Artif Intell Rev (2011) 35:223–240
DOI 10.1007/s10462-010-9192-8

Combining bagging, boosting, rotation forest and random subspace methods

Sotiris Kotsiantis

Published online: 21 December 2010

Abstract Bagging, boosting, rotation forest and random subspace methods are well known re-sampling ensemble methods that generate and combine a diversity of learners using the same learning algorithm for the base-classifiers. Boosting and rotation forest algorithms are considered stronger than bagging and random subspace methods on noise-free data. However, there are strong empirical indications that bagging and random subspace methods are much more robust than boosting and rotation forest in noisy settings. For this reason, in this work we built an ensemble of bagging, boosting, rotation forest and random subspace methods ensembles with 6 sub-classifiers in each one and then a voting methodology is used for the final prediction. We performed a comparison with simple bagging, boosting, rotation forest and random subspace methods ensembles with 25 sub-classifiers, as well as other well known combining methods, on standard benchmark datasets and the proposed technique had better accuracy in most cases.

Keywords Data mining · Machine learning · Pattern recognition · Ensembles of classifiers

1 Introduction

Both empirical studies and specific machine learning applications verify that a given learning algorithm outperforms all others for a particular problem or for a specific subset of the input data, but it is abnormal to find a single expert achieving the best results on the overall problem domain (Dietterich 2001). As a consequence multiple learner systems (an ensemble of classifiers) try to exploit the local different behaviour of the base classifiers to improve the accuracy and the reliability of the overall inductive learning system. There are also hopes that if a learner fails, the overall system can recover the error. Many methods have been proposed for the creation of ensemble of classifiers. Mechanisms that are used to build ensemble of learners include: (i) Using different subset of training data with a single learning technique,

S. Kotsiantis (✉)
Department of Mathematics, University of Patras, Patras, Greece
e-mail: sotos@math.upatras.gr

(ii) Using different learning methods, (iii) Using different training parameters with a single learning method. A handy and informal reasoning, from computational, statistical and representational viewpoints, of why ensembles can get better results is provided in Dietterich (2001). The key for success of ensembles is whether the classifiers in a system are diverse enough from each other, or in other words, that the individual classifiers have a minimum of errors in common. If one classifier makes an error then the others should not be likely to make the same error.

Four of the most popular ensemble algorithms are bagging (Breiman 1996), boosting (Freund et al. 1996), rotation forest (Rodríguez et al. 2006) and random subspace method (Ho 1998). In bagging (Breiman 1996), the training set is randomly sampled k times with replacement, producing k training sets with sizes equal to the original training set. Since the original set is sampled with replacement, some training instances are repeated in the new training sets, and some are not present at all. Theoretical results show that the expected misclassification probability of bagging has the same bias component as a single bootstrap replicate, while the variance component is reduced by a factor N (Fumera et al. 2005). Boosting, on the other hand, induces the ensemble of classifiers by adaptively changing the distribution of the training set based on the accuracy of the previously created classifiers. There are several boosting variants; AdaBoost (Freund et al. 1996, Freund and Schapire 1997) is the most well-known. These methods assign a weight to each example. Initially, all the examples have the same weight. In each iteration a new classifier, named base or weak, is constructed using the base learning method. The construction of the base classifier must take into account the weights distribution. Then, the weight of each example is adjusted, depending on the correctness of the prediction of the base classifier for that example. The final classification is obtained from a weighted vote of the base classifiers.

There are two main differences between bagging and boosting. First, boosting changes adaptively the distribution of the data set based on the performance of previously created learners while bagging changes the distribution of the data set stochastically. Second, boosting uses a function of the performance of a learner as a weight for voting, while bagging utilizes equal weight voting. On the other hand, in random subspace method (Ho 1998) the classifier consists of multiple learners constructed systematically by pseudo-randomly selecting subsets of components of the feature vector, that is, learners constructed in randomly chosen subspaces. The main idea of Rotation Forest (Rodríguez et al. 2006) is to simultaneously encourage diversity by using PCA to do feature extraction for each base classifier and accuracy is sought by keeping all principal components and also using the whole data set to train each base classifier.

Boosting and rotation forest algorithms are considered stronger than bagging and random subspace method on noise-free data; however, bagging and random subspace methods are much more robust than boosting and rotation forest algorithms in noisy settings. For this reason, in this work, we built an ensemble combing bagging, boosting, rotation forest and random subspace version of the same learning algorithm using a voting methodology. We performed a comparison with simple bagging, boosting and random subspace method ensembles as well as other known ensembles on standard benchmark datasets and the proposed technique had better accuracy in most cases. For the experiments, representative algorithms of well known machine learning techniques, such as decision trees, rule learners and Bayesian classifiers were used.

Section 2 presents the most well known algorithms for building ensembles that are based on a single learning algorithm, while Sect. 3 discusses the proposed ensemble method. Experiment results using a number data sets and comparisons of the proposed method with other ensembles are presented in Sect. 4. We conclude with summary and additional research topics in Sect. 5.

2 Ensembles of classifiers

Learning algorithms try to discover a hypothesis in a given space H of hypotheses and in many cases if we have sufficient data they can find the best one for a given problem. But in real cases we have only limited data and sometimes only few instances are available. In these cases the learning algorithm can find different hypotheses that emerge equally accurate with respect to the available training set, and although we can sometimes select among them the simplest or the one with the lowest capacity, we can keep away from the problem combining them to get a good approximation of the unknown true hypothesis.

Hence, there is a growing realization that combinations of learners can be more effective than single learner. Why rely on the best single learner, when a more reliable and accurate result can be obtained from a combination of quite a few? This fundamentally is the reasoning behind the idea of multiple classifier systems. This section provides a short survey of methods for constructing ensembles using a single learning algorithm. This set of ensemble creating techniques relies on varying the data in some manner. Methods of varying the dataset include; sampling, use of different data sources, use of different pre-processing methods, distortion, and adaptive re-sampling.

The Bagging algorithm (Bootstrap aggregating) (Breiman 1996) votes classifiers generated by different bootstrap samples (replicates). Given the parameter T which is the number of repetitions, T bootstrap samples $S_1, S_2, \ldots, S_T$ are generated. From each sample S_i a classifier C_i is induced by the same learning algorithm and the final classifier C* is formed by aggregating T classifiers. A final classification of object x is built by a uniform voting scheme on $C_1, C_2, \ldots, C_T$, i.e., it is assigned to the class predicted most often by these base classifiers, with ties broken arbitrarily.

The core of bagging's potential is found in the majority voting over results from a substantial number of bootstrap samples. As a first approximation, the majority voting helps to cancel out the impact of random variation. However, it still exhibits at least two main open issues: First, there is no clear understanding yet of the conditions under which bagging outperforms an individual classifier. Second, only empirical and rough guidelines have been proposed so far to determine a suitable ensemble size for a task with given computational requirements in terms of memory size and CPU time.

