Computational Statistics and Data Analysis 111 (2017) 1–13

Contents lists available at ScienceDirect

Computational Statistics and Data Analysis

journal homepage: www.elsevier.com/locate/csda

RHSBoost: Improving classification performance in imbalance data

Joonho Gong[a], Hyunjoong Kim[b,*]

[a] Department of Statistics, North Carolina State University, Raleigh, NC 27695, USA

[b] Department of Applied Statistics, Yonsei University, Seoul 03722, South Korea

ARTICLE INFO

Article history:
Received 25 November 2015
Received in revised form 24 January 2017
Accepted 24 January 2017
Available online 3 February 2017

Keywords:
Imbalanced data
AdaBoost
Ensemble
AUC
Undersampling
RHSBoost

ABSTRACT

Imbalance data are defined as a dataset whose proportion of classes is severely skewed. Classification performance of existing models tends to deteriorate due to class distribution imbalance. In addition, over-representation by majority classes prevents a classifier from paying attention to minority classes, which are generally more interesting. An effective ensemble classification method called RHSBoost has been proposed to address the imbalance classification problem. This classification rule uses random undersampling and ROSE sampling under a boosting scheme. According to the experimental results, RHSBoost appears to be an attractive classification model for imbalance data.

1. Introduction

In the real world, the class distribution of data is hardly balanced; to some extent, almost all datasets have a skewed distribution of classes. In classification problems, these datasets are called imbalance data. For example, rare events have been observed in a variety of areas, such as financial fraud detection and the diagnosis of rare diseases in medicine.

Class imbalance is a factor that degrades the classification performance. Recently, several researchers have found that the imbalance problem is responsible for the low performance of traditional classification models, including LDA, soft margin SVM, and classification trees (Xie and Qiu, 2007; Batuwita and Palade, 2013; Cieslak and Chawla, 2008). In particular, for seriously unbalanced data, traditional models tend to predict minority class instances inaccurately. The fitted model is highly biased toward the majority class, but in fact the minority class is of interest. It is very important that classification models can classify instances of the minority class. Throughout this paper, we assume the case of binary classes consisting of a majority class and a minority class.

2. Measures of performance

In class imbalance problems, the accuracy rate defined as the ratio of correctly classified instances among data is less meaningful. For example, suppose a dataset consists of 90% instances from majority class and 10% from minority class. If all data is predicted as majority class according to the classification rule, the rule gets 90% accuracy. In this case, the accuracy rate is not a proper representation of the classification performance because none of the minority class instances

[*] Corresponding author.
E-mail address: hkim@yonsei.ac.kr (H. Kim).

http://dx.doi.org/10.1016/j.csda.2017.01.005
0167-9473/© 2017 Elsevier B.V. All rights reserved.

Table 1
Confusion matrix for binary class.

	Predicted Positive	Predicted Negative
Actual Positive	True Positive (TP)	False Negative (FN)
Actual Negative	False Positive (FP)	True Negative (TN)

are correctly classified. In this article, the area under ROC curve (Huang and Ling, 2005), AUC, is used as the primary measure of classification performance. AUC is a good way to measure the performance of a classifier, regardless of the severity of imbalance.

G-mean (Kubat et al., 1997) is also a useful replacement of the accuracy rate. For example of a binary class with positive and negative labels, the classification model is considered successful if the true positive and the true negative values in Table 1 are large. On the other hand, the large values of false positive and false negative indicate poor performance of the model. By defining G-mean as follows, we can measure the performance of the classification model regardless of the degree of imbalance between classes.

$$G\text{-mean} = \sqrt{TP\text{rate} \times TN\text{rate}},$$
$$\text{where } TP\text{rate} = \frac{TP}{TP + FN} \quad \text{and} \quad TN\text{rate} = \frac{TN}{FP + TN}.$$

3. Previous research

How do we represent the severity of imbalance? In general, most researchers accept an intuitive criterion, called imbalance ratio, adopted in Orriols-Puig and Bernadó-Mansilla (2009). Imbalance ratio (IR) is defined as the ratio between the numbers of majority class and minority class instances.

Common practices for class imbalance problems are balancing classes by using undersampling on majority class instances or oversampling on minority class instances. Researchers studied approaches to using new sampling techniques. These methods intend to keep similar characteristics of the original imbalance data. One of the successful methods in this category is SMOTE (Chawla et al., 2002), which uses k-nearest neighbors to generate new perturbed data near instances of the minority class.

ROSE (Menardi and Torelli, 2014) is an another method in this category. It takes advantage of kernel density estimation to maintain characteristics of each class. Let y be the class variable with class label C_j and let $\mathbf{x}_{ij}$ be the input data of the jth class composed of d-dimensional predictors, where $i = 1, \ldots, n_j$, and n_j be the number of instances of the jth class, $j = 1, 2$, for binary classification. For an instance of the input data $\mathbf{x}$ from the jth class C_j, the ROSE procedure corresponds to sampling from a multivariate kernel density estimate of $f(\mathbf{x} \mid y = C_j)$ as in (1).

$$\hat{f}(\mathbf{x} \mid y = C_j) = \sum_{i=1}^{n_j} p_i Pr(\mathbf{x} \mid \mathbf{x}_{ij}) = \sum_{i=1}^{n_j} \frac{1}{n_j} K_{\mathbf{H}_j}(\mathbf{x} - \mathbf{x}_{ij}), \tag{1}$$

where $K_{\mathbf{H}_j}$ is an estimated kernel function and its smoothing matrix $\mathbf{H}_j$ is defined as

$$\mathbf{H}_j = \text{diag}(h_1^{(j)}, \ldots, h_d^{(j)}), \tag{2}$$

where

$$h_q^{(j)} = \left(\frac{4}{(d+2)n} \right)^{\frac{1}{(d+4)}} \hat{\sigma}_{\text{q}}^{(j)}, \quad q = 1, \ldots, d,$$

and $\hat{\sigma}_q^{(j)}$ is an estimated standard deviation of the qth variable. It is known that the form of smoothing matrices $\mathbf{H}_j$ in (2) minimizes AMISE (Asymptotic Mean Integrated Squared Error) under normal distribution assumption (Bowman and Azzalini, 1997).

It is found that classification models based on newly created data using ROSE performs better than models with original imbalance data. Thanks to its flexible and enlarged decision region associated with the minority class, ROSE outperforms SMOTE in terms of AUC measurement in classification trees and logistic regression models (Menardi and Torelli, 2014). The specific process of ROSE sampling algorithm is described in Algorithm 1.

Researchers have proposed many ingenious strategies for adopting ensemble methods for imbalance data problems. Seiffert et al. (2010) proposed RUSBoost as an ensemble method for class imbalance data. In addition to using the boosting algorithm, they apply undersampling on majority class to mitigate the severity of the imbalance problem. This computationally simple method allows classifiers to focus more on minority class instances by neglecting majority class instances that are less important in each boosting iteration. As a result, a remarkable improvement of classification accuracy can be attained on the minority class instances (Seiffert et al., 2010).

