Information Sciences 512 (2020) 1214–1233

Contents lists available at ScienceDirect

Information Sciences

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

Learning imbalanced datasets based on SMOTE and Gaussian distribution

Tingting Pan, Junhong Zhao, Wei Wu, Jie Yang*

School of Mathematical Sciences, Dalian University of Technology, Dalian 116024, China

ARTICLE INFO

Article history:
Received 16 April 2019
Revised 28 September 2019
Accepted 25 October 2019
Available online 25 October 2019

Keywords:
Imbalanced
Oversample
Gaussian distribution
SMOTE

ABSTRACT

The learning of imbalanced datasets is a ubiquitous challenge for researchers in the fields of data mining and machine learning. Conventional classifiers are often biased towards the majority class, and loss functions attempt to optimize the quantities. In this paper, we present two effective sampling methods that improve the data distributions. One rebalanced method, the Adaptive-SMOTE, improves the SMOTE method by adaptively selecting groups of *Inner* and *Danger* data from the minority class such that a new minority class is compiled based on the selected data, thus preventing an expansion of the category boundary and strengthening the distributional characteristics of the original data. The other method, Gaussian Oversampling, combines dimension reduction with the Gaussian distribution, which makes the tail of the Gaussian distribution thinner. Cross-validation experiments on 15 datasets show that the two sampling methods achieve significant improvements compared with other typical methods. The Adaptive-SMOTE has higher *F*-measure and *Acc* values than other existing sampling methods and higher robustness to classifiers and datasets with different values of **Imb.** Gaussian Oversampling is more efficient when dealing with extremely imbalanced classifications.

1. Introduction

In the real world, there are numerous classification problems that are characterized by imbalanced data, in which one class of data has far more instances than others; these problems include fraud detection [9], medical diagnosis [28], bioinformatics [26], factory production defects, insurance claims, and targeted marketing, among others. This paper focuses on binary imbalanced datasets in which one class outnumbers the other class. Conventional algorithms work well when the numbers of each class are approximately equal, but there is a tendency to make decisions that are biased towards the majority class in imbalanced problems [23]. In extreme cases, classifiers assign all objects to the majority class. Moreover, the focus of interest is typically the minority class. For example, in a dataset that includes both cancer and non-cancer patients, the focused research is most likely to be on cancer patients. Therefore, there would not be a single evaluation criterion for classification performance regarding the accuracy rate (*Acc*) or the ratio of correctly classified instances. To complement performance evaluations, some useful metrics including the *G*-mean [25], *F*-measure [12], *Kappa* statistics [18], and *AUC*/*ROC* [17] have been proposed.

* Corresponding author.
E-mail address: yangjiee@dlut.edu.cn (J. Yang).

https://doi.org/10.1016/j.ins.2019.10.048
0020-0255/

With the development of big data analysis, interest imbalanced problem has increased, and two main types of testing approaches to solve this problem have been developed (data and classifier approaches) [24]. With respect to the data approach, the most common practices are random oversampling (ROS) and random undersampling (RUS) [14]. Classically, the Synthetic Minority Oversampling Technique (SMOTE) [4] randomly creates artificial samples along the line joining a minority sample and one of its nearest neighbors. SMOTE has been modified to produce the Random-SMOTE [7] and ADASYN [16]. All of these methods avoid the overfitting caused by ROS while reducing the decision space of the majority class. In addition, Tomek [2] presents a solution to the noisy sample problem caused by the SMOTE technique. Furthermore, the extrapolation Borderline-SMOTE SVM [27] synthesizes new minority instances by incorporating borderline information, which essentially enlarges the minority decision space. To overcome the shortcomings of the above methods, we present new methods, the Adaptive-SMOTE and Gaussian Oversampling, to strengthen the dataset distribution as much as possible.

