Combining Bagging and Random Subspaces to Create Better Ensembles

Panče Panov and Sašo Džeroski

Department of Knowledge Technologies
Jožef Stefan Institute
Ljubljana, Slovenia
{pance.panov,saso.dzeroski}@ijs.si

Abstract. Random forests are one of the best performing methods 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. We propose to use a combination of concepts used in bagging and random subspaces to achieve a similar effect. 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 our experiments show that the proposed 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.

1 Introduction

Random forests [1] are one of the best performing methods for constructing ensembles of classifiers. They their strength from two aspects: using random subsamples of the training data, on one hand, 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.

We propose to use a combination of bagging [2] and random subspaces [3] to achieve a similar effect. In bagging, we sample the training set and generate random independent bootstrap replicates [4] (random subsamples). The replicates are then used for learning the base-level classifiers of the ensemble. In the random subspace method [3], base-level classifiers are learned from random subspaces of the data feature space. It randomly selects a subset of the features at the start and uses a deterministic version of the base-level algorithm. This procedure is somewhat similar to the randomized version of decision tree classifier used with random forests.

The advantage of this approach over random forests is that it is applicable to any base-level algorithm without the need to randomize the latter. We show that in case of decision trees as base-level algorithm our approach archives comparable results to random forests.

The rest of the paper is organized as follows. Section 2 gives an overview of randomization methods for constructing ensembles of classifiers. In Section 3 we present our combined ensemble method. Section 4 describes the experimental setup. Section 5 covers the results and the discussion and in Section 6 we present the conclusions.

2 Overview of Randomization Methods for Constructing Ensembles of Classifiers

Let us consider the standard supervised learning problem. A learning algorithm is given training examples S of the form $S = \{(X_1, Y_1), ..., (X_n, Y_n)\}$, where n is the number of training samples, for some unknown function $Y = f(X)$. The X_j values are typically vectors of the form $X_j = (x_{j1}, x_{j2}, ..., x_{jp})$ $(j = 1, 2, ..., n)$ where x_{jk} is the value of the k-th feature of example X_j and p is the number of features.

The Y values are typically drawn from a discrete set of classes $\{c_1, ..., c_K\}$ in the case of classification or from the set of numbers in the case of regression. In this work we consider only the task of classification.

The output of the learning algorithm is a classifier which is a hypothesis about the true function f. Given new examples X it predicts the corresponding Y values. An ensemble of classifiers is a set of classifiers whose individual decisions are combined by using some voting scheme (typically by weighted or unweighted voting) to classify new examples. We will denote the ensemble of classifiers by $E = \{C_1, C_2, ..., C_B\}$, where B is the number of classifiers.

Since there is no point in combining classifiers that always make similar decisions, the aim is to be able to find a set of base-level classifiers that will differ in their decisions so that they can complement each other. There are different possibilities how this can be achieved.

One possibility is to have different training sets to train the different base-level classifiers. This can be done randomly by creating random training sets from the given sample as is the case of bagging [2] or by using a random subset of features from the feature set as is the case in the random subspace method [3]. The classifiers can also be trained in series so that instances on which the preceding base-level classifiers are not accurate are given more emphasis in training the next base-level classifiers, like in boosting [5].

Another possibility of inducing ensembles of classifiers is to use randomized versions of the base-level algorithms or to use different algorithms all together to train the base-level classifiers.

Also, combined approaches exist that introduce variations in the training set (e.g bagging) and also use randomized versions of base-level learners (e.g decision trees). This is the case with random forests introduced by Breiman [1].

In this work we focus on combining bagging and the random subspace method to achieve comparable performance to random forests. In that context, in the remaining part of this section we present an overview of bagging, the random subspace method and random forests.

2.1 Bagging

Bagging is based on the concepts of bootstrapping [4] and aggregating, and was introduced by Breiman [2]. Bootstrapping is based on random sampling with replacement. Therefore, taking a bootstrap replicate $S^i = \left(X_1^i, X_2^i, ..., X_n^i\right)$ of the training set $S = (X_1, X_2, ..., X_n)$, one can sometimes have less misleading training instances in the bootstrap training set. Consequently, a classifier constructed on such a training set may have better performance. Aggregating actually means combining of classifiers.

Often, an ensemble of classifiers gives better results than its individual base classifiers because it combines the advantages of the base-level classifiers. Bagging gives good results when unstable learning algorithms (e.g. decision trees) are used as base-level classifiers, where small changes in the training set result in largely different classifiers. The bagging algorithm is presented in Table 1.

