Bagging with Asymmetric Costs for Misclassified and Correctly Classified Examples^{*}

Ricardo Nanculef¹, Carlos Valle¹, Héctor Allende¹, and Claudio Moraga^{2,3}

¹ Universidad Técnica Federico Santa María,
Departamento de Informática, CP 110-V Valparaíso, Chile
{cvalle,jnancu,hallende}@inf.utsfm.cl

² European Centre for Soft Computing 33600 Mieres, Asturias, Spain

³ Dortmund University, 44221 Dortmund, Germany
mail@claudio-moraga.eu

Abstract. Diversity is a key characteristic to obtain advantages of combining predictors. In this paper, we propose a modification of *bagging* to explicitly trade off diversity and individual accuracy. The procedure consists in dividing the bootstrap replicates obtained at each iteration of the algorithm in two subsets: one consisting of the examples misclassified by the ensemble obtained at the previous iteration, and the other consisting of the examples correctly recognized. A high individual accuracy of a new classifier on the first subset increases diversity, measured as the value of the Q statistic between the new classifier and the existing classifier ensemble. A high accuracy on the second subset on the other hand, decreases diversity. We trade off between both components of the individual accuracy using a parameter $\lambda \in [0, 1]$ that changes the cost of a misclassification on the second subset. Experiments are provided using well-known classification problems obtained from UCI. Results are also compared with *boosting* and *bagging*.

Keywords: Ensemble Methods, Bagging, Diversity, Neural Networks, Classification Algorithms.

1 Introduction

Ensemble methods actually constitute an active research area that has lead to a family of learning algorithms applied to a wide range of problems including classification, clustering and regression. These methods are based on the idea of combining a set of simple predictors instead of using only one, maybe more complex. The interesting point is that with an appropriate design, the expected performance of the combined predictor can be better than the average performance of the individual predictors, even if these are weakly good. A key characteristic to obtain such behavior seems to be *diversity* of the combined predictors. This is

^{*} This work was supported in part by Research Grant Fondecyt (Chile) 1070220 and 7070093. Partial support was also received from Research Grant 240725 DGIP-UTFSM (Chile).

intuitively clear, since combining a set of identical models is hardly better than using any single member of the set.

A well-studied method for ensemble generation is *bagging*, introduced by Breiman in [3]. Bagging can be described as an algorithm with only an implicit search for diversity, because no information about the predictions of the other models is incorporated to generate each predictor. Recent works [10] [9] have shown that the action of bagging is to control the effective influence of training examples in the estimation process, which is implicitly implemented by the bootstrapping plan carried out at each round of the algorithm. According to this reasoning, the robustness of bagging to overfitting, is not related to the independence of the different bagging rounds. An explicit search for diversity could hence be incorporated, allowing visibility between the learners while preserving the robustness characteristic of bagging.

In this paper we focus on binary classification, that is, we are given with a set of examples $S = \{(x_k, y_k); k = 1, \dots, m\}$, obtained sampling independently a distribution $P(\mathbf{x}, \mathbf{y})$, where $x_k \in \mathbb{R}^p$ represents a set of features and $y_k \in \{-1, 1\}$ a class label. We are asked to build a classifier $M : \mathbb{R}^p \rightarrow \{-1, 1\}$ capable to recognize the class corresponding to a given $\mathbf{x}$, specifically we are asked to minimize the misclassification probability $E_P(\mathbf{x} \neq M(\mathbf{y}))$.

In this type of problems, diversity can be quantified using the so called Q statistic [13] [11]. For a pair of classifiers, this measure depends on the frequency with which the classifiers coincide and disagree in the recognition and misclassification of the training examples. As stated in [5], diversity can be harmful if this is maximized at the expense of the individual accuracy and hence, an imperfect correlation between diversity and accuracy is observed in practice [11]. In this paper, we propose a modification of bagging, to explicitly trade off diversity and individual accuracy, decomposing the training objective of each learner in a component that contributes to diversity and other that decreases diversity. This algorithm has the same spirit of some algorithms proposed for regression problems, for example the *negative correlation algorithm* [6].

