Proceedings of the IASTED International Conference
Artificial Intelligence and Applications (AIA 2009)
February 16 - 18, 2009 Innsbruck, Austria

REDUCING THE NUMBER OF BASE CLASSIFIERS IN ENSEMBLES

Muhammad Azam[1], Fernando Berzal[2], and Karl Peter Pfeiffer[1]
[1]Department of Medical Statistics, Informatics and Health Economics, Innsbruck, Austria
[2]Department of Computer Science and Artificial Intelligence, University of Granada, Spain
mazam72@yahoo.com, berzal@acm.org, and Karl-Peter.Pfeiffer@i-med.ac.at

ABSTRACT
Bagging is one of the most successful ensemble classification techniques. Ensembles are used to enhance the predictive capability of unstable classifiers. In fact, it has been proven that bagging outperforms single classifiers [1]. We introduce the idea of selecting those base classifiers in the ensemble whose number of misclassified units is less than or equal to the modal number of misclassified units. This automated mode-based bagging technique provides almost the same prediction accuracy than standard bagging, but it does so with a significant reduction in the number of base classifiers in the resulting ensemble. Experimental results using some real-life datasets available from the UCI Machine Learning Repository illustrate the performance of our suggested scheme. Our experiments show that mode-based bagging achieves a significant reduction in the number of base classifiers in the resulting ensembles (44% reduction with respect to standard bagging [1], 25% reduction with respect to trimmed bagging [2]) while the minor increase in the error rates that is observed is not statistically significant. The Friedman test [3] shows that the three studied ensemble methods exhibit a significantly different behavior with respect to the reduction of the number of base classifiers in the ensembles (with $p \leq 0.000041$).

KEY WORDS
Classification, ensemble techniques, bagging, trimmed bagging, decision trees, splitting criteria.

1. Introduction

Ensemble classifiers are used to improve the classification accuracy of single classifiers by aggregating the predictions made by large number of base classifiers.

We can divide classifiers into two general categories, i.e., stable and unstable classifiers. Stable classifiers include support vector machines, linear discriminant analysis, and logistic regression while unstable classifiers include decision trees, like classification and regression trees. Ensemble methods are particularly useful with unstable classifiers. Therefore, we will concentrate our efforts on decision trees, the most common example of unstable classifiers. In fact, decision trees have obtained a lot of appreciation and attention from many authors due to their simple, intuitive and understandable tree structure.

Ensemble methods have been used to improve the predictive capability of decision trees. These methods include bagging [1], boosting [4], and random forests [5]. Bagging builds an ensemble by generating several base classifiers. These base classifiers are then combined to obtain an integrated classifier. This integrated classifier, in the case of bagging, employs a majority vote in the case of classification trees and an average over all the base classifiers in case of regression trees.

The idea of trimmed bagging was introduced in [2]. Trimmed bagging excludes all those base classifiers that yield the highest error rates. It divides the base classifiers into “best” and “worst” base classifiers. The top α portion having highest error rates is trimmed off and the remaining $(1-\alpha)$ portion is used in the ensemble.

In this paper, we introduce an automated method for determining a suitable value for α. Instead of using a fixed value for trimmed bagging, our method automatically adjusts the value of α depending on the properties of the base classifiers obtained for a particular dataset.

Our automated mode-based bagging method reduces the number of base classifiers used in the ensemble process. In automated mode-based bagging, we compute the discrete frequency distribution for the number of misclassified units in the base classifiers. Then, the mode of such distribution is computed to determine the criteria for the inclusion of base classifiers in the ensemble.

The rest of our paper is organized as follows. In Section 2, we will explain different bagging methods, e.g. standard bagging, trimmed bagging, and our proposed automated mode-based bagging. In Section 3, we will discuss some of the different evaluation functions used to split a node during the construction of classification trees. In Section 4, we analyze the experimental results we have obtained. Finally, the final section contains some concluding remarks.

