Hindawi
Mathematical Problems in Engineering
Volume 2018, Article ID 5036710, 13 pages
https://doi.org/10.1155/2018/5036710

Research Article

Classifying Imbalanced Data Sets by a Novel RE-Sample and Cost-Sensitive Stacked Generalization Method

Jianhong Yan and Suqing Han

Department of Computer Science, Taiyuan Normal University, Taiyuan 030012, China

Correspondence should be addressed to Jianhong Yan; yan_jian_hong@163.com

Received 3 June 2017; Revised 1 October 2017; Accepted 6 November 2017; Published 23 January 2018

Academic Editor: Michele Migliore

Learning with imbalanced data sets is considered as one of the key topics in machine learning community. Stacking ensemble is an efficient algorithm for normal balance data sets. However, stacking ensemble was seldom applied in imbalance data. In this paper, we proposed a novel RE-sample and Cost-Sensitive Stacked Generalization (RECSG) method based on 2-layer learning models. The first step is Level 0 model generalization including data preprocessing and base model training. The second step is Level 1 model generalization involving cost-sensitive classifier and logistic regression algorithm. In the learning phase, preprocessing techniques can be embedded in imbalance data learning methods. In the cost-sensitive algorithm, cost matrix is combined with both data characters and algorithms. In the RECSG method, ensemble algorithm is combined with imbalance data techniques. According to the experiment results obtained with 17 public imbalanced data sets, as indicated by various evaluation metrics (AUC, GeoMean, and AGeoMean), the proposed method showed the better classification performances than other ensemble and single algorithms. The proposed method is especially more efficient when the performance of base classifier is low. All these demonstrated that the proposed method could be applied in the class imbalance problem.

1. Introduction

Classification learning becomes complicated if the class distribution of the data is imbalanced. The class imbalance problem occurs when the number of representative instances is much less than that of other instances. Recently, the classification problem of imbalanced data appears frequently and has been widely concerned [1–3].

Usually, imbalanced data sets are grouped into binary class and the majority class and minority class are, respectively, denoted as negative class and positive class. Traditional techniques are divided into four categories. RE-sample technique is to increase the minority class of instances (oversampling) [4] or decrease the majority class of instances (undersampling) [5, 6]. Existing algorithms are improved by increasing the weight of positive instances [7]. Classifier ensemble method is widely adopted to deal with the imbalance problem in the last decade [8]. In cost-sensitive algorithms, data characters are incorporated with misclassification costs in the classification phase [9, 10]. In general, cost-sensitive and algorithms levels are more associated with the imbalance problem, whereas data level and ensemble learning can be used and independent of the single classifier.

Ensemble methods involving training base classifiers, integrating their results, and generating a single final class label can increase the accuracy of classifiers. Bagging algorithm [11] and Ada AdaBoost boost algorithm [12, 13] are the most common ensemble classification algorithms. Ensemble algorithms combined with the other three techniques are widely applied in the classification of imbalanced data sets. Cost-sensitive learning targets the imbalanced learning problem by using different cost matrices that can be considered as a numerical representation of the penalty of misclassifying examples from one class to another. Data characters are incorporated with misclassification costs in the classification phase. Cost-sensitive learning is closely related to learning from imbalanced data. In order to solve the imbalance problem, ensemble algorithms were combined with data preprocessing, cost-sensitive method, and relevant algorithms in previous studies. Four ensemble methods for imbalance data sets are commonly used: boosting-, bagging-, cost-sensitive-boosting and hybrid ensemble methods.

In boosting-based ensembles, the data preprocessing technique is embedded into boosting algorithms. In each iteration, the data distribution is changed by altering the weight to train the next classifier towards the positive class. These algorithms mainly include SMOTEBoost [14], RUSBoost [15], MSMOTEBoost [16], and DataBoost-IM [17] algorithms. In bagging-based ensembles, the algorithms combine bagging with data preprocessing techniques. The algorithms are usually simpler than their integration in boosting because of simplicity and good generalization ability of bagging. The family includes but not limited to OverBagging, UnderOverBagging [18], UnderBagging [19], and IIVotes [20]. In cost-sensitive-boosting ensembles, the general learning framework of AdaBoost is maintained, but a misclassification cost adjustment function is introduced into the weight updating formula. These ensembles are usually different in the modification way of the weight update rule. AdaCost [21], CSB1, CSB2 [22], RareBoost [23], AdaC1, AdaC2, and AdaC3 [24] are the most representative approaches. Unlike previous algorithms, hybrid ensemble methods adopt the double ensemble learning methods. For example, EasyEnsemble and BalanceCascade use bagging as the main ensemble learning method, but each base classifier is trained by AdaBoost. Therefore, the final classifier is an ensemble of ensembles.

Stacking algorithm is another ensemble method and has the same base classifiers to the Bagging and AdaBoost algorithms, but the structure is different. Stacking algorithm has a two-level structure consisting of Level 0 classifiers and Level 1 classifiers. Stacking algorithm involves two steps. The first step is to collect the output of each model into a new set of data. For each instance in the original training set, the data set represents every model's prediction of that instance's class and the models are base classifiers. In the second step, based on the new data and true labels of each instance in the original training set, a learning algorithm is employed to train the second-layer model. In Wolpert's terminology, the first step is referred to as the Level 0 layer and the second-stage learning algorithm is referred to as the Level 1 layer [25].

Stacking ensemble is a general method, in which a high-level model is combined with the lower-level models. Stacking ensemble can achieve the higher predictive accuracy. Chen et al. adopted ant colony optimization to configure stacking ensembles for data mining [26]. Kadkhodaei and Moghadam proposed an entropy-based approach to search for the best combination of the base classifiers in ensemble classifiers based on stack generalization [27]. Czarnowski and Jędrzejowicz focused on the machine classification with data reduction based on stacked generalization [28]. Most of the previous studies were focused on the way to use or generate stacking algorithm. However, the stacking ensemble does not consider data distribution and is suitable for the common data sets other than imbalance data.

In order to solve the imbalance problem, the paper introduces cost-sensitive learning into stacking ensemble and adds a misclassification cost adjustment function into the weight of instance and classifier. In this way, misclassification costs may be considered in the data set as a form of data space weighting to select the best distribution for training. On the other hand, in the combine stage, metatechniques can be integrated with cost-sensitive classifiers to replace standard cost-minimizing techniques. The weights for the misclassification of positive instances are higher and the weights for misclassification of negative instance are relatively lower. The method provides an option for imbalanced learning domains.