Algorithm 1 ROSE sampling algorithm

```
Input: $S_{input} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, $n_j$ = the number of instances for the $j$th class $C_j$, $j = 1, 2$, $n_1 + n_2 = n$, $n^*$ = the
    desired number of instances after ROSE sampling.
1: procedure ROSE
2:     for $i = 1, \ldots, n^*$ do:
3:         Select $C_j$ with $\frac{1}{2}$ probability.
4:         Select one $(\mathbf{x}, y)$ in $S_{input}$ using weight $p_j = \frac{1}{n_j}$ among instances of $y = C_j$.
5:         Sample $\mathbf{x}_i^*$ from $K_{\mathbf{H}_j}(\mathbf{x})$, with $K_{\mathbf{H}_j}$ a probability distribution centered at $\mathbf{x}$ and depending on a smoothing matrix
           $\mathbf{H}_j$ of scale parameters.
6:     end for
7: end procedure
Output: $S_{balanced} = \{(\mathbf{x}_i^*, y_i)\}$, $i = 1, \ldots, n^*$.
```

Table 2
22 ensemble methods for imbalance data problem (Galar et al., 2012).

Cost-Sensitive Ensembles	Cost-sensitive boosting	AdaCost CSB1, CSB2 RareBoost AdaC1 AdaC2 AdaC3
Data Preprocessing + Ensemble Learning	Boosting-based	SMOTEBoost MSMOTEBoost RUSBoost DataBoost-IM
	Bagging-based	OverBagging SMOTEBagging UnderBagging QuasiBagging Asymetric bagging Roughly balanced bagging Partitioning Bagging ensemble variation UnderOverBagging IIVotes
	Hybrid	EasyEnsemble BalanceCascade

Galar et al. (2012) categorized 22 ensemble methods for imbalance data into four categories: cost-sensitive boosting, boosting-based, bagging-based and hybrid clustering, then compared them using 44 datasets. Table 2 arranges the whole methods in accordance with the proposed categorization. According to Galar et al. (2012), SMOTEBagging (Wang and Yao, 2009), RUSBoost (Seiffert et al., 2010), and UnderBagging (Barandela et al., 2003) statistically outperformed the others. Although the three methods are statistically equivalent, RUSBoost is preferred due to simplicity of computation.

4. RHSBoost

Using the intuition of ROSE and RUSBoost, we design and develop the RHSBoost (random hybrid sampling boosting) algorithm. More specifically, we seek a hybrid sampling strategy that uses both undersampling and ROSE sampling in the boosting algorithm to further improve classification accuracy.

4.1. Sampling strategy in RHSBoost

Drummond et al. (2003) have studied the two common strategies for imbalance data, undersampling and oversampling, using C4.5 (Quinlan, 1993) decision tree method. The results for undersampling show reasonable sensitivity to different levels of class distribution, while oversampling makes little difference. In other words, the former sampling scheme effectively influences the classifier to cope with the class imbalance data. The superiority of undersampling-based algorithms, such as RUSBoost (Seiffert et al., 2010) and Underbagging (Barandela et al., 2003), can be explained in this context. In this regard, we design a hybrid undersampling method.

The proposed RHSBoost method employs a hybrid sampling scheme that sequentially combines undersampling on majority class and ROSE sampling for all data. The boosting method is then used on the updated dataset after the hybrid sampling. The weight assignment procedure in boosting algorithm should be considered under RHSBoost sampling scheme, and is given in Section 4.2.

4.2. RHSBoost Algorithm

Since ROSE initially assigns uniform probability to all instances of each class, we designed a weighted ROSE algorithm to allow different weights to the instance. This is necessary because the boosting algorithm generates different weights for the instances at each step of the iteration. Algorithm 2 shows the entire process of weighted ROSE sampling for RHSBoost.

Algorithm 2 Weighted ROSE sampling algorithm

```
Input: $S_{input} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$, $C_j$ is the $j$th class label, $j = 1, 2$, $D = \{D(1), \dots, D(n)\}$ where $D$ is the vector of
    weights that repeatedly vary during the boosting algorithm, $n^*$ = the desired number of instances after ROSE sampling.
1: procedure WEIGHTED ROSE
2:     for $i = 1, \dots, n^*$ do:
3:         Select $C_j$ with $\frac{1}{2}$ probability.
4:         Select one $(\mathbf{x}, y)$ in $S_{input}$ using weight vector $D$ among instances of $y = C_j$.
5:         Sample $\mathbf{x}_i^*$ from $K_{\mathbf{H}_j}(\mathbf{x})$, with $K_{\mathbf{H}_j}$ a probability distribution centered at $\mathbf{x}$ and depending on a matrix $\mathbf{H}_j$ of scale
           parameters.
6:     end for
7: end procedure
Output: $S_{balanced} = \{(\mathbf{x}_i^*, y_i)\}$, $i = 1, \dots, n^*$.
```

Similar to other ensemble-based methods, such as SMOTEBoost (Chawla et al., 2003), RUSBoost (Seiffert et al., 2010), and EUSBoost (Galar et al., 2013), RHSBoost takes advantage of AdaBoost (Freund et al., 1996), one of the most popular ensemble methods in classification problems. The AdaBoost algorithm is a method of developing a final classifier by linearly combining weak learners generated using an iterative process. This method improves the accuracy of the final classification because additional weak learners continue to focus on previously misclassified instances at each iteration. The AdaBoost algorithm is given in Appendix A.

In summary, based on the AdaBoost algorithm, RHSBoost uses a hybrid sampling strategy to improve the classification performance of imbalance data. Algorithm 3 illustrates each step of the RHSBoost algorithm.

Algorithm 3 RHSBoost algorithm

```
Input: $S_{original} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$, $h$ = Base Classifier, $T$ = the number of iterations in boosting procedure.
1: procedure RHSBOOST
2:     Initialize the weight vector $D_1 = \{D_1(1), \dots, D_1(n)\}$ s.t. $D_1(i) = \frac{1}{n}$ for $i = 1, \dots, n$.
3:     Calculate the smoothing matrices $\mathbf{H}_j$ for each class, $j = 1, 2$.
4:     for $t = 1, \dots, T$ do:
5:         Do undersampling on the majority class of $S_{original}$ with $D_t$ to make $S_{under}$.
6:         Call the weighted ROSE algorithm with $D_t$ to oversample $S_{under}$ until it is augmented to $S_{temporary}$ of size n.
7:         Train a Base Classifier $h_t : \mathbf{x} \to y$, using $S_{temporary}$.
8:         $\epsilon_t = \sum_{i=1}^{n} D_t(i)\, \mathbf{I}(h_t(\mathbf{x}_i) \neq y_i)$, where $\mathbf{I}$ is an indicator function and $(\mathbf{x}_i, y_i) \in S_{original}$.
9:         $\beta_t = \frac{1}{2} \log \frac{1-\epsilon_t}{\epsilon_t}$.
10:        $D_{t+1}(i) = \frac{D_t(i)}{Z_t} \cdot \begin{cases} e^{\beta_t} & \text{if } h_t(\mathbf{x}_i) \neq y_i \\ 1 & \text{if } h_t(\mathbf{x}_i) = y_i \end{cases}, \quad i = 1, \dots, n$
       where $Z_t$ is normalizing constant s.t. $\sum_{i=1}^{n} D_{t+1}(i) = 1$.
11:    end for
12: end procedure
Output: The final classifier
       $H(\mathbf{x}) = \arg\max_{y \in Y} \sum_{t=1}^{T} \beta_t \cdot \mathbf{I}(h_t(\mathbf{x}) = y)$.
```

