Enhanced Bagging (eBagging): A Novel Approach for Ensemble Learning

Goksu Tuysuzoglu[1] and Derya Birant[2]
[1]Graduate School of Natural and Applied Sciences, Dokuz Eylul University, Turkey
[2]Department of Computer Engineering, Dokuz Eylul University, Turkey

Abstract: *Bagging is one of the well-known ensemble learning methods, which combines several classifiers trained on different subsamples of the dataset. However, a drawback of bagging is its random selection, where the classification performance depends on chance to choose a suitable subset of training objects. This paper proposes a novel modified version of bagging, named enhanced Bagging (eBagging), which uses a new mechanism (error-based bootstrapping) when constructing training sets in order to cope with this problem. In the experimental setting, the proposed eBagging technique was tested on 33 well-known benchmark datasets and compared with both bagging, random forest and boosting techniques using well-known classification algorithms: Support Vector Machines (SVM), decision trees (C4.5), k-Nearest Neighbour (kNN) and Naive Bayes (NB). The results show that eBagging outperforms its counterparts by classifying the data points more accurately while reducing the training error.*

Keywords: *Bagging, boosting, classification algorithms, machine learning, random forest, supervised learning.*

Received July 31, 2018; accepted December12, 2019
https://doi.org/10.34028/iajit/17/4/10

1. Introduction

Ensemble learning combines several base models, where a traditional algorithm is used to learn each of them. It aggregates the outputs from a set of different classifiers to correctly classify new data points. Constructing ensemble classifiers is extremely useful for high dimensional and large dataset problems that finding an individual classifier in one step is impossible due to the scale and the complexity of the problem.

Bagging is one of the most popular and successful ensemble learning algorithms for improving classification accuracy. Bagging (bootstrap aggregating) was introduced by Breiman [5] as a technique to aggregate various versions of an unstable estimator, each of which is generated from a bootstrap sample. It has provided considerable performance gains over a single learner in many application domains [25]. However, a drawback of bagging is that it generates the training sets by choosing instances completely randomly from the original dataset and so it doesn't consider previously correct or incorrect classified instances in the selection process. In other words, it doesn't focus the underlying learning algorithm on the training examples that have been misclassified previously. In this paper, we present and compare an enhanced version of bagging method which has specifically aimed to solve this drawback of the method.

The major contributions and novelty of this paper are as follows. First, this paper presents a brief survey of different modifications of bagging, which have developed to overcome the limitations of this method. Second, it proposes a novel modified version of bagging, named enhanced bagging (eBagging), which uses a new bootstrapping method, referred to as prediction error-based bootstrapping (eBootstrapping), instead of traditional random bootstrap technique. Third, it provides experimental studies to demonstrate that eBagging often gives a better performance than bagging, random forest and boosting techniques in the context of classification accuracy when tested and compared on 33 well-known benchmark datasets and various artificial (synthetic) datasets using well-known classification methods decision trees, Support Vector Machines (SVM), Naive Bayes (NB) and k-Nearest Neighbor (kNN). This paper also compares eBagging and Bagging techniques in case there is noise in the data.

The remainder of the article is structured as follows: section 2 provides a state of the art review on the distinct modifications of bagging, which have been performed to improve its performance. Section 3 gives background information on bagging, random forest and boosting techniques. This section also defines two novel concepts proposed in this paper: eBootstrapping and eBagging. In section 4, the experimental studies are presented and the obtained results are discussed. This section also answers the question of how well eBagging performs in situations where there is relatively little noise in the data. Finally, section 5 gives future directions and some concluding remarks.

Table 1. Summary of bagging variant methods.

Year	Name of the Bagging Variant	Applied Technique	Advantage / Disadvantage
2017	Actively Balanced Bagging (ABBag) [3]	First learning a bagging classifier and then updating its bootstraps iteratively with batches composed of instances chosen from the training set according to the distribution of weights	Improved classification performance Performing best when it integrates the ensemble disagreement factor with prediction error of component classifiers and information on class distribution in imbalanced data
2015	Positive Lasso Bagging (PL-Bagging) [9]	In the integration step, employing positive Lasso to assign weights to base learners	Encouraging diversity among base learners by excluding redundant ones and at the same time retaining accuracy of base learners Higher classification accuracy as the ensemble size increases
2013	Local-and-Over-All Balanced Bagging [4]	Based on the analysis of local neighborhood of each example, which affect the probability of its selection into bootstrap sample	Competitive results to best known under-sampling bagging extensions
2012	MArgin Distribution based Bagging (MAD-Bagging) [34]	Margin distribution of ensembles to choose the subset of base classifiers for Bagging through acquiring a sparse weight vector of base classifiers, ranking them according to the calculated weights and then combining the base classifiers with high weights	Improved classification accuracy When the base classifiers are diverse, more base classifiers would improve the classification power of the ensembles
2011	Information Gain Based Feature Selection for Bagging (IGF-Bagging) [29]	First bootstrap instances are obtained and then, Information Gain (IG) based feature selection method is applied to remove and identify redundant or irrelevant attributes. Finally, base learners trained from the new sub data sets are combined via majority voting	Improved classification accuracy, introduced diversity Redundant and irrelevant features are reduced Determination of feature selection ratio is a difficult task
2011	Poly-Bagging [23]	Create various classifiers by acquiring the forecasted values from the fitted methods to a number of replicated datasets and then aggregating predictors over a succession of resamplings	Easy to implement, flexible and straightforward structure Better classification accuracy for the two-bagged and the three-bagged models Sequentially, intensifying the decrease of the prediction error
2010	Leveraging Bagging [2]	Two randomization improvements which are increasing resampling using Poisson distribution and using output detection codes where each classier will predict a different function are used	More accurate results with slower execution time for Random Forest classifier compared to other bagging methods; ADWIN Bagging and Online Bagging
2010	Bagging with Rank Aggregation [11]	Out-of bag (OOB) samples are combined over through rank aggregation to obtain the locally best performing classifier given the bootstrap sample	Optimized accuracy, sensitivity and specificity Better predictive performance as judged on test instances Increased computational time
2010	Selecting Base Classifiers on Bagging (SBCB) [37]	The classifier candidates trained at the beginning of the phase may not be the set of base classifiers that the final base classifiers are selected in terms of evaluation according to both diversity and accuracy	Enhanced accuracy and complexity Larger diversity among base classifiers, the method performs better than generic bagging
2008	Robust Bagging [36]	Remove the bootstrapped classifiers producing very high error rates, as predicted by the out-of-bag error rate and to merge over the remaining ones using the robust location estimator, median	Performs better than standard bagging if more stable base classifiers as Nearest Mean Classifier and Fisher Linear Discriminant Analysis are implemented
2007	Class-wise Expert based Bagging (CeBag) [31]	Generation of class-wise experts in each aggregation sample	Improved performance of single SVM, higher diversity Outperforming standard bagging
2007	Trimmed Bagging [10]	Excluding the bootstrapped classification rules that give the biggest error rates, which are predicted by the out-of-bag error rate, and the remaining ones are aggregated	Works well for both unstable and stable classifiers
2005	Post-Bagging [19]	Several classification techniques using a bootstrap sampling procedure on the set of association rules combined with weighted voting approaches and ordinary best rule	Outperforming on average standard decision tree techniques and enhancing the outputs of best rule
2005	Dependency Bagging (DepenBag) [18]	Bootstrap samples are obtained first and each sample is induced using a dependency model mentioned as a Directed Acyclic Graph (DAG); the features without connections to the class attribute in all the DAGs are then eliminated	The causal discovery process performed facilitates constructing accurate but diverse component nearest neighbor classifiers
2005	Bagging with Injecting Randomness (BagInRand) [38]	Creating diverse component nearest neighbor classifiers through perturbing the training set with bootstrap sampling and injecting randomness to distance metrics	Improved accuracy of classifiers, introduced diversity among base classifiers, enhanced performance of stable kNN
2004	BagBoosting [13]	In each boosting iteration, combine the output from various base learners obtained from bootstrap samples, each drawn with replacement from the reweighted training data	The advantageous predictive potential Lower bias, variance and mean squared error
2004	Bagging Multitree (BagMDT) [15]	An optimization method based on sharing the common parts of the models from an ensemble generated by decision trees	The redundancy and high computational cost property of bagging method is reduced, saturation point is reached earlier than classical bagging, enhanced diversity is obtained
2003	Attribute Bagging (AB) [6]	Voting on classifiers induced by (ranked) random attribute subsets	Faster execution time than bagging, improved accuracy and stability
2003	Double Bagging [17]	The out-of-bag sample is used to generate an additional classifier model to integrate with the base learning model	The estimate is unbiased regarding method and variable selection, higher diversity among base classifiers
2003	Bootstrap Robust Aggregating (Bragging) [8]	Instead of the sample mean in the standard bagging algorithm, robust location estimator for the realized bootstrap estimators is used	Improvement on estimation procedure using smoothing effects and taking average over unstable selection of variables
2002	Variance Optimized Bagging (Vogging) [12]	Optimizing weights in linear combinations of classifiers which are acquired by bootstrap sampling	Reduction in variance while retaining high accuracy
2001	Small SubSampled Bagging (S3Bagging) [28]	Classifier induced by decreasing the size of the training set using subsampling and aggregating each classifier's results applying bagging process	Determination of the subsampling rate is tedious The performance is highly depended on subsampling rate
2000	Subsample Aggregating (Subagging) [7]	Subsampling used instead of the bootstrap for the aggregation	Improvement on variance and mean squared error, and gained computational efficiency With very small subsample ratios performance dramatically decreases
1999	Weight Aggregation (Wagging) [1]	Sampling from training set is done via adding Gaussian noise to each weight	Facilitating some control on the bias-variance trade-off Higher diversity among base classifiers
1998	Nice Bagging [27]	Only the bootstrap versions of the applied classifier which obtain lower error rate than the original classifier on the same training set are averaged	More stable results and lower shifting effect compared to bagged classifier; not giving better results than bagging in general