In this paper, RE-sample and Cost-Sensitive Stacked Generalization (RECSG) method is proposed in order to solve the imbalance problem. In the method, preprocessed imbalance data are used to train the Level 0 layer model. Unlike common ensemble algorithm for imbalanced data, the proposed method utilized cost-sensitive algorithm as the Level 1 (meta)layer. Stacking methods combined with imbalance data approaches including cost-sensitive learning had been reported. Kotsiantis proposed a stacking variant methodology with cost-sensitive models as base learners [29]. In the method, the model tree was replaced by MLR in metalayer to determine the class with the highest probability associated with the true class. Lo et al. proposed a cost-sensitive stacking method for audio tag annotation and retrieval [30]. In these methods, cost-sensitive learners are adopted in the base-layer model and the metalayer model is trained by other learning algorithms such as SVM and decision tree. In this paper, Level 0 model generalizer involves resampling the data and training the base classifier. The cost-sensitive algorithm is used to train the Level 1 metalayer classifier. The two layers adopt the imbalanced data algorithms and take the full advantages of mature methods. Level 1 layer model has a bias towards the performance of the minority class. Therefore, the method proposed in the study is more efficient than other methods in which cost-sensitive algorithms are only used in the Level 0 layer.

The method was compared with common classification methods, including other ensemble algorithms. Additionally, the evaluation metrics of the algorithm were analyzed based on the results of statistical tests. Statistical tests of evaluation metrics demonstrated that the proposed new approach could effectively solve the imbalanced problem.

The paper is structured as follows. Related ensemble approaches and cost-sensitive algorithms are introduced in Section 2. Section 3 introduces the details of proposed RECSG approach including Level 0 and Level 1 model generalizers. In Section 4, experiments and corresponding results and analysis are presented. Statistical tests of evaluation metrics of algorithm performance are analyzed and discussed. Finally, Section 5 discusses the advantages and disadvantages of the proposed method.

2. Background

2.1. Performance Evaluation in Imbalanced Domains. The evaluation metric is a vital factor for the classifier model and performance assessment. In Table 1, the confusion matrix demonstrates the results of incorrectly and correctly classified instances of each class in the two classes of problems.

Accuracy is the most popular evaluation metric. However, it cannot effectively measure the correct rates of all the classes, so it is not an appropriate metric for imbalance data sets. For this reason, in addition to accuracy, more suitable metrics should be considered in the imbalance

Table 1: Confusion matrix for performance evaluation.

	Positive prediction	Negative prediction
Positive class	True positive (TP)	False negative (FN)
Negative class	False positive (FP)	True negative (TN)

problem. Other metrics have been proposed to measure the classification performance independently. Based on the confusion matrix (Table 1), these metrics are defined as

$$\begin{aligned} \text{Precision} &= \frac{\text{tp}}{(\text{tp} + \text{fp})} \\ \text{Recall} &= \frac{\text{tp}}{(\text{tp} + \text{fn})} \\ F\text{-Measure} &= \frac{(1+\beta)^2 * \text{Recall} * \text{Precision}}{\beta^2 * \text{Recall} + \text{Precision}}, \end{aligned} \tag{1}$$

where β is a coefficient to adjust the relative importance of precision versus recall (usually $\beta = 1$).

$$\begin{aligned} \text{FPrate} &= \frac{\text{FP}}{(\text{FP} + \text{TN})} \\ \text{TPrate} &= \frac{\text{TP}}{(\text{TP} + \text{FN})}. \end{aligned} \tag{2}$$

The used combined evaluation metrics of these measures include receiver operating characteristic (ROC) graphic [31], the area under the ROC curve (AUC) [2], average geometric mean of sensitivity and specificity (GeoMean) [32] (see (4)), and average adjusted geometric mean (AGeoMean) [33] (see (5)), where N_n refers to the proportion of the majority samples. These metrics are defined as

$$\text{AUC} = \frac{(1 + \text{TPrate} - \text{FPrate})}{2} \tag{3}$$

$$\text{GeoMean} = \sqrt{\text{TPrate} * \text{TNrate}} \tag{4}$$

$$\text{AGeoMean} = \begin{cases} \dfrac{\text{GeoMean} + \text{TNrate} * N_n}{1 + N_n}, & \text{TPrate} > 0 \\ 0, & \text{TPrate} = 0. \end{cases} \tag{5}$$

2.2. Stacking. Bagging and AdaBoost are the most common ensemble learning algorithms. In bagging method, different base classifier models generate different classification results and the final decision is decided by majority voting. In AdaBoost algorithm, a series of base weak classifiers are trained with the whole training set and the final decision is generated by a weighted majority voting scheme. In each round of training iteration, different weights are attributed to each instance. In the two algorithms, the base classifiers are the same.

Stacking ensemble is another ensemble algorithm, in which the prediction result of base classifiers is used as the attribute to train the combination function in metalayer classifier [25]. The algorithm has a two-level structure consisting of Level 0 classifiers and Level 1 classifiers. It was proposed by Wolpert and used by Breiman [34] and Leblanc and Tibshirani [35].

Set a data set:

$$D = \{(X_n, Y_n),\ n = 1, \ldots, N\}, \tag{6}$$

where X_n is a vector representing the attribute values of the instance n and Y_n is the class value. All N instances are randomly split into j equivalent parts and j-fold cross-validation is used to train the model. The prediction results of every time testing set gives a vector which includes n instances:

$$\{y_1, y_2, \ldots, y_n\}'. \tag{7}$$

Set M_j is the difference base learning algorithm model obtained with the data set D, $j = 1, \ldots, k$, and M_j is the part of Level 0 models. Level 0 layer consists of j base classifiers, which are employed to estimate the class probability of each instance.

In M_j, j-fold cross-validation of training data set D gives the prediction result:

$$\left\{y_1^{(j)}, y_2^{(j)}, \ldots, y_n^{(j)}\right\}, \quad j = 1, \ldots, k. \tag{8}$$

All the M_j prediction results generate input space of Level 1 model and the real value of instance is treated as the output space. The model is expressed as follows:

$$\begin{array}{ll} \text{Input} & \text{output} \\ y_1^{(1)}, y_1^{(2)}, \ldots, y_1^{(i)}, \ldots, y_1^{(k)}, & y_1 \\ y_2^{(1)}, y_2^{(2)}, \ldots, y_2^{(i)}, \ldots, y_2^{(k)}, & y_2 \\ \vdots & \vdots \\ y_n^{(1)}, y_n^{(2)}, \ldots, y_n^{(i)}, \ldots, y_n^{(k)}, & y_n. \end{array} \tag{9}$$

The above intermediate data are considered as the training data of the Level 1 layer model. The input data are treated as features and the real value of instance is treated as the output space. The next step is to train the data with some fused leaning algorithms. The process is called Level 1 generalizer and Level 1 model is denoted by $\widetilde{M}$, which can be regarded as the function of $(y^{(1)}, y^{(2)}, \ldots, y^{(i)}, \ldots, y^{(k)})$.

In the stacking process, Level 0 (M_j, $j = 1, \ldots, k$) is combined with Level 1 ($\widetilde{M}$). Given a new instance x', models M_j produce a prediction result vector:

$$y^{(1)}, y^{(2)}, \ldots, y^{(i)}, \ldots, y^{(k)} \quad i = 1, \ldots, k. \tag{10}$$

Then, $\widetilde{M}$ model is used to combine base classifiers and predict the final result of x'.

