Classifier Ensemble Framework Based on Clustering Method

Hamid Parvin, Zahra Rezaei, Sajad Parvin
Nourabad Mamasani Branch
Islamic Azad University
Nourabad Mamasani, Iran

ABSTRACT: *This paper proposes an innovative combinational method how to select the number of clusters in the Classifier Selection by Clustering (CSC) to improve the performance of classifier ensembles both in stabilities of their results and in their accuracies as much as possible. The CSC uses bagging as the generator of base classifiers. Base classifiers are kept fixed as either decision trees or multilayer perceptron during the creation of the ensemble. Then it partitions the classifiers using a clustering algorithm. After that by selecting one classifier per each cluster, it produces the final ensemble. The weighted majority vote is taken as consensus function of the ensemble. Here it is probed how the cluster number affects the performance of the CSC method and how we can switch to a well approximation option for a dataset adaptively. We expand our studies on a large number of real datasets of UCI repository to reach a well conclusion.*

Keywords: Decision Tree, Classifier Ensembles, Bagging, AdaBoosting

Received: 2 October 2011, Revised 26 December 2011, Accepted 30 December 2011

1. Introduction

Voting is a mechanism based on democracy form of government. This mechanism has been proven to be better than dictatorship form of government. The superiority of democracy over dictatorship is not a surprise. It is due to the fact "*all are less probable to get a wrong decision*". So, ensemble methods are used in all fields inspired from the fact. Classification is a field that uses ensemble concept.

Although the more accurate classifier leads to a better performance, there is another approach to use many inaccurate classifiers specialized for a few data in the different problem spaces and using their consensus vote as the classifier. This can lead to a better performance due to the reinforcement of the consensus classifier in the error-prone feature spaces. In General, it is ever-true sentence that combining diverse classifiers usually results in a better classification [1]-[2].

This method uses many inaccurate classifiers, instead of one accurate classifier, specialized for a few data in the different problem spaces and applies their consensus vote as the classifier. In General, it is ever-true sentence that combining diverse classifiers usually results in a better classification [5].

Diversity has a very important role in success of ensemble methods. The diversity assures the undependability of their classifiers; in the other word, the misclassifications of the classifiers don't occur simultaneously. Kuncheva [8] explains that the ensemble of a number of classifiers can always reach a better performance (even can reach a perfect accuracy) as the number of classifiers become greater, provided that they are independent (diverse).

Creating a number of classifiers diverse enough to be appropriate to participate in an ensemble is a familiar challenge. There is

a very large variety of methods to reach a satisfactory diversity. Kuncheva's approach is based on the metrics that represent the amount of similarities or differences of classifier outputs.

Gianito et al. [6] imply a clustering and selection method to deal with the diversity generation. In that work, at first, a large number of classifiers with different initializations are produced, and then they select a subset of them according to their distances in their output space. They don't take into consideration how the base classifiers are created.

In this paper also a framework for development of combinational classifiers is proposed where a number of train data-bags are first bootstrapped from train data-set. Then a pool of weak base classifiers is created; each classifier is trained on one distinct data-bag. After that to get rid of similar base classifiers of the ensemble, using a clustering algorithm, here k-means, the classifiers are partitioned. The partitioning is done considering the outputs of classifiers on train dataset as feature space. In each partition, one classifier, the head of cluster, is selected to participate in final ensemble. Then, to produce consensus vote, different votes (or outputs) are gathered out of ensemble. After that the weighted majority voting algorithm is applied over them. The weights are determined using the accuracies of the base classifiers on train dataset.

The main aim of construction of An Artificial Neural Network (ANN), a model that simulates the structure and properties of biological neurons, is information processing, without necessarily creating a highly complex model of a real biological system. ANN is composed of a large number of interconnected processing elements, so called neurons, working in together to solve specific problems such as pattern recognition or data classification, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons. ANN learns the input/output relationship through training with adapting weights of its connection [7].

