The Role of Combining Rules in Bagging and Boosting

Marina Skurichina and Robert P.W. Duin

Pattern Recognition Group, Department of Applied Physics, Faculty of Applied Sciences,
Delft University of Technology, P.O. Box 5046, 2600GA Delft, The Netherlands.
marina@ph.tn.tudelft.nl

Abstract. To improve weak classifiers bagging and boosting could be used. These techniques are based on combining classifiers. Usually, a simple majority vote or a weighted majority vote are used as combining rules in bagging and boosting. However, other combining rules such as mean, product and average are possible. In this paper, we study bagging and boosting in Linear Discriminant Analysis (LDA) and the role of combining rules in bagging and boosting. Simulation studies, carried out for two artificial data sets and one real data set, show that bagging and boosting might be useful in LDA: bagging for critical training sample sizes and boosting for large training sample sizes. In contrast to a common opinion, we demonstrate that the usefulness of boosting does not directly depend on the instability of a classifier. It is also shown that the choice of the combining rule may affect the performance of bagging and boosting.

1 Introduction

When the training sample size is small compared to the data dimensionality, the training data may often give a distorted representation of the real data distribution. A classification rule, constructed on such training data, may be biased and have a large variance. Consequently, one can get a lousy classifier, having a poor performance [1]. In order to improve a weak classifier by stabilizing its decision, a number of techniques could be used, for instance, regularization [2] or noise injection [3]. Another approach, which allows us to improve a weak classifier, consists in combining classifiers, obtained on the modified versions of the original training set (e.g., by sampling [4] or weighting). This approach is implemented in bagging [5] and boosting [6], however, in different ways. In bagging, one samples the training set, generating random independent bootstrap replicates of the training set, constructs the classifier on each of these bootstrap replicates and aggregates them by simple majority vote in the final decision rule. In boosting, classifiers are constructed on weighted versions of the training set, which are obtained sequentely in the algorithm. Initially, all objects have equal weights, and the first classifier is constructed on this data set. Then weights are changed according to the performance of the classifier. Erroneous classified objects get larger weights and the next classifier is boosted on the reweighted training set. In this way a sequence of training sets and classifiers is obtained, which are then combined by the simple majority or the weighted majority vote in the final decision.

As a rule, bagging and boosting are applied to classification and regression trees (CART) [7],[8],[9],[10], where it is difficult to imply other combining rules than simple majority vote or weighted majority vote. However, bagging and boosting may also perform well in linear discriminant analysis [11], [12]. Linear classifiers allow us to use other combining rules such as the average (when the final classifier is obtained by averaging the coefficients of the combined classifiers), the mean (when decision is made according to the mean of posteriori probabilities given by the combined classifiers) and the product rule (when decision is made by the product of posteriori probabilities presented by the combined classifiers). It may happen that these combining rules perform better than majority vote, especially for bagging. Moreover, the average rule has an advantage to other combining rules, because it requires to keep only the coefficients of classifiers instead of all posteriori probabilities of each combined classifier.

In this paper we investigate the role of five mentioned combining rules (simple majority vote, weighted majority vote, average, mean and product) for bagging and boosting in linear discriminant analysis. The Nearest Mean Classifier (NMC) [13], also known as the Euclidean distance classifier, is used in our study. This choice is made because the NMC is often a weak, unstable classifier for data sets having an other distribution than Gaussian with equal variances. Therefore, bagging and boosting, which we recite in the next section, may be useful in order to improve it's performance. To perform our simulation study, we have chosen two artificial data sets and one real data set, which are described in section 3. The artificial data sets present a 2-class problem. The real data set consists of a 2-class problem and a 4-class problem. The results of our simulation study are presented in section 4. Conclusions are summarized in section 5.

2 Bagging and Boosting

In order to improve the performance of unstable regression or classification rules, a number of combining techniques can be used. In recent years, the most popular ones became bagging and boosting. They both modify the training data set, build classifiers on these modified training sets and then combine them into a final decision rule by simple or weighted majority vote. However, they perform it in a different way.