We start by presenting in section 2 the bagging algorithm [3] and a possible justification of its effectiveness. In section 3 we briefly review the concept of diversity in ensembles. In section 3 we show how to modify bagging to explicitly balance diversity and local accuracy. This algorithm inherits the robustness of bagging to overfitting because it preserves at each iteration, the resampling plan characteristic of bagging. In the final section we provide experimental results on classification problems obtained from UCI[2] and systematically used in the literature to analyze classifier ensembles. We also compare the algorithm with other methods, boosting and bagging, using different numbers of component learners.

2 Bagging

One of the most widely used techniques for creating an ensemble is *bagging* (short for Bootstrap Aggregation Learning), where a base classifier is provided with a set of patterns obtained randomly resampling the original set of examples

and then trained independently of the other classifiers. The final hypothesis is obtained as the signum of the averaged predictions. The algorithm is summarized in figure (1)

```

1: Let  $S = \{(x_i, y_i); i = 1, \dots, m\}$  be training set.
2: Generate  $T$  bootstrap samples  $S^t, t = 1, \dots, T$  from  $S$ .
3: for  $t = 1$  to  $T$ 
4:   Train the classifier  $f_t$  with the set of examples  $S^t$  to minimize the classification error  $\sum_j I(y_j \neq f_t(x_j))$ , where  $I(S)$  is the indicator of the set  $S$ .
5:   Set the ensemble predictor at time  $t$  to be  $F^t(x) = \text{sgn}(\frac{1}{t} \sum_{i=1}^t f_i^t(x))$ .
6: end for

```

Fig. 1. Bagging

Breiman [3] presents bagging as a procedure capable to reduce the variance of predictors mimicking averaging over several training sets.

In [14] it is shown that for well behaved loss functions, bagging can provide generalization bounds with a rate of convergence of the same order as Tikhonov regularization. The key observation is that using bagging, an α -stable algorithm can becomes strongly α -stable with appropriate sampling schemes. Strongly α -stable algorithms provide fast rates of convergence from the empirical error to the true expected prediction error.

The key fact in the previous analysis is that certain sampling plans allow some points to affect only a subset of learners in the ensemble. The importance of this effect is also remarked in [10] and [9]. In these works, empirical evidence is presented to show that bagging equalizes the influence of training points in the estimation procedure, in such a way that points highly influential (the so called leverage points) are down-weighted. Since in most situations leverage points are badly influential, bagging can improve generalization by making robust an unstable base learner. From this point of view, resampling has an effect similar to robust M-estimators where the influence of sample points is (globally) bounded using appropriate robustifying functions.

Since in uniform resampling all the points in the sample have the same probability of being selected, it seems counterintuitive that bagging has the ability to selectively reduce the influence of leverage points. The explanation is that leverage points are usually isolated in the feature space. To remove the influence of a leverage point it is enough to eliminate this point from the sample but to remove the influence of a non-leverage point we must in general remove a group of observations. Now, the probability that a group of size K be completely ignored by bagging is $(1 - K/m)^m$ which decays exponentially with K . For $K = 2$ for example $(1 - K/m)^m \sim 0.14$ while $(1 - 1/m)^m \sim 0.368$. This means that bootstrapping allows the ensemble predictions to depend mainly on “common” examples, which in turns allows to get a better generalization.

3 Diversity for Classifier Ensembles

Several works have revealed that diversity is a very important characteristic to get real advantages of combining predictors [13] [11] [12] [5]. In regression problems, diversity can be rigorously defined by the so called *ambiguity decomposition* [5]. It can be easily proved that if a set of individual regressors $f_1, f_2, \dots, f_m$ are linearly combined as $F = \sum_i w_i f_i$, the quadratic loss of the ensemble prediction $F(x)$ at a fixed pattern (x, y) can be decomposed as

$$(y - F)^2 = \sum_i w_i (y - f_i)^2 - \sum_i w_i (f_i - F)^2 \quad (1)$$

This decomposition states that the ensemble accuracy not only depends on the individual accuracies but also on the variability of the individual predictions around the combined prediction. From this result, is clear that differences between individual outputs contribute toward the overall ensemble accuracy, if the individual accuracies are maintained. We can hence define the level of diversity in the ensemble as the second term of equation (1), computed and averaged over the training examples. Similar measures of diversity can be obtained in terms of the correlation between the individual errors.