In this paper, for imbalance data, we propose a stacked generalization based on cost-sensitive classification. In Level 0 model generalizer layer, we firstly resample the imbalance data. Resampling approaches include oversampling and

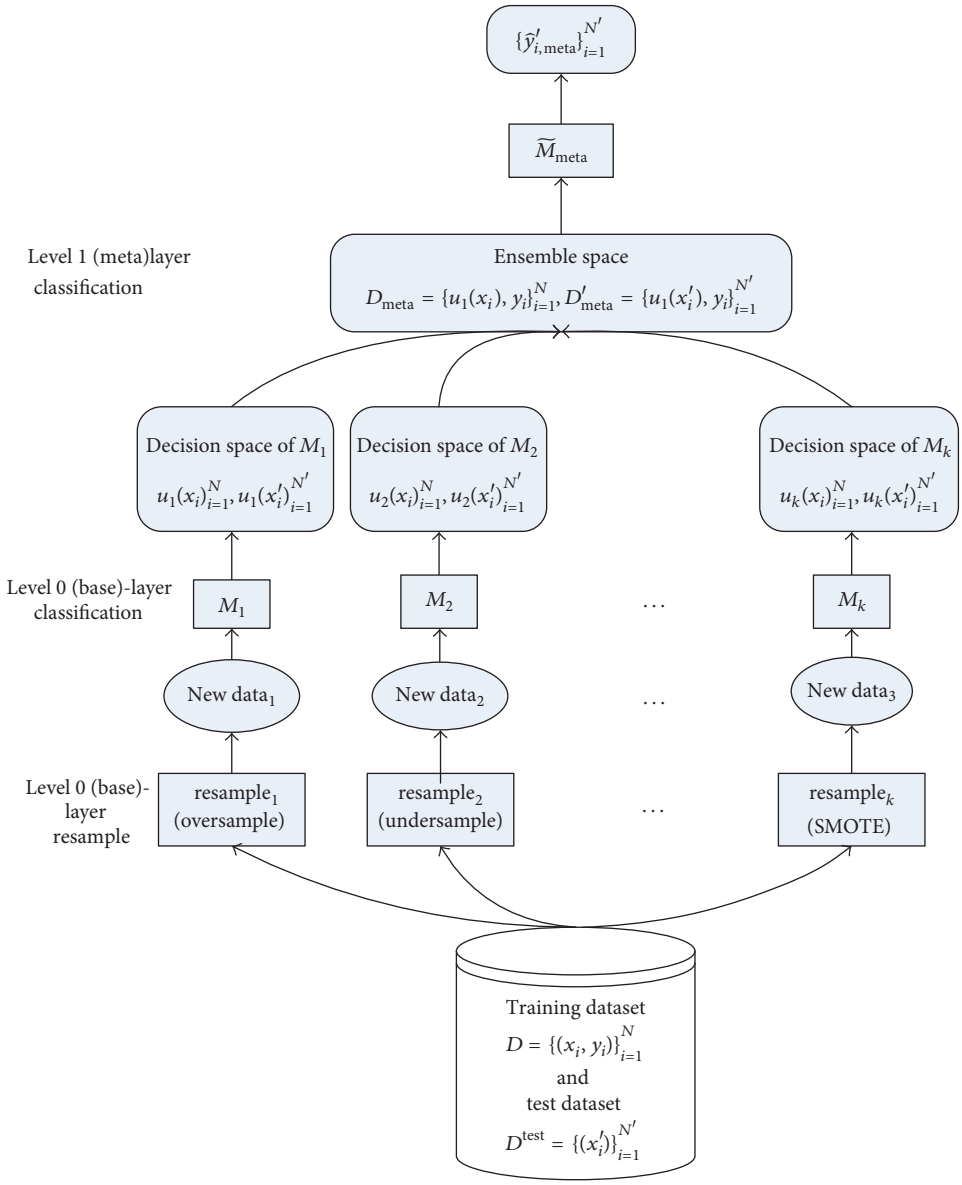

Figure 1: Flowchart of the RECSG architecture.

undersampling. Secondly, Model M_j, $j = 1, \ldots, k$, is trained by the classification with new data, and the prediction output $y_i^{(j)}$ by $j = 1, \ldots, k$ is produced with original data. In Lever 1 model generalizer layer, $\widetilde{M}$ model is generated by cost-sensitive classification based on logistic regression.

3. Stacked Generalization for the Imbalance Problem

The proposed RECSG architecture includes two layers. The first layer (Level 0) consisting of classifiers ensemble is called the base layer; the second layer (Level 1) combined with base classifiers is called the metalayer. The flowchart of the architecture is shown in Figure 1.

3.1. Level 0 Model Generalizers. For the imbalance problem, Level 0 model generalizer step of RECSG includes preprocessing data and training the base classifier. Firstly, the oversampling (SMOTE) method has been used in data preprocessing of base classifier. Secondly, the base classifier model is trained with the new data set. In this level, we employed three base algorithms: Naïve Bayes (NB) [36], decision tree C4.5 [37], and k-nearest neighbors (k-NN) [38].

(1) Naïve Bayes (NB). Given instance x, set $P(i \mid x)$ is the posterior probability of class i, then

$$P_i(x) = \frac{P(i \mid x)}{\Sigma_i P(i \mid x)}, \tag{11}$$

where NB uses an Laplacian estimate for estimating the conditional probabilities.

Table 2: Cost matrix.

	Actual negative	Actual positive
Predict negative	$C(0,0) = c_{00}$	$C(0,1) = c_{01}$
Predict positive	$C(1,0) = c_{10}$	$C(1,1) = c_{11}$

Table 3: Cost matrix of credit data sets.

	Actual bad	Actual good
Predict bad	0	1
Predict good	5	0

(2) Decision Tree (C4.5). Decision tree classification is a method usually used in data mining [39]. A decision tree is a tree, where each input feature labels a nonleaf node, one class, or probability over the class labels each leaf node of the tree. By recursive partitioning, a tree can be "learned" until no prediction value is added or the subset has the same value. When the parameter of training data selects information entropy, the decision tree is named C4.5.

(3) k-NN. k-NN is a nonparametric algorithm used for regression or classification. An instance is classified by a majority vote of its k-nearest neighbors. If $k = 1$, the instance is simply classified in accordance with the class label of the nearest neighbor.

All the above algorithms are simple and have low complexity, so they are applicable weak base classifiers.

3.2. Level 1 Model Generalizers. Prediction results of several base classifiers in Level 0 layer are used as input space and true value class of instance is used as output space. Based on Level 0 layer, Level 1 layer model is trained by other learning algorithms. For imbalanced data, the cost-sensitive algorithm is used to train the Level 1 metalayer classifier in the paper.