Algorithm 3 can be described as follows. First, the algorithm needs to calculate smoothing matrices $\mathbf{H}_j$ for each class. The matrices are based on the original dataset $S_{original}$ and fixed until the algorithm ends. They are used to generate new artificial data in the weighted ROSE sampling step. The RHSBoost algorithm begins its iterative process. The first step is a simple random undersampling on majority class instances. The undersampling makes the number of the majority class equal to that of the minority class. Then, in the second step, the weighted ROSE algorithm is applied to the reduced data. This step increases the number of instances of both classes by adding newly sampled data, while maintaining the balance between

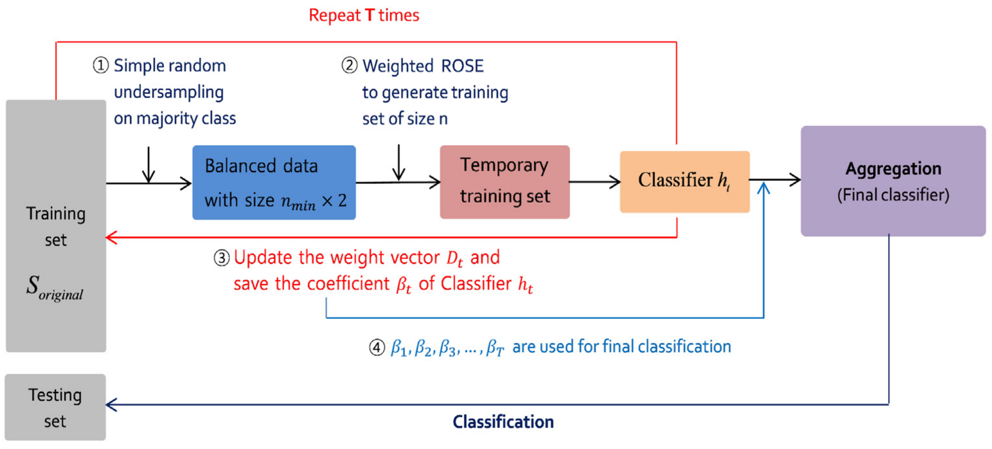
Repeat T times
① Simple random undersampling on majority class
② Weighted ROSE to generate training set of size n
Training set $S_{original}$
Balanced data with size $n_{min} \times 2$
Temporary training set
Classifier h_t
Aggregation (Final classifier)
③ Update the weight vector D_t and save the coefficient β_t of Classifier h_t
④ $\beta_1, \beta_2, \beta_3, \ldots, \beta_T$ are used for final classification
Testing set
Classification

Fig. 1. RHSBoost algorithm.

Table 3
16 real imbalance data.

Data	Description	# obs	Class distribution (minority/majority)	# variables	IR	Source
covt	Forest cover	38 501	Ponderosa pine/ Cottonwood-willow	10	13.02	UCI
page	Page-blocks13vs2	472	graphic/ horiz,line,picture	10	15.85	KEEL
aba2	Abalone9vs18	731	18/9	8	16.68	KEEL
gla2	Glass016vs5	184	tableware/ build-win-float-proc, build-winnon_float-proc headlamp	9	19.44	KEEL
shu	Shuttle2vs4	129	Fpv Open/Bypass	9	20.5	KEEL
yea6	Yeast1458vs7	693	vac/nuc,me2,me3,pox	8	22.10	KEEL
gla1	Glass5	214	tableware/remainder	9	22.81	KEEL
yea5	Yeast2vs8	482	pox/cyt	8	23.10	KEEL
wine	Wine	4 898	$\leq 4/4 \geq$ wine quality	11	25.77	UCI
yea4	Yeast4	1 484	me2/remainder	8	28.41	KEEL
yea3	Yeast1289vs7	947	vac/nuc,cyt,pox,erl	8	30.56	KEEL
yea2	Yeast5	1 484	me1/remainder	8	32.78	KEEL
ecol	Ecoli0137vs26	281	pp,imL/cp,im,imU,imS	7	39.15	KEEL
yea1	Yeast6	1 484	Exc/remainder	8	39.15	KEEL
aba1	Abalone	4 177	$> 20/ \leq 20$ Rings	7	115.03	UCI
info	Innfocamere	11 199	Defaulter/non-defaulter firms	27	151.82	Menardi and Torelli (2014)

the classes. The augmentation ends when the total number of the augmented data reaches n, the exact size of the original dataset. The third step is to build a base classifier h. Classification performance of the classifier on $S_{original}$ updates the weight vector D and determines its corresponding coefficient β. These three steps are iterated T times, after which h_t are aggregated with coefficients β_t as the final classifier $H(\mathbf{x})$. Fig. 1 also graphically represents the entire process of RHSBoost.

5. Experiment

5.1. Data

For the purpose of comparing RHSBoost with other methods, we have collected 16 real imbalance data from KEEL (Alcalá et al., 2010), the UCI machine learning repository (Lichman, 2013), and Menardi and Torelli (2014). Table 3 provides the details in the order of lowest IR. For experiment purposes, only the numeric variables in the data are selected. Also, some data are derived from the same dataset such as *Yeast*, but the way classes are grouped is different. The table provides information on how classes are grouped to generate binary class problems. Except *Infocamere*, all data can be found on Alcalá et al. (2010) and Lichman (2013).

5.2. Experimental design

First, we randomly split original data into training and test datasets with the same number of instances, maintaining the unbalanced class distribution. We apply eight solutions to the imbalance problem to the training data, which includes the method of no treatment as a comparison basis. The test data is not treated at all, thus keeps the original class distribution.

Table 4
Eight classification methods for imbalance data.

1.	No treatment (Nothing)	5.	AdaBoost (AdaB)
2.	Over sampling (Over)	6.	SMOTEBoost (SMOB)
3.	Under sampling (Under)	7.	RUSBoost (RUSB)
4.	Overunder sampling (Ovun)	8.	RHSBoost (RHSB)

Table 5
R packages.

Name of R package	Function
rpart (Therneau et al., 2015)	CART classification tree
DMwR (Torgo, 2010)	SMOTE algorithm
adabag (Alfaro et al., 2013)	AdaBoost.M1 algorithm
ROSE (Lunardon et al., 2014)	ROSE algorithm

Four of the eight methods come from non-ensemble methods and the rest from boosting. Table 4 lists the seven competitors and RHSBoost.