In classification problems, predictors are usually not aggregated using simple linear combinations and hence equation (1) does not hold. Moreover, if base classifiers can only output discrete class labels, quadratic loss cannot properly be used. Definition of diversity for a set of classifiers is hence a more complex task that has been tackled on different ways. Great part of the work examining classifier diversity is due to Kuncheva [13] [11]. In [13], several diversity measures are analyzed, which can be separated between pairwise and non-pairwise measures. Pairwise measures compute a statistic between a pair of classifiers based on the complete set of examples, and then this is averaged over all possible pairings. Non-pairwise measures in contrast, compute a statistic for each example based on the predictions of the set of classifiers and then the average among the examples. The pairwise measures analyzed in [13] depend on the frequency with which a pair of classifiers agree or disagree in the recognition of a training example. Given two classifiers f_i, f_j we can count the number of examples N^{11} for which both are correct, the number of examples N^{00} for which both are incorrect, the cases N^{10} for which f_i is correct but f_j incorrect and the cases N^{01} for which f_j is correct but f_i incorrect. These four types of situations can be summarized in the following table:

	f_j correct	f_j incorrect
f_i correct	N^{11}	N^{10}
f_i incorrect	N^{00}	N^{01}

From these statistics, several metrics of diversity can be derived. Since different measures show great similarity, Kuncheva recommends the use of the so called Q statistic for easy of interpretation. For a pair of classifiers f_i and f_j this is computed as

$$Q(f_i, f_j) = \frac{N^{11}N^{00} - N^{10}N^{01}}{N^{11}N^{00} + N^{10}N^{01}} \quad (2)$$

If the classifiers tend to misclassify and correctly classify the same instances, the Q statistic will take a positive value achieving a maximum of $+1$. If on the other hand, the classifiers show different patterns of errors, the Q statistic will take a negative value with a minimum value of -1 . For independent classifiers $Q = 0$.

4 Bagging with Asymmetric Costs for Misclassified and Correctly Classified Examples

According to the taxonomy proposed in [5] for diversity creation methods, manipulation of the training data is one the most investigated techniques for ensemble generation. In particular, resampling of the training set is part of the two currently most successful ensemble algorithms: boosting [7] and bagging [3]. Bagging can be considered as an implicit method to generate diversity because it does not use any information about the behavior of the other learners to manipulate the training data. This algorithm can however significantly decrease the generalization error of the base learner because, as we have explained in the previous sections, resampling limits the influence of isolated examples on the ensemble, stabilizing the overall prediction. In contrast to bagging, *Adaboost* is an algorithm explicitly designed to focus each learner in the instances of the training set where the previous learners perform bad.

Suppose we generate an ensemble sequentially, adding at each step t a new classifier h_t to the ensemble generated in the previous steps. Suppose we generate the new classifier to maximize the Q statistic between the existing classifier H_{t-1} (resulting from the combination of the previously generated classifiers) and the new classifier. Hence, if the ensemble correctly classifies a subset C of examples of the training set S and misclassifies a subset M , the new classifier h_t has to misclassify all the examples in the set C but correctly recognize the patterns in M . In this case, $Q(H_{t-1}, f_t) = -1$ and diversity is clearly maximized. However, for reasonable aggregation functions this procedure has only sense if the existing classifier has an accuracy lower than 50% or the margin for the classification of the correctly classified examples is wide enough. It would be possible to overcome this problem using instance-dependent aggregation functions, but this is out of the scope of this paper. On the other hand, the problem of determining which classifiers behave well for a given instance is a classification problem itself, which could be even more complex than the original classification problem. As stated by [5], the problem is that diversity can be harmful if individual accuracy is completely sacrificed to obtain differences between the predictors outputs. For this reason it is difficult to speak about a perfect correlation between the diversity measure and the ensemble accuracy [11]. The latter observation holds also for regression ensembles where clear diversity measures can be defined. For example, the ambiguity decomposition and similar decompositions in regression estimation, have lead to algorithms that look for an adequate tradeoff between