3.2.1. Cost-Sensitive Classifier. Aiming at imbalanced data learning problem, in cost-sensitive classifier, different cost matrices are used as the penalty of misclassified instance [40]. For example, a cost matrix C has the following structure in a binary classification scenario in Table 2.

In the cost matrix, the row indicated alternative predicted classer, whereas the column indicates actual classes. The cost of false negative is notated as c_{01} and the cost of false positive is notated as c_{10}. Conceptually, the cost of correctly classified instances should always be less than the cost of incorrectly classified instances. In the imbalance problem, c_{10} is always greater than c_{01}. For the German credit data set previously reported as the part of the Stalog project [41], the cost matrix is provided in Table 3.

The cost of false good prediction is greater than the cost of false bad prediction in the view of economical reason. So, in Level 1 classifier of stacking for the imbalance problem, the cost-sensitive algorithm is adopted.

3.2.2. Logistic Regression. Ting and Witten illustrated that MLR (Multiresponse Linear Regression) algorithm had an advantage over other Level 1 generalizers in stacking [42]. In this paper, the logistic regression classifier is used as the metalayer algorithm. Based on the metalayer, the cost of misclassification is considered in the cost-sensitive algorithm. In logistic regression classifier, the prediction result of Level 0 layer is used as the attribute of Level 1 metalayer and the real value of instance is used as output space. The linear regression for class i is simply obtained as

$$\mathrm{LR}_i(x) = \sum_{j=1}^{K} \alpha_{ji} P_{ji}(x) \quad j = 1, \ldots, k. \tag{12}$$

Details of the implementation of RECSG are presented below.

3.3. Algorithm Pseudocode. Pseudocode 1 presents the proposed RECSG approach. **Input** parameters are two data sets: training set and testing set. **Output** predicts class labels of the test samples. The first step in the process is data preprocessing, resample new instances, and train model M_j ($j = 1, \ldots, k$), where j is the number of base classifiers and $u_j(x_i)$ is the generated function of x_i based on model M_j (lines (1)-(2) in Pseudocode 1).

Then, the metalayer model $\widetilde{M}$ is constructed based on the data D_{meta}, which is firstly predicted with Level 0 layer (line (3) in Pseudocode 1). Finally, Lever 1 layer classification (cost-sensitive and logistic regression) is used to predict the ultimate value of the tested samples.

4. Empirical Investigation

The experiment aims to verify whether the RECSG approach can improve the classification performance for the imbalance problem. In this paper, the RECSG approach was compared with other algorithms involving various combination ensemble techniques and imbalanced approaches. For each method, the same training set, testing set, and validation set were used.

The experimental system has been implemented and is composed of 7 learning methods implemented in Weka [43], namely, Naïve Bayes[(1)] (NB), C4.5(j48)[(2)], k-nearest neighbor (k-NN)[(3)], cost-sensitive[(4)], AdaBoost[(5)], Bagging[(6)], and stacking[(7)]. The brief description, the standard version, and parameters of these methods are shown in Table 4.

The RECSG approach includes 2 layers. Level 1 layer (metalearning) system consists of 2 learning methods implemented in Weka, namely, simple logistic regression[(9)] (MLP), and cost-sensitive classifier[(4)]. MLP is the base classification algorithm of cost-sensitive classifier. Level 0 layer (base - learning) of stacking is composed of 3 classifiers: Naïve Bayes[(1)], C4.5[(2)], and k-NN[(3)]. Evaluation metrics of algorithm performance include AUC, GeoMean, and AGeoMean.

4.1. Experimental Settings. Experiments were implemented with 17 data sets from the UCI Machine Learning Repository [44]. These data sets cover various fields and are based on IR measure values (from 0.54 to 0.014), unique data set names, a varying number of samples (from 173 to 2338),

Input Training set $D = \{(x_i, y_i)\}_{i=1}^{N}$, Test dataset $D^{\text{test}} = \{(x_i')\}_{i=1}^{N'}$
Output Predict class labels of the test samples $\{\widehat{y}'_{i,\text{meta}}\}_{i=1}^{N'}$
For each $j = 1, 2, \ldots, K$ do
(1) Resample imbalance data and generate j–fold cross-validation sets to obtain New data$_j$;
(2) Train and compute $u_j(x_i)_{i=1}^{N}$ and $u_j(x_i')_{i=1}^{N'}$ in Level-0 (base)-layer classifier M_j
end
(3) Construct $D_{\text{meta}} = \{u_1(x_i), y_i\}_{i=1}^{N'}$, and $D'_{\text{meta}} = \{u_1(x_i'), y_i\}_{i=1}^{N'}$
(4) Based on the data D_{meta}, classification (cost-sensitive and Logistic Regression) is used to generate Level-1 (meta)-layer model $\widetilde{M}$, through $\widetilde{M}$ with D'_{meta} to predict $\{\widehat{y}'_{i,\text{meta}}\}_{i=1}^{N'}$

Pseudocode 1: Pseudocodes of RE-sample and Cost-Sensitive Stacked Generalization.

Table 4: Base classifier methods and corresponding ensemble algorithms.

Methods	Descriptions and parameters	Abbreviations
(1) Naïve Bayes	weka.classifiers.bayes.NaiveBayes	NB
(2) C4.5	weka.classifiers.trees.J48 -C 0.25 -M 2	C4.5
(3) k-nearest neighbor	weka.classifiers.lazy.IBk -K 1 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""	KNN
(4) Cost-sensitive with Naïve Bayes	weka.classifiers.meta.AdaBoostM1 -P 100 -S 1 -I 10 -W weka.classifiers.bayes.NaiveBayes	Cost-NB
(5) AdaBoost with Naïve Bayes	weka.classifiers.meta.AdaBoostM1 -P 100 -S 1 -I 10 -W weka.classifiers.bayes.NaiveBayes	Ada-NB
(6) AdaBoost with cost-sensitive and Naïve Bayes	weka.classifiers.meta.AdaBoostM1 -P 100 -S 1-I10 -W weka.Classifiers.meta.CostSensitiveClassifier -- -cost-matrix "[1.0 3.0; 1.0 1.0]" -S 1 -W weka. classifiers.bayes.NaiveBayes	Ada-Cost-NB
(7) Bagging with Naïve Bayes	weka.classifiers.meta.Bagging -P 100 -S 1 -num-slots 1 -I 10 -W weka.classifiers.bayes.NaiveBayes	Bag-NB
(8) Bagging with cost-sensitive and Naïve Bayes	weka.classifiers.meta.Bagging -P 100 -S 1 -num-slots 1 -I 10 -W weka.classifiers.meta.CostSensitive Classifier -- -cost-matrix "[1.0 3.0; 1.0 1.0]" -S 1 -W weka.classifiers.bayes.NaiveBayes	Bag-Cost-NB
(9) Stacking with NB, KNN, C4.5, and logistic	weka.classifiers.meta.Stacking -X 10 -M "weka. classifiers.functions.Logistic -R 1.0E-8 -M -1 –num –decimal-places 4" -S 1 -num-slots 1 -B "weka. classifiers.bayes.NaiveBayes" -B "weka.classifiers. lazy.IBk -K 1 -W 0 -A \"weka.core.neighboursearch. LinearNNSearch -A \ \ \"weka.core.EuclideanDistance -R first-last\ \ \"\"" -B "weka.classifiers.trees.J48 -C 0.25 -M 2"	Stacking-log
(10) Stacking cost-sensitive with NB, KNN, C4.5, and logistic	weka.classifiers.meta.Stacking -X 10 -M "weka. classifiers.meta.CostSensitiveClassifier –cost -matrix \"[1.0 3.0; 1.0 1.0]\" -S 1 -W weka.classifiers. functions.Logistic -- -R 1.0E-8 -M -1 –num –decimal –places 4" -S 1 -num-slots 1 -B "weka. classifiers. bayes.NaiveBayes " -B "weka.classifiers.lazy.IBk -K 1 -W 0 -A \"weka.core.neighboursearch.LinearNN Search -A \ \ \"weka.core.EuclideanDistance -R first-last\ \ \"\"" -B "weka.classifiers.trees.J48 -C 0.25 -M 2"	Stacking-Cost-log