Bagging is based on the **b**ootstrapping [4] and **agg**regat**ing** concepts and presented by Breiman [5]. Both, bootstrapping and aggregating may be beneficial. Bootstrapping is based on random sampling with replacement. Therefore, taking a bootstrap replicate $X^b = (X_1^b, X_2^b, \ldots, X_n^b)$ (the random selection with replacement of N objects from the set of N objects) of the training sample set $X = (X_1, X_2, \ldots, X_n)$, one can sometimes avoid or get less misleading training objects in the bootstrap training set. Consequently, a classifier constructed on such training set may have a better performance. Aggregating actually means combining classifiers. Often a combined classifier gives better results than individual classifiers, because of combining in the final solution advantages of the individual classifiers. Therefore, bagging might be helpful to build better classifier on training sample sets with misleaders. In bagging, bootstrapping and aggregating techniques are implemented in the following way.

1. Repeat for b=1,2,...,B.

a) Take a bootstrap replicate X^b of the training data set X.
b) Construct a classifier $C(X^b)$ on X^b.

2. Combine classifiers by simple majority vote to a final decision rule.

Boosting, proposed by Freund and Schapire [6], is another technique to combine unstable and weak classifiers in order to get a classification rule with a better performance. In contrast to bagging, where bootstrap training sets and classifiers are independent and random, in boosting, classifiers and training sets are obtained in a strictly deterministic way. Both, training data sets and classifiers are obtained sequentely in the algorithm. At each step, training data are reweighted in such way that incorrectly classified objects get larger weights in a new modified training set. By that, one actually maximizes margins between training objects. It suggests the connection between boosting and Vapnik's Support Vector Classifier (SVC) [7],[14]. Boosting is organized in the following way.

1. Repeat for b=1,2,...,B.

a) Construct the classifier $C^b(X^*)$ on the weighted training data set $X^* = (w_1^b X_1, w_2^b X_2, \ldots, w_n^b X_n)$, using weights w_i^b, i=1,...,n ($w_i^b = 1$ for b=1).

b) Compute probability estimates of the error $err_b = \frac{1}{n}\sum_{i=1}^{n} w_i^b \xi_i^b$,
$\xi_i^b = \begin{cases} 0, & X_i \ classified \ correctly \\ 1, & otherwise \end{cases}$, and $c_b = \frac{1}{2}\log\left(\frac{1 - err_b}{err_b}\right)$.

c) Set $w_i^{b+1} = w_i^b \exp(-c_b \xi_i^b)$, i=1,...,n, and renormalize so that $\sum_{i=1}^{n} w_i^{b+1} = n$.

2. Combine classifiers $C^b(X^*)$ by weighted majority vote with weights c_b to a final decision rule.

3 Data

Two artificial data sets and one real data set are used for our experimental investigations.

• The first set is a 30-dimensional correlated Gaussian data set constituted by two classes with equal covariance matrices. Each class consists of 500 vectors. The mean of the first class is zero for all features. The mean of the second class is equal to 3 for the first two features and equal to 0 for all other features. The common covariance matrix is a diagonal matrix with a variance of 40 for the second feature and a unit variance for all other features. The intrinsic class overlap (Bayes error) is 0.064. In order to spread the separability over all features, this data set is rotated using a 30×30 rotation matrix which is $\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$ for the first two features and the identity matrix for all other features. We call these data further "*Gaussian correlated data*". Its first two features are presented in Fig. 1.

• The second data set consists of two 8-dimensional classes. The first two features of the data classes are uniformly distributed with unit variance spherical Gaussian noise along two 2π/3 concentric arcs with radii 6.2 and 10.0 for the first and the second

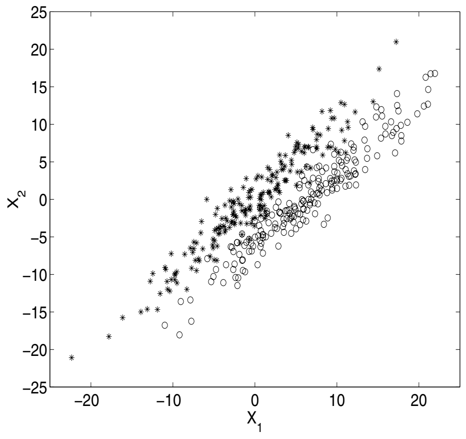

Fig. 1. The scatter plot of a two-dimensional projection of the 30-dimensional Gaussian correlated data

class respectively.

