2008 20th IEEE International Conference on Tools with Artificial Intelligence

Ensemble Learning of Regional Classifiers

Byung-woo Lee, Yong-chan Na, Byonghwa Oh, and Jihoon Yang*
Department Of Computer Engineering, Sogang University, Seoul, Korea
elva1212@naver.com, ycna@sogang.ac.kr, mr-five@hanmail.net, yangjh@sogang.ac.kr

Abstract

We present a new ensemble learning method that employs a set of regional classifiers, each of which learns to handle a subset of the training data. We split the training data and generate classifiers for different regions in the feature space. When classifying new data, we apply a weighted voting among the classifiers that include the data in their regions. We used 10 datasets to compare the performance of our new ensemble method with that of single classifiers as well as other ensemble methods such as bagging and Adaboost. As a result, we found that the performance of our method is comparable to that of Adaboost and bagging when the base learner is C4.5. In the remaining cases, our method outperformed the benchmark methods.

1. Introduction

In machine learning, the improvement of the classification accuracy has been an important objective. Ensemble learning is one of the methods that have been very successful in this aspect. In the supervised learning framework, ensemble learning aims not to find the best single hypothesis for explaining the training data but to create a hypothesis by combining the set of hypotheses produced by individual classifiers in order to classify new data more accurately [1].

There does not exist a single classifier that produces the best result for all kinds of data (or problems or applications). However, if we decompose a complex problem into multiple sub-problems that are possibly easier to understand and to solve using various classifiers, and combine the outputs of multiple classifiers through ensemble methods, we can obtain improved performance. This advantage of ensemble learning is thus worthwhile despite its computational overhead over individual learning by a single classifier. Ensemble classifiers indeed have been experimentally proven to be more accurate than single classifiers [2][3][4].

In common ensemble learning approaches such as bagging [5] and boosting [6], it is not considered which individual classifier is to be used for classifying a new instance. Bagging is nothing but a collection of classifiers on which the majority voting is applied to determine the final classification. Boosting is similar to bagging, but both the classifiers and the training examples are associated with weights which are used in the final classification. In other words, the individual classifiers are applied in either simple (in bagging) or weighted (in boosting) voting. If the set of instances (or the regions where they lie) that each classifier is strong at classification is known, the ensemble classifier will be able to perform more precise classification. Against this background, we present a new ensemble learning method that employs a set of *regional* classifiers each of which learns to handle a subset of the training data in a particular region. In testing a new instance, it votes on the class label among the classifiers that include the instance in its region.

The rest of the paper is organized as follows: Section 2 describes the condition for good ensemble. Section 3 explains our new ensemble learning method. Section 4 presents the experimental results on various real-world data to evaluate the performance of our approach. Section 5 concludes with summary and discussion of some directions for future research.

2. The condition for good ensemble

A learning algorithm is called *unstable* if "small" changes in the training data lead to significantly different classifiers and relatively "large" changes in accuracy. For instance, the decision tree learning algorithms such as ID3 [7], C4.5 [8] are unstable. Such unstable algorithms are known to have a high variance [9]. The opposite concept of the unstable algorithm is the *stable* algorithm. SVM [10] and naïve Bayes [11] are good examples of such learning algorithms.

Generally, ensemble learning such as bagging or boosting works well with unstable algorithms [12]. On the other hand, bagging and boosting does not show any

*This work was supported by the Special Research Grant of Sogang University.

1082-3409/08 $25.00 © 2008 IEEE
DOI 10.1109/ICTAI.2008.140

remarkable improvement on naive Bayes or other stable algorithms based on the linear discriminant analysis [13]. In fact, some research even demonstrated the negative effect of SVM-based ensemble learning [14][15].

The high performance of unstable learner ensembles is related to diversity, meaning that each classifier has errors in different areas in the input space [16]. Suppose that we have an ensemble of three classifiers. If the three classifiers are not diverse, a single incorrect classifier means the other two also have a large chance of being wrong. This will eventually result in an incorrect ensemble.

According to Hansen and Salamon [16], a necessary and sufficient condition for an ensemble to be more accurate than any of its individual classifiers is that the classifiers should be accurate and diverse.