Table 1. Bagging

```
Input: Training examples $S$, Bag size $B$
Output: Ensemble $E$
  $E \Leftarrow 0$
  for $i = 1$ to $B$ do
    $S^i \Leftarrow BootstrapSample(S)$
    $C^i \Leftarrow ConstructClassifier(S^i)$
    $E \Leftarrow E \cup \left\{C^i\right\}$
  end for
  return $E$
```

When bootstrapping the training set $S = (X_1, X_2, ..., X_n)$, the probability that the training instance X_j is selected m times in a bootstrap sample S^i is given in Equation 1.

$$b\left(m|n, \frac{1}{n}\right) = C_n^m \left(\frac{1}{n}\right)^m \left(1 - \frac{1}{n}\right)^{n-m}, C_n^m = \frac{n!}{m!(n-m)!} \tag{1}$$

For large n, the binomial distribution can be approximated by the Poisson distribution, so that each instance has a probability of approximately $\frac{1}{e}$ of being left out of a bootstrap sample. Therefore, on average, approximately 37% of the instances are not present in the bootstrap sample. This means that possible outliers in the training set sometimes do not show up in the bootstrap sample. By that, better classifiers (with smaller error) may be obtained from the bootstrap sample than from the original training set.

2.2 The Random Subspace Method

The random subspace method (RSM) is an ensemble construction technique proposed by Ho [3]. In the RSM, the training set is also modified as in bagging. However, this modification is performed in the feature space (rather than example space).

Let each training example X_j in the training sample set S be a p-dimensional vector $X_j = (x_{j1}, x_{j2}, ..., x_{jp})$. In the RSM, we randomly select p^* features from the training set S, where $p^* < p$. By this, we obtain the p^* dimensional random subspace of the original p-dimensional feature space. Therefore, the modified training set $\tilde{S} = \left(\tilde{X}_1, \tilde{X}_2, ..., \tilde{X}_n\right)$ consists of p^*-dimensional training examples $\tilde{X}_j = (x_{j1}, x_{j2}, ..., x_{jp^*})\,(j = 1, 2, ..., n)$. Afterwards, base-level classifiers are constructed from the random subspaces $\tilde{S}^i$ (of the same size), $i = 1, 2, ..., B$, and they are combined by a voting scheme to obtain a final prediction. The RSM algorithm is presented in Table 2.

Table 2. Random subspace method

```
Input: Training examples $S$, Number of subspaces $B$, Dimension of subspaces $p^*$
Output: Ensemble $E$
  $E \Leftarrow 0$
  for $i = 1$ to $B$ do
    $\tilde{S}^i \Leftarrow SelectRandomSubspace(S, p^*)$
    $C^i \Leftarrow ConstructClassifier(\tilde{S}^i)$
    $E \Leftarrow E \cup \{C^i\}$
  end for
  return $E$
```

The RSM may benefit from using both random subspaces for constructing the classifiers and aggregating the classifiers. When the number of training examples is relatively small as compared with the data dimensionality, by constructing classifiers in random subspaces one may solve the small sample problem. In this case the subspace dimensionality is smaller than in the original feature space, while the number of training objects remains the same. When the dataset has many redundant features, one may obtain better classifiers in random subspaces than in the original feature space. The combined decision of such classifiers may be superior to a single classifier constructed on the original training set in the complete feature space.

The RSM was originally developed for decision trees, but the methodology can be used to improve the performance of other unstable classifiers (e.g. rules, neural networks etc.). The RSM is expected to perform well when there is a certain redundancy in the data feature space [3]. It is noticed that the performance of the RSM is affected by problem complexity (feature efficiency, length of class boundary etc.)[6]. When applied to decision trees, the RSM is superior to a single classifier and may outperform both bagging and boosting [3].

2.3 Random Forests

Random forests, introduced by Breiman [1], have been shown to be a powerful classification and regression technique. In bagging, single models are induced over bootstrap samples of the training data, and the classification is made by using some voting scheme. Random forests is a particular implementation of bagging in which each model is a random tree. A random tree is grown according to the CART [7] algorithm with one exception: for each split, rather than considering all possible splits, only a small subset of randomly selected splits is considered (e.g. a random subset of input features), and the best split is chosen from this subset. There are two random steps when inducing the trees: the bootstrap sample for each tree, and random selection of features to split on at every node of the tree.