Table 5: Summary of imbalanced data sets.

Data sets	#sam.	(#min., #maj.)	IR	#fea.
wisconsin	683	(239, 444)	0.54	9
pima	768	(268, 500)	0.53	8
haberman	306	(81, 225)	0.36	3
Vehicle1	846	(217, 629)	0.345	18
Vehicle0	846	(199, 647)	0.307	18
segment0	2308	(329, 1979)	0.17	19
yeast-0-3-5-9_vs_7-8	506	(50, 456)	0.11	8
yeast-0-2-5-6_vs_3-7-8-9	1004	(99, 905)	0.11	8
vowel0	988	(90, 898)	0.10	13
led7digit-0-2-4-5-6-7-8-9_vs_1	443	(37, 406)	0.09	7
Glass2	214	(17, 197)	0.08	10
cleveland-0_vs_4	173	(13, 160)	0.08	13
glass-0-1-6_vs_5	184	(9, 175)	0.05	9
car-good	1728	(69, 1659)	0.04	6
flare-F	1066	(43, 1023)	0.04	11
car-vgood	1728	(65, 1663)	0.039	6
abalone-17_vs_7-8-9-10	2338	(58, 2280)	0.0254	9

and variations in the amount of class overlap (see the KEEL repository [45]). Multiclass data sets were modified to obtain two-class imbalanced problems so that the union of one or more classes became the positive class and the union of one or more of the remaining classes was labeled as the negative class. A brief description of the used date set is presented in Table 5. It includes the total number of instances (#Sam.), the total number of each class instances (#Min., #Maj.), the imbalance ratio (IR = the ratio of the number of minority class instance to majority class instance), and the number of features (#Fea.).

Our system was compared with other ensemble algorithms including AdaBoost with Naïve Bayes, AdaBoost with cost-sensitive, bagging with Naïve Bayes, bagging with cost-sensitive, and stacking cost-sensitive with NB, k-NN, C4.5, and logistic regression. All the experiments were performed by 10-fold cross-validation.

4.2. Experimental Results. Tables 6, 7, and 8, respectively, show the results of the three metrics (AUC, GeoMean, and AGeoMean) for the algorithms in Table 4 obtained with the data set in Table 5. The best results are emphasized in boldface on each data set in these tables.

The results showed that the performance of the proposed RECSG method was the best for 12 of 17 data sets in terms of GeoMean and AGeoMean and for 10 of 17 data sets in terms of AUC. Some methods are better than others in some evaluation metrics, but not in all metric and most data sets.

In Table 6 (AUC), the performance of the RECSG method is better than that of the 3 single-base classification methods for 14 of 17 data sets and better than that of other 5 ensemble algorithms for 13 of 17 data sets. In Table 7 (GeoMean), the RECSG method outperforms all the 3 single-base classification methods in 15 of 17 data sets and outperforms other 5 ensemble algorithms in 15 of 17 data sets. In Table 8 (AGeoMean), the RECSG method outperforms all the 3 single-base classification methods in 14 of 17 data sets and outperforms other 5 ensemble algorithms in 16 of 17 data sets. For the 17 data sets, the evaluation metrics of GeoMean and AGeoMean are better than AUC.

4.3. Statistical Tests. Statistical test [46] is adopted to compare different algorithms. In the paper, we use two types of comparisons: pairwise comparison (between a pair of algorithms) and multiple comparisons (among a group of algorithms).

4.3.1. Pair of Algorithms. We performed statistic T-tests to explore whether RECSG approach is the significantly better than other algorithms in the three metrics (AUC, GeoMean, and AGeoMean). Table 9 shows the results of RECSG approach compared with the other methods in terms of AUC, GeoMean, and AGeoMean. The values in the square brackets indicate the number of the metrics with statistically significant difference in the T-test performed with the confidence level $\alpha = 0.05$ in the three evaluation metrics. As for the evaluation metric of AUC, the RECSG outperformed NB for 11 of 17 data sets and 5 data sets showed the statistically significant difference with the confidence level of $\alpha = 0.05$.

4.3.2. Multiple Comparisons. We performed Holm post hoc test [47] and selected multiple groups to test the three metrics (AUC, GeoMean, and AGeoMean). The post hoc procedures can determine whether a comparison hypothesis should be rejected at a specified confidence level α. Statistical experiment was performed in the platform on the website http://tec.citius.usc.es/stac/. Table 10 provides the results that RECSG approach is significantly different with the confidence level of $\alpha = 0.05$ in terms of AUC, GeoMean, and AGeoMean. H_0 indicates that there is no significant difference. According

Table 6: Results of other methods in terms of AUC.