With regard to the former issue, currently the main explanation of bagging operation is given in terms of its capability to reduce the variance component of the misclassification probability, which was related to the degree of "instability" of the base classifier (Breiman 1996), informally defined as the tendency of undergoing large changes in its decision function as a result of small changes in the training set: The more unstable a classifier, the higher the variance component of its misclassification probability and, thus, the improvement attained by bagging. Theoretical investigations of why bagging works has also been found in Bühlman and Yu (2002), Buja (2006), Freidman and Hall (2007). With the influence equalization viewpoint, bagging is interpreted as a perturbation technique aiming at improving the robustness against outliers (Grandvalet 2004).

With regard to the latter issue above, it is well known that bagging misclassification rate tends to an asymptotic value as the ensemble size increases. Works in the literature focused on determining the ensemble size sufficient to reach the asymptotic misclassification rate, empirically showing that suitable values are between 10 and 20 depending on the particular data set and base classifier (Bauer and Kohavi 1999; Opitz and Maclin 1999). Fumera et al. (2008) applied an analytical framework for the analysis of linearly combined classifiers to ensembles generated by bagging. This provides an analytical model of bagging misclassification probability as a function of the ensemble size. This allowed the authors to derive a novel and theoretically grounded guideline for choosing bagging ensemble size according to the dataset characteristics.

If there is too little data, the gains achieved via a bagged ensemble cannot recompense for the decrease in accuracy of individual models, each of which now sees an even less training data. On the other end, if the data set is extremely large and computation time is not an issue, even a single classifier can be quite adequate. Important shortcomings of ensemble methods are the loss of speed in classification with respect to the base classifier and the growth in storage needs with increasing numbers of inducers. In order to remedy these drawbacks, one can try to retain only those classifiers that are essential to solve the classification task at hand and eliminate those whose contribution is redundant. Ensemble pruning reduces the storage needs, speeds up the classification process and has the potential of improving the classification accuracy of the original ensembles. Bootstrap samples are simple random samples of size n selected with replacement from the original n sized sample, so not all bootstrap samples are equally informative, due to the randomness associated to the number of distinct original observations in the bootstrap sample. The variability of this number is neither necessary nor desirable, having negative effects on the performance of the bootstrap technique in certain applications. Dagging (Ting and Witten 1997) produces a number of disjoint, stratified folds out of the data and feeds each chunk of data to a copy of the supplied base learner. Predictions are made via majority vote.

Quite well known is Random Subspace Method (Ho 1998), which consists of training several classifiers from input data sets constructed with a given proportion k of features picked randomly from the original set of features—the author of this method suggested in his experiment to select around 50% of the original set of features. Latinne et al. (2000) proposed to combine the standard bagging with random selection of feature subsets over each bootstrap sample. They showed that this combination outperformed each of the single approach.

Bryll et al. (2003) presented attribute bagging (AB), a similar technique for improving the accuracy and stability of classifier ensembles induced using random subsets of features. AB is a wrapper method that can be used with any learning algorithm. It establishes an appropriate attribute subset size and then randomly selects subsets of features, creating projections of the training set on which the ensemble classifiers are built. The induced classifiers are then used for voting. Bryll et al. (2003) compared the performance of AB method with bagging and other algorithms on a hand-pose recognition dataset. It is shown that AB gives consistently better results than bagging, both in accuracy and stability. Bryll et al. (2003) also demonstrated that ranking the attribute subsets by their classification accuracy and voting using only the best subsets further improves the resulting performance of the ensemble. Stefanowski (2007) studied to integrate bootstrap sampling with more advanced methods of feature selection, which are based on evaluation of the relationship between single feature, or feature subsets, and the target class. Shirai et al. (2008) introduced a method for selecting both samples and features at the same time and demonstrate the effectiveness of the method.

Random forests (Breiman 2001) are another method for constructing ensembles. They derive their strength from two aspects: using random subsamples of the training data (as in bagging) and randomizing the algorithm for learning base-level classifiers (decision trees). The base-level algorithm randomly selects a subset of the features at each step of tree construction and chooses the best among these. Panov and Dzeroski (2007) proposed to use a combination of concepts used in bagging and random subspaces. The latter randomly select a subset of the features at the start and use a deterministic version of the base-level algorithm (and is thus somewhat similar to the randomized version of the algorithm). The results of the authors' experiments show that this approach has a comparable performance to that of random forests, with the added advantage of being applicable to any base-level algorithm without the need to randomize the latter. In classification scenarios, the random resampling procedure in bagging induces some classification margin over the dataset. In addition, when perform bagging in different feature subspaces; the resulting classification margins are likely to be diverse. Cai et al. (2008) took into account the diversity of classification margins in feature subspaces for improving the performance of bagging. Cai et al. (2008) first studied the average error rate of bagging, convert the task into an optimization problem for determining some weights for feature subspaces, and then assigned the weights to the subspaces via a randomized technique in classifier construction.

The resampling mechanism in Bagging leads to that the performance of each weak classifier is independent of one another and the weak classifiers can be built in parallel. However, the training set for each ensemble member of Boosting relies on the performance of the earlier trained classifiers. The probability of a specific instance being selected to be part of a particular classifier's training set depends on the performance of the previously built classifiers on that instance, that is, the probability distribution in each iteration is updated so that the instances being incorrectly predicted by previous classifiers can be chosen more often than those being correctly predicted. Thus, Boosting attempts to generate new classifiers that are able to better classify the "hard" instances for the previous ensemble members. There are several boosting variants; AdaBoost (Freund et al. 1996; Freund and Schapire 1997) is the most well-known. In boosting-by-resampling, the fixed training sample size and training instances resampled according to a probability distribution are used in each iteration; whereas in boosting-by-reweighting, all training instances with weights assigned to each instance are used in each iteration to train the base classifier and this technique is only useful when the weak learning algorithm can handle weighted instances.

Kuncheva et al. (2002) carried out experiments on seven data sets for different sample sizes, different number of classifiers in the ensembles, and the two linear classifiers. On each of these, the authors calculated the measures of diversity and the accuracy of the eight different combination methods, averaged over 50 runs. The results confirmed in a quantitative way the intuitive explanation behind the success of Boosting for linear classifiers for increasing training sizes, and the poor performance of Bagging in this case. Diversity measures indicated that Boosting succeeds in inducing diversity even for stable classifiers whereas Bagging does not.