The Multi Layer Perceptrons (MLP), the most representatives of ANNs, is a linear classifier for classifying data specified by parameters and an output function. Its parameters are adapted similar to stochastic steepest gradient descent. The units each perform a biased weighted sum of their inputs and pass this activation level through a transfer function to produce their output, and the units are arranged in a layered feedforward topology. The network thus has a simple interpretation as a form of input-output model, with the weights and thresholds (biases) the free parameters of the model. Important issues in Multilayer Perceptrons design include specification of the number of hidden layers and the number of units in these layers [7].

One way is to set the weights explicitly, using a prior knowledge. Another way is to '*train*' the MLP, feeding it by teaching patterns and then letting it change its weights according to some learning rule. In this paper the MLP is used as one of the base classifiers.

Decision tree is one of the most versatile classifiers in the machine learning field. Decision tree is considered as one of the unstable classifiers that can be suitable for ensemble construction. It uses a tree-like graph or model of decisions. The kind of representation is appropriate for experts to analyze what classifier does [10]. The ensemble of a number of decision trees is a well-known ensemble called Random Forest which is one of the most powerful ensemble algorithms. The algorithm of random forest was first developed by Breiman [2]. In this paper, decision tree is totally used as one of the base classifiers.

Rest of this paper is organized as follows. Section 2 is related works. In section 3, we explain the proposed method. Section 4 demonstrates results of our proposed method against traditional ones comparatively. Finally, we conclude in section 5.

2. Background

Generally, there are two important challenging approaches to combine a number of classifiers that use different train sets. They are Bagging and Boosting. Both of them are considered as two methods that are sources of diversity generation.

The term Bagging is first used by [2] abbreviating for Bootstrap AGGregatING. The idea of Bagging is simple and interesting: the ensemble is made of classifiers built on bootstrap copies of the train set. Using different train sets, the needed diversity for ensemble is obtained.

Breiman [3] proposes a variant of Bagging which it is called Random Forest. Random Forest is a general class of ensemble building methods using a decision tree as the base classifier. To be labeled a "*Random Forest*", an ensemble of decision trees should be built by generating independent identically distributed random vectors and use each vector to grow a

decision tree. Bagging involves having each classifier in the ensemble vote with equal weight. In order to promote model diversity, bagging trains each model in the ensemble using a randomly-drawn subset of the training set. As an example, the random forest algorithm combines random decision trees with bagging to achieve very high classification accuracy. In this paper Random Forest algorithm [8] is implemented and compared with the proposed method.

Boosting is inspired by an online learning algorithm called Hedge(β) [4]. Boosting involves incrementally building an ensemble by training each new classifier to emphasize the training instances that previous classifiers misclassified. This algorithm allocates weights to a set of strategies used to predict the outcome of a certain event. At this point we shall relate Hedge(β) to the classifier combination problem. Boosting is defined in [4] as related to the "*general problem of producing a very accurate prediction rule by combining rough and moderately inaccurate rules of thumb*." The main boosting idea is to develop the classifier team D incrementally, adding one classifier at a time. The classifier that joins the ensemble at step k is trained on a dataset selectively sampled from the train dataset Z. The sampling distribution starts from uniform, and progresses towards increasing the likelihood of *"difficult"* data points. Thus the distribution is updated at each step, increasing the likelihood of the objects misclassified at step *k*-1. Here the correspondence with Hedge(β) is transposed. The classifiers in D are the trials or events, and the data points in *Z* are the strategies whose probability distribution we update at each step. The algorithm is called AdaBoost which comes from ADAptive BOOSTing. In some cases, boosting has been shown to yield better accuracy than bagging, but it also tends to be more likely to overfit the training data. By far, the most common implementation of Boosting is AdaBoost, although some newer algorithms are reported to achieve better results. One version of these algorithms is arc-x4 which outperforms the common ADAboost [8].