individual accuracy and diversity. In the *negative correlation algorithm* [6] this achieved training each learner with the objective function

$$E_i = \sum_{(x_i, y_i) \in S} (y_i - f(x_i))^2 + \lambda \sum_{i \in S} (f_i - \bar{F})^2 \quad (3)$$

where $\bar{F}$ is the ensemble output. It can be shown that diversity can decrease the ensemble accuracy if this is not balanced with other components of the ensemble error. In this spirit, we propose an algorithm to explicitly trade off diversity and individual accuracy. The procedure works iteratively, by adding a new classifier h_t to the existing ensemble H_{t-1} at each step $t = 1, 2, \dots, M$. Following bagging, each learner is trained using a bootstrap sample S^t of the original set of examples S . Bootstrapping at each iteration allows us to inherit the robustness of bagging to overfitting, which is not based on the independence of the different bagging steps, but on the effects of the resampling plan. Instead of generating each learner independently of the others, we propose to train the learner at step t with the following objective function

$$E_t = \sum_{(x_i, y_i) \in M_{t-1}} I(y_i \neq h_t(x_i)) + \lambda \sum_{(x_i, y_i) \in C_{t-1}} I(y_i \neq h_t(x_i)) \quad (4)$$

where M_{t-1} is the set of examples incorrectly classified by the ensemble obtained at step $t-1$ and C_{t-1} the set of correctly recognized examples. Note that $I(S)$ is the indicator of the set S . Hence, the first component of equation (4) is the total classification error on the set M_{t-1} and the second is the total error on the set C_{t-1} . The greater the first term, the greater the Q statistic between the new classifier and the existing ensemble. Optimization of the second term in contrast, monotonically decreases diversity measured as the value of the Q statistic. Both components of E_t contribute to the individual accuracy of the new classifier. In other words, equation (4) represents a tradeoff between individual accuracy that contributes to generate diversity and individual accuracy that decreases diversity. Note that if $\lambda = 1$, the algorithm coincides with bagging and accuracy is optimized independently of the diversity the new classifier incorporates to the ensemble. If λ is allowed to increase beyond 1, then accuracy that damages diversity would be preferred over accuracy compatible with diversity generation. If λ is allowed to go below 0, then it leads to a better optimization of the Q statistic, but both individual and ensemble accuracy can be seriously damaged, especially when the existing ensemble has performance over the 50%.

Algorithm (2) summarizes the proposal of the paper. The implementation of this procedure can be carried out in several ways. The most natural of them is to consider weights for each example: a weight $\lambda_i = 1$ for the examples (x_i, y_i) misclassified by the previous ensemble and a weight $\lambda_i = \lambda$ for the correctly recognized examples. Another possibility, if the base learner cannot manipulate weights for each example, is to use a sampling plan with probabilities coinciding with the desired weights. This approach is used for example in the *adaBoost* algorithm [7] [8].

The resultant algorithm is similar in spirit to the half-and-half bagging algorithm proposed by Breiman [4]. In this method predictors are generated

- 1: Let $S = \{(x_i, y_i); i = 1, \dots, m\}$ be training set.
- 2: Generate T bootstrap samples $S^t, t = 1, \dots, T$ from S .
- 3: Train the classifier f_1 with S^1 to minimize $\sum_j I(y_j \neq f_1(x_j))$
- 4: **for** $t = 2$ to T
- 5: Classify each example x_j of S^t using the classifier ensemble F^{t-1} . If the example is correctly classified put it in the set C_{t-1} , otherwise put it in the set M_{t-1} .
- 6: Train the classifier f_t to minimize

$$\begin{aligned}
 E_t &= \sum_{j \in M_{t-1}} I(y_j \neq h_t(x_j)) + \lambda \sum_{j \in C_{t-1}} I(y_j \neq h_t(x_j)) \\
 &= \sum_{j \in S_t} \lambda_j I(y_j \neq h_t(x_j))
 \end{aligned}$$

where $\lambda_j = 1$ if $(x_j, y_j) \in M_{t-1}$ and $\lambda_j = \lambda$ if $(x_j, y_j) \in C_{t-1}$.