Suppose an ensemble classifier is created by combining five individual classifiers, each of which is completely independent of each other and has 70% of classification accuracy. And suppose the simple majority voting is used to combine the prediction of individual classifiers. For this ensemble classifier to classify a pattern correctly, more than 3 individual classifiers should be correct. We can calculate this probability as follows:

$$\sum_{i=3}^{5}\binom{5}{i}\cdot(0.7)^{i}\cdot(0.3)^{5-i}=0.84 \quad (1)$$

As seen in Equation (1), the accuracy of this ensemble classifier is approximately 84%, which is 14% higher than that of its individual classifiers. The misclassification rate of this ensemble classifier is approximately 16%, which is almost a half of its individual classifiers' error rate, 30%. If an ensemble classifier is created with 5 classifiers of 90% accuracy, it leads to Equation (2).

$$\sum_{i=3}^{5}\binom{5}{i}\cdot(0.9)^{i}\cdot(0.1)^{5-i}=0.99 \quad (2)$$

As seen in Equation (1) and (2), the improvement of the ensemble of 5 classifiers with 70% is bigger than the ensemble of 5 classifiers with 90%. However, the latter shows better performance than the former. Therefore, even though the gain in accuracy from forming an ensemble is not as significant, we may as well use more accurate classifiers to generate a more accurate ensemble classifier.

Note that the complete diversity is fairly strong assumption to obtain in most of the real-world applications. Therefore, we relax the assumption and make use of classifiers with a certain degree of diversity to create more accurate ensemble classifier.

3. Ensemble of regional classifiers

In our study, we attempt to design a new ensemble method which splits the training set and allocate the splits of the training data to different classifiers. This ensemble method can be represented as a tree in which each node corresponds to a classifier that is assigned to a particular region in the feature space. In this ensemble method, not only the leaf nodes, but also all the internal nodes are classifiers. We use the training samples in its region to generate the corresponding classifier. In addition, each classifier in the tree can be trained by a different learning algorithm. When classifying a new sample, we apply a weighted voting among the classifiers that include this sample in their region. We call this method the *Ensemble Learning of Regional Classifiers (ELRC)*. Figure 1 depicts ELRC.

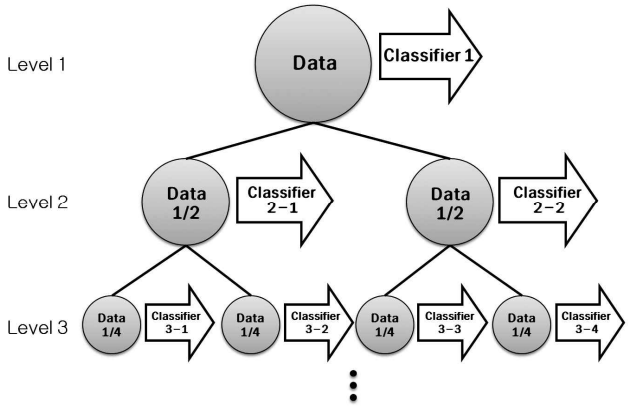

Figure 1. Overview of ELRC

3.1. Learning algorithm

ELRC generates classifiers of different regions in the feature space. Among the predefined group of learning algorithms, we choose the one that produces the best performance in each region. We can consider as many learning algorithms as we wish in order to obtain good performance. Currently, we consider three algorithms (C4.5 [8], SVM [10], and naïve Bayes [11]). For each region, we create three classifiers by the three learning algorithms and select the best one. The three algorithms are chosen since they belong to the set of machine learning algorithms that are most common and widely used nowadays. C4.5 is a decision tree learning algorithm that determines the nodes in the tree based on the *entropy* (as described below) in order to find the most relevant feature with respect to the class, and thus to induce the most succinct tree. SVM is a kernel classifier that finds the class-separating hyperplane with the maximal margin. Naïve Bayes computes the conditional probabilities that can be used to compute the posterior probability of the

classes, under the strong independence assumption among the features given the class. (See references for more detailed descriptions on the algorithms.)

The performance is measured through 10-fold cross-validation, where we choose the learning algorithm with the highest accuracy. If there is a tie, the learning algorithm with the highest training accuracy is selected. In case of a tie in training accuracy, we choose the learning algorithm that is selected the most frequently by the ancestor nodes when there are ancestor nodes, or follow a predefined priority for the root node. Since we choose the best learning algorithm at each step, we expect to generate accurate classifiers. Meanwhile, the diversity is also maintained through utilization of different learning algorithms.

The target regions are iteratively split as learning progresses, and the corresponding classifiers are generated according to the patterns in each region. The detailed learning process of ELRC for a sample data set is illustrated in Figure 2.