2. Related Work

The aggregation of various classifiers, referred as a classifier ensemble, has previously managed to enhance classification accuracy in lots of application areas compared to single classifiers. Bagging is one of the most commonly used ensemble learning methods because of its proven performance improvements in classification task [21, 24, 25]. Therefore, remarkable amount of research has been put forth in recent years by introducing new bagging variants. Almost every year, a new bagging algorithm is developed in order to make up for the missing properties of the previous ones. Table 1 presents a brief summary of the previously proposed bagging versions up to the present. It is apparent that there are two key issues which make the difference among the models. At first, the way of splitting the original training data set into subsets, i.e. the bootstrap technique to be performed, should be taken into consideration. The second one is the determination of the learning method to train base classifiers [20].

Differently from the previous works, in this study, the proposed eBagging algorithm, which is a novel version of the standard bagging method, presents a new direction to the existing instance-based ensemble learning models by performing a new bootstrap technique, eBootstrap. In the following section, its implementation is explained in detail.

3. Materials and Method

3.1. Background Information

Bagging, random forest and boosting are among the mostly applied ensemble learning methods due to their good performance in improving classification accuracy. These techniques construct diverse classifier ensembles by manipulating the training dataset and then classify new data by a voting mechanism.

3.1.1. Bagging

Bagging is an ensemble learning approach which creates multiple exemplars of a learner to result in an aggregated predictor whose output is obtained using a combination rule (i.e., majority voting) on the outputs of each constructed subspace. Generation of the multiple exemplars is done via making bootstrap replicates of the learning set in which instances are randomly drawn from the entire training data with replacement by placing the same number of instances to each ensemble subspace. Bagging provides a way of presenting variability between different models within a committee.

3.1.2. Random Forest

It is an ensemble learning method which is comprised of a number of individual decision trees. The difference from the decision tree algorithm is that the operations of finding the root node and partition of nodes run randomly. The procedure is practised as follows. Firstly, the original data is randomly sampled as in bagging to build each tree. Then, randomly selected features as in the random subspace method are used to produce the best split point. In other words, two types of randomness are present in the selection of either feature or instance. Instead of looking for the most significant feature when partitioning a node, it searches the best attribute within a random subset of attributes. In this way, a wide diversity is obtained in addition to a better model. If there are enough trees in the forest, the risk of overfitting is reduced.

3.1.3. Boosting

It is another commonly known ensemble learning strategy where weighted resampling procedure is performed by sequentially updating selected instances to the ensemble subspace by giving more weight to difficult examples, i.e. the most informative instances, which are not correctly classified in the previous steps. Weighted majority voting is applied as the combination rule for the ensemble outputs. Boosting facilitates the reduction of bias of otherwise stable learners such as univariate decision trees also known as decision stumps or linear classifiers. The most popular boosting algorithm is AdaBoost, which is an abbreviation for Adaptive Boosting. It is basically a machine learning method where multiple "weak classifiers" are combined into a single "strong classifier" by optimizing the weights in each iteration.

When bagging and boosting are compared, boosting leads to greater reduction in error [25] because it concentrates on the challenging elements in the training set during the sample selection. However, boosting suffers from being hard to parallelize because the inputs of one model depend on the outputs of others. In this work, we present a new ensemble method (eBagging) which takes the best characteristics of these two techniques. While both approaches substantially improve predictive accuracy, eBagging shows the greater benefit.

3.2. Error-based Bootstrapping

Fundamentally, bagging relates to the bootstrap approach where the training sets are randomly selected with replacement from the original instances. By following this procedure, it is possible that several records may appear more than once as a result of resampling while others may not be present in the training set. The disadvantage with traditional bootstrap method is that training subsets produced by random selection with replacement are not especially concentrated on misclassified instances. In other words, the hard-to- classify examples may not present in the training sets, so the learning algorithm cannot

focus on those data points to minimize the training errors. To overcome this drawback, we propose a novel bootstrapping model, named eBootstrapping, that ensures the presence of misclassified instances in training sets to encourage their correct classification.

Consider the dataset $D=\{o_1, o_2, ..., o_n\}=\{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$ with y_i as the class output and x_i as a p-dimensional explanatory variable for the i^{th} object o_i, so $o_i = (x_i, y_i)$ for $(i = 1, \ldots, n)$, where n is the number of objects (instances) in the dataset. Assume that we have k different class labels, in this case, $y_i \in Y = \{1, ..., k\}$.

- *Definition* 1. (Bootstrapping). If we have n data instances denoted by

$$D = \{o_1, o_2, ..., o_n\} \quad (1)$$

Then randomly generated sample with the same size n and with the equal probability ($1/n$ for each observation)

$$D^* = \{o_1^*, o_2^*, ..., o_n^*\} \quad (2)$$

Called the bootstrap sample or bootstrap resample and denoted by adding a star to the symbols. The star symbol * represents zero or more instances. This means that D^* is likely to contain repeats. Similarly, just as $\bar{o}$ is the mean of the original dataset, we write $\bar{o}^*$ for the mean of the resamples' data.

To create an ensemble E, we get b independent bootstrap samples and denote by

$$E = \{D_1^*, D_2^*, ..., D_b^*\} \quad (3)$$

For a given dataset of n examples, each example has a probability $(1\text{-}1/n)^n$ of not be included in the bootstrap sample. If n is large, the probability approaches 1/e = 0.368, which means 36.8% of the original examples would not be selected, called out-of-bag instances.

$$\lim_{n\to\infty}\left(1-\frac{1}{n}\right)=\frac{1}{e}\approx 0.368 \quad (4)$$

This also means that on average 36.8% of the "difficult" examples (misclassified) would not take part in training set and for this reason the underlying learning algorithm may not provide a good compromise between bias and variance on classification. To get over this drawback, we propose a novel bootstrapping method, eBootstrapping, which gives priority to the difficult instances.

- *Definition* 2. (eBootstrapping). Given a dataset with size n,

$$D = \{o_1, o_2, ..., o_n\} \quad (5)$$

Misclassified instances are identified by a *prior classifier* in the pre-training step and denoted by M

$$M = \{o_1, o_2, ..., o_m\} \quad (6)$$

Where m is the number of incorrectly classified instances and it is smaller than instance size n.

Classified instances are the rest of the dataset (correctly classified instances) and denoted by C

$$C = \{o_{m+1}, o_{m+2}, ..., o_n\} \quad (7)$$

Where $D = M \cup C$ and $M \cap C = \emptyset$.

eBootstrapping is a prediction error-based bootstrapping method which generates a dataset that includes all misclassified instances in M and some correctly classified instances taken with replacement from C.

An eBo*otstrap sample* of D is a collection of n instances

$$D^* = \{o_1, o_2, ..., o_m, o_{m+1}, o_{m+2}, ..., o_n\} \quad (8)$$

Where each value o_i is a distinct instance from M for $(1 \le i \le m)$, and each value c_j^* is a randomly selected instance from C with replacement and with equal probability $\Pr(c_j^* = o_t) = 1/(n\text{-}m)$ for $(m+1 \le j, t \le n)$ and $(m+1 \le t \le n)$. The star symbol * represents zero or more instances. In particular, repeated values $c_{j1}^* = c_{j2}^* = o_t$ are allowed. Since the sample size of D^* is n, repeated values c_j^* means that some values in C must be left out. To form b independent eBootstrap samples, we repeat this process b times and denote by $E = \{D_1^*, D_2^*, ..., D_b^*\}$.