- 7: Set the ensemble predictor at time t to be $F^t(x) = \text{sgn}(\frac{1}{t} \sum_{i=1}^t f_i^t(x))$.
- 8: **end for**

Fig. 2. The proposed algorithm

sequentially using different subsets of the original training set. At each round, an example is randomly selected and classified by the learners previously generated that did not use the example in their training sets. If the example is misclassified by majority voting on the latter predictions this is incorporated in a set M . Otherwise the example is incorporated in a set C . This process is repeated until the sizes of M and C are equal to $s = n/4$, where n is the total number of examples. The next classifier is trained on the examples of M and C with equal costs. It should be noted that the number of correctly classified examples will be in general greater than the number of misclassified examples, especially when several rounds of the algorithm have been elapsed. Hence, this method implicitly puts more focus than standard bagging in the misclassified examples. The constant proportion of the correctly classified examples in the next training sample on the other hand, ensures each learner achieves a reasonable individual accuracy of the original training set and that misclassified examples do not dominate excessively the next training round. This is a clear difference with the *adaBoost* algorithm [7] [8]. In contrast to standard bagging, *adaBoost* forces the learners to focus on the difficult training examples and pay less attention to the examples correctly classified by the last predictor. This is achieved by computing example weights and then bootstrapping the training data with probabilities coinciding with these weights. Although very effective in practice, *adaBoost* is highly more sensitive to leverage points (in particular outliers) than bagging [1]. In some rounds, *adaBoost* could be completely dominated by few isolated examples more than in local patterns present in the data. With the algorithm proposed in this section, the relative importance of a misclassified example with respect to an already recognized example is kept constant and hence the former cannot dominate excessively.

5 Experimental Results and Discussion

In this section we present results of empirical studies to evaluate the proposed algorithm. In the whole set of experiments, two real and well-known data sets were used, namely *Phoneme* and *Diabetes*. A detailed description of these data

Table 1. Analysis of the parameter λ

	Phoneme		Diabetes	
λ	Training Error	Testing Error	Training Error	Testing Error
0.0	0.2730 ± 0.0810	0.2793 ± 0.0758	0.3503 ± 0.0383	0.3935 ± 0.0368
0.1	0.1687 ± 0.0154	0.1810 ± 0.0111	0.1939 ± 0.0048	0.2600 ± 0.0101
0.2	0.1643 ± 0.0073	0.1777 ± 0.0102	0.1927 ± 0.0049	0.2465 ± 0.0113
0.3	0.1676 ± 0.0090	0.1762 ± 0.0094	0.1896 ± 0.0042	0.2359 ± 0.0107
0.4	0.1659 ± 0.0062	0.1786 ± 0.0077	0.1911 ± 0.0040	0.2384 ± 0.0097
0.5	0.1661 ± 0.0077	0.1774 ± 0.0110	0.1884 ± 0.0035	0.2375 ± 0.0097
0.6	0.1709 ± 0.0081	0.1811 ± 0.0103	0.1892 ± 0.0036	0.2329 ± 0.0105
0.7	0.1691 ± 0.0059	0.1814 ± 0.0107	0.1891 ± 0.0040	0.2314 ± 0.0098
0.8	0.1717 ± 0.0072	0.1805 ± 0.0104	0.1900 ± 0.0033	0.2338 ± 0.0093
0.9	0.1675 ± 0.0065	0.1811 ± 0.0082	0.1908 ± 0.0040	0.2316 ± 0.0095
1.0	0.1730 ± 0.0045	0.1850 ± 0.0118	0.1919 ± 0.0040	0.2333 ± 0.0115