As shown in Figure 2, classifier 1 is generated using the entire data D by the selected learning algorithm. Then the training data D is divided into two regions D_1, D_2 containing nearly the same number of instances. In order to divide the data close to halves, we compute the *Score* of each feature A in regard to the division and select the feature that has the maximum *Score*:

$$Entropy(X) = -\sum_{i=1}^{n} p(x_i)\log p(x_i)$$
$$Score(D, A) = Entropy(D) - \frac{||D_1| - |D_2||}{|D|}\sum_{i=1}^{2}\frac{|D_i|}{|D|}\cdot Entropy(D_i) \quad (3)$$

The entropy characterizes the (im)purity of an arbitrary collection of examples [17]. Equation (3) resembles the definition of information gain [7]. Simply put, information gain is the expected reduction in entropy caused by partitioning the examples according to the selected feature [17]. We add $||D_1| - |D_2|| / |D|$ to information gain, which measures how equally the data is divided. Thus we can consider the balanced division of data as well as the reduction in entropy. If the partitions become imbalanced, the size of the tree becomes unnecessarily bigger. That is, the number of classifiers becomes larger. Meanwhile, a small subset with insufficient data might cause the classifier insufficient learning. That is, when the training data is not sufficient, the probability of selecting the correct hypothesis decreases since there are more likely to be several hypotheses that have the same maximum accuracy in the hypothesis space of the learning algorithm. Therefore, we need to divide the data as equally as possible.

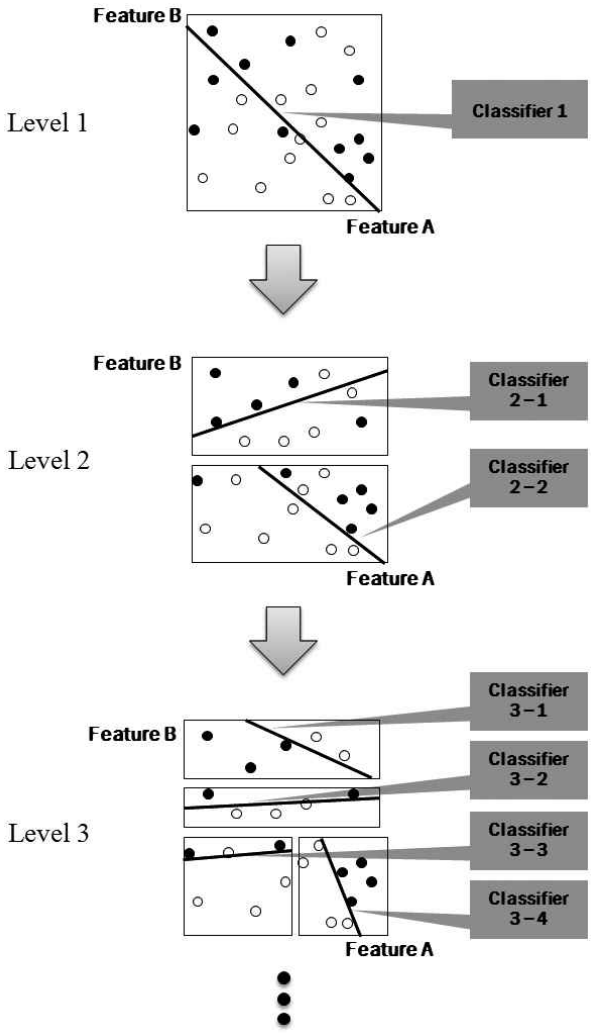

Figure 2. Learning process of ELRC

We choose the feature with the maximum *Score*, so as to induce subsets of the data to be easier to classify and construct accurate classifiers. At the same time, since each classifier is generated from different training data, we can keep the classifiers diverse.

Through the repetition of this process on the divided data, succeeding classifiers 2-1 and 2-2 are generated. Similarly, classifiers 3-1, 3-2, 3-3, and 3-4 are generated with the data in the four regions in level 3. ELRC keeps repeating this process until at least one of the following stopping conditions are satisfied. Since the performance of a learning algorithm is measured through 10-fold cross-validation in ELRC, we need the last criterion.