As shown in Figure 1, an ebootstrap sample includes members of the original dataset with all misclassified members appearing, some correctly classified members appearing zero times, some appearing only once, some appearing twice, and so on. Replacement results in repetition of correctly classified values in ebootstrap samples. eBootstrap samples have the same sample size as the original sample. As an example, consider a dataset D ={1, 2, 3, 4, 5, 6, 7, 8, 9, 10} with size n =10. The eBootstrap sample D^* ={1, 2, 3, 8, 6, 6, 4, 10, 9, 6} has three repeats of the same number (6). The order of the elements in the resample plays no role. The out-of-bag set consists of two elements {5, 7}.

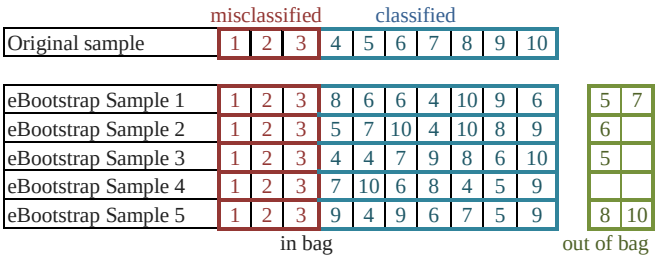


Figure 1. An example of eBootstrapping.

3.3. Enhanced Bagging (eBagging)

eBagging improves the traditional bagging technique by replacing the bootstrap method by eBootstrap. The key difference is the creation of training sets by giving

higher chance for selection to hard-to-classify instances which are misclassified by the prior learner. As shown in Figure 2, the proposed eBagging technique contains four steps:

- *Step*1: *Pre-training* A prior classifier is applied on the original dataset and the dataset is divided into two parts: one part that contains correctly classified instances and the others (incorrectly classified).
- *Step* 2: *eBootstrapping* eBootstrapping is used to construct different training sets by directly pushing misclassified instances and resampling with replacement from classified instances. So, the difficult instances are always included in each subset of data. This step both provides diversity and allows the learning algorithm to focus on hard-to-classify examples and so it gives us a reasonable starting point.
- *Step* 3: *Training* The base classifiers are trained on different subsets of the training patterns. In this study, the learning algorithms used in pre-training and training steps (in other words the learning algorithms of prior classifier and base classifier) are planned to be same, but different learning algorithms can also be tried in the future studies.
- *Step* 4: *Combining* (*Aggregating*) The base classifiers perform classification task and the final prediction is made by applying majority voting to the outputs of each ensemble subset. If classifiers disagree with each other, then the incorrect errors of the different classifiers can be removed by the voting mechanism.

Boosting technique (i.e., AdaBoost algorithm) also tries to classify the hard-to-classify examples accurately, and ignores the ones which are easy-to-classify. However, our proposed method is different from boosting in two-folds. First, eBagging generates training sets from the original dataset in parallel, so it is not an iterative approach as boosting. Second, eBagging doesn't assign weight values to each instance as boosting. Instead of that, eBagging is directly copied all difficult examples into all training sets.

Algorithm 1 eBagging

Inputs
Dataset $D=\{(x_1,y_1),(x_2,y_2),\ldots,(x_n,y_n)\}$ *where* $y_i \in Y=\{1,\ldots,k\}$
L:Learning algorithm for prior and base classifiers
t:ensemble size
n:the number of instances
k:the number of classes
x:an unlabelled instance to classify

Process
#Step1: Pre-training
$h=L(D)$ *#train to build prior classifier*
$C=\emptyset$, $M=\emptyset$ *#generate new datasets*
for(i=1 to n)
{
  if($h(x_i)=y_i$) *#correctly classified instances*
    $C.add(x_i,y_i)$
  else
    $M.add(x_i,y_i)$ *#misclassified instances*
}
#Step2: eBootstrapping
for(i=1 to t)
{
  $D_i=\emptyset$
  $D_i= D_i \cup M$
  while(D.length!=m)
  {
    r=random.next(1,C.length)
    $D_i.add(C_r)$
  }
}
#Step3: Training
$h_i=L(D_i)$

Output
#Step4:Combining
$H(x)=voting(h_1(x), h_2(x),\ldots, h_n(x))$ *#final hypothesis*

$$= \underset{y \in Y}{argmax} \sum_{i} 1 : y = h_i(x)$$

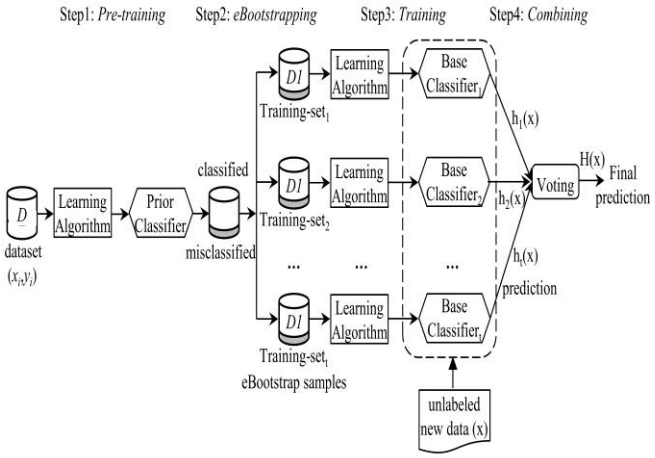


Figure 2. The general framework of Enhanced Bagging (eBagging) algorithm.

The effectiveness of our method depends on constructing a diverse, yet accurate, collection of classifiers. By eBagging, we obtain a good compromise between a lack of diversity and a premature overfitting situation. Algorithm 1 demonstrates the pseudo code of the proposed eBagging algorithm. A classifier (or a hypothesis) is a mapping from x to y. Given set D containing n examples, misclassified instances are identified by a prior classifier in the pre-training step and denoted by M. Classified instances are the rest of the dataset and denoted by C. Datasets $D_1 \ldots D_t$ are generated by including all misclassified instances in M and some correctly classified data points taken with replacement from C. After training, a new instance is classified by voting.

In Table 2, there is a brief summary of the properties of eBagging, bagging, random forest and boosting techniques in terms of given criteria.

Table 2. Comparison table of eBagging, Bagging, Random Forest and Boosting techniques.

	eBagging	Bagging	Random Forest	Boosting
Training set construction	misclassified samples + random resampling	random resampling	random resampling+random feature subset	weighted resampling
Model construction	parallel	parallel	parallel	iterative (incremental)
Weight assignment to each instance	no	no	no	yes
Classifiers	inducer independent	inducer independent	inducer dependent (decision tree)	inducer independent
Training datasets	giving higher preference to misclassified samples	random	random	giving higher preference to misclassified samples
Goal to achieve	increase the performance of the base learners with a prior learner	minimize variance	minimize variance	increase predictive force
Aim	find a good trade-off between variance and bias that minimizes prediction error	decrease variance	increase diversity	decrease bias
Methods where this is used	error-based random subspace	random instance subspace	random instance subspace and random feature subspace	gradient descent
Function to combine	majority voting	majority voting	majority voting	weighted majority voting
Inducer number	single inducer	single inducer	single inducer	single inducer
Base learner	simple or complex	complex	simple	simple
Effect	loss minimization	risk minimization	risk minimization	margin maximization

4. Experimental Work

In this study, the proposed eBagging method was compared with single (no ensemble strategy used), standard bagging ensemble, random forest and AdaBoost learners. As a base learner for ensemble methods, SVM, kNN, Decision Tree (C4.5) and NB algorithms were individually applied. Many studies have taken these methods as base learners in bagging [16, 30, 33] even though bagging is not generally favorable to work with stable algorithms such as SVM, kNN and NB. We aimed to handle this stiuation with our additional improvements in eBagging to see whether it is useful or not.

Weka open source data mining library was used to develop the models [32]. Classification accuracies, pairwise comparisons showing win/tie/loss status and average error rates of the applied algorithms are presented in this section. Statistical test results were also given to verify the obtained outputs. In addition, we explored the effect of classification noise on the performance of eBagging technique.