2. Bagging Methods

Ensemble classifiers are used to improve the classification accuracy of unstable classifiers by aggregating the predictions made by large numbers of base classifiers. Several authors have pointed out that ensemble classifiers can significantly improve the performance as compared to a single classifier. Empirical as well as experimental studies on ensemble classifiers have been made by [9] and

[10]. These studies confirm the substantial improvement in classification accuracy of ensemble classifier as compared to single classifier.

Let us suppose that we have a dataset $D = \{X, Y\}$, where $X = (x_1, x_2, \cdots, x_K)$ is the set of K observed predictors and Y is the response variable. In classification problems, Y belongs to a predefined set of discrete class labels $Y \varepsilon \{1, 2, \cdots, C\}$, while Y is a continuous variable in regression problems.

Using the above dataset, our problem consists of building a classifier $C(x)$ that, given a set of values for the input variables X, is able to predict a proper value for Y. The bagging term is actually derived from "**B**OOTSTRAP **AGG**REGAT**ING**" and its experimental and theoretical justification can be found in [1].

We take B bootstrap samples [6] from the dataset D of size N in order to obtain B datasets D_B of size N (i.e. sampling with replacement). Each bootstrap sample consists, on average, of 63.2% of the units. The remaining 36.8% of the units are known as out-of-bag sample. A detailed discussion about the use of out-of-bag sample units can be found in [7] and [8].

Bagging builds an integrated classifier using the base classifiers obtained from the D_B datasets instead of just employing the single classifier that could be built from D. Bagging provides an integrated classifier $C(x)$ from the base classifiers $C(x, D_B)$ in the following way. If the response variable Y is continuous, then $C(x) = average_B[C(x, D_B)]$. Otherwise, the integrated classifier demands majority vote from the individual base classifiers $C(x, D_B)$ employed to obtain the integrated classifier $C(x)$.

In the following paragraphs, we will discuss the original bagging procedure as well as two modified forms of the original bagging procedure.

2.1 Bagging

We take B bootstrap samples in order to obtain B datasets D_B of size N from the original dataset D. Then, we construct a base classifier against each bootstrap sample.

In classification problems, where we have a predefined set of class labels, we can gather all the information resulting from the application of the B base classifiers to the N units in a matrix consisting of B rows and N columns. Each column of this matrix is used to obtain the class label that is dominating or in majority for classifying a particular individual.

As described, bagging is suitable for unstable classifiers (since it usually improves the predictive performance of individual unstable classifiers) but it is unsuitable for stable classifiers (bagging might even degrade the predictive performance of the individual classifier) [1] [2] [10].

Suppose our ensemble consists of B base classifiers, i.e., $C_1(x), C_2(x), \ldots, C_B(x)$. For a given individual, each base classifier votes for one of the classes and the individual is classified on the basis of majority vote. Mathematically, bagging can be described as

$$C_{Bagging}(x) = \underset{1 \le b \le B}{majority\ vote}\ [C_b(x)].$$

2.2 Trimmed Bagging

The error rates obtained by the B bootstrap samples can be ordered. Given their individual error rates, base classifiers are then partitioned into "best" and "worst" classifiers. The idea of trimmed bagging [2] is to trim off the portion α of "worst" classifiers, i.e. those having the highest error rates, and to average only over the best base classifiers in the ensemble. Note that the trimming strategy is one-sided: only those classifiers having low error rates will be included in the resulting ensemble. As a rule of thumb, the proponents of trimmed bagging suggested that the top 25% error rates should be excluded from the ensemble.

Mathematically, trimmed bagging can be described as

$$C_{Trimmed\ bagging}(x) = \underset{1 \le b \le [(1-\alpha)B]}{majority\ vote}\ [C_b(x)].$$

2.3 Automated Mode-Based Bagging

As always, we build B base classifiers using the B bootstrap samples. As in trimmed bagging, we compute the number of misclassified units M for each one of the B base classifiers. There might be as many discrete values of M as the number of bootstrap samples.