- If the predefined level is reached
- If the entropy of the region is 0
- If the classification error in the region is 0
- If the number of instances in the region is less than 10

3.2. Classification algorithm

When ELRC classifies a test pattern, the classifiers in the ensemble that include the test pattern in their allocated region vote on the class label. In other words, a pattern is classified by the classifiers in a path of the tree, each of which is in charge of the region that includes the test pattern. Figure 3 illustrates the classification process in ELRC.

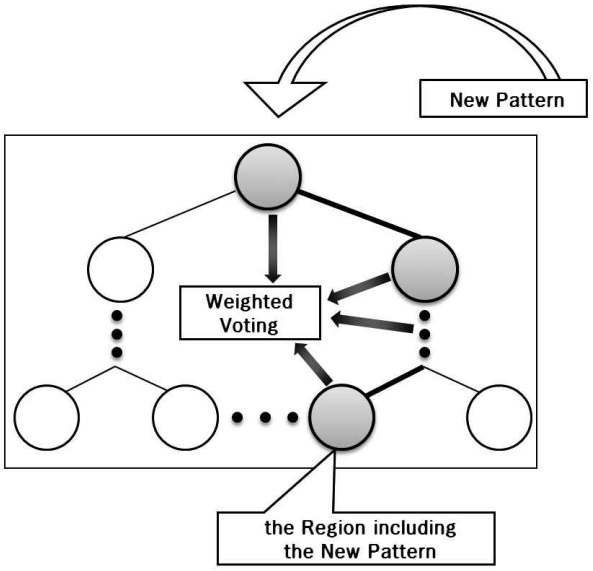

Figure 3. Classification process in ELRC

As shown in Figure 3, classifiers that fall in the path from the root to the leaf node that include the new pattern in their regions vote on the class. Only these classifiers are trained with the patterns in the same regions during the learning process, hence, other classifiers are not allowed to vote. As a result, we can assure good performance of the classifiers that participate in the voting process.

Our expected result was that the data subsets will be easier to classify at the lower level of the tree. However, experiments show that this is not always true. The cases where the number of errors increased after the data split were often observed. Therefore, we calculate the weight of each classifier according to its accuracy rather than giving heavier weights to the lower nodes. We calculate the weight of classifier as Equation (4).

$$Weight(N) = \frac{|I|}{|PI|} \cdot PW \tag{4}$$

where N is a classifier, I is the set of instances that are classified correctly by N among the instances in the allocated region of N, PI is the set of instances that are classified correctly by the parent of N among the instances in region of N, and PW is the weight of N's parent. The weight of the classifier in root node is 1 and each classifier has a different weight according to the accuracy.

Instead of the discrete classifications, we use the class probability of each classifier to decide the final prediction of ELRC. The use of class probability increases the number of possible outcomes of classification, which results in higher diversity. The votes of each class are determined according to the weighted class probability. The class with the most votes is decided as ELRC's prediction value. We calculate the vote as Equation (5)

$$Voting(x,P) = \sum_{i=1}^{L} Weight(N_i) \cdot Class\_\Pr ob(x,P,N_i) \tag{5}$$

and decide the prediction of ELRC as Equation (6)

$$Classification(P) = \arg\max_{i \in C} Voting(i,P) \tag{6}$$

where x is the class, P denotes the pattern, L represents the number of classifiers which vote on the class, N_i is the classifier which votes on the class, $Class\_Prob(x,P,N_i)$ is class probability that classifier N_i predicts the pattern P as class x, and C stands for the set of classes.

Overfitting can be caused by the classifier that does not rely on the distribution of the whole data but on the distribution of data in the regionally split areas instead. That is, classifiers in the lower nodes have the possibility of overfitting. The classifiers in the upper nodes in the ELRC can minimize the risk of overfitting because they provide more generalization than the classifiers in the lower nodes. That is why the upper nodes participate in the vote.

4. Experiments

4.1. Experimental Setup

In our experiments, we used 10 data sets from the UCI Machine Learning Data Repository [18]. Table 1 summarized the data sets. We chose naive Bayes [11], C4.5 [8] and SMO (which is the sequential minimal optimization algorithm for training a support vector classifier) [19] as the base learning algorithm and a priority among the algorithms is given in the order of SMO, naive Bayes, and C4.5

In order to measure the performance of ELRC in relation to existing methods, a single classifier with ELRC's base learning algorithm, an Adaboost [20] (the representative boosting algorithm) ensemble classifier with 21 classifiers, and a bagging ensemble classifier with 21 classifiers are compared in terms of the classification accuracy. For the base learner of the two benchmark