2.1 Artificial Neural Network

A first wave of interest in ANN (also known as '*connectionist models*' or '*parallel distributed processing*') emerged after the introduction of simplified neurons by McCulloch and Pitts in 1943. These neurons were presented as models of biological neurons and as conceptual components for circuits that could perform computational tasks. Each unit of an ANN performs a relatively simple job: receive input from neighbors or external sources and use this to compute an output signal which is propagated to other units. Apart from this processing, a second task is the adjustment of the weights. The system is inherently parallel in the sense that many units can carry out their computations at the same time. Within neural systems it is useful to distinguish three types of units: input units (indicated by an index i) which receive data from outside the ANN, output units (indicated by an index o) which send data out of the ANN, and hidden units (indicated by an index h) whose input and output signals remain within the ANN. During operation, units can be updated either synchronously or asynchronously. With synchronous updating, all units update their activation simultaneously; with asynchronous updating, each unit has a (usually fixed) probability of updating its activation at a time t, and usually only one unit will be able to do this at a time. In some cases the latter model has some advantages.

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Figure 1. An exemplary raw data

An ANN has to be configured such that the application of a set of inputs produces the desired set of outputs. Various methods to set the strengths of the connections exist. One way is to set the weights explicitly, using a priori knowledge. Another way is to '*train*' the ANN by feeding it teaching patterns and letting it change its weights according to some learning rule. For example,

the weights are updated according to the gradient of the error function. For further study the reader must refer to an ANN book such as Haykin's book on theory of ANN (Haykin 1999).

2.2 Decision Tree Learning

DT as a machine learning tool uses a tree-like graph or model to operate deciding on a specific goal. DT learning is a data mining technique which creates a model to predict the value of the goal or class based on input variables. Interior nodes are the representative of the input variables and the leaves are the representative of the target value. By splitting the source set into subsets based on their values, DT can be learned. Learning process is done for each subset by recursive partitioning. This process continues until all remain features in subset has the same value for our goal or until there is no improvement in Entropy. Entropy is a measure of the uncertainty associated with a random variable.

Data comes in records of the form: (x,Y) = (x1, x2, x3,…, xn,Y). The dependent variable, Y, is the target variable that we are trying to understand, classify or generalize. The vector x is composed of the input variables, x1, x2, x3 etc., that are used for that task. To clarify that what the DT learning is, consider Figure 1. Figure 1 has 3 attributes Refund, Marital Status and Taxable Income and our goal is cheat status. We should recognize if someone cheats by the help of our 3 attributes. To do learn process, attributes split into subsets. Figure 2 shows the process tendency. First, we split our source by the Refund and then MarSt and TaxInc.

For making rules from a decision tree, we must go upward from leaves as our antecedent to root as our consequent. For example consider Figure 2. Rules such as following are apprehensible. We can use these rules such as what we have in Association Rule Mining.

Refund = Yes $\Rightarrow$ cheat = No

TaxInc < 80, MarSt = (Single or Divorce), Refund = No $\Rightarrow$ cheat = No

TaxInc > 80, MarSt = (Single or Divorce), Refund = No $\Rightarrow$ cheat = Yes

Refund = No, MarSt = Married $\Rightarrow$ cheat = No

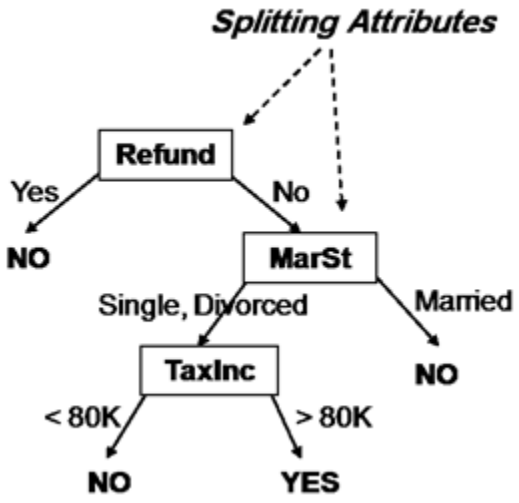


Figure 2. The process tendency for Figure 1

3. Proposed Method

In our proposed method, the aim is to use the most diverse set of classifiers obtained by Bagging or Boosting mechanism. In this method first, a number of classifiers are trained by the two well-known mechanisms: Bagging or Boosting and then the produced classifiers are partitioned according their outputs. Finally, the nearest classifier to the head of each produced cluster is selected.