The first four methods are fitted with the CART (Breiman et al., 1984) model. The latter four methods are boosting based models using CART as a base classifier. The classification performance is then evaluated in terms of AUC and G-means. This whole process is repeated 100 times with different training and test data combinations. To compare performance, the AUCs of two specific methods are calculated using test data. Then, a paired t-test is performed using two AUC values from each of 100 repeated test data. Since eight methods are paired by two, a total of $\binom{8}{2} = 28$ tests are performed on each dataset.

The experiments are performed using a number of R (R Core Team, 2015) packages developed to implement various classification methods. Table 5 lists the name and the function of specific packages.

5.3. Issues

According to Lunardon et al. (2014), if there is a possibility that sampled instances of different classes are crowded on the boundary of distribution, it is necessary to cautiously select the shrinking or inflating argument provided by the ROSE package. However, it is difficult to meticulously tune the factors each time a new dataset is handled. Therefore, we equally and simultaneously set a 1/2 shrinkage factor for majority and minority classes of all real data.

The number of iterations is another issue to clarify; this term means the ensemble size or the number of fitted classifiers in an ensemble. There is no specific criterion for this parameter. For a fair comparison, we used the same number of iterations for each boosting-based algorithm. Specifically, we tried iteration values of 10, 25, 50, 75, and 100.

5.4. Results

When the number of iterations increases, the AUC values of RHSBoost also increases in most datasets. For example, Fig. 2 presents the AUC values of RHSBoost from 10 to 100 iterations for 9 datasets. Although the AUC tends to increase as RHSBoot adds iteration, it is difficult to generalize that AUC always increases. In fact, in the case of *yea1* data, the highest AUC occurs at a small number of iterations. Nevertheless, it can be seen that the overall performance of RHSBoost is better with a larger number of iterations.

Fig. 3 shows the pairwise comparison result of RHSBoost with other ensemble methods. We calculate the paired t-test statistics for 16 real data for each ensemble size. One boxplot in Fig. 3 is drawn using 16 t-statistics. According to the boxplots, most t-statistics are positive and many are over 2, a rough criterion of 5% significance level. In summary, the classification performance of RHSBoost appears to be better in many real datasets. However, the dominance of RHSBoost is slightly weaker for large iterations such as 75 and 100.

Fig. 4 shows the performance of the eight methods in terms of mean AUC and G-mean for 16 datasets when the number of iterations in an ensemble is $T = 50$. These plots clearly show the improved performance of RHSBoost. Due to the limited length of the paper, we have attached boxplots for other levels of iterations, 10, 25, 75 and 100, in Appendix B.

Table 6 provides the win–loss statistics by summarizing the results of the paired t-tests. The numbers in each cell mean how many times the method in the column will beat the method in the row among 16 dataset. The first number in the cell is the number of datasets where the t statistic is positive, and the second number is where the value is greater than 2. For example, let us compare RHSBoost and AdaBoost. The numbers in the cell of column RHSBoost and row AdaBoost are 12 | 9. It can be interpreted that RHSBoost has better prediction accuracy for 12 among 16 dataset, while 9 of them are statistically significant. The sums of significant wins and losses are recorded in Table 7 to calculate the dominance rank for eight methods. The dominance rank is defined as the difference between the number of significant wins and the number of significant losses. For example, according to Table 7, RHSBoost has a total of 90 t-statistics greater than 2. The number 90 comes from the sum of the numbers to the right of the last column of Table 6. On the other hand, according to the t-statistics, AdaBoost, SMOTEBoost, and RUSBoost significantly win over RHSBoost 4, 2, and 5 times, respectively, so RHSBoost of Table 7

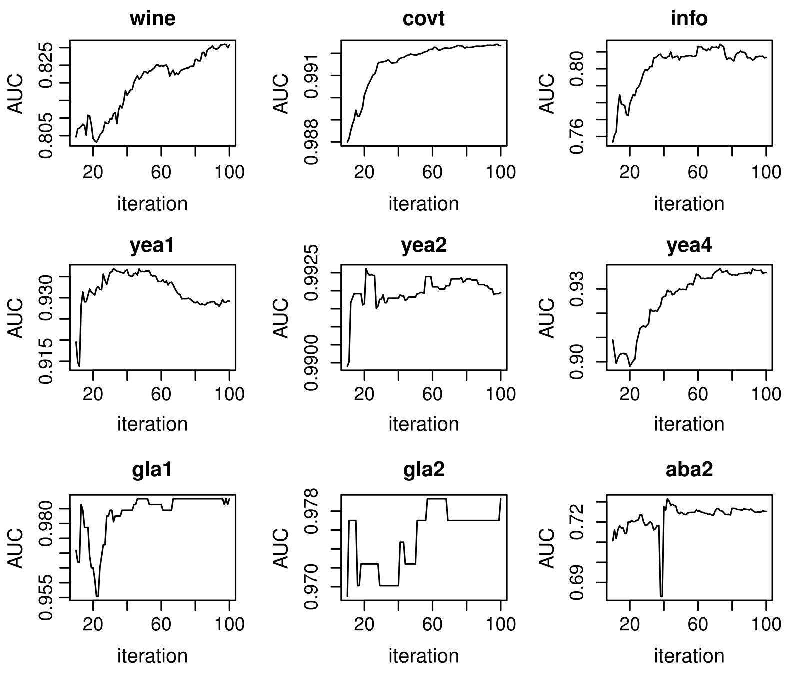

Fig. 2. Trend of AUC of RHSBoost with increasing number of iterations.

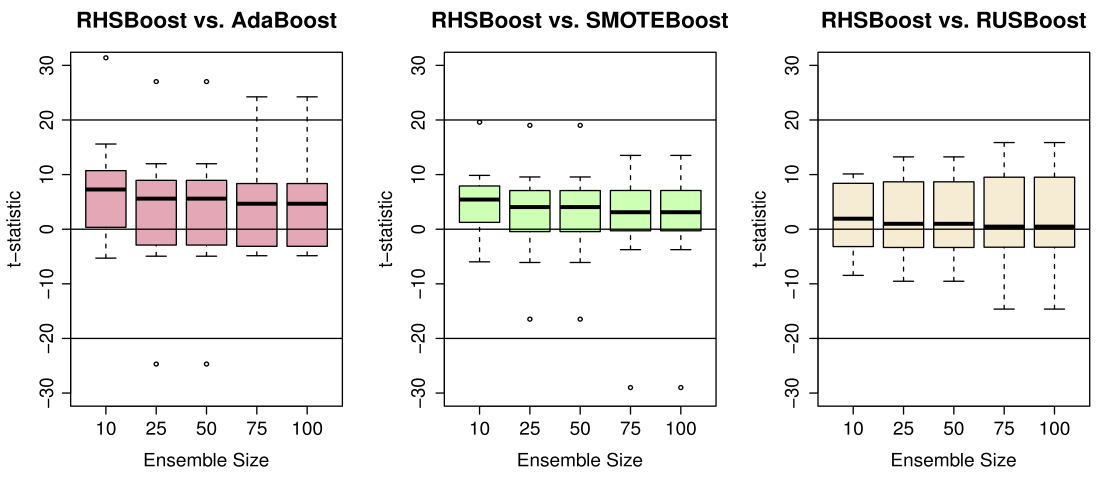

Fig. 3. Paired t-test statistics.