$$\vec{X}^{(1)} = \begin{pmatrix} X_1^{(1)} \\ X_2^{(1)} \end{pmatrix} = \begin{pmatrix} \rho_1 \cos(\gamma_1) + \xi_1 \\ \rho_1 \cos(\gamma_1) + \xi_2 \end{pmatrix}, \quad \vec{X}^{(2)} = \begin{pmatrix} X_1^{(2)} \\ X_2^{(2)} \end{pmatrix} = \begin{pmatrix} \rho_2 \cos(\gamma_2) + \xi_3 \\ \rho_2 \cos(\gamma_2) + \xi_4 \end{pmatrix},$$

where $\rho_1 = 6.2$, $\rho_2 = 10$, $\xi_i \sim N(0, 1)$, $\gamma_k \sim U\left(-\frac{\pi}{3}, \frac{\pi}{3}\right)$, $i = \overline{1, 4}$; $k = 1,2$. The other six features have the same spherical Gaussian distribution with zero mean and variance 0.1 for both classes. Both classes consist of 500 objects each. We will call these data *"banana-shaped data"* (BSD). Its first two features are presented in Fig. 2.

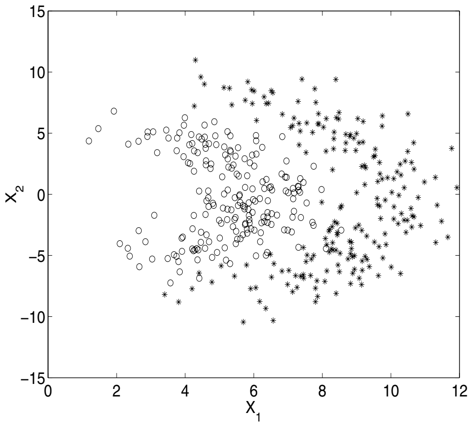

Fig. 2. Scatter plot of a two-dimensional projection of the 8-dimensional banana-shaped data

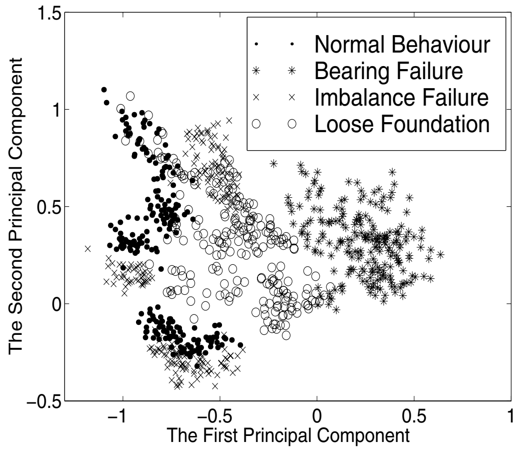

Fig. 3. Scatter plot of the first two principal components of the pump data with four classes

• The last data set consists of the measurements obtained from four kinds of water-pump operating states: a normal behaviour (NB), a bearing fault (BF) (a fault in the outer ring of the uppermost single bearing), an imbalance failure (IF) and a loose foundation failure (LFF), where the running speed (46, 48, 50, 52 and 54 Hz) and machine load (25, 29 and 33 KW) are varied. To measure pump vibrations, a ring accelerometer is used. For the obtained time series of the pump vibration patterns, we determined the coefficients of an order 128 autoregressive model. The 128 coefficients of this model are used as the features describing the pump vibration patterns. For each operating mode 15 128-dimensional vectors are obtained, which are normalized w.r.t. the mean and the standard deviation. Then the data are combined either in 4 classes (a normal behaviour, a bearing fault, an imbalance failure and a loose foundation failure) consisted of 225 observations each, or in 2 classes (the normal behaviour and the abnormal behaviour). In the latter case, the normal behaviour class consisted of 225 observations the abnormal behaviour class consisted of 675 observations representing all three failures: bearing, imbalance and loose foundation. The first two principal components of the autoregressive model coefficients for four operating states are presented in Fig. 3. We call these data *"pump data"* in the experiments.