Selection of one classifier from each cluster, and usage of them as an ensemble, can produce a diverse ensemble that outperforms the traditional Bagging and Boosting, i.e. usage of all classifiers as an ensemble, while each cluster is produced according to classifiers' outputs.

Figure 3 illustrates the training phase of the Bagging method in general. In proposed method, it is bootstrapped *n* subsets of dataset with *b* percent of the train dataset. Then a classifier is trained on each of those subsets. In addition, it is tested each decision tree over the whole of train dataset and calculated its accuracy. O_i and P_i , denoted as *i*th output of classifier over train dataset and its accuracy, respectively.

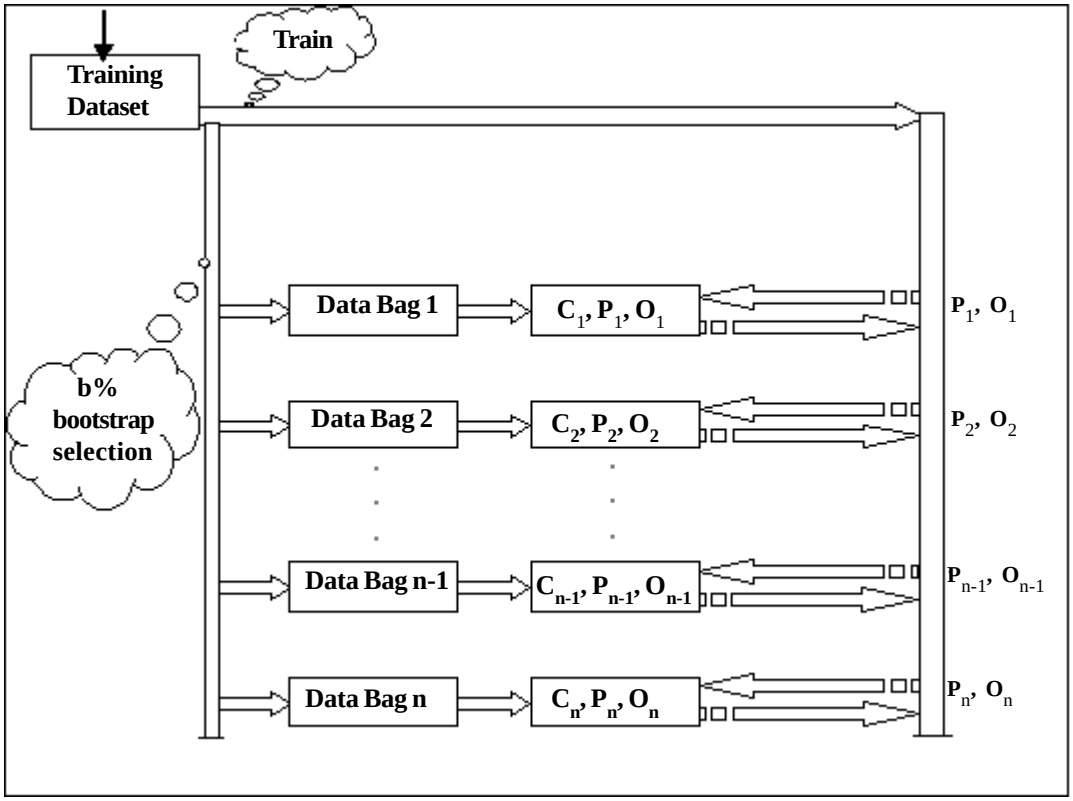


Figure 3. Training phase of the Bagging method

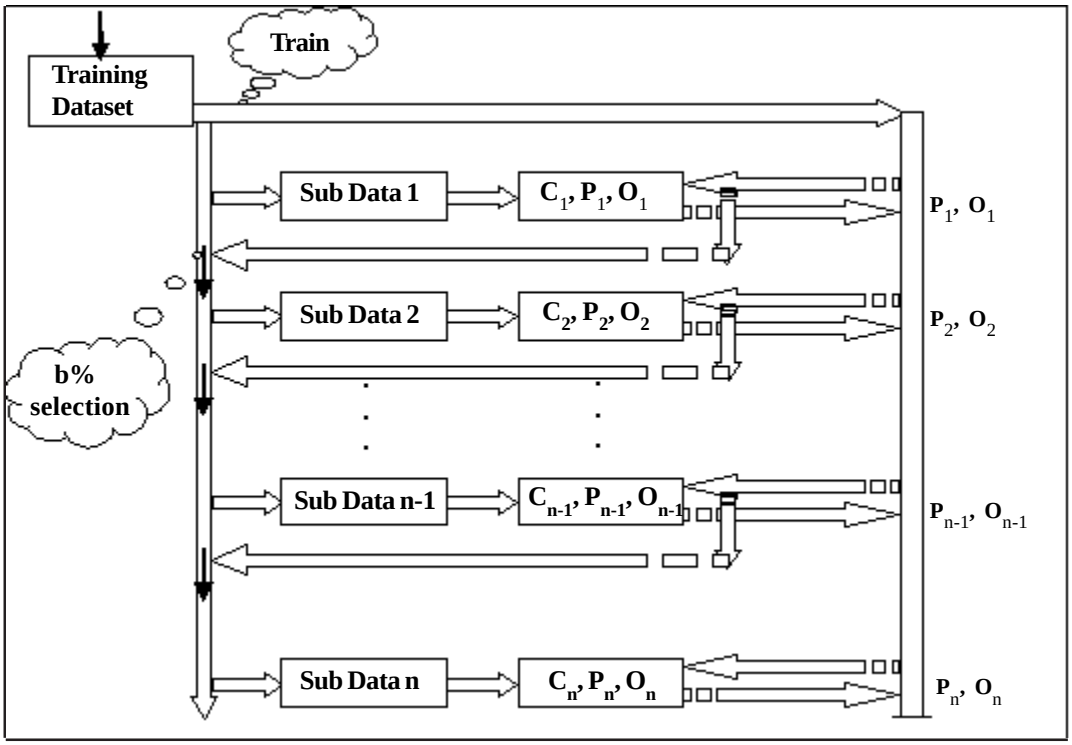
Figure 4 illustrates the training phase of the Boosting method, too. We again select a subset of dataset containing *b* percent of train dataset. Then the first classifier is trained on this subset. After that the first classifier is tested on the whole train dataset which this results in producing the O_1 and P_1. Using O_1, the next subset of *b* percent of train dataset is obtained. This mechanism is continued in such a way that obtaining *i*th subset of *b* percent of train dataset is produced considering the O_1 ,O_2, …, O_{i-1}. For more information about the mechanism of Boosting, the reader can refer to Kuncheva [8].