Schapire et al. (1998) identified two scenarios where AdaBoost is likely to fail: (i) when there is insufficient training data relative to the "complexity" of the base classifiers, and (ii) when the training errors of the base classifiers become too large too quickly. Schapire and Singer (1999) proposed Real AdaBoost, a generalized version of AdaBoost, in which weak classifiers are piece-wise functions whose output is a real value representing the confidence-rated prediction. Normally, to construct such weak classifiers, one splits the input space X into non-overlapping blocks (or subspaces) $X_1, X_2, \ldots, X_N$ so that the predictions of the weak classifier are the same for all instances falling into the same block. In the case of one-feature-based weak classifiers, this is equivalent to dividing the real line into intervals. Meanwhile, determining the appropriate number of bins for weak classifiers learned by Real AdaBoost is a challenging task because small ones might not accurately approximate the real distribution while large ones might cause over-fitting, increase computation time and waste storage space. Freidman et al. (2000) imported an additive logistic regression model into AdaBoost and exactly explained the boosting algorithm from a statistical view. From the perspectives of additive regression model and exponential loss function, Freidman et al. (2000) proposed some new boosting algorithms including "GentleBoost" and "LogitBoost".

Although Adaboost has become a very popular classification method over the past years for its simplicity and adaptivity, there is increasing evidence to show that it is quite susceptible to noise. One of the most convincing experimental studies that find this phenomenon has been reported in Dietterich (2000). In his experiments, Dietterich (2000) compares the performance of Adaboost and bagging on some standard learning benchmarks and investigates the dependence of the performance of these methods on the addition of classification noise to the training data. As expected, the prediction accuracy of both Adaboost and bagging deteriorates as the noisy level increases. However, the increase in the error is much more significant in Adaboost.

Yin et al. (2005) introduced a strategy of boosting based feature combination, where a variant of boosting is proposed for integrating different features. Different from the general boosting, at each round of this variant boosting, some weak classifiers are built on different feature sets, one of which is trained on one feature set. And then these classifiers are combined by weighted voting into a single one as the output classifier of this round. Redpath and Lebart (2005) identified feature subset by the regularized version of Boosting, i.e., AdaboostReg. Additionally, their search strategy is the floating feature search.

Garcıa-Pedrajas and Ortiz-Boyer (2008) proposed a boosting approach to random subspace method (RSM) to achieve an improved performance and avoid some of the major drawbacks of RSM. RSM is a successful method for classification. However, the random selection of inputs, its source of success, can also be a major problem. For several problems some of the selected subspaces may lack the discriminant ability to separate the different classes. These subspaces produce poor classifiers that harm the performance of the ensemble. Garcıa-Pedrajas and Ortiz-Boyer (2008) search subspaces that optimize the weighted classification error given by the boosting algorithm, and then the new classifier added to the ensemble is trained using the obtained subspace.

Shirai et al. (2008) compared boosting with bagging in different strengths of learning algorithms. The authors' conclusions are (1) boosting is effective for simpler algorithms whose difference between training and testing errors is small and bagging is effective for more complicated algorithms whose difference between training and testing errors is large, (2) boosting algorithm using random feature subset selection works almost always regardless of base algorithms, and (3) bagging using random feature subset selection works better than simple bagging on samples.

MultiBoosting (Webb 2000) is another technique of the same category that can be considered as wagging committees formed by AdaBoost. Wagging is a variant of bagging; bagging uses re-sampling to get the datasets for training and producing a weak hypothesis, whereas wagging uses re-weighting for each training instance, pursuing the effect of bagging in a different way. Based on Principal Component Analysis (PCA), Rodríguez et al. (2006) proposed a new ensemble classifier generation technique Rotation Forest. Its main idea is to simultaneously encourage diversity and individual accuracy within an ensemble classifier. Specifically, diversity is promoted by using PCA to do feature extraction for each base classifier and accuracy is sought by keeping all principal components and also using the whole data set to train each base classifier.

Melville and Mooney (2003) presented a new meta-learner (DECORATE, Diverse Ensemble Creation by Oppositional Re-labeling of Artificial Training Examples) that uses an existing "strong" learner (one that provides high accuracy on the training data) to construct a diverse committee. This is accomplished by adding different randomly constructed instances to the training set when building new committee members. These artificially constructed instances are given category labels that disagree with the current classification of the committee, thereby directly increasing diversity when a new learner is trained on the augmented data and added to the committee.

3 Proposed methodology

Several authors (Breiman 1996; Freund et al. 1996; Rodríguez et al. 2006; Ho 1998) have proposed theories for the effectiveness of bagging, boosting, rotation forest and random subspace method based on bias plus variance decomposition of classification error. In this decomposition we can view the expected error of a learning algorithm on a particular target function and training set size as having 3 components:

- A bias term measuring how close the average classifier produced by the learning algorithm will be to the target function;
- A term measuring the minimum classification error associated with the Bayes optimal classifier for the target function (this term is sometimes referred to as the intrinsic target noise).
- A variance term measuring how much each of the learning algorithm's guesses will vary with respect to each other (how often they disagree)

Unlike bagging and random subspace method, which is largely a variance reduction method, boosting appears to reduce both bias and variance. After a base model is trained, misclassified training instances have their weights increased and correctly classified examples have their weights decreased for the purpose of training the next base model. Clearly, boosting attempts to correct the bias of the most recently constructed base model by focusing more attention on the instances that it misclassified. This skill to reduce bias enables boosting to work especially well with high-bias, low-variance base models.

It must be mentioned that weak classifiers can create problems in bagging, especially when the distribution of the response is highly unbalanced. Weak classifiers are sometimes defined as those that do no better than the marginal distribution. Suppose the marginal distribution of the response is unbalanced so that it is very difficult for a model using the predictors to perform better than the marginal distribution. Under those circumstances the rare class will likely be misclassified most of the time because votes will be typically being won by the class that is far more common. To illustrate this point, suppose there is a binary response variable, and for the moment, we are interested in a single observation that happens to be a "success." For a given set of trees, that observation is classified as a success about two times out of ten. So, the classification for that observation will be wrong about 80% of the time. But if one classifies by majority vote, the class assigned would be a failure and that would be wrong 100% of the time. Because the classifier does a poor job, the majority vote produces a disappointing result.

As the boosting algorithms iterate, the error rates on the new training sets for each new learner tends to rise because the examples used to train on become more difficult to learn although the ensemble training error rate decreases. For the two-class case, keeping the error rate below 50% is not too difficult with any reasonable choice of weak learner. It should be made clear that even if the weak learner has a terrible error rate such as 80%, we could always reverse classifications and turn this weak learner into one with an error rate of 20%. Therefore, when we say that we want the error rate less than 50% in the two-class case we mean that weak learner should do better than a random decision. In the multi-case, the error rate still must remain below 50%. For the n-class case, a random classifier will give an error rate of $(n-1)/n$ which can be far from 50%. Therefore, for the multi-class case, we must investigate weak learners that are more powerful than those for the two-class case.