4.1. Dataset Description

In the experimental studies, 33 datasets taken from UCI Machine Learning Repository [22] were used. Table 3 shows the basic properties of the datasets by noting the release year of the dataset, their respective number of records, the number of dimensions including the class attribute and how many classes they contain. In addition to well-known benchmarks, synthetic datasets with different characteristics were also handled and their properties are given in the next section.

Table 3. Experimental datasets and their characteristics.

ID	Dataset Name	Year	Attributes	Instances	Number of Classes	ID	Dataset Name	Year	Attributes	Instances	Number of Classes
1	arrhythmia	1998	280	452	16	18	letter	1991	17	20000	26
2	audiology	1992	70	226	24	19	liver-disorders	1990	7	345	2
3	breast-cancer	1988	10	286	2	20	lymph	1988	19	148	4
4	car	1997	7	1728	4	21	nursery	1997	9	12960	5
5	dermatology	1998	34	366	6	22	page-blocks	1995	11	5473	5
6	diabetes	1990	9	768	2	23	segment	1990	20	2310	7
7	ecoli	1996	8	336	4	24	sick	1987	30	3772	2
8	glass	1987	10	214	7	25	sonar	1988	61	208	2
9	haberman	1999	4	306	2	26	soybean	1988	36	683	19
10	heart-c	1988	14	303	5	27	spambase	1999	58	4601	2
11	heart-h			294	5	28	tae	1997	6	151	3
12	heart-statlog	1992		270	2	29	tic-tac-toe	1991	10	958	3
13	hepatitis	1988	20	155	2	30	vehicle	1987	19	846	4
14	hypothyroid	1987	30	3772	4	31	wine	1991	14	178	3
15	ionosphere	1989	35	355	2	32	waveform-5000	1988	41	5000	3
16	iris	1988	5	150	3	33	zoo	1990	17	101	7
17	kr-vs-kp	1989	37	3196	2						

4.2. Comparison of the Applied Methods

Five different cases were taken into consideration as classifiers' performances were compared: classification accuracy for benchmark datasets and artificial datasets (shown in Table 4 and Table 7), win/tie/loss status showing pairwise comparisons of the applied methods (displayed in Table 5), the average error rates relative to each other (demonstrated in Figure 3), the results of statistical tests (Friedman and Quade) and the effect of classification noise (in Figure 5). Classification accuracies of the applied algorithms were obtained

using 10-fold cross-validation. In the implementations of the ensemble learners, the number of iterations to be performed (ensemble size) were determined as Weka's default parameter, 10. The number of neighbors, *N* for kNN classifier was selected as $\log_2(n)$ where *n* indicates the number of instances in the respective dataset. The default *N* parameter of Weka is 1, however, it does generally not make sense to choose the parameter *N* so small when large numbers of data points are available in the dataset. $N=\log_2(n)$ is a reasonable choice because if *N* is determined as too large or too small, the probability of overfitting dramatically increases [14, 26, 35]. The classifier parameters of SVM, C4.5 and NB classifiers were left as default Weka parameters.

Table 4. Comparison table of the applied classifiers in terms of classification accuracies (%).

DATASET / METHOD	C4.5					NB			
	eBagging	Bagging	Single	RF	AdaBoost	eBagging	Bagging	Single	AdaBoost
arrhythmia	78.89	75.44	64.38	64.82	70.58	70.42	64.60	62.39	62.83
audiology	83.18	81.42	77.88	76.55	84.96	75.44	71.68	73.45	77.88
breast-cancer	73.64	73.78	75.52	69.23	69.58	73.46	72.73	71.68	64.69
car	92.89	93.11	92.36	93.58	96.12	85.76	85.53	85.53	90.16
dermatology	94.64	96.17	93.99	95.90	95.63	98.28	96.99	97.27	96.45
diabetes	79.21	74.61	73.83	74.35	72.40	76.17	76.56	76.30	76.17
ecoli	87.92	84.52	84.23	84.82	81.25	86.25	86.01	85.42	85.42
glass	81.31	74.30	66.82	74.30	74.30	50.75	50.00	48.60	49.07
haberman	75.20	72.88	72.88	68.63	70.26	75.26	75.16	76.14	75.16
heart-c	83.83	78.88	77.56	79.21	82.18	84.22	83.17	83.50	84.16
heart-h	81.39	79.93	80.95	77.55	78.57	84.80	83.33	83.67	86.39
heart-statlog	85.07	78.52	76.67	77.78	80.37	85.48	83.33	83.70	82.59
hepatitis	86.26	83.87	83.87	83.87	85.81	85.81	85.81	84.52	85.81
hypothyroid	99.62	99.60	99.58	99.02	99.58	95.55	95.47	95.28	95.28
ionosphere	94.10	92.88	91.45	92.88	93.16	83.65	83.19	82.62	92.02
iris	96.33	94.67	96.00	95.33	93.33	96.33	96.00	96.00	93.33
kr-vs-kp	99.41	99.37	99.44	98.72	99.50	87.56	87.73	87.89	94.96
letter	91.29	92.47	87.98	93.97	95.54	64.41	64.17	64.12	64.11
liver-disorders	76.52	69.28	68.70	68.41	71.59	57.36	57.10	55.36	67.25
lymph	84.46	79.73	77.03	75.00	81.08	83.92	82.43	83.11	80.41
nursery	97.18	97.33	97.05	98.05	99.51	90.20	90.28	90.32	91.82
page-blocks	97.68	97.20	96.88	97.20	97.02	89.73	90.50	90.85	90.85
segment	97.66	97.45	96.93	97.53	98.48	80.06	80.48	80.22	80.22
sick	99.15	98.59	98.81	98.14	99.18	93.08	92.71	92.60	93.96
sonar	87.50	77.88	71.15	77.40	77.88	71.78	68.27	67.79	80.77
soybean	93.05	92.68	91.51	91.22	92.83	92.78	92.97	92.97	92.83
spambase	94.73	94.18	92.98	94.48	95.09	79.85	79.70	79.29	79.29
tae	69.54	60.26	59.60	65.56	68.87	53.18	54.30	54.30	54.30
tic-tac-toe	89.77	93.32	84.55	92.80	95.82	71.06	70.15	69.62	83.92
vehicle	85.15	74.47	72.46	76.00	76.24	46.16	45.63	44.80	44.80
wine	95.39	93.82	93.92	97.75	96.63	98.15	97.19	96.63	96.07
waveform-5000	87.70	81.20	75.08	81.20	80.48	79.99	79.98	80.00	80.00
zoo	96.53	93.07	92.08	95.05	95.05	96.63	95.05	95.05	97.03
Average Accuracy	88.37	85.66	83.76	85.04	86.33	80.11	79.34	79.12	80.91

DATASET / METHOD	kNN				SVM			
	eBagging	Bagging	Single	AdaBoost	eBagging	Bagging	Single	AdaBoost
arrhythmia	60.97	58.41	58.85	58.85	80.55	71.9	70.35	65.93
audiology	78.76	59.73	56.19	59.73	90.71	76.99	81.86	82.74
breast-cancer	77.52	74.48	73.43	73.43	71.82	72.03	69.58	69.58
car	93.82	92.53	93.29	93.29	93.51	93.46	93.52	94.50
dermatology	96.89	96.17	95.63	94.26	98.52	96.72	95.36	95.36
diabetes	78.35	73.57	71.09	71.09	76.71	77.47	77.34	77.34
ecoli	85.24	86.90	86.31	86.31	85.36	83.63	83.33	84.82
glass	75.65	66.36	64.02	64.02	57.29	57.48	56.07	57.01
haberman	76.21	73.20	73.53	72.22	73.14	73.53	72.88	73.86
heart-c	82.77	82.51	83.50	82.18	83.96	84.49	84.16	84.82
heart-h	85.20	83.33	82.31	79.59	83.98	83.67	82.65	82.31
heart-statlog	82.00	78.15	81.11	75.56	84.37	82.96	84.07	84.07
hepatitis	83.03	83.23	84.52	84.52	86.00	85.81	85.16	81.29
hypothyroid	92.90	93.05	93.08	90.54	93.60	93.64	93.58	94.99
ionosphere	89.77	84.90	85.47	86.04	89.40	89.46	88.60	88.60
iris	96.40	96.67	96.67	96.00	95.87	95.33	96.00	98.00
kr-vs-kp	96.44	95.18	94.96	97.09	96.31	95.87	95.56	97.18
letter	96.89	94.44	94.25	94.25	82.47	82.65	82.35	82.34
liver-disorders	73.13	62.90	61.74	61.74	58.14	60.00	58.26	62.03
lymph	86.15	84.46	83.78	85.14	90.27	85.14	86.49	83.78
nursery	97.95	97.52	98.36	98.49	93.08	93.07	93.09	93.07
page-blocks	96.73	95.34	95.05	94.98	93.06	93.60	92.93	92.93
segment	97.45	94.37	94.46	94.46	92.66	92.86	93.07	93.29
sick	96.65	95.94	96.00	95.68	94.09	93.93	93.88	94.41
sonar	88.89	75.48	79.33	83.17	82.40	78.37	75.96	75.96
soybean	92.59	88.73	88.58	88.87	95.40	92.39	93.70	92.68
spambase	92.82	90.07	88.85	85.98	90.60	91.02	90.42	90.76
tae	69.93	53.64	51.66	51.66	52.05	54.97	52.98	52.98
tic-tac-toe	98.16	98.33	98.85	98.64	98.33	98.33	98.33	98.12
vehicle	76.94	70.09	70.21	70.21	74.81	75.30	74.59	74.35
wine	97.58	96.07	94.94	94.38	98.43	98.31	98.31	98.31
waveform-5000	83.41	81.94	81.34	81.34	86.85	86.26	86.70	86.68
zoo	95.94	89.11	89.11	93.07	97.13	92.08	93.07	97.03
Average Accuracy	87.06	83.24	83.04	82.93	85.48	84.32	84.07	84.28