Table 6
Win–loss table ($T = 50$).

Method	Nothing	Over	Under	Ovun	AdaBoost	SMOTEBoost	RUSBoost	RHSBoost
Nothing	0	14|13	14|14	15|14	16|16	16|16	16|15	16|16
Over	2|1	0	11|11	15|11	15|15	15|15	15|15	16|16
Under	2|1	5|5	0	6|5	15|15	16|16	16|16	16|16
Ovun	1|1	1|0	10|8	0	15|15	15|15	15|15	16|16
AdaBoost	0|0	1|1	1|0	1|0	0	6|5	9|6	12|9
SMOTEBoost	0|0	1|1	0|0	1|0	10|7	0	8|8	13|10
RUSBoost	0|0	1|0	0|0	1|0	7|5	8|5	0	9|7
RHSBoost	0|0	0|0	0|0	0|0	4|4	3|2	7|5	0

is recorded with 11 losses. Therefore, the dominance rank of RHSBoost is calculated as $90 - 11 = 79$. The average ranks on the last column are the mean values of the rankings among the eight methods based on 100 repeated experiments.

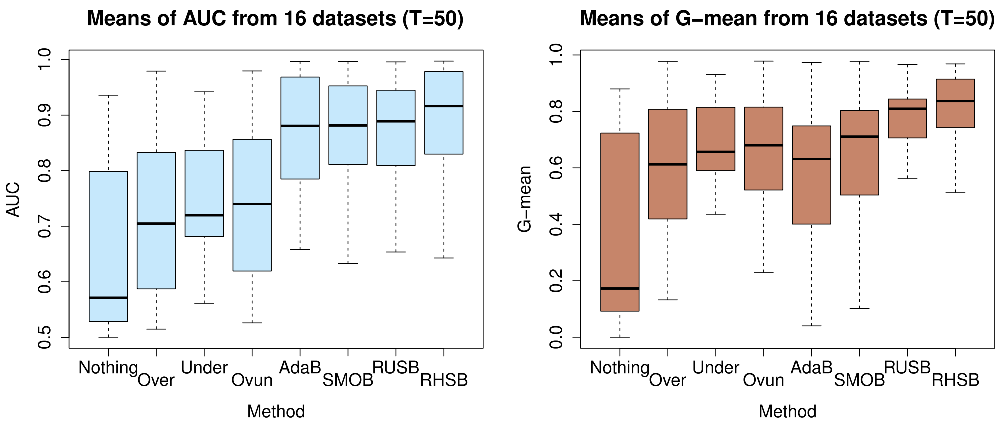

Fig. 4. Boxplot of mean AUC and *G*-mean.

Table 7
Dominance rank (Wins–losses for $T = 50$).

Method	Dominance	Wins	Losses	Avgerage rank
Nothing	−101	3	104	7.056
Over	−64	20	84	6.283
Under	−41	33	74	5.606
Ovun	−40	30	70	5.832
AdaBoost	56	77	21	2.934
SMOTEBoost	48	74	26	2.988
RUSBoost	63	80	17	2.847
RHSBoost	79	90	11	2.453

Table 8
Distribution of data.

Variable	Distribution
X_1	$N(0,1)$
X_2	$\chi^2(1)$
X_3	Exp(1)
X_4	Unif(0,1)
X_5	Discrete Unif(1,2,3,4)

Tables 6 and 7 show that RHSBoost has the best classification performance for imbalance data compared to other methods. The results for the remaining four ensemble sizes are included in the Appendix.

6. Simulation study

The experiment with real data shows that the performances of RHSBoost and RUSBoost are quite close. This section aims to understand the behavior and characteristics of the two algorithms. In order to compare these two classification methods in detail, a simulation study is conducted with different iteration numbers and various ratios of minority class in dataset.

A dataset $(\mathbf{x}, y)$ is generated by a logit model with 5 predictor variables. Each of predictors has a different distribution as specified in Table 8, then the entries of the vector $\mathbf{x}$ are independently sampled from the corresponding distributions. After $\mathbf{x}$ is drawn, a binary variable y is created with the probability of p, where p is calculated from the function

$$logit(p) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 + \beta_5 x_5.$$

In this simulation, we set $(\beta_1, \beta_2, \beta_3, \beta_4, \beta_5) = (5, 4, 3, 2, 1)$. Although 10,000 instances are generated at a time, only 2000 of them are chosen in order to adjust the degree of imbalance in the dataset. The ratios of the minority class are set to be 25%, 20%, 15%, 10%, 7.5%, and 5%. The 2000 instances are divided into two random parts with the same ratio of minority so that a half takes the role of the training data and the other half works as the test data. This simulation is repeated 100 times with different random numbers.

For each given ratio, ensemble sizes of 10, 25, 40, 55, 70, and 85 are tried. In summary, the two methods are compared under 36 different conditions which result from 6 ratios of minority class and 6 ensemble sizes. To compare the classification performances, the mean AUC of the two methods are computed, in addition, the difference of two AUCs are also calculated.

From the simulation study, some characteristics that were not identified in Section 5 are evident. First, in terms of the mean AUC value, RHSBoost outperforms RUSBoost for all 36 different conditions as in Fig. 5. As the ratio of the minority

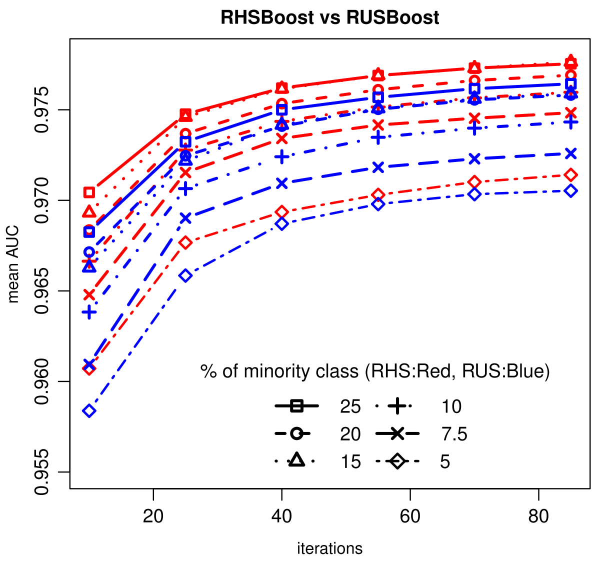

Fig. 5. Trends of mean AUCs of RHSBoost and RUSBoost.