As mentioned in Buja (2006) the main trouble with boosting seems to be robustness to noise. This is expected because noisy examples tend to be misclassified, and the weight will increase for these instances. They present several examples were the performance of boosting algorithms degraded compared to the original algorithms. Bagging uses a voting technique which is not capable to take into account the heterogeneity of the instance space. When the majority of the base classifiers give a wrong prediction for a new instance then the majority vote will result in a wrong classification. The trouble may consist in discarding base classifiers (by assigning small weights) that are highly accurate in a restricted region of the instance space because this accuracy is swamped by their inaccuracy outside the restricted area. It may also consist in the use of classifiers that are accurate in most of the space but still unnecessarily confuse the whole learning committee in some restricted areas of the space. The gain of boosting over bagging is that boosting acts directly to reduce error cases, whereas bagging works indirectly.

Rotation Forest is another successful ensemble classifier generation technique (Rodríguez et al. 2006), in which the training set for each base classifier is formed by applying PCA to rotate the original attribute axes. Specifically, to create the training data for a base classifier, the attribute set F is randomly split into K subsets and PCA is applied to each subset. All principal components are retained in order to preserve the variability information in the data. Thus, K axis rotations take place to form the new attributes for a base classifier. The main idea of Rotation Forest is to simultaneously encourage diversity and individual accuracy within the ensemble: diversity is promoted through doing feature extraction for each base classifier and accuracy is sought by keeping all principal components and also using the whole data set to train each base classifier.

For additional improvement of the prediction of a classifier, we suggest combing bagging, boosting, rotation forest and random subspace methodology with sum rule voting (Vote B&B&R&R). The approach is presented briefly in Fig. 1.

It has been observed that for bagging, boosting, rotation forest and random subspace method, an increase in committee size (sub-classifiers) usually leads to a decrease in prediction error, but the relative impact of each successive addition to a committee is ever diminishing. Most of the effect of each technique is obtained by the first few committee members (Breiman 1996; Freund et al. 1996; Ho 1998). We used 6 sub-classifiers for each sub-ensemble for the proposed algorithm.

The presented ensemble is effective owing to representational reason. The hypothesis space h may not contain the true function f (mapping each example to its real class), but several good approximations. Then, by taking combinations of these approximations, classifiers that lie outside of h may be represented. The core of Vote B&B&R&R algorithm's potential is found in the majority voting over results from a substantial number of samples. The majority voting helps to cancel out the impact of random variation. In addition, when different feature subspaces are used; the resulting classification margins are likely to be diverse. Boosting attempts to generate new classifiers that are able to better classify the "hard" instances for the previous ensemble members.

4 Comparisons and results

For the comparisons of our study, we used 33 well-known datasets mostly from many domains from the UCI repository (Frank and Asuncion 2010). These data sets were selected so as to

```
(input LS learning set; T(=6) number of bootstrap samples; LA learning algorithm output C* classifier)
begin
 for i =1 to T do
  begin
  $S_i$ := bootstrap sample from LS; {sample with replacement}
  $C_i$ := LA ($S_i$) ; {generate a base classifier}
  end;{endfor}
begin
 for i =T+1 to T+6 do
  begin
  $S_i$ := random projection from the d-dimensional input space to a k-dimensional subspace;
  $C_i$ := LA ($S_i$) ; {generate a base classifier}
  end;{endfor}
              Initialize the observation weights $w_k$=1/n, i=1,2,..n
for i =T+7 to T+12  do
begin
         Produce from LA classifier $C_i$ to the training data using weights $w_k$
         Compute $err^{(i)} = \sum_{i=1}^{n} w_k (y \neq C_i) / \sum_{i=1}^{n} w_k$
         Compute $a^{(i)} = \log \frac{1 - err^{(i)}}{err^{(i)}}$
         Set $w_k = w_{k*}\exp(a^{(i)}(y \neq C_i))$
         for i=1 to n Re-normalize $w_k$
      end;{endfor}
for i =T+13 to T+18  do
begin
         The input variables are randomly grouped.
         For each group of input variables:
         – Consider a data set formed by this input variables and all the examples.
         – Eliminate from the data set all the examples from a proper subset of the classes.
         – Eliminate from the data set a subset of the examples.
         – Apply PCA (Principal Component Analysis) with the remaining data set.
         – Consider the components of PCA as a new set of variables. None of the components is discarded.
         $S_i$ := the training data set using as new variables the components selected by PCA for each group
         $C_i$ := LA ($S_i$) ; {generate a base classifier}
  end;{endfor}
Output = $C^* = \arg\max_{y \in K_j} \sum_{i=1}^{T} (C_i(x) = y)$   {the most often predicted class}
End
```

Fig. 1 The Vote B&B&R&R algorithm

come from real-world problems and to vary in characteristics. Thus, we have used data sets from the domains of: pattern recognition (anneal, mushroom, iris, zoo), image recognition (ionosphere, sonar), medical diagnosis (breast-cancer, colic, breast-w, diabetes, heart-c, heart-h, heart-statlog, hepatitis, lymphotherapy, primary-tumor) commodity trading (autos, credit-g) music composition (waveform), computer games (monk1, monk2, kr-vs-kp), various control applications (balance), language morphological analysis (dimin) (Bosch and Daelemans 1999) and prediction of student dropout (student) (Kotsiantis et al. 2003).

Table 1 is a brief description of these data sets presenting the number of output classes, the type of the attributes and the number of instances. In order to calculate the classifiers'

Table 1 Description of the data sets

Data sets	Instances	Categ. features	Numer. features	Classes
Anneal	898	32	6	6
Audiology	226	69	0	24
Autos	205	10	15	6
Badge	294	4	7	2
Breast-cancer	286	9	0	2
Breast-w	699	0	9	2
Colic	368	15	7	2
Credit-g	1,000	13	7	2
Diabetes	768	0	8	2
Dimin	3,949	12	0	5
Haberman	306	0	3	2
Heart-c	303	7	6	5
Heart-h	294	7	6	5
Heart-statlog	270	0	13	2
Hepatitis	155	13	6	2
Hypothyroid	3,772	22	7	4
Ionosphere	351	34	0	2
Iris	150	0	4	3
kr-vs-kp	3,196	35	0	2
Lymphotherapy	148	15	3	4
monk1	124	6	0	2
monk2	169	6	0	2
Primary-tumor	339	17	0	21
Segment	2,310	0	19	7
Sick	3,772	22	7	2
Sonar	208	0	60	2
Soybean	683	35	0	19
Student	344	11	0	2
Titanic	2,201	3	0	2
Vote	435	16	0	2
Vowel	990	3	10	11
Waveform	5,000	0	40	3
Zoo	101	16	1	7

accuracy, the whole training set was divided into ten mutually exclusive and equal-sized subsets and for each subset the classifier was trained on the union of all of the other subsets. Then, cross validation was run 10 times for each algorithm and the average value of the 10-cross validations was calculated.