Data set	NB	C4.5	KNN	Cost-NB	Ada-NB	Ada-Cost-NB	Bag- NB	Bag-Cost- NB	Stacking-log	Stacking-Cost-log
wisconsin	**0.986**	0.957	0.953	0.989	0.981	0.982	0.99	0.989	0.982	0.982
pima	0.819	0.751	0.662	0.819	0.804	0.808	0.817	0.819	0.819	**0.819**
haberman	0.64	0.615	0.53	0.64	0.667	0.625	0.638	**0.647**	0.636	0.63
Vehicle1	0.711	0.722	0.671	0.711	0.769	0.723	0.715	0.712	0.782	**0.782**
Vehicle0	0.814	0.93	0.92	0.814	0.784	0.781	0.808	0.808	0.977	**0.977**
segment0	0.987	0.971	0.996	0.987	0.944	0.959	0.987	0.986	0.999	**0.999**
yeast-0-3-5-9_vs_7-8	0.75	0.675	0.653	0.75	0.751	0.686	0.75	0.754	0.769	**0.773**
yeast-0-2-5-6_vs_3-7-8-9	0.832	0.687	0.782	0.832	0.79	0.808	0.84	0.842	**0.847**	0.82
vowel0	0.98	0.974	1	0.98	0.996	0.994	0.981	0.98	1	**1**
led7digit-0-2-4-5-6-7-8-9_vs_1	**0.954**	0.925	0.827	0.954	0.898	0.893	0.953	0.949	0.909	0.904
Glass2	0.716	**0.812**	0.587	0.716	0.737	0.735	0.748	0.737	0.713	0.721
cleveland-0_vs_4	0.909	0.824	0.754	0.909	0.826	0.891	0.915	0.913	0.979	**0.976**
glass-0-1-6_vs_5	0.934	0.989	0.825	0.934	0.975	0.905	0.983	0.985	0.9	**0.99**
car-good	0.976	0.493	0.648	0.976	0.968	0.971	0.976	0.976	0.653	**0.977**
flare-F	0.908	0.525	0.59	0.911	0.876	0.893	0.908	**0.912**	0.825	0.806
car-vgood	0.997	0.992	0.69	0.997	0.997	0.997	0.997	0.997	0.998	**0.998**
abalone-17_vs_7-8-9-10	0.775	0.613	0.624	0.775	**0.817**	0.73	0.773	0.774	0.752	0.726
Mean	0.864	0.791	0.748	0.864	0.858	0.846	0.869	0.869	0.855	**0.875**

Table 7: Results of other methods in terms of GeoMean.

Data set	NB	C4.5	KNN	Cost-NB	Ada-NB	Ada-Cost-NB	Bag- NB	Bag-Cost- NB	Stacking-log	Stacking-Cost-log
wisconsin	0.965	0.954	0.953	0.968	0.959	0.967	0.968	0.966	0.964	**0.970**
pima	0.719	0.697	0.649	0.736	0.715	0.739	0.723	0.731	0.703	**0.741**
haberman	0.419	0.566	0.455	0.586	0.466	0.573	0.434	0.594	0.361	**0.605**
Vehicle1	0.677	0.622	0.652	0.674	0.607	0.677	0.673	0.661	0.561	**0.712**
Vehicle0	0.724	0.912	0.919	0.726	0.735	0.733	0.719	0.733	0.894	**0.946**
segment0	0.897	0.977	0.995	0.892	0.897	0.906	0.902	0.895	**0.994**	0.993
yeast-0-3-5-9_vs_7-8	0.467	0.463	0.609	0.464	0.466	0.550	0.445	0.485	0.444	**0.603**
yeast-0-2-5-6_vs_3-7-8-9	0.762	0.590	0.762	0.793	0.762	0.792	0.762	0.787	0.706	**0.798**
vowel0	0.920	0.969	1.000	0.913	0.944	0.929	0.921	0.914	1.000	**1.000**
led7digit-0-2-4-5-6-7-8-9_vs_1	0.853	0.874	0.818	0.869	0.863	0.870	0.845	0.850	0.846	**0.874**
Glass2	0.578	0.472	0.470	0.564	0.578	**0.610**	0.586	0.568	0.000	0.607
cleveland-0_vs_4	**0.899**	0.724	0.722	0.936	0.611	0.863	0.866	0.861	0.866	0.866
glass-0-1-6_vs_5	**0.932**	0.877	0.810	0.932	0.932	0.872	0.872	0.869	0.814	0.877
car-good	0.000	0.000	0.558	0.794	0.520	0.783	0.000	0.778	0.561	**0.813**
flare-F	0.751	0.000	0.450	**0.781**	0.541	0.647	0.738	0.781	0.000	0.774
car-vgood	0.496	0.888	0.626	0.986	0.917	0.925	0.481	0.971	0.957	**0.988**
abalone-17_vs_7-8-9-10	0.628	0.321	0.506	0.640	0.319	0.368	0.627	0.630	0.293	**0.664**
Mean	0.687	0.641	0.703	0.779	0.696	0.753	0.680	0.769	0.645	**0.814**

Table 8: Results of other methods in terms of AGeoMean.

Data set	NB	C4.5	KNN	Cost-NB	Ada-NB	Ada-Cost-NB	Bag- NB	Bag-Cost- NB	Stacking-log	Stacking-Cost-log
wisconsin	0.961	0.960	0.961	0.962	0.963	0.962	0.962	0.960	0.967	**0.967**
pima	**0.768**	0.743	0.706	0.734	0.767	0.736	0.775	0.729	0.770	0.726
haberman	0.642	0.680	0.601	**0.718**	0.660	0.694	0.653	0.721	0.613	0.663
Vehicle1	0.690	0.721	0.728	0.658	0.722	0.669	0.690	0.654	0.709	**0.740**
Vehicle0	0.666	0.929	0.934	0.658	0.681	0.663	0.664	0.663	0.924	**0.936**
segment0	0.859	0.986	0.996	0.850	0.859	0.871	0.863	0.854	0.995	**0.995**
yeast-0-3-5-9_vs_7-8	0.715	0.706	**0.782**	0.709	0.713	0.737	0.703	0.721	0.700	0.770
yeast-0-2-5-6_vs_3-7-8-9	0.856	0.778	0.855	0.857	0.856	0.855	0.856	0.854	0.839	**0.866**
vowel0	0.929	0.981	1.000	0.909	0.966	0.948	0.931	0.910	1.000	**1.000**
led7digit-0-2-4-5-6-7-8-9_vs_1	0.853	0.874	0.818	0.869	0.863	0.870	0.845	0.850	0.846	**0.874**
Glass2	0.511	0.701	0.695	0.493	0.511	0.551	0.539	0.515	0.000	**0.744**
cleveland-0_vs_4	**0.927**	0.845	0.841	0.943	0.783	0.914	0.918	0.910	0.918	0.918
glass-0-1-6_vs_5	**0.954**	0.932	0.894	0.954	0.954	0.923	0.923	0.919	0.902	0.932
car-good	0.000	0.000	0.763	0.878	0.747	0.870	0.000	0.873	0.769	**0.890**
flare-F	0.840	0.000	0.705	0.842	0.751	0.795	0.836	0.842	0.000	**0.845**
car-vgood	0.743	0.934	0.793	0.986	0.953	0.956	0.725	0.980	0.974	**0.990**
abalone-17_vs_7-8-9-10	0.736	0.655	0.745	0.716	0.650	0.672	0.734	0.711	0.641	**0.804**
Mean	0.744	0.731	0.813	0.808	0.788	0.805	0.742	0.8039	0.739	**0.862**

Table 9: Comparison between RECSG and other algorithms in terms of wins/losses of the results.