Table 4 displays the comparisons of the applied methods (eBagging, single learner, standard Bagging, random forest and AdaBoost) with respect to classification accuracy by separately using C4.5, NB, kNN and SVM as a base learner. Random Forest is only under C4.5 column because it is inducer dependent and the decision tree is employed as a base learner. The results show that eBagging has generally the best average classification accuracies 88.37%, 80.11%, 87.06%, and 85.48%, respectively. The highest classification accuracy for each dataset is shown by bold background. According to the results, eBagging clearly ahead of other methods by outperforming its counterparts in 21, 14, 25 and 13 out of 33 when one of the C4.5, NB, kNN or SVM methods was used as a base learner, respectively. As seen in the results, the performance of standard bagging algorithm, which managed to be the best learner in 1, 5, 2, and 10 out of 33 for the same base learners, is significantly enhanced when the ensemble subspace is constructed with the proposed eBagging algorithm. On the other hand, eBagging also achieved classifying instances more accurately than the most powerful ensemble learning methods, AdaBoost, which is the winner in 9, 11, 2, and 9 out of 33 datasets for the corresponding base learners. Furthermore, Random Forest performs the best among other methods in only one dataset under C4.5 base classifier.

In addition to the results of classification accuracy, it is important to extend experimental work on the pairwise comparisons of the performed algorithms. Table 5 indicates the (win-tie-loss) status of the paired algorithms where each cell is read by looking at the algorithm in the respective row and then in the respective column.

It is clear that eBagging is the winner among others when one of the methods from C4.5, SVM and kNN is used as a base learner in the generation of ensembles. Besides, when NB is used as the classifier of the ensembles, both eBagging and AdaBoost perform equally well for classifying instances.

Table 5. Pairwise comparisons of the applied methods by showing in each cell (wins - ties - loses) between the method in that row and the method in that column for the respective classifier (C4.5, SVM, NB or kNN).

C4.5	eBagging	Single	Bagging	AdaBoost	Random Forest
eBagging		**31** - 0 - 2	**27** - 0 - 6	**22** - 0 - 11	**27** - 0 - 6
Single			6 - 2 - **25**	6 - 1 - **26**	11 - 1 - **21**
Bagging				11 - 2 - **20**	**16** - 5 - 12
AdaBoost					**24** - 2 - 7
Random Forest					
SVM	eBagging	Single	Bagging	AdaBoost	
eBagging		**24** - 1 - 8	**17** - 1 - 15	**21** - 0 - 12	
Single			10 - 2 - **21**	10 - 9 - **14**	
Bagging				**16** - 2 - 15	
AdaBoost					
Random Forest					
NB	eBagging	Single	Bagging	AdaBoost	
eBagging		**24** - 0 - 9	**25** - 1 - 7	15 - 2 - **16**	
Single			11 - 5 - **17**	10 - 8 - **15**	
Bagging				**16** - 3 - 14	
AdaBoost					
Random Forest					
kNN	eBagging	Single	Bagging	AdaBoost	
eBagging		**25** - 0 - 8	**28** - 0 - 5	**28** - 0 - 5	
Single			13 - 2 - **18**	12 - **13** - 8	
Bagging				**19** - 1 - 13	
AdaBoost					
Random Forest					

Figure 3 shows average error rates (results from all datasets are averaged) obtained from the pairwise comparisons of the applied algorithms. The calculation of the average error rate can be explained through an example (eBagging vs single classifier): For each dataset, the ratio between the mean error rate of eBagging algorithm and of single classifier when C4.5 is applied are calculated over the 10-fold cross-validation. For instance, in the case of NB algorithm, the classification errors (%) of arrhythmia dataset are 29.58 and 37.61 for eBagging and single classifier, respectively. The ratio of eBagging vs single classifier is calculated as 1 - (37.61 - 29.58) / 37.61 = 0.79. After all of the ratio values for each dataset are computed, their mean value gives the average error rate between the compared algorithms.

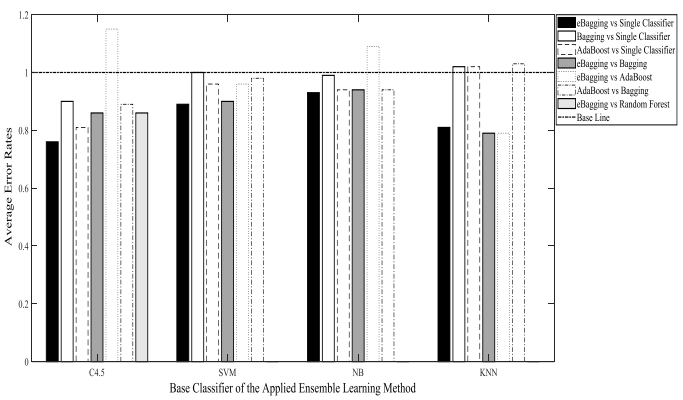


Figure 3. Average error rates of the proposed algorithms in terms of pairwise comparisons.

According to the Figure 3, the base line, y=1, means that the performance of the compared methods is approximately equal in classifying instances. The average error rate below 1 states the enhancement case in which the first method in the legend improves classification performance compared with the second applied algorithm. For example, when SVM is used as a base learner, eBagging obtains average error rates of 0.89, 0.90, and 0.96 compared to the algorithms, Single classifier, bagging and AdaBoost, respectively. There is a noticeable improvement in the experimental results when eBagging is selected as an ensemble strategy for SVM classifier. In the part of C4.5 algorithm, there is also the comparison of eBagging versus Random Forest differently from the comparisons of other base learners. The average error rate is decreased 13.87% when eBagging is applied instead of Random Forest. Apart from this, eBagging provides a visible improvement in the most of the comparisons for all base learners. At the same time, standard bagging algorithm is apparently enhanced by executing eBagging regardless of the choice of the base classifier.

Even though eBagging performs better than AdaBoost in terms of classification accuracy, when NB and C4.5 classifiers are taken as base classifiers of the ensembles (in half of the cases), average enhancement rate reduces below 1. In this case, average error rates are 1.15 and 1.09, respectively, if eBagging applied instead of AdaBoost as an ensemble strategy. It is because that AdaBoost results in great increases in classification accuracy of a number of datasets although eBagging manages to classify much more datasets accurately.

Figure 4 shows the classifiers' average ranks. Initially for each dataset, classifier models are rated interms of their classification accuracies. It is done by giving rank 1 to the method with the highest classification accuracy and raising the rank for each evaluated classifier until assigning rank *c* to the lowest one for *c* applied methods. The rank value is given as the mean rank in case of a tie. In the next step, average ranks per classifiers are calculated by taking the mean value of the outputs of 33 datasets. Finally, the obtained mean values of all applied models for each base classifier are again ranked from 1 to *c*. In Figure 4, the output is given in terms of base classifiers using the ensemble models, eBagging, bagging, random forest (in C4.5 model) and AdaBoost; and using single classifier models. According to the results, when eBagging is performed, it obtains the lowest rank value which means that it attains the best performance among the others.