Table 1. Datasets

Dataset	# instance	# class	# attribute
Ablone	4177	28	8
Ecoli	336	8	7
Glass	213	6	9
Hypothyroid	215	3	5
Ionosphere	351	2	34
Page-Blocks	5473	5	10
SPECTF	267	2	44
Segment	2310	7	19
Spambase	4601	2	57
Vehicle	846	4	18

ensemble methods, we used the same algorithm as ELRC. In ELRC, we have to generate classifiers with three learning algorithms to select the best learning algorithm. Thus we generated 21 classifiers in order to learn ELRC having three levels, which has 7 nodes. For a fair comparison, boosting and bagging also generated 21 classifiers. Weka [21], the open data mining software, was used for the base learning algorithms, Adaboost, and bagging.

4.2. Results

Table 2 shows the average accuracies of single classifiers and the ensemble methods such as bagging, Adaboost, and ELRC that used SMO, naive Bayes and C4.5 as the base learning algorithm. 10-fold cross-validation was used to measure the performance. The most accurate classifiers among various methods are shown in boldface.

First, in comparison to single classifiers, ELRC always produced higher accuracy than single classifiers regardless of the base learning algorithm. This verifies the outstanding performance of ELRC. While bagging and Adaboost generally outperformed or were comparable to the single classifiers, they also produced poorer results for some data, as well.

ELRC was also generally more accurate than bagging and Adaboost with SMO or naive Bayes as the base learning algorithm. When either SMO or naive Bayes were used as the base learning algorithm, Adaboost and bagging failed to improve the performance of the single classifier. The poor performance of Adaboost and bagging is due to the fact that SMO and naive Bayes are stable learning algorithms. Note that ELRC hold more diversity in terms of the base learners than Adaboost and bagging because each classifier in the ELRC can use a different learning algorithm. Therefore, ELRC is more accurate than Adaboost and bagging on datasets that are best classified by SMO or naive Bayes single classifiers.

When C4.5 was used as the base learning algorithm, ELRC was comparable to bagging and Adaboost. ELRC was composed of 7 classifiers, while Adaboost and bagging were composed of 21 classifiers. When classifying a pattern, ELRC uses 3 classifiers and Adaboost and bagging use 21 classifiers. Thus, Adaboost and bagging can make more diverse predictions of classifiers than ELRC. Moreover, Adaboost and bagging apply sampling for each classifier and thus learn on possibly very different training data, while this is not the case in ELRC because the dataset which was assigned to a child node must be assigned to its parent nodes as well. Thus, Adaboost and bagging yielded more accurate results than ELRC in some datasets when C4.5 is used as the base learning algorithm.

Table 2. Accuracy comparison of ELRC and other methods

Dataset	SMO			naive Bayes			C4.5			ELRC
	Single	Ada	Bagging	Single	Ada	Bagging	Single	Ada	Bagging	
Abalone	25.31	25.31	25.58	23.8	23.8	23.64	20.9	22.15	24.47	**27.10**
Ecoli	84.22	86.3	83.63	86.29	86.23	86.27	70.3	82.43	84.5	**86.62**
Glass	53.94	58.14	55.22	47.45	47.92	50.76	69.39	**77.42**	76.56	74.18
Hypothyroid	89.85	95.81	89.77	96.75	95.81	97.21	94.91	94.42	93.49	**97.23**
Ionosphere	88.03	87.78	88.04	82.9	91.76	82.62	88.05	**93.18**	91.20	92.33
Page-Blocks	92.91	92.91	93.42	90.15	90.15	90.02	97.06	97.17	**97.37**	97.17
SPECTF	79.42	76.4	79.4	67.86	68.54	70.41	74.54	80.15	79.78	**81.30**
Segment	92.90	93.2	92.9	79.65	79.65	79.96	96.8	**98.27**	97.45	97.45
Spambase	90.39	90.78	90.81	79.5	79.5	79.7	93.04	**95.35**	94.24	93.74
Vehicle	74.55	74.43	74.9	46.28	46.28	46.52	75.02	76.92	74.32	**77.27**

5. Conclusion

We proposed a new ensemble method - ensemble of regional classifiers, ELRC. Compared with the existing methods such as bagging and boosting, our algorithm demonstrated outstanding performance when combined with stable learning algorithms as the base learner. ELRC was also comparable to bagging and Adaboost with unstable algorithms like C4.5 as the base learner.