Let n be the number of training examples, and p the number of variables in the classifier. Let f be the number of input variables to be used to determine the decision at each node of the tree and $f << p$ (usually $f = \sqrt{p}$ or $f = \lfloor log_2(p) + 1 \rfloor$). The algorithm for constructing random forests is presented in Table 3.

Table 3. Random forest

```
**Input:** Training examples $S$,Bag size $B$, Proportion of features considered $f$
**Output:** Ensemble$E$
  $E \Leftarrow 0$
  **for** $i = 1$ to $B$ **do**
    $S^i \Leftarrow BootstrapSample(S)$
    $C^i \Leftarrow BuildRandomTreeClassifier(S^i, f)$
    $E \Leftarrow E \cup \{C^i\}$
  **end for**
  **return** $E$
```

3 Combining Bagging and Random Subspace Method

In this section, we present an ensemble construction scheme in which new learning sets are generated on the basis of both bagging and random subspaces.

In this combined method, the training set is modified in two ways. First, the modification is performed in the training set by taking bootstrap replicates $S^i = \left(X_1^i, X_2^i, ..., X_n^i\right)$ of the training set $S = (X_1, X_2, ..., X_n)$. After that, a modification is performed in the feature space (like in the RSM) on every bootstrap replicate taken from the training set.

Let each example X_j^i $(j = 1, 2, ..., n; i = 1, 2, ...B)$ of a bootstrap replicate $S^i = \left(X_1^i, X_2^i, ..., X_n^i\right)$ be a p-dimensional vector $X_j^i = \left(x_{j1}^i, x_{j2}^i, ..., x_{jp}^i\right)$. We randomly select $p^* < p$ features from every bootstrap replicate X^i. By this, we obtain the p^* dimensional random subspace of the original p-dimensional feature space. Therefore, the modified training set $\tilde{S}^i = \left(\tilde{X}_1^i, \tilde{X}_2^i, ..., \tilde{X}_n^i\right)$ consists of p^*-dimensional training examples $\tilde{X}_j^i = \left(x_{j1}^i, x_{j2}^i, ..., x_{jp^*}^i\right)$ $(j = 1, 2, ..., n)$, where the p^* components x_{jk}^i $(k = 1, 2, ..., p^*)$ are randomly selected from p components

Table 4. SubBag

```
Input: Training examples $S$, Bag size $B$, Dimension of the subspaces $p^*$
Output: Ensemble $E$
  $E \Leftarrow 0$
  for $i = 1$ to $B$ do
    $S^i \Leftarrow BootstrapSample(S)$
    $\tilde{S^i} \Leftarrow SelectRandomSubspace(S^i, p^*)$
    $C^i \Leftarrow ConstructClassifier(\tilde{S^i})$
    $E \Leftarrow E \cup \{C^i\}$
  end for
  return $E$
```

x^i_{jk} $(j = 1, 2, ..., p)$ of the training vector X^i_j (the selection is the same for each training vector). One then constructs base-level classifiers in the random subspaces $\tilde{S^i}$ (of the same size), $i = 1, 2, ..., B$, and combines them with a voting scheme in the final prediction rule. We name this algorithm SubBag and it is presented in Table 4.

4 Implementation and Experimental Setup

For the experimental evaluation of the proposed method, we used the WEKA [8] data mining environment. In the original implementation some of the ensemble methods like bagging and random forests were already implemented. We implemented the random subspace method and our proposed combined method labeled as SubBag. The reason why the WEKA environment was chosen is that it offers a variety of learning algorithms and a well defined framework for development of new algorithms. The WEKA Experimenter was used for testing and comparing the performances of different learning algorithms.

Experiments were performed on 19 datasets taken from the UCI repository [9]. We selected datasets from the repository that have different number of training instances, different number of features, and different number of classes. A summary of the datasets used is presented in Table 5.

In this work, we compared four ensemble learning algorithms: SubBag, the random subspaces method (RSM), bagging and random forests. Experiments were conducted using three base-level algorithms: J48 which is WEKA's implementation of C4.5 [10] decision tree, JRip which is WEKA's implementation of the RIPPER [11] rule learner and the nearest neighbor algorithm IBk [12].