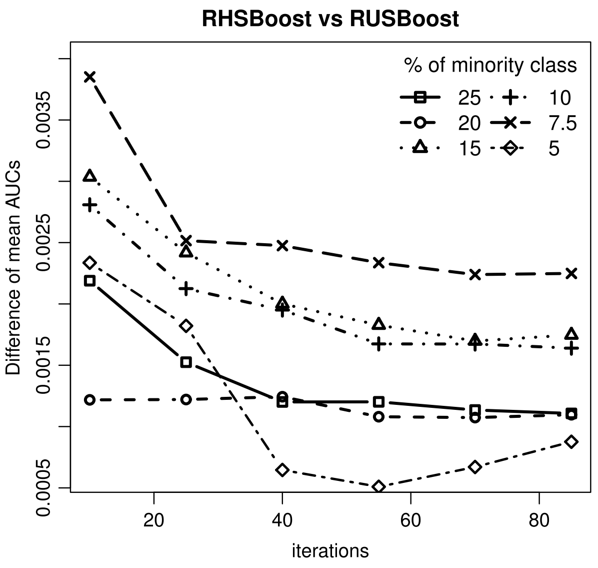

Fig. 6. Difference of mean AUCs between RHSBoost and RUSBoost.

class decreases, the performance of RUSBoost declines rapidly. On the other hand, RHSBoost maintains its classification performance relatively high. Therefore, we can conclude that RHSBoost is less vulnerable to the imbalance of data. It is interesting to note that the mean AUCs of RHSBoost at minority ratios of 25%, 20%, and 15% are still higher than that of RUSBoost at minority ratio of 25%.

Fig. 6 reveals the difference of AUCs between RHSBoost and RUSBoost. Although the differences might not be significant in some cases, all of the differences are positive. As in Section 5, it is clear that the small ensemble size is advantageous to RHSBoost and the difference between the two methods gradually decreases as the ensemble size increases.

7. Conclusion

RHSBoost is proposed as an ensemble method to deal with the problem of imbalance classification. It uses a hybrid sampling strategy that combines undersampling with ROSE sampling. As an ensemble technique, AdaBoost algorithm is adopted in the proposed method. In short, the RHSBoost method is a combination of AdaBoost and hybrid sampling.

We carried out a considerable comparisons using 16 real data of various IR levels. For small numbers of ensemble sizes, such as 10, 25, and 50, RHSBoost considerably outperforms the other methods. As the ensemble size increases to 100, the amount of improvement due to RHSBoost decreases, RUSBoost is close to the performance of RHSBoost for some data.

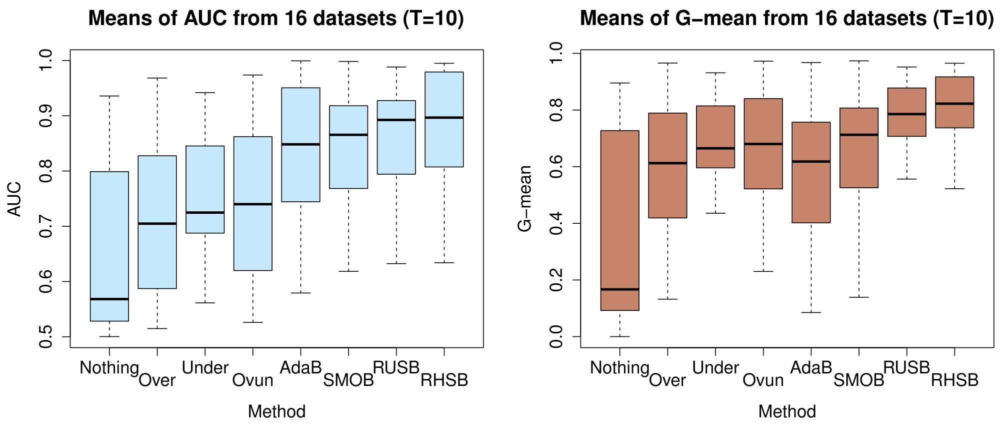

Fig. B.7. Boxplot of mean AUC and *G*-mean.

However, over various IR levels, RHSBoost provides reliable and high classification performance in real data. In terms of dominance rank, RHSBoost shows the best results. In addition, the mean value of total ranks for RHSBoost is also recorded as the lowest. These remarkable results of RHSBoost do not vary much with the levels of the ensemble size. From the simulation study, we find that RHSBoost is less sensitive to ratios of minority class than RUSBoost.

RHSBoost is an attractive option for imbalance data problems. RHSBoost shows its robustness against various types of data and various levels of IR. Although it does not significantly excel in some data, we can argue that considerable improvement on most datasets can be achieved by RHSBoost.

Acknowledgment

Hyunjoong Kim's work was supported by Basic Science Research program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science, and Technology (No. 2016R1D1A1B02011696).

Appendix A. The AdaBoost algorithm

Algorithm 4 AdaBoost algorithm

```
Input: $S_{original} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$, $h$ = Base Classifier, $T$ = the number of iterations in boosting procedure.
1: procedure AdaBoost
2:     Initialize the weight vector $D_1 = \{D_1(1), \dots, D_1(n)\}$ s.t. $D_1(i) = \frac{1}{n}$ for $i = 1, \dots, n$.
3:     for $t = 1, \dots, T$ do:
4:         Train a Base Classifier $h_t: \mathbf{x} \to y$, using $S_{original}$.
5:         $\epsilon_t = \sum_{i=1}^{n} D_t(i)\, \mathbf{I}(h_t(\mathbf{x}_i) \neq y_i)$, where $\mathbf{I}$ is an indicator function and $(\mathbf{x}_i, y_i) \in S_{original}$.
6:         $\beta_t = \frac{1}{2} \log \frac{1-\epsilon_t}{\epsilon_t}$.
7:         $D_{t+1}(i) = \frac{D_t(i)}{Z_t} \cdot \begin{cases} e^{\beta_t} & \text{if } h_t(\mathbf{x}_i) \neq y_i \\ 1 & \text{if } h_t(\mathbf{x}_i) = y_i \end{cases}, \quad i = 1, \dots, n$
          where $Z_t$ is normalizing constant s.t. $\sum_{i=1}^{n} D_{t+1}(i) = 1$.
8:     end for
9: end procedure
Output: The final classifier
          $H(\mathbf{x}) = \arg\max_{y \in Y} \sum_{t=1}^{T} \beta_t \cdot \mathbf{I}(h_t(\mathbf{x}) = y)$.
```

Appendix B. Results of the experiment with different levels of ensemble size

See Figs. B.7–B.10. See Tables B.9–B.16.