In the following Tables, we represent with "*" that the specific ensemble looses from the proposed ensemble. That is, the proposed algorithm performed statistically better than the specific ensemble according to t-test with $p < 0.01$. In addition, in Tables, we represent with "v" that the proposed ensemble looses from the specific ensemble according to t-test with $p < 0.01$. In all the other cases, there is no significant statistical difference between the results (Draws). It must be mentioned that the conclusions are based on the resulting differences for $p < 0.01$ because a p value of 0.05 is not strict enough, if many classifiers are compared in numerous data sets (Salzberg 1997).

In the last rows of the Tables one can see the aggregated results in the form (a/b/c). In this notation "a" means that the specific ensemble algorithm is significantly more accurate than the proposed ensemble in a out of 33 data sets, "c" means that the proposed ensemble is significantly more accurate than the specific ensemble in c out of 33 data sets, while in the remaining cases (b), there is no significant statistical difference between the results.

For bagging, boosting and random subspace methodology, much of the reduction in error appears to have occurred after ten to fifteen classifiers. But Adaboosting continues to measurably improve their test-set error until around 25 classifiers (Bauer and Kohavi 1999). For this reason, we used 25 sub-learners for all the tested ensembles of classifiers. The time complexity of the proposed ensemble is about the same with simple bagging, boosting, rotation forest and random subspace methodology with 25 sub-learners. This happens because we use 6 sub-classifiers for each sub-ensemble (totally 24). The proposed ensemble also use less time for training than Multiboost (Webb 2000) and Decorare (Melville and Mooney 2003) combining methods.

In the following subsection, we present the experiment results for different base classifiers. For the experiments, representative algorithms of well known machine learning techniques, such as decision trees, rule learners and Bayesian classifiers were used. We tried to minimize the effect of any expert bias by not attempting to tune any of the algorithms to the specific data set. Wherever possible, default values of learning parameters were used. This naive approach results in lower estimates of the true error rate, but it is a bias that affects all the learning algorithms uniformly.

4.1 Using decision tree as base classifier

Firstly, we used a decision tree algorithm as base classifier in the ensemble. Decision trees are trees that classify examples by sorting them based on attribute values (Murthy 1998). Each node in a decision tree represents an attribute in an example to be classified, and each branch represents a value that the node can take. Examples are classified starting at the root node and sorting them based on their attribute values. The attribute that best divides the training data would be the root node of the tree. The same process is then repeated on each partition of the divided data, creating sub trees until the training data sets are divided into subsets of the same class. However, a decision tree is said to overfit training data if there exists another hypothesis h' that has a larger error than h when tested on the training data, but a smaller error than h when tested on the entire data set. For this reason, there are two common approaches that decision tree algorithms can use to avoid overfitting training data: (1) Stop the training algorithm before it reaches a point in which it perfectly fits the training data, (2) Prune the induced decision tree. The most commonly used C4.5 algorithm (Murthy 1998) was the representative of the decision trees in our study. At each level in the partitioning process a statistical property known as information gain is used by C4.5 algorithm to determine which attribute best divides the training examples. The approach that C4.5 algorithm uses to avoid overfitting is by converting the decision tree into a set of rules (one for each path from the root node to a leaf) and then each rule is generalized by removing any of its conditions that will improve the estimated accuracy of the rule. Decision trees are very unstable in this regard as small perturbations in the training data set can produce large differences in the structure (and predictions) of a model.

Table 2 Comparing the proposed ensemble with other well known ensembles that uses as base classifier the C4.5

Dataset	Vote B&B &R&R C4.5	Bagging C4.5	Dagging C4.5	Boosting C4.5	Multi-Boost C4.5	Decorate C4.5	Rotation forest C4.5	Random-Subspace C4.5
Anneal	99.22	98.79	83.73*	99.61	99.62	98.72	99.22	98.73
Audiology	84.87	80.76*	46.69*	84.62	85.27	81.97*	80.57*	79.08*
Autos	86.24	83.95	50.19*	86.05	86.24	83.38*	84.31	85.37
Badges	100	100	100	100	100	100	82*	99.32*
Breast-cancer	73.16	72.64	71.84	66.6*	68.61*	70.3*	72.76	73.68
Breast-w	96.67	96.12	96.08	96.51	96.55	96.17	97.14	96.5
Colic	85.32	85.29	81.76	81.76*	83.56	84.31	84.51	84.85
Credit-g	75.04	74.27	70.71*	72.79	74.59	72.53	75.8	74.44
Diabetes	75.55	76.38	75.48	72.81	74.67	74.91	76.44	74.61
Dimin	97.36	97.16	89.83*	96.03	96.33	96.95	97.39	95
Haberman	72.16	72.62	73.16	71.12	71.08	72.66	72.86	73.07
Heart-c	81.46	79.47	82.34	79.6	80.16	78.58	83.46	81.39
Heart-h	80.82	80.11	81.92	78.25	79.97	79.09	81.69	81.48
Heart-statlog	82.41	81.19	83.44	80.15	80.93	80.56	83.33	83.59
Hepatitis	83.50	81.5	79.38	82.74	82.93	82.19	83.13	82.81
Hypothyroid	99.64	99.59	98.65	99.67	99.68	98.65	99.66	95.62*
Ionosphere	93.62	92.45	81.05*	93.62	93.5	92.6	93.17	93.71
Iris	94.53	94.67	81*	94.47	94.4	95.33	94.67	94.33
kr-vs-kp	99.51	99.43	94.48*	99.62	99.62	99.53	99.41	96.97
Lymphography	81.71	78.75*	77.41*	83.09	82.42	80.38	86.48v	79.76
monk1	92.18	82.99*	58.71*	96.54	93.49	90.13	96.67	85.97*
monk2	64.58	60.33	60.87	61.86	60.44*	59.19*	70.48v	61.9
Primary-tumor	45.73	45.16	31.74*	41.65*	41.71*	44.22	45.12	45.4
Segment	98.27	97.55	92*	98.42	98.34	98.18	98.23	97.54
Sick	98.90	98.85	97.55	99.06	99.01	97.55	98.91	95.83*
Sonar	82.59	79.78	70.45*	83.03	83.33	78.33*	85.14	82.08
Soybean	94.19	93.15	62.9*	93.21	93.28	94.29	94.58	94.36
Students	85.70	86.29	86.81	81.46*	82.78	80.52*	86.08	86.19
Titanic	78.56	78	77.6	78.89	78.65	79.05	78.83	78.19
Vote	96.64	96.5	95.61	95.33	95.52	94.94	96.79	95.42
Vowel	95.17	91.6*8	56.57*	95.42	95.11	95.96	98.79	95.26
Waveform	83.59	83.02	83.24	83.32	83.64	79.48*	85.52	83.78
Zoo	95.07	93	46.37*	95.38	95.37	93.18	91.18*	94.27
W/D/L		0/29/4	0/17/16	0/29/4	0/30/3	0/26/7	2/28/3	0/28/5