Let M_i be the number of misclassified units with corresponding frequencies f_i (i ranges from 1 to h, where h represents the highest integer value of M) in B number of base classifiers. A modal value $\hat{M}$ is computed from the discrete frequency distribution of M.

Finally, we consider only those base classifiers having $M_i \le \hat{M}$

In other words, the discrete value of $\hat{M}$ determines the list of base classifiers to be included or excluded from the ensemble. As in trimmed bagging, we are not interested in the base classifiers that lead to the higher numbers of misclassified units.

We have chosen the mode as a decision point for the selection of final list of base classifiers because we do not want to lose the base classifiers corresponding to the most common number of misclassified units. By selecting an arbitrary specific percentage of base classifiers among all the base classifiers, as in trimmed bagging, this could easily happen.

Hence, our overall strategy can be explained as follows: Let I and E represent the lists of base classifiers to be included and excluded from the ensemble, respectively.

I: List of base classifiers whose $M_i \le \hat{M}$ and

E: List of base classifiers whose $M_i > \hat{M}$.

We obtain the class labels on the basis of a majority vote from the I list of base classifiers.

Sometimes, it is possible that we might get more than one mode values, especially when the dataset is large and we have many different values for M. In that case, our automated mode-based bagging procedure selects the smaller mode value (in order to reduce the number of base classifiers as much as possible). It should be noted that a similar problem appears when classifying instances using ensembles: ties can then be resolved by selecting the first most frequent class label or just by choosing one of them randomly.

Mathematically, automated mode-based bagging can be viewed as

$$C_{Mode\ based\ bagging}(x) = \underset{B_{f_1} \le b \le B_{f_m}}{majority\ vote}\ [C_b(x)],$$

where the list of base classifiers is sorted by their error rate in ascending order, B_{f_1} denotes the base classifier having the lowest number of misclassified units and B_{f_m} denotes the last base classifier having $\hat{M}$ misclassified units.

3. Decision Tree Splitting Rules

Decision trees are widely used in the form of classification and regression trees. Decision trees provide hierarchical structures that recursively partition the data from their root nodes to their terminal nodes. Common usages of decision trees include description (summarization) and prediction (generalization) [11].

During the construction of decision trees, an evaluation function plays an important role, since it decides how to split nodes. Although the selected approach for the construction of classification or regression trees might affect the resulting misclassification rates, it has been found in practice that almost all the evaluation functions proposed in the literature yield very similar results [12]. A detailed overview of such evaluation functions and their properties can be found in [13].

In this section, we describe some of the most common families of evaluation functions i.e., impurity-based and non-impurity-based functions. Later, we will explore whether those evaluation functions play any role in reducing the error rate of bagging methods.

3.1 Gini Index

The Gini index for a node t is given by

$$i(t)_G = 1 - \sum_{j=1}^{J} P_j^2, \tag{1}$$

where j is total number of classes or categories and P_j is the proportion of j^{th} class in a node t, such that $\sum_{j=1}^{J} P_j = 1$.

For the left and right child nodes in binary decision trees, we can define the Gini index function as

$$i(t_L)_G = 1 - \sum_{j=1}^{J} P_{j|L}^2 \quad \text{and} \quad i(t_R)_G = 1 - \sum_{j=1}^{J} P_{j|R}^2, \tag{2}$$

where t_L and t_R correspond to the left and right descendants, and $P_{j|L}$ and $P_{j|R}$ are the proportions of j^{th} class on left and right descendent node respectively, such that $\sum_{j=1}^{J} P_{j|L} = \sum_{j=1}^{J} P_{j|R} = 1$.