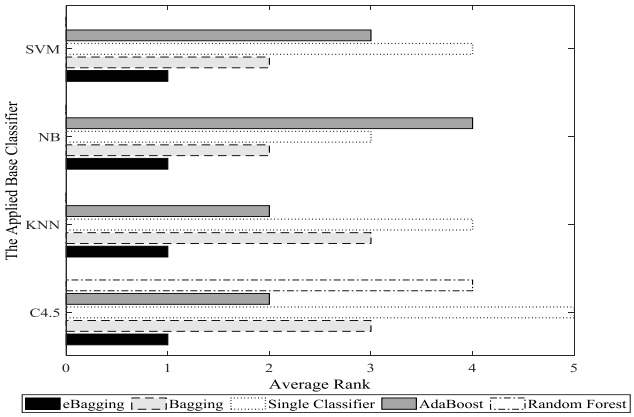


Figure 4. Average ranks of the methods (eBagging, Bagging, Single Classifier, Random Forest and AdaBoost) using four different base classifiers (SVM, NB, kNN and C4.5).

Another important thing is to apply statistical tests to verify the experimental results and to support decision making process. Therefore two non-parametric tests were applied as Friedman and Quade ranking tests at the significance level of $\alpha = 0.05$. Accuracy results in Table 4 were used in each test for each base classifier. The null hypothesis, H_0, is that the means of the experimental results of different classifiers (eBagging, single classifier, bagging, random forest and AdaBoost) are the same, meaning that all the groups behave similarly. According to the results, the null hypothesis is rejected by both of the statistical tests in other words the results are considered as statistically significant. The obtained p-values of the tests with the base classifiers C4.5, kNN, NB and SVM are presented in Table 6 by also showing their significance level.

Table 6. The number of wins obtained from classification.

Base Classifier / Statistical Test	Friedman		Quade	
	p-value	Significance Level	p-value	Significance Level
C4.5	0.00001	Very Strong	<0.00001	Very Strong
KNN	<0.0000	Very Strong	<0.00001	Very Strong
SVM	0.03841	Strong	0.00140	Very Strong
NB	0.03182	Strong	0.01458	Strong

In addition to the experiments on the benchmark datasets, evaluations with arficial (synthetic) datasets were also performed. Experiments were categorized with four different aspects:

a) The effect of the number of dimensions
b) The impact of the imbalanced data generated using different class weights
c) The variation when the class separation factor is changed
d) The effect of the outliers on the performance of learners.

Table 7 (a, b, c, and d) displays the evaluation results (% accuracy) of each factor using 10-fold cross-validation.

Table 7. Evaluation results of artificial datasets.

C4.5						SVM					
Method	Num of Features					Method	Num of Features				
	50	100	150	200	250		50	100	150	200	250
eBagging	97.09	97.19	97.3	97.07	97.36	eBagging	95.23	94.84	95.23	94.39	93.37
Bagging	95.5	95.1	95.6	95.5	95.1	Bagging	95.3	93.9	92.8	91.7	91.4
Single	94.4	93.6	93.8	94.4	94.5	Single	95.5	94.2	93.9	91.4	92.2
AdaBoost	95.9	95.2	95.1	95.2	95.3	AdaBoost	95	91.3	90.6	88.5	86.7
RF	95.1	93.4	91	87.7	88						
NB						**KNN**					
Method	Num of Features					Method	Num of Features				
	50	100	150	200	250		50	100	150	200	250
eBagging	94.14	94.21	93.97	93.98	94.11	eBagging	87.94	83.63	84.59	82.68	79.29
Bagging	94.1	94.2	93.8	93.6	93.4	Bagging	90.2	85	83.4	82.1	79.4
Single	94.3	94	94	93.5	93.9	Single	89.6	83	84	81.3	77.7
AdaBoost	92.2	92.4	89.3	90.6	90.2	AdaBoost	89.6	83	84	81.3	77.7

Table 8. The experimental results of the artificial dataset with different class imbalances.

C4.5						SVM					
Method	Balanced	10%-90%	20%-80%	30%-70%	40%-60%	Method	Balanced	10%-90%	20%-80%	30%-70%	40%-60%
eBagging	97.19	98.42	97.86	97.06	96.66	eBagging	94.84	97.31	96.34	95.56	95.57
Bagging	95.1	98	97.5	96.2	96.1	Bagging	93.9	96.7	95.3	94.9	94
Single	93.6	97	96.1	93.4	94.8	Single	94.2	96.9	95.2	94.8	94.3
AdaBoost	95.2	97.6	97.3	95.6	95.6	AdaBoost	91.3	95.1	93.1	91.8	91.6
RF	93.4	96	95.9	94.5	94.3						
NB						**KNN**					
Method	Balanced	10%-90%	20%-80%	30%-70%	40%-60%	Method	Balanced	10%-90%	20%-80%	30%-70%	40%-60%
eBagging	94.21	97.57	96.05	95.11	94.17	eBagging	83.63	90.92	86.59	86.76	85.5
Bagging	94.2	96.8	95.7	94.8	94	Bagging	85	90.1	85.2	86	86
Single	94	97.5	95.7	95.3	93.9	Single	83	90.4	87.5	86.6	85.3
AdaBoost	92.4	95.9	93.5	92.5	92.7	AdaBoost	83	89.8	84.1	86.6	85.3

Table 9. The experimental results of the artificial dataset with different class separation values.

C4.5								SVM							
	0.5	0.75	1	1.25	1.5	1.75	2		0.5	0.75	1	1.25	1.5	1.75	2
eBagging	83.92	85.75	87.32	89.73	91.41	94.11	95.24	eBagging	68.61	75.84	81.35	85.48	89.22	92.56	94.89
Bagging	74.4	79.9	84.3	88.3	89.1	93	94.2	Bagging	67.3	74.8	80.8	84.4	89.2	91.9	94.6
Single	69.7	72.5	76.2	81.3	86.6	86.3	91	Single	67.2	76.1	80.9	85	88.3	92.2	94.9
AdaBoost	71.9	77.9	83.9	87.9	91	94	96	AdaBoost	68.4	75.1	80.6	85	87.8	92.1	93.2
RF	70.9	72.4	80.8	83.9	90.6	91.5	94.5								
NB								**KNN**							
	0.5	0.75	1	1.25	1.5	1.75	2		0.5	0.75	1	1.25	1.5	1.75	2
eBagging	72.45	77.74	82.36	85.98	88.73	90.76	92.73	eBagging	81.73	84.48	87.76	91.53	94.15	96.47	97.71
Bagging	70.8	76.7	81.6	85.1	87.7	90.3	92.7	Bagging	78.4	83	87.7	91.5	95.4	97.3	98.5
Single	72	77.3	82.2	85.4	88.2	90.8	93	Single	76.8	80.9	86	91.7	94.8	97	98.9
AdaBoost	73.1	77.6	82.1	85.3	88.6	92.6	93.7	AdaBoost	76.8	80.9	86	90.1	92	95.8	97.3

Table 10. The experimental results of the artificial dataset with extreme outliers.

C4.5					SVM			
eBagging	Bagging	Single	AdaBoost	RF	eBagging	Bagging	Single	AdaBoost
93.16	92.1	89	91.2	89.6	90.48	89.7	89.5	88.4
NB					**KNN**			
eBagging	Bagging	Single	AdaBoost		eBagging	Bagging	Single	AdaBoost
89.71	89.8	90	88		85.24	84.9	84.7	84.7

The artificial datasets were created on the Pyhton platform. All the generated datasets have 2 class labels and they include 1000 instances. The dataset used in Table 7-a includes variable feature number for each experiment as 50, 100, 150, 200 and 250. In Table 7-b, an artificial dataset with 100 features is shown. For different evaluations, the number of instances per class changes with a ratio of 10%-90%, 20%-80%, 30%-70%, 40%-60% and balanced class condition is also examined. Furthermore, the results according to different class separation factors in Table 7-c were obtained using a dataset with 50 features and several class separation values were used as 0.5 to 2 with 0.25 increase. Lastly, the dataset which includes 50 features and 2% of which consists of extreme outliers was created for the experiments in Table 7-d.

According to the results of Table 7-a, eBagging outperforms other applied methods in most of the cases
when C4.5, SVM and NB base classifiers were selected. When kNN is applied, Bagging performs better if the dataset has 50, 100 or 250 features.

When different class imbalances as in Table 7-b were applied, the evaluation results were found similar to the outputs of Table 7-a. Even though eBagging achieved a comparable performance with bagging when kNN is the base classifier, it is the best among others when the base classifiers, SVM, C4.5 and NB are applied. It managed to classify the same number of datasets as bagging did.