Table 2. Experimental results on the *Phoneme* dataset

N	Boosting	Bagging	Proposal $\lambda = 0.3$
Training Set			
2	0.1992 ± 0.0178	0.1973 ± 0.0186	0.1893 ± 0.0165
4	0.1885 ± 0.0098	0.1846 ± 0.0134	0.1665 ± 0.0077
6	0.1822 ± 0.0111	0.1776 ± 0.0046	0.1726 ± 0.0167
8	0.1840 ± 0.0139	0.1748 ± 0.0043	0.1648 ± 0.0067
10	0.1786 ± 0.0075	0.1777 ± 0.0092	0.1662 ± 0.0088
12	0.1769 ± 0.0071	0.1746 ± 0.0066	0.1654 ± 0.0117
14	0.1753 ± 0.0074	0.1760 ± 0.0076	0.1609 ± 0.0067
16	0.1751 ± 0.0065	0.1737 ± 0.0051	0.1642 ± 0.0087
18	0.1754 ± 0.0062	0.1725 ± 0.0046	0.1587 ± 0.0062
20	0.1746 ± 0.0075	0.1712 ± 0.0048	0.1570 ± 0.0049
Testing Set			
2	0.2058 ± 0.0193	0.2093 ± 0.0205	0.1972 ± 0.0155
4	0.1978 ± 0.0125	0.1965 ± 0.0158	0.1764 ± 0.0101
6	0.1910 ± 0.0161	0.1904 ± 0.0116	0.1822 ± 0.0174
8	0.1937 ± 0.0190	0.1865 ± 0.0112	0.1752 ± 0.0099
10	0.1882 ± 0.0094	0.1863 ± 0.0110	0.1768 ± 0.0118
12	0.1862 ± 0.0093	0.1848 ± 0.0083	0.1765 ± 0.0039
14	0.1863 ± 0.0101	0.1849 ± 0.0082	0.1733 ± 0.0131
16	0.1872 ± 0.0122	0.1853 ± 0.0084	0.1729 ± 0.0091
18	0.1886 ± 0.0123	0.1846 ± 0.0095	0.1729 ± 0.0129
20	0.1867 ± 0.0130	0.1837 ± 0.0091	0.1674 ± 0.0101

Table 3. Experimental results on the *Diabetes* dataset

N	Boosting	Bagging	Proposal $\lambda = 0.7$
Training Set			
2	0.2247 ± 0.0068	0.2083 ± 0.0058	0.2092 ± 0.0062
4	0.2080 ± 0.0061	0.1986 ± 0.0046	0.1966 ± 0.0043
6	0.2013 ± 0.0053	0.1944 ± 0.0045	0.1921 ± 0.0037
8	0.1956 ± 0.0052	0.1921 ± 0.0044	0.1894 ± 0.0042
10	0.1920 ± 0.0054	0.1919 ± 0.0040	0.1891 ± 0.0040
12	0.1896 ± 0.0053	0.1905 ± 0.0038	0.1890 ± 0.0036
14	0.1869 ± 0.0055	0.1899 ± 0.0038	0.1889 ± 0.0039
16	0.1850 ± 0.0054	0.1895 ± 0.0037	0.1880 ± 0.0035
18	0.1835 ± 0.0055	0.1892 ± 0.0035	0.1863 ± 0.0036
20	0.1818 ± 0.0053	0.1892 ± 0.0035	0.1860 ± 0.0036
Testing Set			
2	0.2616 ± 0.0126	0.2467 ± 0.0106	0.2465 ± 0.0107
4	0.2650 ± 0.0125	0.2386 ± 0.0087	0.2386 ± 0.0104
6	0.2621 ± 0.0117	0.2354 ± 0.0105	0.2349 ± 0.0109
8	0.2561 ± 0.0110	0.2343 ± 0.0111	0.2314 ± 0.0101
10	0.2589 ± 0.0125	0.2333 ± 0.0115	0.2314 ± 0.0098
12	0.2588 ± 0.0121	0.2340 ± 0.0104	0.2316 ± 0.0105
14	0.2573 ± 0.0112	0.2305 ± 0.0108	0.2297 ± 0.0104
16	0.2560 ± 0.0116	0.2302 ± 0.0097	0.2319 ± 0.0097
18	0.2596 ± 0.0117	0.2321 ± 0.0099	0.2323 ± 0.0093
20	0.2590 ± 0.0110	0.2306 ± 0.0102	0.2317 ± 0.0095

sets can be obtained from [2]. For comparison purposes, three algorithms will be evaluated: Boosting, Bagging, and the proposed algorithm. In addition, neural networks with five sigmoidal hidden units and trained with standard backpropagation were employed as base learners. For each experiment, *t*-student confidence intervals will be reported with a significance of 0.02 obtained after 50 simulations. The estimation process is carried out with a 75% of the available observations and testing with the rest 25%.

