accuracy Estimation and Model Selection. In Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, 1137-1143. Morgan Kaufmann.

Kwok, S.W. and Carter, C. 1990. Multiple Decision Trees. In Schachter, R.D., Levitt, T.S., Kanal, L.N., and Lemmer, J.F. (eds.), *Uncertainty in Artificial Intelligence*, 327-335. Elsevier Science.

Merz, C.J. and Murphy, P.M. 1997. UCI Repository of machine learning databases [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: Univ of California, Dept of Info and Computer Science.

Quinlan, J.R. 1993. *C4.5: Program for Machine Learning.* San Mateo, CA: Morgan Kaufmann.

Quinlan, J.R. 1996. Bagging, Boosting, and C4.5. In Proceedings of the Thirteenth National Conference on Artificial Intelligence, 725-730. AAAI Press.

Schapire, R.E. 1990. The Strength of Weak Learnability. *Machine Learning* 5: 197-227.

Schapire, R.E., Freund, Y., Bartlett, P., and Lee, W.S. 1997. Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods. In Proceedings of the 14th International Conference on Machine Learning, 322-330. Morgan Kaufmann.

Webb, G.I. 1998. Idealized Models of Decision Committee Performance and Their Application to Reduce Committee Error. Tech Report (TR C98/11), School of Computing and Mathematics, Deakin University, Australia (available at http://www3.cm.deakin.edu.au/~webb/papers/TRC9811.ps.Z).

Zheng, Z. and Webb, G.I. 1998. Stochastic Attribute Selection Committees. Technical Report (TR C98/08), School of Computing and Mathematics, Deakin University (available at http://www3.cm.deakin.edu.au/~zijian/Papers/sasc-tr-C98-08.ps.gz).

Zheng, Z., Webb, G.I., and Ting, K.M. 1998. Integrating Boosting and Stochastic Attribute Selection Committees for Further Improving the Performance of Decision Tree Learning. Technical Report (TR C98/12), School of Computing and Mathematics, Deakin University, Australia (available at http://www3.cm.deakin.edu.au/~zijian/Papers/sascb-tr-C98-12.ps.gz).