The proposed method is generally illustrated in the Figure 5. In the proposed method we first produce a dataset whose *i*th dataitem is O_i. Features of this dataset are real dataitems of under-leaning dataset. Then we have a new dataset having *n* classifiers and *N* features, where *n* is a predefined value showing the number of classifiers produced by Bagging or Boosting and *N* is the cardinality of under-leaning datasets. After producing the mentioned dataset, we partition that dataset by use of a clustering algorithm that this results in some clusters of classifiers. Each of the classifiers of a cluster has similar outputs on the train dataset; it means these classifiers have low diversities, so it is better to use one of them in the final ensemble rather than all of them. For escaping from outlier classifiers, we ignore from the clusters which contain number of classifiers smaller than a threshold.

Let us assume that *E* is the ensemble of *n* classifiers $\{C_1 ,C_2 ,C_3 \ldots C_n\}$. Also assume that there are *m* classes in the case. Next,

assume applying the ensemble over data sample *d* results in a binary *D* matrix like equation 1.

$$D = \begin{bmatrix} d_{1\ 1} & d_{1\ 2} & \cdot & d_{1\ n} \\ \cdot & \cdot & \cdot & \cdot \\ d_{m-1\ 1} & d_{m-1\ 2} & \cdot & d_{m-1\ n} \\ d_{m\ 1} & d_{m\ 2} & \cdot & d_{m\ n} \end{bmatrix} \quad (1)$$



d

where $d_{i,j}$ is one if classifier *j* votes that data sample *d* belongs to class *i*. Otherwise it is equal to zero. Now the ensemble decides the data sample *d* to belong to class *q* according to equation 2.

$$q = \arg \max_{i=1}^{m} \left| \sum_{j=1}^{n} w_j * d_{ij} \right| \quad (2)$$

where w_j is the weight of classifier *j* which is obtained optimally according to equation 3 [8].

$$w_j = \log \frac{P_j}{1 - P_j} \quad (3)$$

where p_j is accuracy of classifier *j* over total train set. Note that a tie breaks randomly in equation 2.