The comparisons of the ensemble algorithms were made with same number of classifiers (B=50). For the random subspace method and SubBag 75% of the input features were randomly selected for every classifier. With random forests, we choose $\lfloor log_2(p) + 1 \rfloor$ (where p is the number of input variables to the classifier) variables to be randomly selected for determining the spliting attribute at every node of the tree. Base-line algorithms were used with parameters as specified here: J48 algorithm was used to induce unpruned trees, JRip was used to produce unpruned rules and IBk algorithm was used with 1 nearest neighbour and the search

Table 5. Datasets used in the experiments

Dataset	Training set size	Number of features	Number of classes
arrhythmia	452	279	16
audiology	226	69	24
autos	205	25	7
cylinder-bands	540	39	2
dermatology	336	34	6
hepatitis	155	19	2
hypothyroid	3772	29	4
ionosphere	351	34	2
optdigits	5620	64	10
sick	3772	29	2
sonar	208	60	2
spambase	4601	57	2
kr-vs-kp	3196	36	2
lung-cancer	32	57	3
mfeat	2000	76	10
molecular-biology-promoters	106	58	2
mushroom	8124	22	2
splice	3190	61	3
sponge	76	45	2

was performed with linear nearest neighbour search algorithm with Eucilidian distance function. For all datasets a stratified 10-fold cross-validation was performed. A paired T-test was employed for testing the difference in performances of the algorithms on every dataset seperately. For testing whether the difference in predictive perfomance between the different methods is statisticaly significant over all datasets, we use the Wilcoxon test [13] that is sugested in [14].

5 Results and Discussion

In this section we present the results of the experiments. We first present the results of using the J48 decision tree algorithm as base-level classifier then the the JRip rule learner as a base-level classifier and nearest neighbor algorithm IBk.

In Table 6, we present the percentage of correctly classified instances using J48 as base-level classifier. A statistical comparison was performed with the SubBag method. From the results we can see that the SubBag method is on average comparable with random forests and performs better than random subspace method and bagging. Random forests preform statistically better than SubBag for the cylinder-bands and sonar datasets. Bagging performs statistically better than SubBag on one dataset, kr-vs-kp. The last column shows the percentage of correctly classified instances for the baseline algorithm J48[1]. It is obvious from the results that all ensemble methods improve the accuracy of the baseline classifier.

[1] The baseline algorithm was run with confidence parameter C=0.25 and minimum number of instances in a leaf parameter M=2.

Table 6. The accuracy of the methods using J48 as base-level classifier (the method that has the best performance for a given dataset is marked in bold, results from the performed T-test are marked with ∘ if there is statistically significant improvement or with • if there is statistically significant degradation with as compared to to SubBag method)

Dataset	SubBag	Random subspaces	Bagging	Random forest	J48
arrhythmia	**72.36**	68.81	71.92	67.48 •	64.38 •
audiology	86.23	**86.70**	84.45	80.04 •	77.87 •
autos	**87.76**	86.76	86.31	86.24	81.88
cylinder-bands	64.26	64.81	62.59	**75.37** ∘	57.78 •
dermatology	**97.55**	94.55 •	95.65	96.46	94.00
hepatitis	84.46	81.96	83.21	**84.54**	83.79
hypothyroid	**99.63**	99.58	99.60	99.36	99.58
ionosphere	93.46	92.32	93.17	**94.03**	91.46
optdigits	97.70	97.38	96.23 •	**98.11**	90.69 •
sick	98.67	98.99	**99.05**	98.41	98.81
sonar	80.79	76.93	77.43 •	**87.55** ∘	71.17
spambase	95.37	95.28	94.76	**95.70**	92.98 •
kr-vs-kp	99.44	99.34	**99.72** ∘	99.25	99.44
lung-cancer	70.83	**77.50**	70.83	67.50	**77.50**
mfeat	**84.45**	82.15 •	82.15 •	83.55	75.25 •
molecular promoters	89.55	83.73	85.45	**90.45**	80.82
mushroom	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
splice	**95.27**	94.83	94.61	93.48	94.08
sponge	93.57	93.57	93.57	**93.75**	92.50

Table 7. Comparison of the predictive performance of ensemble methods using J48 as base-level classifier with Wilcoxon test

SubBag > Random subspaces	$p = 0.040043$
SubBag > Bagging	$p = 0.001324$
SubBag > Random forests	$p = 0.489783$
SubBag > J48	$p = 0.001713$

In Table 7, we present the results of applying the Wilcoxon statistical test to the accuracies achieved using J48 as a base level classifier. We compared our proposed method with random subspaces, bagging, random forests, and the baseline method. In the results, $M1 > M2$ means that method $M1$ has better predictive performance than method $M2$. The significance is reported by adding the corresponding p-value. From the results we can see that SubBag archives statistically significant improvement over random subspaces, bagging and the baseline method. SubBag performs equally well on these datasets when compared to random forests.