It is a known fact that larger values of class separation factor makes the classification task easier. This condition holds for the applied experiments in Table 7-c. For each base classifier, eBagging achieved the most accurate results, especially in C4.5.

Table 7-d demonstrates the experimental results of the dataset with extreme outliers. Except from the results of NB classifier, eBagging with other base classifiers managed to classify data points correctly.

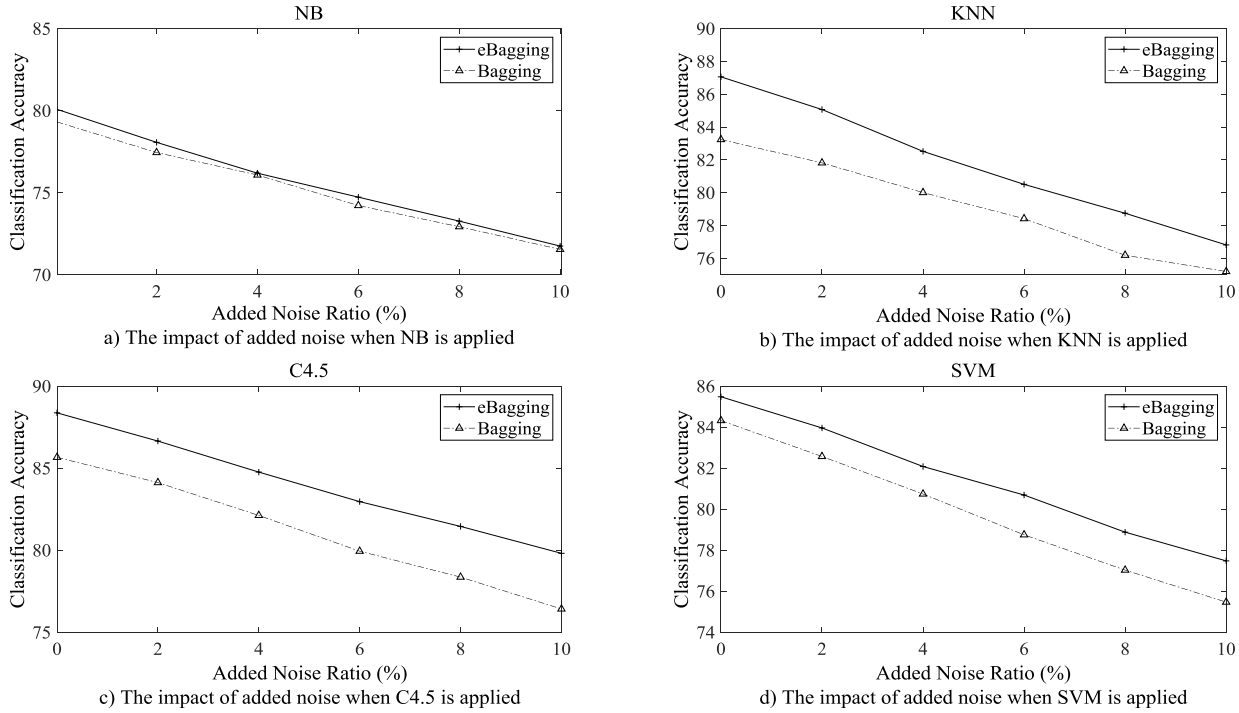
As a consequence of the experimental work, eBagging is proved to be a novel and improved version of the standard bagging algorithm and it has succeeded to compete with powerful ensemble learning methods, random forest and AdaBoost, as well. The idea is easy to implement, simple and provides major benefits compared the other applied algorithms.

4.3. The Effects of the Classification Noise

Robustness with respect to noise is a desirable property, because some noise in the data is often present. In this experimental study, we explored the effect of classification noise on the performance of eBagging technique.

We added random class noise to the 33 datasets described in Table 3 for exploring the effect of classification noise. To include classification noise at the rate of p percent, p% of the data instances were chosen randomly without replacement and class labels of them were changed to be incorrect (alternated to class label chosen uniformly from the other labels).

Figure 5 shows the average classification performances of eBagging and Bagging techniques at the six noise levels (0%, 2%, 4%, 6%, 8% and 10%). From this analysis, we can conclude that eBagging is still better than Bagging in the presence of noise in the data. The evaluation results also proved that in the case of Naive Bayes algorithm, eBagging is slightly superior to Bagging. However, in the case of decision tree, kNN and SVM, eBagging is much better than Bagging. In Table 8, the number of wins and ties for eBagging and bagging methods is shown. With 10% noise, eBagged C4.5 has a certain advantage over bagged C4.5 (28 wins, 5 losses).



a) The impact of added noise when NB is applied
b) The impact of added noise when KNN is applied
c) The impact of added noise when C4.5 is applied
d) The impact of added noise when SVM is applied

Figure 5. Influence of added noise on the average accuracies of eBagging and Bagging methods when noise is added at the rate of (0% - 10%).

Table 8. The number of wins obtained from classification accuracy of eBagging and bagging methods in terms of added classification noise.

Noise Ratio	C4.5			NB			kNN			SVM		
	eBagging	**Bagging**	**Tie**	**eBagging**	**Bagging**	**Tie**	**eBagging**	**Bagging**	**Tie**	**eBagging**	**Bagging**	**Tie**
0%	27	6	0	25	7	1	28	5	0	17	15	1
2%	26	7	0	22	11	0	24	8	1	21	10	2
4%	27	6	0	18	15	0	23	10	0	22	10	1
6%	26	7	0	23	10	0	20	13	0	26	7	0
8%	25	8	0	21	12	0	22	11	0	23	10	0
10%	28	5	0	19	14	0	19	14	0	24	9	0

Classification noise destroys the effectiveness of eBagged kNN and bagged decision tree methods at most compared to the other applied algorithms. There are more dramatic decreases in the classification accuracies of the mentioned methods when the noise ratio is increased. However, when noise increases, eBagging still has certain advantages to noise.

From this analysis, we can conclude that the best method in the presence of classification noise is eBagged SVM. In contrast, eBagged kNN is not a good choice in such applications. Because, kNN changes markedly with noise, while SVM procedures generally show small changes.

A plausible explanation for the better response of eBagging to noise is that misclassified examples due to noise will present in the training set. Hence, the training examples with misclassified examples will tend to make each classifier more accurate. EBagging overcomes classification noise problem since it emphasizes on hard points.

However, in the experiments, we only considered ensembles of size 10. Larger ensembles might be able to overcome the effects of fairly high levels of noise.

5. Conclusions

The principal purpose of this study is to present a novel ensemble learning technique, eBagging, by modifying and optimizing standard bagging algorithm. The key difference is the creation of the ensemble subsets by giving higher chance for selection to the most informative and challenging instances which are misclassified by the prior learner. This essential step canalizes the algorithm to deal with hard-to-classify instances in depth so that training errors can be minimized.

In the experimental studies, which were verified by statistical tests, four commonly used classifiers, which are SVM, NB, kNN and C4.5, are used as base classifiers of the applied ensemble learning methods. eBagging was compared with single learners, standard bagging, random forest and AdaBoost algorithms. According to the experimental results, eBagging outperforms its counterparts by classifying the data points more accurately while reducing the training error. Furthermore, average error rate significantly decreases when eBagging is performed compared to single classifiers and standard bagging algorithm, while it leads close results with AdaBoost in half of the cases. As a consequence, the proposed eBagging method shows promising applicability in classifying data samples. The experiments also show that over 33 datasets, eBagging gives better results than Bagging as long as there is little or no noise in the data.

In addition to benchmark datasets, four different evaluations were also done on synthetic datasets according to the effect of the number of dimensions, the performance on the imbalanced data, the influence of the size of class separation values and the impact of the outliers. As a result, eBagging dealt with these exteme cases better than other applied methods.

In the future, the following studies is going to be performed:

- Generation of the ensemble subsets in eBootstrap step by randomly selecting the instances from the whole dataset instead of the classified instances after placing all the misclassified instances identified by a prior classifier to the ensemble subsets,
- Determination of the optimal ensemble size for eBagging.

References

[1] Bauer E. and Kohavi R., "An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants," *Machine learning*, vol. 36, no. 1, pp. 105-139, 1999.

[2] Bifet A., Holmes G., and Pfahringer B., "Leveraging Bagging for Evolving Data Streams," *in Proceedings of European Conference on Machine Learning and Knowledge Discovery in Databases*, Berlin, pp. 135-150, 2010.