ELRC's automatic selection of the best learning algorithm frees the user from the algorithm consideration. The user is only responsible for deciding the candidate group for the base learning algorithm. We expect splitting data would induce subsets of data easier to be classified and generate different training data for regional classifiers. This will lead to the construction of an ensemble with accurate base learners. Meanwhile, in the unfortunate cases where the correctness of the classifiers dropped from the splits, we lowered the weight of the regional classifier in order to reduce the influence of such classifiers. As well, we are able to create a variety of classifiers from different data subsets.

When classifying a new pattern, ELRC utilizes the classifier allocated to the region that includes the pattern, and its ancestors which are also trained with the patterns in the region. This rationale behind this idea is to obtain high classification accuracy as well as to prevent overfitting.

The accuracy of ELRC may start decreasing from a certain level. So it needs to be examined further to develop a well-defined method to determine the exact level where the performance starts to drop. For example, methods that remove unnecessary classifiers from ELRC, similar to the pruning method for decision tree classifiers, can be a good remedy for this problem. This work is left as a topic for future research.

6. References

[1] Dietterich, T. G., "Ensemble method in Machine learning", *Lecture Notes in Computer Science* 1857, 2000, pp. 1-15.

[2] Bauer, E., and Kohavi, R., "An Empirical Comparison of Voting Classification Algorithm: Bagging, Boosting, and Variants", *Machine Learning*, 1999, vol. 36, no. 1-2, pp. 105-142.

[3] Dietterich, T. G., "An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, and Randomization", *Machine Learning*, 2000, vol. 40, no.2, pp. 139-157.

[4] Optiz, D., and Maclin, R., "Popular Ensemble Methods: An Empirical Study", *Journal of Artificial Intelligence Research*, 1999, vol. 11, pp. 169-198.

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

[6] Freund, Y., and Schapire, O., "Experiments with a new boosting algorithm", *Proceedings of the Thirteenth International Conference on Machine Learning*, 1996, pp. 148-156.

[7] Quinlan, J., "Induction of Decision Tree", *Machine Learning*, 1986, vol. 1, no. 1, pp. 81-106.

[8] Quinlan, J., *C4.5: Programs for Machine learning*, Morgan Kaufmann, 1993.

[9] Duda, R. O., Hart, P. E. and Stork, D. G., *Pattern Classification*, Second edition, Wiley-interscience, 2000.

[10] Vapnik, V., *The Nature of Statistical Learning Theory*, Springer, 1995.

[11] Domingos, P. and Pazzani, M. "On the optimality of the simple Bayesian classifier under zero-one loss", *Machine Learning*, Vol. 29, pp. 103-137, 1997.

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

[13] Breiman, L., "Bias, Variance, and Arcing Classifiers", *Technical Report TR 460*, UC Berkeley, 1996.

[14] Buciu, I., Kotropoulos, C., and Pitas, I., "Combining Support Vector Machines for Accuracy Face Detection", *Proceedings Of ICIP'01*, 2001, pp. 1054-1057.

[15] Evgeniou, T., Perez-Breva, L., Pontil, M., and Poggio, T., "Bound on the Generalization Performance of kernel Machine Ensembles", Proceedings of the seventeenth International Conference on Machine Learning, 2000, pp. 271-278.

[16] Hansen, L. K., and Salamon, P., "Neural Network Ensembles", *IEEE Trans. Pattern Analysis and Machine Intell.*, 1990, vol. 12, pp. 993-1001.

[17] Mitchell, T. M., *Machine learning*, McGRAW-HILL, 1997.

[18] Asuncion, A., and Newman, D., UCI Machine Learning Repository [http://www.ics.uci.edu/~mlearn/ML Repository.html], Irvine, CA: University of California, School of Information and Computer Science, 2007.

[19] Platt, J., "Fast Training of Support Vector machines using Sequential Minimal Optimization", in Schölkopf, B., Burges, C., Smola, A. (Eds.), Adavances in Kernel Methods, The MIT press, 1999, pp. 185-208.

[20] Freund, Y., and Schapire, R., "A Decision Theoretic Generalization of On-Line Learning and an Application to Boosting", *Journal of Computer and System Science*, 1997, vol. 55, pp. 119-139.

[21] Witten, I., Frank, E., *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementation*, 2nd Ed., Morgan Kaufmann, San Francisco, 2005.