In Table 8, we present the percentage of correctly classified instances using JRip as base-level classifier. In this case SubBag was statistically compared with

Table 8. The accuracy of the methods using JRip as base-level classifier (for notation see Table 6)

Dataset	SubBag	Random subspaces	Bagging	JRip
arrhythmia	**72.82**	72.14	74.81	70.80
audiology	78.28	76.05	**78.70**	72.98
autos	**83.86**	**83.86**	85.83	73.10 ●
cylinder-bands	78.15	79.07	**79.26**	65.19 ●
dermatology	**95.63**	94.80	92.89 ●	86.88 ●
hepatitis	**83.88**	83.21	80.63	78.00
hypothyroid	95.57	95.71	**99.42** ○	99.34 ○
ionosphere	**94.03**	93.46	92.89	89.75 ●
optdigits	**98.17**	97.94	97.88	90.78 ●
sick	95.97	96.24	**98.67** ○	98.22 ○
sonar	**86.55**	83.10	83.67	73.07 ●
spambase	93.28	92.78	**94.61** ○	92.39
kr-vs-kp	93.74	93.09	**99.47** ○	99.19 ○
lung-cancer	**80.83**	**80.83**	**80.83**	78.33
mfeat	83.60	**83.75**	83.60	73.15 ●
molecular promoters	**88.55**	80.36	87.55	82.91
mushroom	**100.00**	**100.00**	**100.00**	**100.00**
splice	93.73	92.79	**96.11** ○	93.70
sponge	**92.50**	**92.50**	**92.50**	**92.50**

random subspace method and bagging as well as the baseline method JRip[2]. From the results we can see that the SubBag method is comparable to the random subspace method. Bagging performs statistically better on several datasets (hypothyroid, sick, spambase, kr-vs-kp and splice) that have a large number of training examples. The baseline method also performs statistically better on three datasets (hypothyroid, sick and kr-vs-kp) possibly because of the included phase of optimization and revision of the produced rules that is a part of the RIPPER algorithm.

In Table 9 we present the results of applying the Wilcoxon statistical test to the performance of ensemble methods using JRip as a base level classifier. From the results we can see that SubBag achives statistically significant improvement over random subspaces and the baseline method. SubBag on these datasets performs statistically worse than bagging.

In Table 10, we present the percentage of correctly classified instances using IBk as base-level classifier. In this case SubBag was statistically compared with the random subspace method, bagging and the baseline method IBk [3]. From

[2] The baseline algorithm was run with the following parameters: number of folds for pruning F=3, minimum total weight of the instance in a rule N=2 and number of optimizations O=2.

[3] IBk algorithm was used with 1 nearest neighbor and the search was performed with linear nearest neighbor search algorithm with Eucilidian distance function

Table 9. Comparison of ensemble methods using JRip as base-level classifier using Wilcoxon test

SubBag > Random subspaces	$p = 0.008590$
SubBag < Bagging	$p = 0.061953$
SubBag > JRip	$p = 3.38454e^{-4}$

Table 10. The accuracy of the methods using IBk as base-level classifier (for notation see Table 6)

Dataset	SubBag	Random subspaces	Bagging	IBk
arrhythmia	**58.84**	58.40	53.55 ●	52.88 ●
audiology	**80.04**	78.26	76.94	77.81
autos	78.88	**79.86**	75.93	75.93
cylinder-bands	73.70	**74.44**	**74.44**	**74.44**
dermatology	**96.72**	95.65	94.54	94.54
hepatitis	**85.13**	83.88	80.63	80.63
hypothyroid	94.35	**94.41**	91.73 ●	91.52 ●
ionosphere	91.17	**92.60**	86.33 ●	86.33 ●
optdigits	98.88	**99.02**	98.61	98.61
sick	96.29	**96.58**	96.24	96.18
sonar	88.50	**89.48**	86.10	86.57
spambase	**94.07**	93.91	91.26 ●	90.78 ●
kr-vs-kp	**96.81**	96.62	96.62	96.28
lung-cancer	**70.83**	67.50	64.17	67.50
mfeat	**82.25**	82.20	80.10 ●	80.20 ●
molecular promoters	**87.82**	85.73	86.09	82.27
mushroom	**100.00**	**100.00**	**100.00**	**100.00**
splice	**92.04**	91.82	77.27 ●	74.67 ●
sponge	**95.00**	**95.00**	93.57	92.14

Table 11. Comparison of ensemble methods using kNN as base-level classifier using Wilcoxon test

SubBag > Random subspaces	$p = 0.331723$
SubBag > Bagging	$p = 2.7271e^{-4}$
SubBag > kNN	$p = 2.7271e^{-4}$

the results we can see that the SubBag method is comparable to the random subspace method.