3.2 Entropy Function

[15] proposed an evaluation function for the measurement of node impurity based on the definition of entropy from information theory. For a node t, its entropy is defined by

$$i(t)_Q = -\sum_{j=1}^{J} P_j \ln P_j, \tag{3}$$

In binary decision trees, the entropy function for the left and right child nodes is

$$i(t_L)_Q = -\sum_{j=1}^{J} P_{j|L} \ln(P_{j|L}) \text{ and } i(t_R)_Q = -\sum_{j=1}^{J} P_{j|R} \ln(P_{j|R}). \tag{4}$$

The interpretation of all the symbols is the same as above.

3.3 Twoing Rule

A non-impurity based node splitting criteria that measures the goodness of split value directly was introduced by [14]. It measures the difference in probabilities that a class appears in the left descendant rather than the right descendant node. The criterion is thus based on a concept of class separation rather than node heterogeneity. The criteria select a best splitting value, which maximizes

$$g(x,s,t)_T = \frac{P_L P_R}{4}\left[\sum_{j=1}^{J} \left|P_{j|L} - P_{j|R}\right|\right]^2. \tag{5}$$

The aim is to get a probability that a class j unit goes to the left as different as possible from the probability that it goes to the right. The criterion sums the absolute value of the probability differences over all j classes. The factor (P_L P_R) is designed to favor relatively even splits. This factor takes a maximum value of 0.25 when $P_L = P_R = 0.5$, it declines if any of the proportion is close to 0 or 1.

3.4 Exponent-Based Function

An idea to use an exponent-based function can be found in [13], which yields very similar or improved results as compared to standard node splitting node splitting approaches. The exponent based function to split nodes for the construction of classification trees is

$$i(t)_{\exp} = 1 - \frac{1}{e}\sum_{j=1}^{J} P_j e^{P_j}, \qquad (6)$$

For its left and right descendants, we obtain

$$i(t_L)_{\exp} = 1 - \frac{1}{e}\sum_{j=1}^{J} P_{j|L} e^{P_{j|L}} \text{ and } i(t_R)_{\exp} = 1 - \frac{1}{e}\sum_{j=1}^{J} P_{j|R} e^{P_{j|R}}, \qquad (7)$$

Once the values of the evaluation function for a given node and its descendants are computed, the general approach to measure the goodness of a given split consists of computing

$$g(x,s,t)_f = i(t)_f - P_L\ i(t_L)_f - P_R\ i(t_R)_f,$$

where f may be any of the aforementioned evaluation functions.

Decision trees are built so that we always branch the tree by selecting the best split s^* at each node, which is provided by the splitting variable x^* that maximizes the goodness of split measure.

$$g(x^*, s^*, t)_f = \text{Maximum of } g(x,s,t)_f .$$

4. Experimental Results

We have used eleven real life datasets available from the UCI Machine Learning Repository to evaluate the performance of the proposed automated mode-based bagging method. We have chosen datasets from the repository that have no missing values and include a categorical class variable (i.e. we have built decision trees for classification, and not regression, purposes). Table 1 summarizes the datasets we have used in our study, which are freely available from the following URL:

ftp://ftp.ics.uci.edu/pub/machine-learning-databases/

We have implemented bagging, trimmed bagging, and the proposed automated mode-based bagging in **R** to evaluate their performance. We have also implemented the different evaluation functions mentioned in Section 3. Our experimental results have been obtained from 200 bootstrap samples. Our experiments show that mode-based bagging achieves a significant reduction in the number of base classifiers in the resulting ensembles (44% reduction with respect to standard bagging [1], and 25% with respect to trimmed bagging [2]) while the minor increase in the error rates that is observed is not statistically significant. The Friedman as well as Iman & Davenport test [3] show that the three studied ensemble methods exhibit a significantly different behavior with respect to the reduction of the number of base classifiers in the ensembles (with $p \leq 0.000041$).

In order to check the usefulness of trimmed bagging, and automated mode-based bagging in particular, a two sample t test has also been applied at two splits of the number of misclassified units i.e., split 1 consist of base classifiers having $M_i \leq \hat{M}$ and split 2 consist of base classifiers having $M_i > \hat{M}$. It has found that both splits differ significantly at *alpha*=0.05.