[3] Błaszczyński J. and Stefanowski J., "Actively Balanced Bagging for Imbalanced Data," *in Proceedings of International Symposium on Methodologies for Intelligent Systems*, Cham, pp. 271-281, 2017.

[4] Błaszczyński J., Stefanowski J. and Idkowiak Ł., "Extending Bagging for Imbalanced Data," *in Proceedings of the 8th International Conference on Computer Recognition Systems*, Heidelberg, pp. 269-278, 2013.

[5] Breiman L., "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123-140, 1996.

[6] Bryll R., Gutierrez-Osuna R. and Quek F., "Attribute Bagging: Improving Accuracy of Classifier Ensembles by Using Random Feature Subsets, " *Pattern recognition*, vol. 36, no. 6, pp. 1291-1302, 2003.

[7] Bühlmann P. L. and Yu B., "Explaining Bagging," Technical Report, 2000.

[8] Bühlmann P. L., "Bagging, Subagging and Bragging for Improving Some Prediction Algorithms," Research report, 2003.

[9] Chung D. and Kim H., "Accurate Ensemble Pruning with PL-Bagging," *Computational Statistics and Data Analysis*, vol. 83, pp. 1-13, 2015.

[10] Croux C., Joossens K., and Lemmens A., "Trimmed bagging," *Computational Statistics and Data Analysis*, vol. 52, no. 1, pp. 362-368, 2007.

[11] Datta S., Pihur V. and Datta S., "An Adaptive Optimal Ensemble Classifier Via Bagging and Rank Aggregation with Applications to High Dimensional Data," *BMC bioinformatics*, vol. 11, no. 1, pp. 1-11, 2015.

[12] Derbeko P., El-Yaniv R., and Meir R., "Variance Optimized Bagging," *in Proceedings of European Conference on Machine Learning*,

Berlin, pp. 60-72, 2002.

[13] Dettling M., "Bagboosting for Tumor Classification with Gene Expression Data," *Bioinformatics*, vol. 20, no. 18, pp. 3583-359, 2004.

[14] Dexun J., Peijun M., Xiaohong S. and Tiantian W., "Distance Metric Based Divergent Change Bad Smell Detection and Refactoring Scheme Analysis," *International Journal of Innovative Computing, Information and Control*, vol. 10, no. 1, pp. 1519-1531, 2014.

[15] Estruch V., Ferri C., Hernández-Orallo J., and Ramírez-Quintana M., "Bagging Decision Multi-Trees, Multiple Classifier Systems," *in Proceedings of International Workshop on Multiple Classifier Systems*, Berlin, pp. 41-51, 2004.

[16] Hieu P. and Olafsson S., "Bagged Ensembles with Tunable Parameters," *Computational Intelligence*, vol. 35, no. 1, pp. 184-203, 2019.

[17] Hothorn T. and Lausen B., "Double-Bagging: Combining Classifiers by Bootstrap Aggregation," *Pattern Recognition*, vol. 36, no. 6, pp. 1303-1309, 2003.

[18] Jiang Y., Ling J., Li G., Dai H., and Zhou Z., "Dependency Bagging," *in Proceedings of International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing*, Berlin, pp. 491-500, 2005.

[19] Jorge A. and Azevedo P., "An Experiment with Association Rules and Classification: Post-Bagging and Conviction," *in Proceedings of International Conference on Discovery Science*, Berlin, pp. 137-149, 2005.

[20] Jurek A. et al., "A Survey of Commonly Used Ensemble-Based Classification Techniques," *The Knowledge Engineering Review*, vol. 29, no. 5, pp. 551-581, 2014.

[21] Kilimci Z. and Omurca S., "Enhancement of The Heuristic Optimization Based Extended Space Forests with Classifier Ensembles," *The International Arab Journal of Information Technology*, vol. 17, no. 2, pp. 188-195, 2020.

[22] Lichman M., "UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]," *Irvine, CA: University of California, School of Information and Computer Science*, Last Visited 2013.

[23] Louzada F., Anacleto-Junior O., Candolo C., and Mazucheli J., "Poly-Bagging Predictors for Classification Modelling for Credit Scoring," *Expert Systems with Applications*, vol. 38, no. 10, pp. 12717-12720, 2011.

[24] Mohamed H., Negm A., Zahran M., and Saavedra O., "Assessment of Ensemble Classifiers Using the Bagging Technique for Improved Land Cover Classification of multispectral Satellite Images," *The International Arab Journal of Information Technology*, vol. 15, no. 2, pp. 270-277, 2018.

[25] Quinlan J., "Bagging, Boosting, and C4.5," *in Proceedings of the 13th National Conference on Artificial Intelligence*, Portland, pp. 725-730, 1996.

[26] Shmulevich I. and Dougherty D., Genomic Signal Processing, Princeton University Press, 2007.

[27] Skurichina M. and Duin R., "Bagging for Linear Classifiers," *Pattern Recognition*, vol. 31, no. 7, pp. 909-930, 1998.

[28] Terabe M., Washio T., and Motoda H., "The Effect of Subsampling Rate on S3bagging Performance," *in Proceedings of 4th International Conference Advances in Intelligent Data Analysis*, Portugal, 2001.

[29] Wang G., Ma J. and Yang S., "IGF-Bagging: Information Gain Based Feature Selection for Bagging," The *International Journal of Innovative Computing, Information and Control*, vol. 7, no. 11, pp. 6247-6259, 2011.

[30] Wang G., Sun J., Ma J., Xu K. and Gu J., "Sentiment Classification: The Contribution of Ensemble Learning," *Decision Support Systems*, vol. 57, pp. 77-93, 2014.

[31] Wang Y. and Lin C., "Learning by Bagging And Adaboost Based on Support Vector Machine," *in Proceedings of 5th IEEE International Industrial Informatics Conference*, Vienna, pp. 663-668, 2007.

[32] Witten L., Frank E., Hall M., and Pal C., Practical Machine Learning Tools and Techniques, Morgan Kaufmann, 2016.

[33] Xiaoyuan S., Taghi M., and Xingquan Z., "Vob Predictors: Voting on Bagging Classifications," *in Proceedings of 19th International Conference on Pattern Recognition*, Tampa, pp. 1-4, 2008.

[34] Xie Z., Xu Y., Hu Q., and Zhu P., "Margin Distribution Based Bagging Pruning," *Neurocomputing*, vol. 85, pp. 11-19, 2012.

[35] Zaidi N ., Metric Learning and Scale Estimation in High Dimensional Machine Learning Problems with An Application to Generic Object Recognition, Thesis, Monash University, 2011.

[36] Zaman F. and Hirose H., "A Robust Bagging Method Using Median As A Combination Rule," *in Proceedings of Computer and Information Technology Workshops*, Sydney, pp. 55-60, 2008.

[37] Zeng X., Chao S., and Wong D., "Optimization of Bagging Classifiers Based on SBCB Algorithm," *in Proceedings of International Conference on Machine Learning and Cybernetics*, Qingdao, pp. 262-267, 2010.

[38] Zhou Z. and Yu Y., "Adapt Bagging to Nearest Neighbor Classifiers," *Journal of Computer Science and Technology*, vol. 20, no.1, pp. 48-54, 2005.

Goksu Tuysuzoglu received her BS in Information Systems Engineering in 2013 at Dogus University, Turkey. In the same year, she also received her double major in Industrial Engineering. Then, she received her MS in the department of computer engineering from Istanbul Technical University, Turkey in 2016. At the same time, she worked as a research and teaching assistant in there between the years 2014 and 2016. She is currently a PhD student in the department of computer engineering at Dokuz Eylul University, Turkey. She has also been working as a research and teaching assistant in the same department since 2016. She has BS graduation awards with ranking 1st in the Department and ranking 3rd in the Faculty and University. Her research interests include data mining and machine learning.

Derya Birant received her B.S., M.S. and Ph.D. degrees in Computer Engineering from Dokuz Eylul University, Turkey in 2000, 2002 and 2006 respectively. Since 2017 she has been an Associate Professor at the Computer Engineering Department of Dokuz Eylul University. She is the vice-chair of the Computer Engineering Department. She was a Visiting Lecturer at the South East European University in 2006 and Ege University between 2010 and 2012. She is the author of 5 book chapters and more than 70 publications (i.e. journal articles, conference papers). Dr. Birant has also served as an organizing committee member in several conferences. She has been involved in more than 20 long-term interdisciplinary R&D projects. Dr. Birant has several Most Downloaded Article certifications and has graduation awards, ranking 2nd in the Faculty and the Department. She was also the recipient of the Outstanding Achievement Award in 2010.