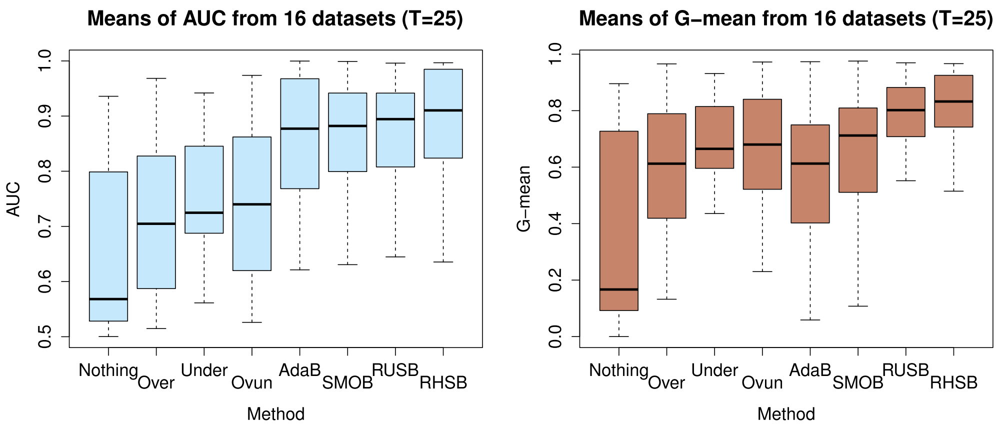

Fig. B.8. Boxplot of mean AUC and *G*-mean.

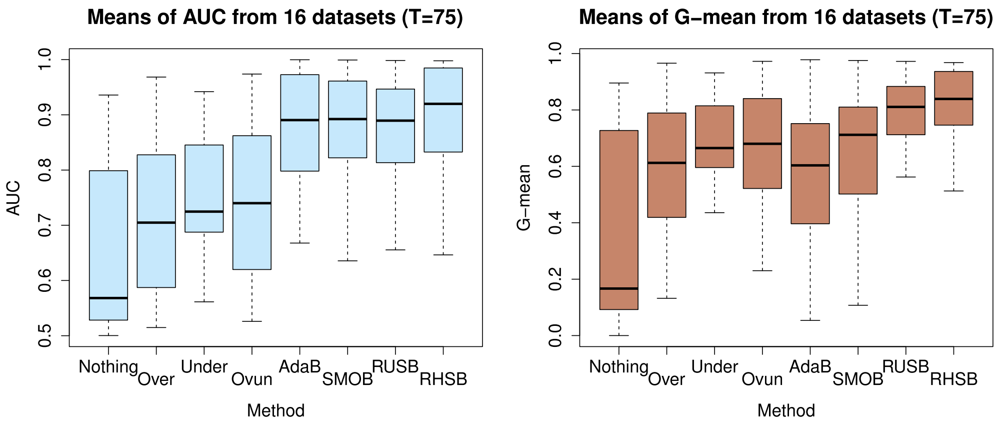

Fig. B.9. Boxplot of mean AUC and *G*-mean.

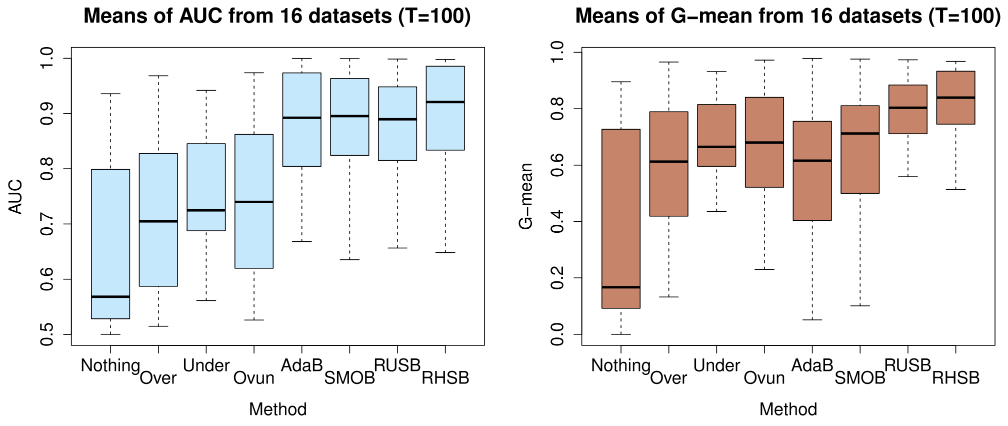

Fig. B.10. Boxplot of mean AUC and *G*-mean.

Table B.9
Win–loss table ($T = 10$).

Method	Nothing	Over	Under	Ovun	AdaBoost	SMOTEBoost	RUSBoost	RHSBoost
Nothing	0	14|13	13|13	15|14	16|16	16|16	16|15	16|16
Over	2|1	0	11|11	15|11	15|15	15|15	15|15	16|16
Under	3|1	5|5	0	6|5	13|13	15|15	16|16	16|16
Ovun	1|1	1|0	10|8	0	16|15	16|15	15|15	16|16
AdaBoost	0|0	1|0	3|2	0|0	0	10|6	9|9	12|10
SMOTEBoost	0|0	1|0	1|1	0|0	6|5	0	10|8	13|11
RUSBoost	0|0	1|1	0|0	1|1	7|4	6|4	0	11|8
RHSBoost	0|0	0|0	0|0	0|0	4|2	3|2	5|5	0

Table B.10
Dominance rank (Wins–losses for $T = 10$).

Method	Dominance	Wins	Losses	Average rank
Nothing	−100	3	103	7.9715625
Over	−65	19	84	6.1953125
Under	−36	35	71	5.393125
Ovun	−39	31	70	5.6953125
AdaBoost	43	70	27	3.3296875
SMOTEBoost	48	73	25	3.144375
RUSBoost	65	83	18	2.8275
RHSBoost	84	93	9	2.443125

Table B.11
Win–loss table ($T = 25$).

Method	Nothing	Over	Under	Ovun	AdaBoost	SMOTEBoost	RUSBoost	RHSBoost
Nothing	0	14|13	13|13	15|14	16|16	16|16	16|15	16|16
Over	2|1	0	11|11	15|11	15|15	15|15	15|15	16|16
Under	3|1	5|5	0	6|5	15|15	16|16	16|16	16|16
Ovun	1|1	1|0	10|8	0	15|15	16|15	15|15	16|16
AdaBoost	0|0	1|1	1|1	1|0	0	7|5	9|7	11|10
SMOTEBoost	0|0	1|1	0|0	0|0	9|6	0	8|8	12|10
RUSBoost	0|0	1|1	0|0	1|1	7|7	8|5	0	9|7
RHSBoost	0|0	0|0	0|0	0|0	5|5	4|3	7|5	0

Table B.12
Dominance rank (Wins–losses for $T = 25$).

Method	Dominance	Wins	Losses	Avgerage rank
Nothing	−100	3	103	7.0246875
Over	−63	21	84	6.2565625
Under	−41	33	74	5.54375
Ovun	−39	31	70	5.7796875
AdaBoost	55	79	24	3.0221875
SMOTEBoost	50	75	25	2.995625
RUSBoost	60	81	21	2.880625
RHSBoost	78	91	13	2.496875

Table B.13
Win–loss table ($T = 75$).