Training data sets with 3 to 400 (with 3 to 200 for pump data) samples per class are chosen randomly from a total set. The remaining data are used for testing. These and all other experiments are repeated 50 times for independent training sample sets. In all figures the averaged results over 50 repetitions are presented and we do not mention that anymore.

The standard deviations of the mean generalization errors of the NMC, the bagged NMC, the boosted NMC and the SVC were of the similar order for each data set. When increasing the training sample size, they were decreasing approximately from 0.015 to 0.005, from 0.015 to 0.009, from 0.01 to 0.008 and from 0.02 to 0.01 for 30-dimensional Gaussian correlated data, for 8-dimensional banana-shaped data and for 128-dimensional pump data with 4 and 2 classes, respectively.

4 The Effect of the Combining Rule in Bagging and Boosting

Let us now study the usefulness of bagging and boosting in LDA on the example of the NMC and look at the effect of the combining rule on their performance. In order to understand better, when bagging and boosting might be beneficial, it may be useful to consider the instability of a classifier [11]. The instability of a classifier is measured by us by calculating the changes in classification of a training data set caused by the bootstrap replicate of the original learning data set. Repeating this procedure several times on the training set (we did it 25 times) and averaging the results an estimate of the classifier instability is obtained. The mean instability of the NMC (on 50 independent training sets) defined in this way is presented in Fig. 4 for the data sets described in the previous section. One can see that the instability of the NMC is distinct for different data sets. However, for all data sets, the classifier is the most unstable when the training sample size is small. Then the instability of the classifier decreases as the training sample size increases.

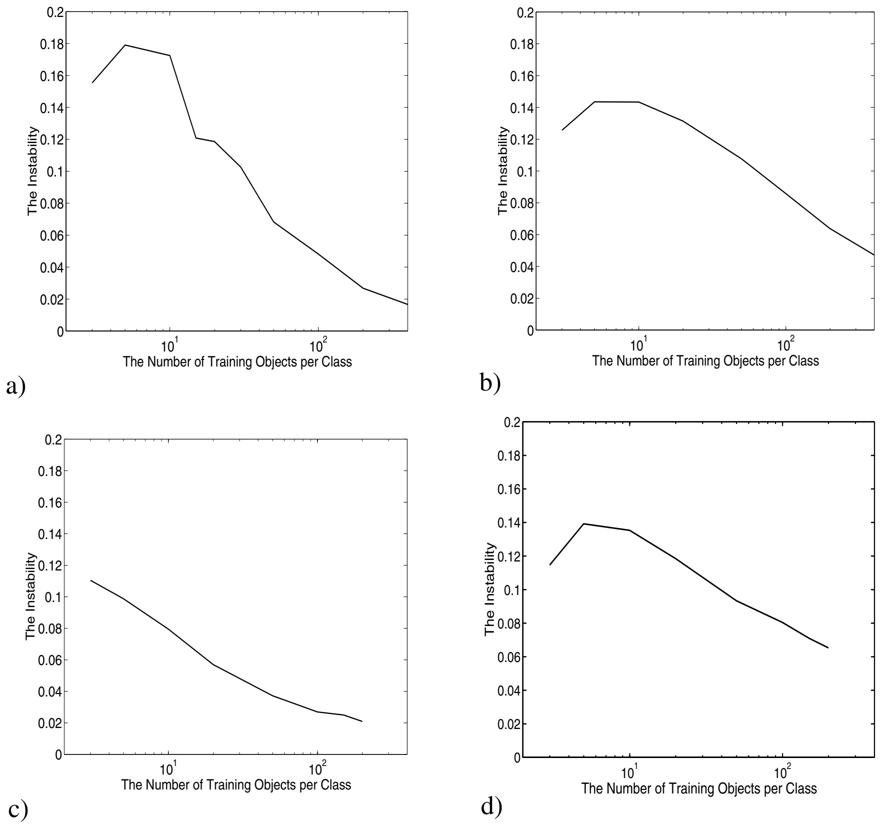

Fig. 4. The instability of the NMC for 30-dimensional Gaussian correlated data (a), 8-dimensional banana-shaped data (b), 128-dimensional pump data with 2 classes (c) and 4 classes (d)