We compare the presented ensemble with bagging, boosting, Random-SubSpace and MultiBoost version of C4.5 (using 25 sub-classifiers), as well as, with DECORATE, Dagging and Rotation Forest combining method using C4.5 as base classifier. In the last raw of the

Table 2 one can see the concentrated results. The presented ensemble is significantly more accurate than Bagging C4.5 and Boosting C4.5 in 4 out of the 33 data sets, while it has significantly higher error rates in none data set. The presented ensemble is significantly more accurate than MultiBoost C4.5 in 3 out of the 33 data sets whilst it has significantly higher error rates in none data set. Furthermore, Dagging C4.5 has significantly lower error rates in none out of the 33 data sets than the proposed ensemble, whereas it is significantly less accurate in 16 data sets. What is more, Rotation Forest C4.5 is significantly more accurate than the proposed ensemble in 2 out of the 33 data sets whilst it has significantly higher error rates in 3 data sets. Moremore, Random-SubSpace C4.5 has significantly lower error rates in none out of the 33 data sets than the proposed ensemble, whereas it is significantly less accurate in 5 data sets. The presented ensemble is significantly more accurate than DECORATE C4.5 in 7 out of the 33 data sets whilst it has significantly higher error rates in none data set.

To sum up, the performance of the presented ensemble is much more accurate than the other well-known ensembles that use only the C4.5 algorithm.

4.2 Using Bayesian algorithm as base classifier

Secondly, we used a Bayesian method as base classifier in the ensemble. A Bayesian network is a graphical model for probabilistic relationships among a set of attributes. The Bayesian network structure S is a directed acyclic graph (DAG) and the nodes in S are in one-to-one correspondence with the attributes. The arcs represent casual influences among the variables while the lack of possible arcs in S encodes conditional independencies. Moreover, an attribute (node) is conditionally independent of its non-descendants given its parents. Using a suitable training method, one can induce the structure of the Bayesian Network from a given training set. In spite of the remarkable power of the Bayesian Networks, there is an inherent limitation. This is the computational difficulty of exploring a previously unknown network. Given a problem described by n attributes, the number of possible structure hypotheses is more than exponential in n. Naive Bayes algorithm was the representative of the Bayesian networks (Domingos and Pazzani 1997). It is a simple learning that captures the assumption that every attribute is independent from the rest of the attributes, given the state of the class attribute.

We compare the presented ensemble with bagging, boosting, Random-SubSpace and MultiBoost version of NB (using 25 sub-classifiers), as well as, with DECORATE, Dagging and Rotation Forest combining method using NB as base classifier. In the last raw of the Table 3 one can see the aggregated results. The presented ensemble is significantly more accurate than Bagging NB in 5 out of the 33 data sets, while it has significantly higher error rates in none data set. The presented ensemble is significantly more accurate than Boosting NB in 4 out of the 33 data sets whilst it has significantly higher error rates in 2 data sets. Furthermore, Dagging NB has significantly lower error rates in one out of the 33 data sets than the proposed ensemble, whereas it is significantly less accurate in 13 data sets. What is more, Rotation Forest NB is significantly more accurate than the proposed ensemble in 3 out of the 33 data sets whilst it has significantly higher error rates in 11 data sets. The presented ensemble is significantly more accurate than DECORATE NB in 6 out of the 33 data sets, while it has significantly higher error rates in one data set. The presented ensemble is significantly more accurate than Random-Subspace NB in 5 out of the 33 data sets whilst it has significantly higher error rates in none data set. To sum up, the performance of the presented ensemble is more accurate than the other well-known ensembles that use only the NB algorithm.

Table 3 Comparing the proposed ensemble with other well known ensembles that uses as base classifier the NB

Dataset	Vote B&B &R&R NB	Bagging NB	Boosting NB	RandomSubSpace NB	Dagging NB	Decorate NB	Rotation Forest NB
Anneal	90.43	86.91*	95.2v	88.8	79.33*	86.59*	90.31
Audiology	77.58	71.79*	78.2	68.48*	28.34*	72.42*	72.11*
Autos	60.00	57.56	57.12*	59.27	46.7*	58.16	63.29v
Badges	99.66	99.69	99.66	99.05	76.77*	96.4*	84.08*
Breast-cancer	72.28	72.84	68.57*	72.14	72.57	73.05	71
Breast-w	96.08	96.08	95.55	96.05	95.78	95.95	95.85
Colic	80.22	78.81	77.46	79.98	77.28	77.99	75.83*
Credit-g	75.34	75.14	75.09	73.83	73.11	75	70.6*
Diabetes	76.02	75.7	75.88	74.89	74.83	75.31	75.66
Dimin	94.41	92.9	92.93	91.93*	82.19*	92.86	47.99*
Haberman	74.93	74.86	73.94	74.31	73.66	74.74	74.81
Heart-c	83.50	83.27	83.14	83.23	81.19	83.31	83.46
Heart-h	84.19	84.02	84.67	83.75	80.62	83.95	83.69
Heart-statlog	83.74	83.48	82.3	83.93	84.67	83.33	83.33
Hepatitis	85.17	84.39	84.23	84.81	79.38*	82.66	83.75
Hypothyroid	95.39	95.46	95.27	94.45	95.69	94.45	81.65*
Ionosphere	86.92	81.91*	91.12	83.08*	91.03	82.62*	89.17
Iris	95.93	95.67	95.07	95.4	92.6	94	96.67
kr-vs-kp	90.49	87.78*	95.1	86.57*	86.81*	87.8*	76.69*
Lymphography	82.85	83.76	80.67	82.54	75.34*	84.38	83
monk1	73.69	73.23	72.37	73.69	70.67	75.77	78.01v
monk2	56.73	56.38	56.83	59.93	57.56	57.43	55.15
Primary-tumor	48.06	49.77	49.71	48.2	41.6*	50.13	38.38*
Segment	81.43	80.31	80.17	79.05	83.35	80.17	84.5v
Sick	92.74	92.77	93.7	95.9	96.1v	95.9v	77.44*
Sonar	70.50	68.11	81.21v	68.33	67.18*	69.31	71.71
Soybean	93.57	92.78	92.02	91.85	71.86*	92.23	93.55
Students	85.91	85.76	85.12	84.74	82.59*	85.51	84.92
Titanic	77.76	77.81	77.86	77.46	77.83	78.33	75.56
Vote	91.70	90.05	95.19v	89.75	89.77	89.9	94.49
Vowel	68.54	63.25*	81.32v	62.71*	70.74	63.13*	60.3*
Waveform	80.30	79.98	80.01	80.06	80.21	80.34	81.3
Zoo	97.33	94.97	97.23	95.35	70.42*	95.09	92.18*
W/D/L		0/28/5	2/27/4	0/28/5	1/19/13	1/26/6	3/19/11

4.3 Using Rule learner as base classifier

Thirdly, we used a rule based algorithm as base classifier in the ensemble. In rule induction systems, a decision rule is defined as a sequence of Boolean clauses linked by logical AND operators that together imply membership in a particular class (Frank and Witten 1998). The general goal is to construct the smallest rule-set that is consistent with the training data. A large number of learned rules is usually a sign that the learning algorithm tries to "remember" the training set, instead of discovering the assumptions that govern it. During classification, the left hand sides of the rules are applied sequentially until one of them evaluates to true, and then the implied class label from the right hand side of the rule is offered as the class prediction. PART algorithm was the representative of the rule based algorithms (Frank and Witten 1998) in our experiments.