Method	Nothing	Over	Under	Ovun	AdaBoost	SMOTEBoost	RUSBoost	RHSBoost
Nothing	0	14|13	13|13	15|14	16|16	16|16	15|15	16|16
Over	2|1	0	11|11	15|11	15|15	15|15	15|15	16|16
Under	3|1	5|5	0	6|5	16|15	16|16	16|15	16|16
Ovun	1|1	1|0	10|8	0	15|15	15|15	15|15	16|16
AdaBoost	0|0	1|1	0|0	1|0	0	6|5	9|6	10|9
SMOTEBoost	0|0	1|1	0|0	1|0	10|7	0	8|8	12|9
RUSBoost	1|0	1|1	0|0	1|1	7|7	8|7	0	8|8
RHSBoost	0|0	0|0	0|0	0|0	6|5	4|3	8|6	0

Table B.14
Dominance rank (Wins–losses for $T = 75$).

Method	Dominance	Wins	Losses	Avgerage rank
Nothing	−100	3	103	7.0484375
Over	−63	21	84	6.2946875
Under	−41	32	73	5.6590625
Ovun	−39	31	70	5.82875
AdaBoost	59	80	21	2.91
SMOTEBoost	52	77	25	2.9415625
RUSBoost	56	80	24	2.8634375
RHSBoost	76	90	14	2.4540625

Table B.15
Win–loss table ($T = 100$).

Method	Nothing	Over	Under	Ovun	AdaBoost	SMOTEBoost	RUSBoost	RHSBoost
Nothing	0	14|13	13|13	15|14	16|16	16|16	15|15	16|16
Over	2|1	0	11|11	15|11	15|15	15|15	15|15	16|16
Under	3|1	5|5	0	6|5	16|16	16|16	16|15	16|16
Ovun	1|1	1|0	10|8	0	15|15	15|15	15|15	16|16
AdaBoost	0|0	1|1	0|0	1|0	0	6|5	8|6	10|9
SMOTEBoost	0|0	1|1	0|0	1|1	10|7	0	9|8	12|10
RUSBoost	1|0	1|1	0|0	1|1	8|7	7|6	0	8|8
RHSBoost	0|0	0|0	0|0	0|0	6|6	4|3	8|6	0

Table B.16
Dominance rank (Wins–losses for $T = 100$).

Method	Dominance	Wins	Losses	Avgerage rank
Nothing	−100	3	103	7.0509375
Over	−63	21	84	6.3040625
Under	−42	32	74	5.66875
Ovun	−38	32	70	5.8396875
AdaBoost	61	82	21	2.876875
SMOTEBoost	49	76	27	2.9515625
RUSBoost	57	80	23	2.8428125
RHSBoost	76	91	15	2.4653125

References

Alcalá, J., Fernández, A., Luengo, J., Derrac, J., García, S., Sánchez, L., Herrera, F., 2010. Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. J. Mult.-Valued Logic Soft Comput. 17 (2–3), 255–287.

Alfaro, E., Gámez, M., García, N., 2013. adabag: An R package for classification with boosting and bagging. J. Stat. Softw. 54 (2), 1–35. URL http://www.jstatsoft.org/v54/i02/.

Barandela, R., Valdovinos, R.M., Sánchez, J.S., 2003. New applications of ensembles of classifiers. Pattern Anal. Appl. 6 (3), 245–256.

Batuwita, R., Palade, V., 2013. Class imbalance learning methods for support vector machines. In: Imbalanced Learning. Wiley-Blackwell, pp. 83–99. http://dx.doi.org/10.1002/9781118646106.ch5.

Bowman, A.W., Azzalini, A., 1997. Applied Smoothing Techniques for Data Analysis: The Kernel Approach with S-Plus Illustrations: The Kernel Approach with S-Plus Illustrations. Oxford University Press.

Breiman, L., Friedman, J., Stone, C.J., Olshen, R.A., 1984. Classification and Regression Trees. CRC press.

Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P., 2002. Smote: synthetic minority over-sampling technique. J. Artificial Intelligence Res. 321–357.

Chawla, N.V., Lazarevic, A., Hall, L.O., Bowyer, K.W., 2003. Smoteboost: Improving prediction of the minority class in boosting. In: Knowledge Discovery in Databases: PKDD 2003. Springer, pp. 107–119.

Cieslak, D.A., Chawla, N.V., 2008. Learning decision trees for unbalanced data. In: Machine Learning and Knowledge Discovery in Databases. Springer, pp. 241–256.

Drummond, C., Holte, R.C., et al. 2003. C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In: Workshop on Learning from Imbalanced Datasets II. Vol. 11. Citeseer.

Freund, Y., Schapire, R.E., et al. 1996. Experiments with a new boosting algorithm. In: ICML. Vol. 96. pp. 148–156.

Galar, M., Fernandez, A., Barrenechea, E., Bustince, H., Herrera, F., 2012. A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. 42 (4), 463–484.

Galar, M., Fernández, A., Barrenechea, E., Herrera, F., 2013. Eusboost: Enhancing ensembles for highly imbalanced data-sets by evolutionary undersampling. Pattern Recognit. 46 (12), 3460–3471.

Huang, J., Ling, C.X., 2005. Using auc and accuracy in evaluating learning algorithms. IEEE Trans. Knowl. Data Eng. 17 (3), 299–310.

Kubat, M., Holte, R., Matwin, S., 1997. Learning when negative examples abound. In: Machine Learning: ECML-97. Springer, pp. 146–153.

Lichman, M., 2013. UCI machine learning repository. URL http://archive.ics.uci.edu/ml.

Lunardon, N., Menardi, G., Torelli, N., 2014. ROSE: a package for binary imbalanced learning. R J. 6 (1), 82–92.

Menardi, G., Torelli, N., 2014. Training and assessing classification rules with imbalanced data. Data Min. Knowl. Discov. 28 (1), 92–122.

Orriols-Puig, A., Bernadó-Mansilla, E., 2009. Evolutionary rule-based systems for imbalanced data sets. Soft Comput. 13 (3), 213–225.

Quinlan, J., 1993. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Francisco, CA.

R Core Team. 2015. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Seiffert, C., Khoshgoftaar, T.M., Van Hulse, J., Napolitano, A., 2010. Rusboost: A hybrid approach to alleviating class imbalance. IEEE Trans. Syst. Man. Cybern. A 40 (1), 185–197.

Therneau, T., Atkinson, B., Ripley, B., 2015. rpart: Recursive Partitioning and Regression Trees. R package version 4.1–10. URL http://CRAN.R-project.org/package=rpart.

Torgo, L., 2010. Data Mining with R, Learning with Case Studies. Chapman and Hall/CRC, URL http://www.dcc.fc.up.pt/~ltorgo/DataMiningWithR.

Wang, S., Yao, X., 2009. Diversity analysis on imbalanced data sets by using ensemble models. In: IEEE Symposium on Computational Intelligence and Data Mining, 2009. CIDM'09. IEEE, pp. 324–331.

Xie, J., Qiu, Z., 2007. The effect of imbalanced data sets on lda: A theoretical and empirical analysis. Pattern Recognit. 40 (2), 557–562.