Simulation results obtained on the 30-dimensional Gaussian correlated data (see Fig. 5) show that bagging and boosting are very useful for the NMC on this data set. Bagging improves almost twice the generalization error of the NMC for critical training sample sizes, when the data dimensionality is comparable with the number of training objects, and the classifier is unstable. When training sets are very small, often they represent the distribution of the entire data set incorrectly. Bootstrapping such training sets, one can hardly get a better training set. Therefore, bagging is useless for very small training sample sizes. When the training sample size is large, the classifier is stable. Large training sets represent the distribution of the entire data accurately. Therefore, perturbations in the composition of the training set do not change the training set very much. By this reason, bagging is useless for large training sample sizes. One also can see that the performance of bagging is strongly affected by the choice of the combining rule. Bagging with the simple majority vote rule, which is

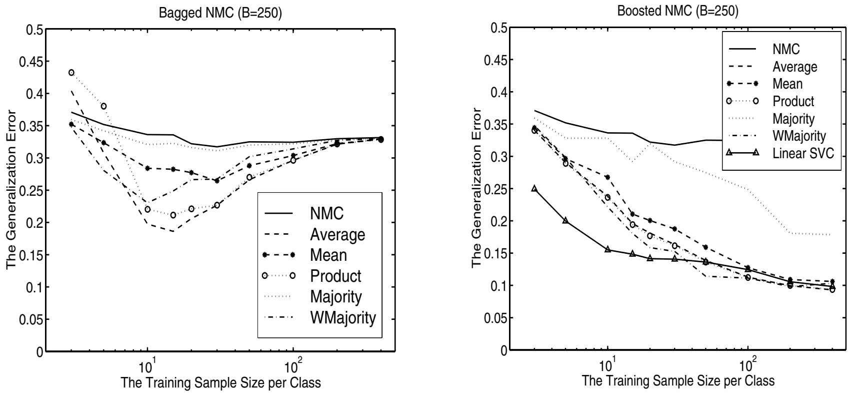

Fig. 5. The generalization error of the NMC, the bagged NMC (left plot) and the boosted NMC (right plot) using different combining rules for 30-dimensional Gaussian correlated data

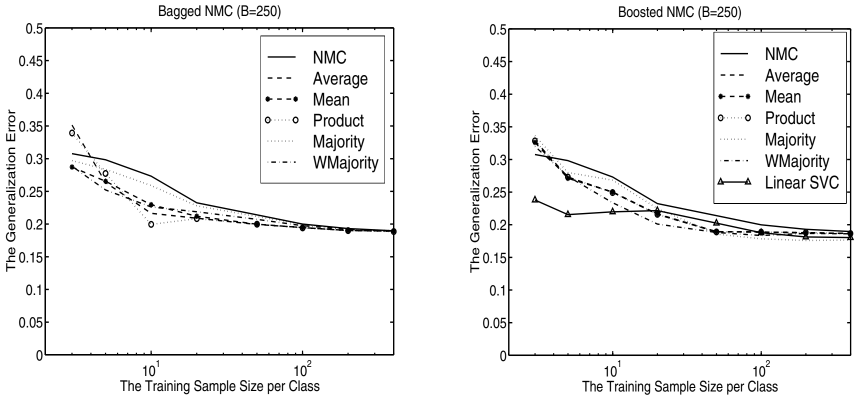

Fig. 6. The generalization error of the NMC, the bagged NMC (left plot) and the boosted NMC (right plot) using different combining rules for 8-dimensional banana-shaped data

usually used as a combing rule in bagging, performs the worst. For this data set, the best combining rules for bagging are the average, the weighted majority vote and the product. Comparing the left plot for bagging and the right plot for boosting in Fig. 5, one can clearly see that boosting outperforms bagging for each combining rule respectively. In boosting, wrongly classified objects get larger weights. Mainly, they are objects on the border between classes. Therefore, boosting performs the best for large training sample sizes, when the border between classes becomes more informative. In this case, boosting the NMC performs similar to the linear SVC [14]. However, when the training sample size is large, the NMC is stable. It puts us on an idea that, in contrast to bagging, the usefulness of boosting does not depend directly on the stability of the classifier. It depends on the "quality" of the wrong classified objects (usually, the border between data classes) and on the ability of the classifier (its complexity) to distinguish them correctly. As concerns combining rules, we see that