We compare the presented ensemble with bagging, boosting, Random-SubSpace and MultiBoost version of PART (using 25 sub-classifiers), as well as, with DECORATE, Dagging and Rotation Forest combining method using PART as base classifier. In the last raw of the Table 4 one can see the aggregated results. The presented ensemble is significantly more accurate than Bagging PART in 2 out of the 33 data sets, while it has significantly higher error rates in one data set. The presented ensemble is significantly more accurate than Boosting PART in 3 out of the 33 data sets whilst it has significantly higher error rates in none data set. Furthermore, Dagging PART has significantly lower error rates in none out of the 33 data sets than the proposed ensemble, whereas it is significantly less accurate in 19 data sets. What is more, Rotation Forest PART is significantly more accurate than the proposed ensemble in 2 out of the 33 data sets whilst it has significantly higher error rates in 3 data sets. The presented ensemble is significantly more accurate than DECORATE PART in 9 out of the 33 data sets, while it has significantly higher error rates in none data set. The presented ensemble is significantly more accurate than Random-Subspace PART in 5 out of the 33 data sets whilst it has significantly higher error rates in none data set. To sum up, the performance of the presented ensemble is more accurate than the other well-known ensembles that use only the PART algorithm.

5 Conclusion

An ensemble of learners is a set of classifiers whose individual decisions are combined in some way (typically by weighted or unweighted voting) to classify new examples. One of the most active areas of research in supervised machine learning has been to study methods for constructing good ensembles of learners. The main discovery is that ensembles are often much more accurate than the individual learners that make them up. The main reason is that many learning algorithms apply local optimization techniques, which may get stuck in local optima. For example, decision trees employ a greedy local optimization approach, and neural networks apply gradient descent techniques to minimize an error function over the training set. As a consequence even if the learning algorithm can in principle find the best hypothesis, we really may not be able to find it. Building an ensemble may achieve a better approximation, even if no assurance of this is given (Bauer and Kohavi 1999).

Boosting and rotation forest algorithms are considered stronger than bagging and random subspace method on noise-free data, however, bagging and random subspace methods are much more robust than boosting and rotation forest in noisy settings. In this work we built an ensemble using a voting methodology of bagging, boosting, rotation forest and random subspace ensembles. It was proved after a number of comparisons with other ensembles, that the proposed methodology gives better accuracy in most cases. The proposed ensemble has been demonstrated to (in general) achieve lower error than either boosting or bagging or random subspace or rotation forest method or other well known ensembles methods when applied to a base learning algorithm and learning tasks for which there is sufficient scope for both bias and variance reduction.

Table 4 Comparing the proposed ensemble with other well known ensembles that uses as base classifier the PART

Dataset	Vote B&B &R&R PART	Bagging PART	Boosting PART	Random-subSpace PART	Dagging PART	Multi-Boost PART	Decorate PART	Rotation forest PART
Anneal	99.29	98.69	99.6	98.73	84.63*	99.45	98.89	99.33
Audiology	85.15	82.68*	85.28	79*	46.55*	85.5	81.32*	81.86*
Autos	84.34	81.98	84.38	83.75	50.56*	83.75	82.33	83.83
Badges	100	100	100	98.92*	100	100	99.67*	83.75*
Breast-cancer	72.07	71.33	68.93*	73.13	72.29	69.38	68.92*	74.16
Breast-w	96.54	96.37	96.5	96.72	96.08	96.61	95.57	96.99
Colic	85.02	85.21	81.3*	84.69	81.9*	83.99	83.96	84.5
Credit-g	75.58	75.18	74.03	75.81	71.56*	74.62	72.3*	77.8
Diabetes	76.03	75.95	73.78	74.96	74.79	74.88	75.53	76.43
Dimin	97.50	97.75	96.51	95.46	85.13*	96.89	95.37	97.09
Haberman	72.97	73.3	72.18	72.84	73.46	72.18	74.48	73.52
Heart-c	82.09	81.95	80.47	83.27	82.61	81.03	79.49	83.8
Heart-h	82.06	82.25	81.47	82.45	82.13	81.53	78.62*	83.71
Heart-statlog	81.74	81.7	80.78	83.44	83.33	81.22	77.78*	81.48
Hepatitis	84.35	83.71	83.52	83.83	79.38*	84.48	82.66	84.46
Hypothyroid	99.63	99.63	99.66	96.26*	98.6	99.66	98.6	99.71
Ionosphere	93.65	92.77	93.51	93.51	80.97*	93.54	92.32	95.17
Iris	94.87	94.6	94.93	94.67	79.8*	94.73	95.33	95.33
kr-vs-kp	99.57	99.46	99.67	97.28	95.68*	99.68	98.59	99.62
Lymphography	82.66	83.2	84.13	83.01	77.21*	83.19	81.67	84.38
monk1	98.65	98.57	98.81	84.74*	61.87*	98.49	91.79*	95.83
monk2	69.10	67.75	66.24	62.49*	60.12*	65.78	58.05*	68.09
Primary-tumor	46.41	45.34	42.21*	46.11	31.77*	42.6*	44.82	44.52
Segment	98.25	97.8	98.45	97.69	91.95*	98.28	97.92	98.18
Sick	98.87	98.79	98.92	96.69	97.44	98.85	97.44	98.65
Sonar	83.18	81.41	83.05	83.75	70.45*	82.95	85.6	88v
Soybean	94.14	93.26	93.22	94.13	56.79*	93.43	93.26	93.99
Students	84.28	84.08	81.37	85.44	85.73	82.43	78.5*	84.92
Titanic	78.90	78.37	78.98	78.27	77.6	78.78	78.92	78.96
Vote	96.39	96.52	94.94	95.56	95.61	95.28	94.49	96.33
Vowel	94.60	91.33	94.32	96.15	57.08*	93.71	96.36	97.98v
Waveform	84.29	84.42	84.19	84.94	83.82	84.19	83.32	86.34
Zoo	95.16	93.2	95.75	94.28	46.37*	94.28	93.27	91.18*
W/D/L		0/32/1	0/30/3	0/28/5	0/14/19	0/32/1	0/24/9	2/28/3

Our approach answers to some extent such questions as generating uncorrelated classifiers and control the number of learners needed to improve accuracy in the ensemble of learners. While ensembles provide very accurate learners, too many learners in an ensemble may limit their practical application. To be feasible and competitive, it is important that the learning algorithms run in reasonable time. In our method, we limit the number of sub-classifiers to 6 in each sub-ensemble.