	NB	C4.5	KNN	Cost-NB	Ada-NB	Ada-Cost-NB	Bag-NB	Bag-Cost-NB	Stacking-log
Stacking-Cost-log (AUC)	11 [5]/6 [3]	15 [11]/2 [2]	17 [15]/0 [2]	11 [6]/6 [5]	13 [7]/4 [2]	14 [9]/3 [1]	10 [6]/7 [5]	10 [6]/7 [5]	11 [3]/6 [2]
Stacking-Cost-log (GeoMean)	15 [13]/2 [2]	17 [13]/0 [0]	15 [13]/2 [1]	14 [8]/3 [2]	16 [12]/1 [1]	16 [10]/1 [0]	17 [13]/0 [0]	16 [10]/3 [0]	16 [12]/1 [0]
Stacking-Cost-log (AGeoMean)	14 [10]/3 [1]	15 [9]/2 [2]	15 [12]/2 [1]	13 [7]/4 [3]	15 [11]/2 [2]	15 [10]/2 [2]	16 [10]/1 [1]	15 [8]/2 [1]	16 [10]/1 [1]

Table 10: Holm post hoc test to show differences of Stacking-Cost-log (AUC) and other algorithms.

	Rejected (H_0)	Accepted (H_0)
Stacking-Cost-log (AUC) versus other algorithms	3	6
Stacking-Cost-log (GeoMean) versus other algorithms	8	1
Stacking-Cost-log (AGeoMean) versus other algorithms	9	0

to the evaluation metric of GeoMean, the RECSG is significantly better than other 8 methods with the confidence level of $\alpha = 0.05$.

4.4. Discussion. According to the experiments performed with 17 different imbalance data sets and the comparison with other 9 classification methods, the proposed RECSG method showed the higher performance than other methods in terms of AUC, GeoMean, and AGeoMean, as illustrated in Tables 7, 8, and 9. The RECSG method showed the better performance for 10 of 17 cases in terms of AUC and for 12 of 17 cases in terms of GeoMean and AGeoMean. The method outperformed other 5 ensemble algorithms for 16 of 17 data sets in terms of GeoMean and AGeoMean and for 13 of 17 data sets in terms of AUC. GeoMean and AGeoMean are better than AUC in evaluation metrics in 17 data sets. The means of RECSG method in terms of GeoMean, AGeoMean, and AUC are all higher than other methods. The Holm post hoc test shows that RECSG method significantly outperforms the rest with the confidence level $\alpha = 0.05$ in terms of AGeoMean, for 8 of 9 methods in terms of AGeoMean, and for 3 of 9 methods in terms of AUC.

Experimental results and statistical tests show that the RECSG approach has improved the classification performance of imbalanced data sets. The reasons can be explained as follows.

Firstly, stacking algorithm uses $\widetilde{M}$ to combine the result of base classifier, whereas bagging employs majority vote. Bagging is only a simple decision combination method which requires neither cross-validation nor Level 1 learning. In this paper, stacked generalization $\widetilde{M}$ adopts logistic regression, thus providing the simplest linear combination of pooling the Level 0 models' confidence.

Secondly, cost-sensitive algorithm can affect imbalance data sets in two aspects. Firstly, cost-sensitive modifications can be applied in the probabilistic estimation. Moreover, cost-sensitive factors change the weight of instance, and the weight of the minority class is higher than that of the majority class. Therefore, the method directly changes the number of common instances without discarding or duplicating any of the rare instances.

Thirdly, RECSG method introduces cost-sensitive learning into stacking ensemble. The cost-sensitive function in $\widetilde{M}$ replaces the error-minimizing function, thus allowing Level 1 learning to be prone to focus on the minority class.

The results in Tables 7 and 8 demonstrate that the RECSG method has the higher performance when the evaluation performance metric of base classifier is weaker (such as Vehicle1, Vehicle0, and car-vgood). The reason is that the alternative methods of NB, C4.5, and k-NN have shortcomings. The independence assumption hampers the performance of NB in some data sets. In C4.5 tree construction, the selection of attributes affects the model performances. Error ratio of k-NN is relatively high when data sets are in imbalance. Therefore, logistic regression adopted in $\widetilde{M}$ can improve the performance when base classifier is weaker.

The performance of the RECSG method is generally better when the IR is low (such as Glass2, car-good, flare-F, car-vgood, and abalone-17_vs_7-8-9-10). The performance of RECSG method is probably related to the setting of the cost matrix. Different data sets should use different cost matrices. For the purpose of simplicity in the paper, we have adopted the same cost matrices, which may be more suitable to low IR values.

5. Conclusions

In this paper, in order to solve the class imbalance problem, we proposed the RECSG method based on 2-layer learning models. Experimental results and statistical tests showed that the RECSG approach improved the classification performance. The proposed RECSG approach may have the relatively high computational complexity in the training stage because the approach involves 2-layer classifier models

which consist of several base classifiers and metaclassifiers. The number and kinds of the base-level classifiers are closely related to the performance of stacking algorithm. In the fusion stage of base classifiers, the selection of the metaclassifier is also important. In this paper, in order to validate the performance improvement compared with other current classification algorithms, we only randomly selected 3 classification algorithms (NB, k-NN, and C4.5) as base classifier and cost-sensitive algorithm as metaclassifier in the fuse stage. Selection of the number or kind of the base classifiers and metaclassifier was not discussed. Therefore, we will explore the diversity and quality of base classifier in the future. The adoption of these strategies would improve the prediction performance and reduce the training time of stacking algorithm for imbalance problems.

Conflicts of Interest

The author declares that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research is supported by the Natural Science Foundation of Shanxi Province under Grant no. 2015011039.

References

[1] O. Loyola-González, J. F. Martínez-Trinidad, J. A. Carrasco-Ochoa, and M. García-Borroto, "Effect of class imbalance on quality measures for contrast patterns: an experimental study," *Information Sciences*, vol. 374, pp. 179–192, 2016.

[2] C. Beyan and R. Fisher, "Classifying imbalanced data sets using similarity based hierarchical decomposition," *Pattern Recognition*, vol. 48, no. 5, pp. 1653–1672, 2015.

[3] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[4] V. C. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.

[5] X.-Y. Liu, J. X. Wu, and Z.-H. Zhou, "Exploratory under-sampling for class-imbalance learning," in *Proceedings of the 6th International Conference on Data Mining (ICDM '06)*, pp. 965–969, IEEE, Hong Kong, December 2006.

[6] N. Japkowicz, "The class imbalance problem: significance and strategies," in *Proceedings of the International Conference on Artificial Intelligence*, 2000.

[7] B. Zadrozny and C. Elkan, "Learning and making decisions when costs and probabilities are both unknown," in *Proceedings of the the seventh ACM SIGKDD international conference*, pp. 204–213, San Francisco, Calif, USA, August 2001.

[8] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, "A review on ensembles for the class imbalance problem: bagging, boosting, and hybrid-based approaches," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 42, no. 4, pp. 463–484, 2012.

[9] N. V. Chawla, D. A. Cieslak, L. O. Hall, and A. Joshi, "Automatically countering imbalance and its empirical relationship to cost," *Data Mining and Knowledge Discovery*, vol. 17, no. 2, pp. 225–252, 2008.