4. Experimental Results

Evaluation metric based on which an output of a classifier is computed is discussed in the first subsection of this section. The details of the used datasets are given in the subsequent section. Then the settings of experimentations are given. Finally the experimental results are presented.

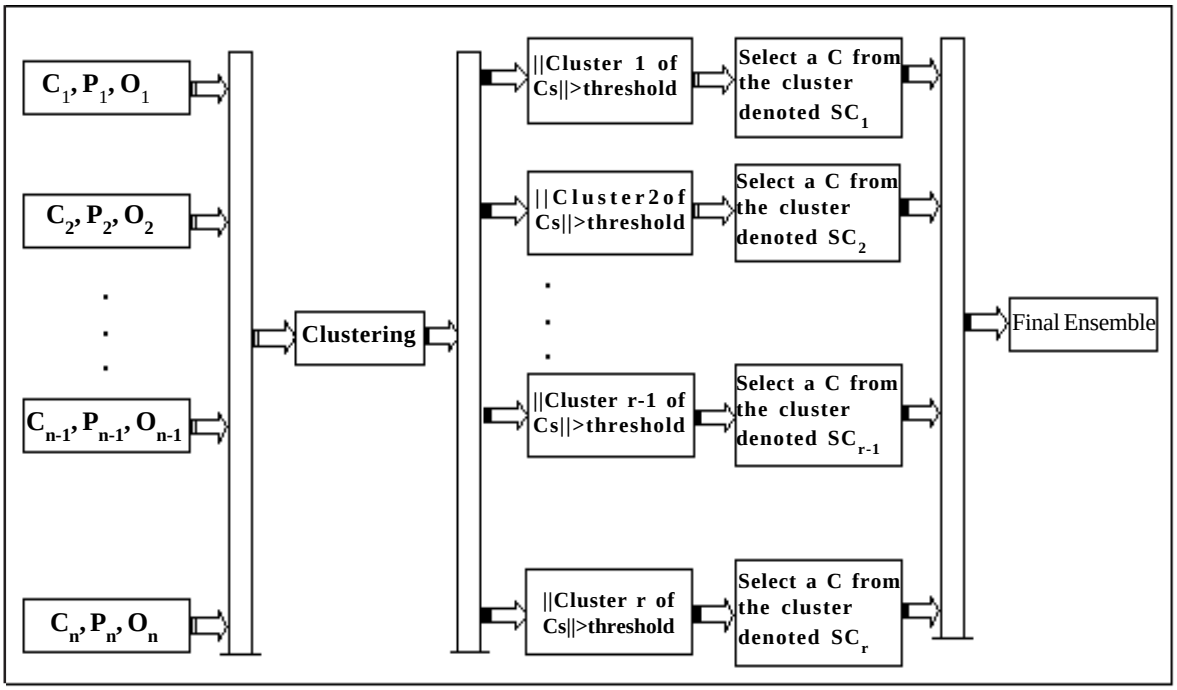


Figure 5. Proposed method for selecting the final ensemble from a pool of classifier generated by Bagging or Boosting

Dataset Name	# of dataitems	# of features	# of classes	Data distribution per classes
Breast Cancer	404	9	2	444-239
Bupa	345	6	2	145-200
Glass	214	9	6	70-76-17-13-9-29
Galaxy	323	4	7	51-28-46-38-80-45-35
half-ring	400	2	2	300-100
Heart	462	9	2	160-302
Ionosphere	351	34	2	126-225
Iris	150	4	3	50-50-50
test Monk 1	412	6	2	216-216
test Monk 2	412	6	2	216-216
test Monk 3	412	6	2	216-216
train Monk 1	124	6	2	62-62
train Monk 2	169	6	2	105-64
train Monk 3	122	6	2	62-60
Wine	178	13	3	59-71-48

Table 1. Details of used dataset

The accuracy is taken as the evaluation metric throughout all the paper. All the experiments are done using 4-fold cross validation. The results obtained by 4-fold cross validation are repeated as many as 10 independent runs. The averaged accuracies

over the 10 independent runs are reported.