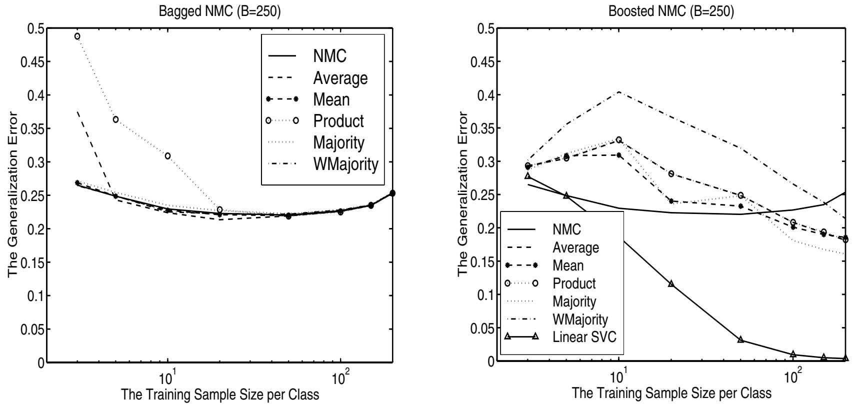

Fig. 7. The generalization error of the NMC, the bagged NMC (left plot) and the boosted NMC (right plot) using different combining rules for 128-dimensional pump data with 2 classes

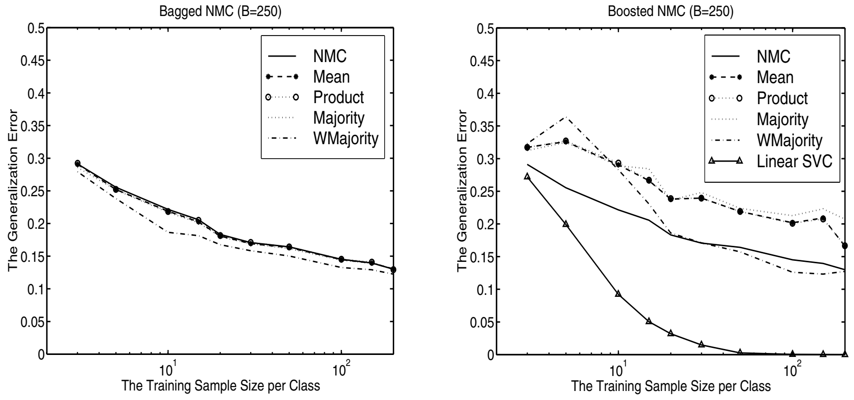

Fig. 8. The generalization error of the NMC, the bagged NMC (left plot) and the boosted NMC (right plot) using different combining rules for 128-dimensional pump data with 4 classes

the choice of the combining rule is less important for boosting, than for bagging. When boosting the NMC on the 30-dimensional Gaussian correlated data, all combining rules perform similar to each other with the exception of simple majority vote, which is reasonably worse.

On the 8-dimensional banana-shaped data (see Fig. 6), bagging and boosting are also useful for the NMC. However, due to non-Gaussian data distribution and a lower instability of the NMC, the obtained improvement is not so spectacular as in the previous case. Bagging outperforms boosting for critical training sample sizes. However, boosting performs slightly better on large training sample sizes achieving the performance of the SVC. One can also see that some small difference exists when different combining rules are used in bagging and boosting. Simple majority vote is again the worst combining rule for bagging. For this data set, the product combing rule is the best when bagging the NMC. In boosting, the weighted majority vote combining rule is slightly better than other combining rules when training sample sizes are not large.