In Table 11 we present the results of applying Wilcoxon statistical test to the performance of ensemble methods using IBk as a base level classifier. From the results we can see that SubBag achieves statistically significant improvement over bagging and the baseline method. SubBag on these datasets performs comparably well to random subspaces.

6 Conclusion and Further Work

In this work we present a new method for constructing ensembles of classifiers in which new learning sets are generated on the basis of bagging and random subspaces. This method was compared with other ensemble methods that use randomization to induce classifiers (bagging, random subspace method and random forests). Comparison was performed on 19 datasets from the UCI domain. As base-level classifiers we used J48 decision tree, JRip rule learner and the IBk nearest neighbor learner. By application of a paired T-test we investigated the statistical differences between these methods in terms of classification accuracy on every dataset separately. We also employed the Wilcoxon test to test the statistical significance of the methods over all datasets. The experimental results obtained show that in the case of J48 as a base-level classifier, SubBag performs comparably to random forests. The added advantage of this combination scheme is that it is applicable to any base-level algorithm without the need to randomize the algorithm itself. In case of JRip as a base-level classifier our method is statistically comparable to bagging and better than random subspace method. For the case of IBk as a base level classifier it performs better than random subspaces and bagging.

As further work, we plan to investigate the diversity of the members of the ensemble of classifiers induced by our combined approach and compare it to other ensemble methods in this respect. Another possibility to investigate is using a different combination of bagging and random subspaces method (e.g. bags of RSM ensembles and RSM ensembles of bags). Finally, a comparison of bagged ensembles of randomized base-level classifiers (e.g. a randomized version of JRip) would be of interest.

Acknowledgments. This work was supported by the EU FP6-516169 project "Inductive Queries for Mining Patterns and Models". The authors would like to thank Dragi Kocev and Ivica Slavkov for providing valuable suggestions for improving this manuscript.

References

1. Breiman, L.: Random forests. Mach. Learn. 45(1), 5–32 (2001)
2. Breiman, L.: Bagging predictors. Machine Learning 24(2), 123–140 (1996)
3. Ho, T.K.: The random subspace method for constructing decision forests. IEEE Transactions on Pattern Analysis and Machine Intelligence 20(8), 832–844 (1998)
4. Efron, B., Tibshirani, R.J.: An introduction to the Bootstrap. Monographs on Statistics and Applied Probability, vol. 57. Chapman and Hall (1993)
5. Schapire, R.E.: The strength of weak learnability. Machine Learning 5, 197–227 (1990)
6. Ho, T.K.: Complexity of classification problems and comparative advantages of combined classifiers. In: MCS '00: Proceedings of the First International Workshop on Multiple Classifier Systems, London, UK, pp. 97–106. Springer, Heidelberg (2000)

7. Breiman, L., Friedman, J., Olshen, R., Stone, C.: Classification and Regression Trees. Wadsworth and Brooks, Monterey, CA (1984)
8. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques (Morgan Kaufmann Series in Data Management Systems), 2nd edn. Morgan Kaufmann, San Francisco (2005)
9. Newman, D.J., Hettich, S., Blake, C., Merz, C.: UCI repository of machine learning databases (1998)
10. Quinlan, J.R.: C4.5: programs for machine learning. Kaufmann Publishers Inc., San Francisco, CA, USA (1993)
11. Cohen, W.W.: Fast effective rule induction. In: Prieditis, A., Russell, S. (eds.) Proc. of the 12th International Conference on Machine Learning, Tahoe City, CA, pp. 115–123. Morgan Kaufmann, San Francisco (1995)
12. Aha, D.W., Kibler, D., Albert, M.K.: Instance-based learning algorithms. Mach. Learn. 6(1), 37–66 (1991)
13. Wilcoxon, F.: Individual comparisons by ranking methods. Biometrics 1, 80–83 (1945)
14. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. 7, 1–30 (2006)