Nevertheless, there are still some interesting problems deserved to be investigated further, which include but are not limited to the following items.

- Evaluation of the performance of the proposed algorithm by adopting other algorithms such as a neural network as the base learning algorithm.
- How will the proposed method perform if it is extended to solve regression problems?
- How to automatically select the optimal number of learners in each sub-ensemble.

References

Bauer E, Kohavi R (1999) An empirical comparison of voting classification algorithms: bagging, boosting, and variants. Mach Learn 36:105–139

Bosch A, Daelemans W (1999) Memory-based morphological analysis. In: Proceedings of 37th annual meeting of the ACL. University of Maryland, pp 285–292 (http://ilk.kub.nl/~antalb/ltuia/week10.html)

Breiman L (1996) Bagging predictors. Mach Learn 24(3):123–140

Breiman L (2001) Random forests. Mach Learn 45(1):5–32

Bryll R, Gutierrez-Osuna R, Quek F (2003) Attribute bagging: improving accuracy of classifier ensemble by using random feature subsets. Pattern Recognit 36:1291–1302

Bühlman P, Yu B (2002) Analyzing bagging. Ann Stat 30:927–961

Buja WS (2006) Observations on bagging. Statistica Sinica 16:323–351

Cai Q-T, Peng C-Y, Zhang C-S (2008) A weighted subspace approach for improving bagging performance. In: IEEE ICASSP, pp 3341–3344

Dietterich T (2000) An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization. Mach Learn 40:139–157

Dietterich TG (2001) Ensemble methods in machine learning. In: Kittler J, Roli F (eds) Multiple classifier systems. LNCS, vol 1857, pp 1–15

Domingos P, Pazzani M (1997) On the optimality of the simple Bayesian classifier under zero-one loss. Mach Learn 29:103–130

Frank E, Witten IH (1998) Generating Accurate Rule Sets Without Global Optimization. In: Fifteenth international conference on machine learning, pp 144–151

Frank A, Asuncion A (2010) UCI machine learning repository [http://www.archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science

Freidman JH, Hall P (2007) On bagging and nonlinear estimation. J Stat Plan Inference 137(3):669–683

Freidman J, Hastie T, Tibshirani R (2000) Additive logistic regression: a statistical view of boosting. Ann Stat 28:307–337

Freund Y, Robert E Schapire (1996) Experiments with a new boosting algorithm. In: Proceedings of ICML'96, pp 148–156

Freund Y, Schapire RE (1997) A decision-theoretic generalization of on-line learning and an application to boosting. J Comput Syst Sci 55(1):119–139

Fumera G, Roli F, Serrau A (2005) Dynamics of variance reduction in bagging and other techniques based on randomisation. MCS 2005, LNCS vol 3541, pp 316–325

Fumera G, Roli F, Serrau A (2008) A theoretical analysis of bagging as a linear combination of classifiers. IEEE Trans Pattern Anal Mach Intell 30(7):1293–1299

Garcıa-Pedrajas N, Ortiz-Boyer D (2008) Boosting random subspace method. Neural Netw 21:1344–1362

Grandvalet Y (2004) Bagging equalizes influence. Mach Learn 55:251–270

Ho TK (1998) The random subspace method for constructing decision forests. IEEE Trans Pattern Anal Mach Intell 20(8):832–844

Kotsiantis S, Pierrakeas C, Pintelas P (2003) Preventing student dropout in distance learning systems using machine learning techniques. In: Proceedings of 7th international conference on knowledge-based intelligent information and engineering systems (KES), Oxford, Sept. 3–5, Lecture notes series, vol 2774. Springer, pp 267–274

Kuncheva LI, Skurichina M, Duin RPW (2002) An experimental study on diversity for bagging and boosting with linear classifiers. Inf Fusion 3:245–258

Latinne P, Debeir O, Decaestecker Ch (2000) Mixing bagging and multiple feature subsets to improve classification accuracy of decision tree combination. In: Proceedings of the 10th Belgian-Dutch conference on machine learning, Tilburg University

Melville P, Mooney R (2003) Constructing diverse classifier ensembles using artificial training examples. In: Proceedings of IJCAI-2003, pp 505–510, Acapulco, Mexico, August 2003

Murthy (1998) Automatic construction of decision trees from data: a multi-disciplinary survey. Data Min Knowl Discov 2:345–389

Opitz D, Maclin R (1999) Popular ensemble methods: an empirical study. Artif Intell Res 11:169–198

Panov P, Dzeroski S (2007) Combining bagging and random subspaces to create better ensembles, IDA 2007, LNCS 4723, pp 118–129

Redpath DB, Lebart K (2005) Boosting feature selection. In: The third international conference on advances in pattern recognition, Bath, UK, Springer, Berlin, pp 305–314

Rodríguez JJ, Kuncheva LI, Alonso CJ (2006) Rotation forest: a new classifier ensemble method. IEEE Trans Pattern Anal Mach Intell 28(10):1619–1630

Salzberg S (1997) On comparing classifiers: pitfalls to avoid and a recommended approach. Data Min Knowl Discov 1:317–328

Schapire RE, Freund Y, Bartlett P, Lee WS (1998) Boosting the margin: a new explanation for the effectiveness of voting methods. Ann Stat 26:1651–1686

Schapire RE, Singer Y (1999) Improved boosting algorithms using confidence-rated predictions. Mach Learn 37:297–336

Shirai S, Kudo M, Nakamura A (2008) Bagging, random subspace method and biding. SSPR&SPR 2008, LNCS vol 5342, pp 801–810

Stefanowski J (2007) Combining answers of sub-classifiers in the bagging-feature ensembles, RSEISP 2007, LNAI vol 4585, pp 574–583

Ting KM, Witten IH (1997) Stacking bagged and dagged models. In: Fourteenth international conference on machine learning, San Francisco, CA, pp 367–375

Webb GI (2000) MultiBoosting: a technique for combining boosting and wagging. Mach Learn 40:159–196

Yin X-C, Liu C-P, Zhi H (2005) Feature combination using boosting. Pattern Recognit Lett 26:2195–2205