When considering 128-dimensional pump data for a 2-class and a 4-class problem, one can see that the NMC is more stable (Fig. 4) on this data set than on other data sets, and bagging is almost useless (Fig. 7 and Fig. 8). Boosting becomes useful only when the number of training objects is larger than the data dimensionality. In this case, boosting performs better for a 2-class problem than for a 4-class problem, because to solve a 2-class problem is easier than a 4-class problem. However, to make more conclusions about the performance of boosting for large training sample sizes is difficult, as only limited amount of data is available (225 objects per class). Therefore, it is impossible to check whether the boosted NMC performs similar to the SVC, when the number of training objects exceeds 200 per class. Nevertheless, the results also show that the choice of the combining rule might be important. In a 4-class problem, using the weighted majority vote in bagging and boosting is more preferable than using other combining techniques. In a 2-class problem, boosting with the simple majority vote combining rule performs better than with the weighted majority vote combining rule, which is surprisingly the worst for this data set. It seems that it does not exist the unique combining rule which is the best for all data sets and for all training sample sizes.

5 Conclusions

Summarizing simulation results presented in the previous section, we can conclude the following:

Bagging and boosting may be useful in linear discriminant analysis.

Bagging helps in unstable situations, for critical training sample sizes.

Boosting is useful for large training sample sizes, when the objects on the border between data classes are enough representative to separate data classes correctly and the classifier is able (by its complexity) to distinguish them well. By that, boosting sometimes allows us to achieve the performance of the support vector classifier. The performance of boosting does not depend on the instability of the classifier.

The choice of the combining rule might be important. However, it strongly depends on the data and the training sample size.

When comparing the performance of bagging and boosting, it should be done on the fair background, when the same combining rule is used in both methods.

As a rule, simple majority vote is the worst possible choice of the combining rule. The weighted majority vote rule is often a good choice as for bagging as for boosting. The average, mean, and product combining rules may also perform well and sometimes better than the weighted majority vote combining rule.

Acknowledgment

This work is supported by the Foundation for Applied Sciences (STW) and the Dutch Organization for Scientific Research (NWO).

We are also thankful to Alexander Ypma for providing us the pump data.

References

1. Jain, A.K., Chandrasekaran, B.: Dimensionality and Sample Size Considerations in Pattern Recognition Practice. In: Krishnaiah, P.R., Kanal, L.N. (eds.): Handbook of Statistics, Vol. 2. North-Holland, Amsterdam (1987) 835-855
2. Friedman, J.H.: Regularized Discriminant Analysis. Journal of the American Statistical Association (JASA) **84** (1989) 165-175
3. An, G.: The Effects of Adding Noise During Backpropagation Training on a Generalization Performance. Neural Computation **8** (1996) 643-674
4. Efron, B., Tibshirani, R.: An Introduction to the Bootstrap. Chapman and Hall, New York (1993)
5. Breiman, L.: Bagging predictors. Machine Learning Journal **24**(2) (1996) 123-140
6. Freund, Y., Schapire, R.E.: Experiments with a New Boosting Algorithm. In: Machine Learning: Proceedings of the Thirteenth International Conference (1996) 148-156
7. Schapire, R.E., Freund, Y., Bartlett, P., Lee, W.: Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods. The Annals of Statistics **26**(5) (1998) 1651-1686
8. Breiman, L.: Arcing Classifiers. Annals of Statistics, **26**(3) (1998) 801-849
9. Friedman, J., Hastie, T., Tibshirani, R.: Additive Logistic Regression: a Statistical View of Boosting. Technical Report (1999)
10. Dietterich, T.G.: An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, and Randomization. Machine Learning, to appear
11. Skurichina, M., Duin, R.P.W.: Bagging for Linear Classifiers. Pattern Recognition **31**(7) (1998) 909-930
12. Skurichina, M., Duin, R.P.W.: Boosting in Linear and Quadratic Discriminant Analysis. In preparation for submission to the First Int. Workshop on Multiple Classifier Systems
13. Fukunaga, K.: Introduction to Statistical Pattern Recognition. Academic Press (1990) 400-407
14. Cortes, C., Vapnik, V.: Support-Vector Networks. Machine Learning **20** (1995) 273-297