We obtained almost the same results with the four evaluation functions used to split decision tree nodes during the construction of the base classifiers used by the ensemble methods. On average, the performance of the entropy function is slightly better than other evaluation functions, but the difference is not significant.

While bagging and trimmed bagging consider 100% and 75% of the base classifiers that take part in the ensemble process, respectively, our proposed method employs a variable number of base classifiers. Table 2 shows the percentage of base classifiers that take part in the ensemble when using automated mode-based bagging for the different datasets using a variety of node splitting functions.

Tables 3 through 6 provide the detailed accuracy results we have obtained in our experiments. Apart from the aforementioned significant reduction in the number of base classifiers in the resulting ensembles, a minor increase (twoing rule, gini function, and exponent-based function) and even a light decrease (entropy) in the error rates can be observed. In any case, the accuracy differences are not statistically significant for any of evaluation functions used in our experiments.

In summary, there is a substantial reduction in the number of base classifiers in favor of the automated mode-based bagging approach (44% and 25% fewer base classifiers than standard bagging and trimmed bagging, respectively) while the minor increase in the error rates that is observed is not statistically significant. Hence, our experiments corroborate that we can achieve the goal of reducing the number of base classifiers without compromising the accuracy of the ensemble classifier.

Table 1
Dataset used in our experiments

Dataset	Size	Predictors	Classes
Iris	150	4	3
Haberman	306	3	2
Lympho	148	18	4
Bal. Scale	625	4	3
Pima	768	8	2
Wine	178	13	3
Yeast	1484	8	10
Glass	214	9	6
Housing	506	13	9
Bupa	345	6	2
Breast Cancer	569	30	2

Table 2
Percentage of base classifiers employed by automated mode-based bagging

Dataset	Gini	Twoing	Entropy	Exponent
Iris	54	54	51	54
Haberman	71	27	35	15
Lympho	87	85	72	89
Bal. Scale	28	74	39	28
Pima	75	75	73	75
Wine	57	34	59	44
Yeast	56	24	58	33
Glass	41	52	52	33
Housing	85	98	49	93
Bupa	56	56	96	55
Breast Cancer	50	50	26	48
Mean Ratio	60.00	57.18	55.45	51.55

Table 3
Error rates using the Gini function

Dataset	Mode Bagging	Trim. Bagging	Bagging
Iris	2.00	2.67	2.67
Haberman	15.03	15.36	15.03
Lympho	19.59	18.24	18.24
Bal. Scale	13.12	13.28	12.80
Pima	21.09	21.09	20.83
Wine	0.00	0.00	0.00
Yeast	39.49	39.69	39.96
Glass	19.62	20.09	20.09
Housing	26.08	27.27	25.69
Bupa	26.09	26.09	25.22
Breast Cancer	1.93	1.93	1.93
Overall Average	16.73	16.88	16.59

Table 4
Error rates using the Twoing rule

Dataset	Mode Bagging	Trim. Bagging	Bagging
Iris	2.00	2.67	2.67
Haberman	15.36	15.36	15.03
Lympho	23.65	21.62	21.62
Bal. Scale	12.32	12.32	12.16
Pima	20.96	20.96	20.83
Wine	0.00	0.00	0.00
Yeast	41.17	41.04	41.24
Glass	18.69	18.22	17.76
Housing	26.68	27.67	26.48
Bupa	26.38	26.09	25.22
Breast Cancer	1.93	1.93	1.93
Overall Average	17.19	17.08	16.81

Table 5
Error rates using the Entropy function

Dataset	Mode Bagging	Trim. Bagging	Bagging
Iris	1.33	2.00	2.00
Haberman	15.36	14.71	14.38
Lympho	25.68	25.00	26.35
Bal. Scale	12.16	12.32	11.84
Pima	20.05	20.05	20.05
Wine	0.5600	0.5600	0.5600
Yeast	42.18	41.98	41.98
Glass	19.62	19.16	18.69
Housing	16.01	17.00	17.00
Bupa	24.64	24.93	24.93
Breast Cancer	1.58	1.76	1.93
Overall Average	16.29	16.32	16.34