Table (1) shows the proposed algorithm performance by using different values of parameter λ in the *Phoneme* and *Diabetes* datasets.

Table (2) and table (3) show confidence intervals for the *mse* of Boosting, Bagging and the proposed algorithm versus the number of learners in the ensemble, obtained with the *Phoneme* and *Diabetes* dataset respectively. In this experiment, we use the test error optimal values of λ found in the previous experiment. It should be remarked that these values of λ are the optimal found with 10 base learners.

For this paper only two benchmark problems have been chosen, which however allow us to draw preliminary representative conclusions. Table (1) shows clearly that the optimal value of λ is problem dependent, as it was already shown in the case of *Negative Correlation* applied to regression problems [6]. In both benchmarks however, the optimal value of λ was lower than 1. It should be

remarked $\lambda = 1$ is equivalent to bagging. This parameter may be selected using a validation set or adjusted with an evolutionary algorithm. Table (2) illustrates that the proposed method, with $\lambda = 0.3$ reaches a lower mean square error than Boosting and Bagging on the test set. It is known that the Diabetes classification problem suffers from “missing information” and mse results at the level of 0.25 are already considered to be good. Boosting reaches this level with 16 base learners in the ensemble. Bagging obtains the best result (mse 0.2302) with an ensemble of the same size. The proposed method, with $\lambda = 0.7$, obtains a result (mse 0.2314) which is better than Boosting and slightly weaker than Bagging using however only 8 base learners.

References

1. Bauer, E., Kohavi, R.: An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning* 36(1-2), 105–139 (1999)
2. Blake, C.L., Merz, C.J.: UCI repository of machine learning databases (1998)
3. Breiman, L.: Bagging predictors. *Machine Learning* 26(2), 123–140 (1996)
4. Breiman, L.: Half and half bagging and hard boundary points. Technical report, Statistics Department, University of California (1998)
5. Brown, G., Wyatt, J., Harris, R., Yao, X.: Diversity creation methods: A survey and categorisation. *Information Fusion Journal* 6(1), 5–20 (2004)
6. Brown, G., Wyatt, J.L.: Negative correlation learning and the ambiguity family of ensemble methods. In: Windeatt, T., Roli, F. (eds.) *MCS 2003. LNCS*, vol. 2709, Springer, Heidelberg (2003)
7. Freud, Y., Schapire, R.: A decision-theoretic generalization of on-line learning and application to boosting. *J. of Computer and System Sciences* 55(1), 119–137 (1997)
8. Freud, Y., Schapire, R.: A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence* 14(5), 771–780 (1999)
9. Grandvalet, Y.: Bagging down-weights leverage points. In: *IJCNN*, vol. IV, pp. 505–510 (2000)
10. Grandvalet, Y.: Bagging equalizes influence. *Mach. Learning* 55(3), 251–270 (2004)
11. Kuncheva, L.: That elusive diversity in classifier ensembles. In: Perales, F.J., Campilho, A., Pérez, N., Sanfeliu, A. (eds.) *IbPRIA 2003. LNCS*, vol. 2652, pp. 1126–1138. Springer, Heidelberg (2003)
12. Kuncheva, L.: *Combining Pattern Classifiers: Methods and Algorithms*. Wiley InterScience, Chichester (2004)
13. Whitaker, C., Kuncheva, L.: Measures of diversity in classifier ensembles. *Machine Learning* 51, 181–207 (2003)
14. Poggio, T., Rifkin, R., Mukherjee, S.: Bagging regularizes. Technical Report 214/AI Memo 2002-003, MIT CBCL (2002)