The proposed method is examined over 13 different standard datasets and one artificial dataset. It is tried for datasets to be diverse in their number of true classes, features and samples. A large variety in used datasets can more validate the obtained results. Brief information about the used datasets is available in Table 1. These real datasets are available at UCI repository [9]. The details of half-ring dataset can be available in [10].

Note that some of datasets which are marked with star (*) in Table 1 are normalized. All experiments are done over the normalized features in the stared dataset. It means each feature is normalized with mean of 0 and variance of 1, N(0, 1).

The measure of decision for each employed decision tree is taken as Gini measure. The threshold of pruning is set to 2. Also the classifiers' parameters are fixed in all of their usages.

All multilayer perceptions which are used in the experiments have two hidden layers including 10 and 5 neurons respectively in the first and second hidden layers, as well as they are iterated 100 epochs.

In all experiments *n*, *b* and threshold of accepting a cluster are set to 151, 30 and 2 (i.e. only the clusters with one classifier is dropped down) respectively. All the experiments are done using 4-fold cross validation. Clustering is done by k-means clustering algorithm with different *k* parameters.

Table 2 shows the accuracies of different methods by considering a DT as each of the base classifiers. Table 3 shows the accuracies of different methods by considering a MLP as each of the base classifiers. The parameter *r* is set to 33 to reach the results of the Table 2 and Table 3.

	Arc-X4	Random Forest	Classifier Selection By RF	Classifier Selection By Arc-X4	Cluster and Selection
Breast Cancer*	95.74	96.32	**96.47**	95.05	93.68
Balance Scale*	94.44	93.60	**94.72**	94.24	94.44
Bupa*	70.64	72.09	**72.97**	66.28	64.53
Glass*	65.04	**70.28**	**70.28**	62.26	60.85
Galaxy*	70.59	**73.07**	72.45	70.28	70.94
Half-Ring*	**97.25**	95.75	**97.25**	95.75	95.75
SAHeart*	70.00	71.30	**72.61**	69.70	68.04
Ionosphere*	90.31	92.31	**91.45**	89.74	87.64
Iris*	**96.62**	95.27	**96.62**	95.95	94.59
Monk problem1**	98.11	97.49	**98.76**	97.37	98.34
Monk problem2**	97.01	86.64	**97.62**	86.73	97.14
Monk problem3**	87.29	96.92	**96.97**	96.34	87.31
Wine*	96.07	**97.19**	**97.19**	95.51	92.61
Yeast*	53.17	**53.98**	**53.98**	52.09	54.51
Average	84.45	85.16	**86.38**	83.38	82.88

Table 2. Comparison of the results by considering Decision Tree as base classifier. * shows the dataset is normalized, and 4 fold cross validation is taken for performance evaluation. ** shows that the train and test sets are predefined and averaged over 10 independent runs

While we choose only at most 22 percent of the base classifiers of Bagging, the accuracy of their ensemble outperforms the full ensemble of them, i.e. Bagging Method. Also it outperforms Boosting method and proposed method based on Boosting method.

Because the classifiers selected in this manner (by Bagging along with clustering), have different outputs, i.e. they are as diverse as possible, they are more suitable than ensemble of all them. It is worthy to mention that the Boosting is inherently diverse enough to be an ensemble totally; and the reduction of ensemble size by clustering destructs their Boosting effect. Take it in the consideration that in Boosting ensemble, each member covers the drawbacks of the previous ones.