Table 6
Error rates using the Exponent-based function

Dataset	Mode Bagging	Trim. Bagging	Bagging
Iris	2.00	2.67	2.67
Haberman	15.03	15.03	15.03
Lympho	18.92	16.89	16.89
Bal. Scale	13.60	12.96	12.32
Pima	21.22	20.83	21.09
Wine	0.00	0.00	0.00
Yeast	40.57	38.81	39.02
Glass	19.16	21.03	20.56
Housing	39.72	36.56	39.72
Bupa	26.38	26.38	25.51
Breast Cancer	1.93	1.93	1.93
Overall Average	18.05	17.55	17.70

5. Conclusion

In this paper, we have proposed a modified form of bagging called automated mode-based bagging. Our approach includes in the ensemble only those base classifiers having an error rate lower than the modal error rate. This approach for building ensemble classifiers significantly decreases the number of base classifiers in the ensemble, while preserving the resulting ensemble accuracy

It should be noted that the proposed approach is applicable not only for classification problems (where a discrete class label is predicted) but also to regression problems (where the response variable is continuous).

Acknowledgements

This work was supported by HEC Pakistan and Medical University Innsbruck, Austria.

References

[1] L. Breiman, Bagging predictors, *Machine Learning 24*, 1996a, 123-140.

[2] C. Croux, K. Joosens, & A. Lemmens, Trimmed bagging, *Computational Statistics & Data Analysis, 52*, 2007, 362-368.

[3] J. Demsar, Statistical comparison of classifiers over multiple datasets, *Journal of Machine Learning Research, 7*, 2006, 1-30.

[4] R. E. Schapire, Y. Freund, P. Barlett, & W. S. Lee, Boosting the margin: a new explanation for the effectiveness of voting methods, *Ann. Stat. 26* (3), 1998, 801-824.

[5] L. Breiman, Random forests, *Machine Learning*, *45*, 2001, 5-32.

[6] B. Efron, & R. J. Tibshirani, *An Introduction to the Bootstrap* (Chapman and Hall, London, UK, 1993).

[7] L. Breiman, Out-of-bag estimation. Technical report on, Statistics Department, University of California Berkley, Berkley, CA 94708, 1996b.

[8] T. Hothorn, & B. Lausen, Double bagging: combining classifiers by bootstrap aggregation, *Pattern Recognition, 36* (6), 2003, 1303-1309.

[9] E. Bauer, & R. Kohavi, An empirical comparison of voting classification algorithms: Bogging, Boosting and Variants, *Machine Learning, 36*(1-2), 1999, 105-139.

[10] T. G. Dietterich, An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, Boosting and Randomization, *Machine Learning*, 1998, 1-22.

[11] S. K. Murthy, Automatic construction of decision trees from data: A multi disciplinary survey, *Data Mining & Data Discovery, 2*, 1998, 345-389.

[12] F. Berzal, J. C. Cubero, F. Cuenca, F. & M. J. Martin-Bautista, On the quest of easy to understand splitting rules, *Data & Knowledge Engineering*, *44*, 2003, 31-44.

[13] M. Azam, Q. Zaman, & K. P. Pfeiffer, Improved classification trees with two or more classes. *Proceedings of The 9th Islamic Countries Conference on Statistical Sciences,* Kuala Lumpur, Malaysia, 2007, 608-626.

[14] L. Breiman, J. H. Friedman, R. A. Olshen, & C. J. Stone, *Classification and regression trees* (Wadsworth International Group, Belmont, CA, 1984).

[15] R. J. Quinlan, Induction of decision trees, *Machine Learning, 1*, 1986, 81-106.