[10] A. Freitas, A. Costapereira, and P. Brazdil, "Cost-Sensitive Decision Trees Applied to Medical Data," in *International Conference on Data Warehousing and Knowledge Discovery*, pp. 303–312, 2007.

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

[12] R. E. Schapire, "The strength of weak learnability," *Machine Learning*, vol. 5, no. 2, pp. 197–227, 1990.

[13] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, part 2, pp. 119–139, 1997.

[14] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, "SMOTEBoost: improving prediction of the minority class in boosting," *in Proceeding of Knowledge Discovery in Databases*, vol. 2838, pp. 107–119.

[15] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "RUSBoost: A hybrid approach to alleviating class imbalance," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 40, no. 1, pp. 185–197, 2010.

[16] S. Hu, Y. Liang, L. Ma, and Y. He, "MSMOTE: Improving classification performance when training data is imbalanced," in *Proceedings of the 2nd International Workshop on Computer Science and Engineering, WCSE 2009*, pp. 13–17, China, October 2009.

[17] H. Guo and H. L. Viktor, "Learning from imbalanced data sets with boosting and data generation," *ACM SIGKDD Explorations Newsletter*, vol. 6, no. 1, pp. 30–39, 2004.

[18] S. Wang and X. Yao, "Diversity analysis on imbalanced data sets by using ensemble models," in *Proceedings of the 2009 IEEE Symposium on Computational Intelligence and Data Mining, CIDM 2009*, pp. 324–331, USA, April 2009.

[19] R. Barandela, J. S. S'anchez, and R. M. Valdovinos, "New applications of ensembles of classifiers," *PAA. Pattern Analysis and Applications*, vol. 6, no. 3, pp. 245–256, 2003.

[20] J. Błaszczyński, M. Deckert, J. Stefanowski, and S. Wilk, "Integrating selective pre-processing of imbalanced data with Ivotes ensemble," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface*, vol. 6086, pp. 148–157, 2010.

[21] W. Fan, S. J. Stolfo, J. Zhang, and P. K. Chan, "Adacost: Misclassication cost-sensitive boosting," in *the 6th Int. Conf. Mach. Learning*, pp. 97–105, San Francisco, CA, USA, 1999.

[22] K. M. Ting, "A comparative study of cost-sensitive boosting algorithms," in *Proc. 17th Int. Conf. Mach. Learning*, pp. 983–990, Stanford, CA, USA, 2000.

[23] M. Joshi, V. Kumar, and R. Agarwal, "Evaluating boosting algorithms to classify rare classes: comparison and improvements," in *Proceedings of the 2001 IEEE International Conference on Data Mining*, pp. 257–264, San Jose, CA, USA.

[24] Y. Sun, M. S. Kamel, A. K. C. Wong, and Y. Wang, "Cost-sensitive boosting for classification of imbalanced data," *Pattern Recognition*, vol. 40, no. 12, pp. 3358–3378, 2007.

[25] D. H. Wolpert, "Stacked generalization," *Neural Networks*, vol. 5, no. 2, pp. 241–259, 1992.

[26] Y. Chen, M. L. Wong, and H. Li, "Applying Ant Colony Optimization to configuring stacking ensembles for data mining," *Expert Systems with Applications*, vol. 41, no. 6, pp. 2688–2702, 2014.

[27] H. Kadkhodaei and A. M. E. Moghadam, "An entropy based approach to find the best combination of the base classifiers in ensemble classifiers based on stack generalization," in *Proceedings of the 4th International Conference on Control, Instrumentation, and Automation, ICCIA 2016*, pp. 425–429, Iran, January 2016.

[28] I. Czarnowski and P. Jędrzejowicz, "An approach to machine classification based on stacked generalization and instance selection," in *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, SMC 2016*, pp. 4836–4841, Hungary, 2017.

[29] S. Kotsiantis, "Stacking cost sensitive models," in *Proceedings of the 12th Pan-Hellenic Conference on Informatics, PCI 2008*, pp. 217–221, Greece, August 2008.

[30] H. Y. Lo, J. C. Wang, H. M. Wang, and S. D. Lin, "Cost-sensitive stacking for audio tag annotation and retrieval," in *Proceedings of the 36th IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2011*, pp. 2308–2311, Czech Republic, May 2011.

[31] J. Huang and C. X. Ling, "Using AUC and accuracy in evaluating learning algorithms," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 3, pp. 299–310, 2005.

[32] M. Kubat and S. Matwin, "Addressing the curse of imbalanced training sets: one sided selection in," in *Proceedings of the 14th International Conference on Machine Learning (ICML'97*, pp. 179–186, 1997.

[33] R. Batuwita and V. Palade, "Adjusted geometric-mean: A novel performance measure for imbalanced bioinformatics datasets learning," *Journal of Bioinformatics and Computational Biology*, vol. 10, no. 4, Article ID 1250003, 2012.

[34] L. Breiman, "Stacked regressions," *Machine Learning*, vol. 24, no. 1, pp. 49–64, 1996.

[35] M. Leblanc and R. Tibshirani, "Combining estimates in regression and classification," *Journal of the American Statistical Association*, vol. 91, no. 436, pp. 1641–1650, 1996.

[36] B. Cestnik, "Estimating Probabilities: A Crucial Task in Machine Learning," in *Proceedings of the European Conference on Artificial Intelligence*, pp. 147–149, 1990.

[37] J. R. Quinlan, *C4.5: Program for machine learning*, Morgan Kaufmann, 1993.

[38] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.

[39] J. R. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, no. 1, pp. 81–106, 1986.

[40] C. Elkan, "The foundations of cost-sensitive learning," in *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI '01)*, pp. 973–978, New York, NY, USA, August 2001.

[41] D. Michie, D. Spiegel halter J, C. Taylor et al., *Machine learning, neural and statistical classification*, Ellis Horwood, neural and statistical classification, 1995.

[42] K. M. Ting and I. F. Witten, "Issues in stacked generalization," *Artif. Intell. Res*, vol. 10, no. 1, pp. 271–289, 1999.

[43] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: an update," *ACM SIGKDD Explorations Newsletter*, vol. 11, no. 1, pp. 10–18, 2009.

[44] A. Frank and A. Asuncion, *UCI Machine Learning Repository*, University of California, School of Information and Computer Science, Irvine, CA, USA, http://archive.ics.uci.edu/ml.

[45] J. Alcalá-Fdez, A. Fernández, J. Luengo et al., "KEEL data-mining software tool: data set repository, integration of algorithms and experimental analysis framework," *Journal of Multiple-Valued Logic and Soft Computing*, vol. 17, no. 2-3, pp. 255–287, 2011.

[46] S. García, A. Fernández, J. Luengo, and F. Herrera, "A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability," *Soft Computing*, vol. 13, no. 10, pp. 959–977, 2009.

[47] S. Holm, "A simple sequentially rejective multiple test procedure," *Scandinavian Journal of Statistics*, vol. 6, no. 2, pp. 65–70, 1979.