	Arc-X4	Random Forest	Classifier Selection By Bagging	Classifier Selection By Arc-X4	Cluster and Selection
Breast Cancer*	**97.06**	96.91	96.91	96.47	96.19
Balance Scale*	93.27	91.99	91.35	92.95	**95.75**
Bupa*	70.06	71.22	**72.09**	68.02	71.98
Glass*	66.04	**66.98**	**67.45**	66.04	67.05
Galaxy*	**87.00**	85.62	85.62	84.52	**87.00**
Half-Ring*	**97.25**	95.75	**97.25**	97.75	**97.25**
SAHeart*	**73.04**	72.39	71.52	69.70	68.04
Ionosphere*	90.31	92.31	**91.45**	71.09	87.64
Iris*	**96.62**	96.62	**97.97**	97.33	97.33
Monk problem1**	98.06	92.23	**98.43**	97.87	98.34
Monk problem2**	87.35	85.68	**87.41**	87.23	87.21
Monk problem3**	97.09	95.87	**97.33**	96.99	96.77
Wine*	96.59	96.06	**97.19**	95.51	95.23
Yeast*	60.85	**61.19**	**61.19**	60.85	60.56
Average	86.45	85.50	**86.57**	85.70	86.10

Table 3. Comparison of the results by considering MLP as base classifier. * shows the dataset is normalized, and 4 fold cross validation is taken for performance evaluation. ** shows that the train and test sets are predefined and averaged over 10 independent runs

5. Conclusion

In this paper, we have proposed a new method to improve the performance of classification. The proposed method uses Bagging as generator of the base classifiers. Then using k-means we partition the classifiers. After that we select one classifier per a validated cluster.

Using the decision tree as base classifier increases the gap between the three approaches to generate the base classifiers. It is due to special feature of the decision tree. Because it is very sensitive to its train set, the use of decision tree as base classifier is very consistent with the Bagging mechanism.

While we choose only at most 22 percent of the base classifiers of Bagging, the accuracy of their ensemble outperforms the full ensemble of them. Also it outperforms Boosting.

As a future work, one can turn to research on the variance of the method. Since it is said about Bagging can reduce variance and Boosting can simultaneously reduce variance and error rate.

References

[1] Günter, S., Bunke, H. (2002). Creation of Classifier Ensembles for Handwritten Word Recognition Using Feature Selection Algorithms, *In*: Proceedings of the Eighth International Workshop on Frontiers in Handwriting Recognition (IWFHR'02), p.183, August 06-08.

[2] Kuncheva, L. I. (2005). Combining Pattern Classifiers, Methods and Algorithms. New York: Wiley.

[3] Giacinto, G., Roli, F. (2001). An approach to the automatic design of multiple classifier systems. Pattern Recognition Letters, 22, 25–33.

[4] Breiman, L. (1996). Bagging Predictors, Journal of Machine Learning, 24(2), 123-140.

[5] Breiman, L. (2001). Random Forests. Machine Learning 45(1), 5-32.

[6] Yang, T. (2006). Computational Verb Decision Trees. International Journal of Computational Cognition, 4(4), 34–46.

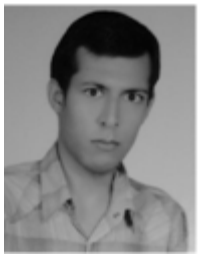
[7] Freund,Y., Schapire, R. E. (1997). A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. J. Comput. Syst. Sci. 55(1), 119-139.

[8] Parvin, H., Minaei-Bidgoli, B., Beigi, A. (2011). A New Classifier Ensembles Framework. Knowledge-Based and Intelligent Information and Engineering Systems, p. 110-119.

[9] Blake, C. L., Merz, C. J. (1998). UCI Repository of machine learning databases, http://www.ics.uci.edu/~mlearn/MLRepository.html

[10] Minaei-Bidgoli, B., Topchy, A. P., Punch, W. F. Ensembles of Partitions via Data Resampling. ITCC: 188-192.

Author's Profile



Hamid Parvin received his B.Sc. and MSc. degrees in computer engineering from Chamran University, Ahvaz, in 2006 and Iran University of Science and Technology (IUST), Tehran, in 2008, respectively. Now, he is a PhD student in computer engineering, artificial intelligence in Iran University of Science and Technology, Iran. His research interests are in pattern recognition, machine learning, particularly, data clustering, classification, and ensemble methods.