Ensemble learning is a general method for classifying data streams and has shown the potential to increase the classification accuracy beyond that of an individual classifier [29]. Due to this property, these methods are integrated with other techniques and then applied to classifying imbalanced datasets. EasyEnsemble and BalanceCascade [20] are two hybrid ensemble undersampling methods that combine bagging and boosting. SMOTEBoost [5] combines boosting with undersampling to produce balanced training sets for the base classifiers that are applied to the final ensemble. Combined with boosting, RHSBoost [11] uses RUS and ROSE [21] for binary classification. The Bagging of Extrapolation Borderline-SMOTE SVM [27] combines an ensemble of SVMs [6] with borderline information. Above all, when combined with sampling methods, ensemble methods obtain relatively accurate predictive performance.

As a classic oversampling method, SMOTE is improved by Random-SMOTE [7], ADASYN [16], and Borderline-SMOTE SVM [27], among others. For the purposes of this paper, these improved SMOTE methods are denoted as Border-SMOTEs, which oversample the boundary to both reduce the risk of overfitting in many oversampling methods [14] and achieve greater prediction accuracy. However, the robustness of Border-SMOTEs still needs to be improved, since they are easily affected by imbalanced datasets in different fields. The main limitations of Border-SMOTEs are the expansion of the minority data and the neglect of the distributional characteristics of minority samples. Focusing on the distribution density of minority samples, we present an innovative oversampling method, Adaptive-SMOTE, to address the issue of unbalanced classification. To strengthen the distributional nature of minority data, Adaptive-SMOTE adaptively divides minority data into two subsets, *Inner* and *Danger*, which are both used to balance the original data by generating new minority data. This sampling method fully reflects the distributional characteristics of minority data and improves the classification accuracy of the majority data.

In addition, undersampling and oversampling methods lead to information loss [20] and the overestimation of minority data [4], respectively. A fundamental drawback of these sampling methods is that newly balanced samples do not effectively grasp the distribution of the original minority data. In statistics and probability theory, a Gaussian distribution is continuous and therefore offers a proper description of data that cluster around the mean [10]. Based on the Gaussian distribution, another novel method, Gaussian Oversampling, is proposed in this paper. Since the distribution of intraclass data is more concentrated and the distribution of interclass data is more dispersed after proper dimensional reduction, when combined with dimension reduction, the Gaussian model will have a thin tail when fitted using training data and substantially fewer computations. New minority samples are synthesized in accordance with the area under the Gaussian density function.

The remainder of this paper is structured as follows. Section 2 describes the two oversampling methods. Section 3 presents the experimental design, results, and evaluation. Section 4 concludes the paper.

2. Two oversampling algorithms

Given a minority (positive) training set P and a majority (negative) training set N, where $P \bigcup N = D$ and $|P| \ll |N|$, the two oversampling methods each synthesize a new positive set P_{new} from P such that a new positive data set $P' = P \cup P_{new}$ is obtained and N is balanced, i.e., $|P'| = |N|$.

2.1. Adaptive-SMOTE algorithm

The Adaptive-SMOTE is described as follows.

Step 1: For each point $x \in P$, correct classification depends on location. If most neighbors of a point are positive, the point is easily classified as positive (called an *Inner* point); otherwise the point can be classified as positive or negative (called a *Danger* point). Then, a given positive set P is divided adaptively into two subsets, *Inner* and *Danger*, using the k-nearest-neighbor algorithm, but k is a variable ranging from K to $K + C$, where K and C are two fixed parameters. For each positive sample, if there is a $k \in [K, K + C]$ such that more than half of the k nearest neighbors are positive, then it belongs to the *Inner* subset (Fig. 1b); otherwise, it belongs to *Danger* (Fig. 1a).

Step 2: SMOTE synthesizes a new positive sample $P^i_{new} \in P_{new}$ in two cases.

Case 1. If $Inner \neq \emptyset$, then for each $P^i \in Inner$, SMOTE synthesizes a new positive data P^i_{new} between P^i and P^i_{nb} according to

$$P^i_{new} = P^i + \delta(P^i_{nb} - P^i), \tag{2.1}$$

where δ is a random number ranging from [0,1], and P^i_{nb} is the nearest neighbor of P^i in *Danger*. If $Danger = \emptyset$, then P^i_{nb} is the nearest neighbor of $P^i \in Inner$. Fig. 2a shows an example in 2D space.

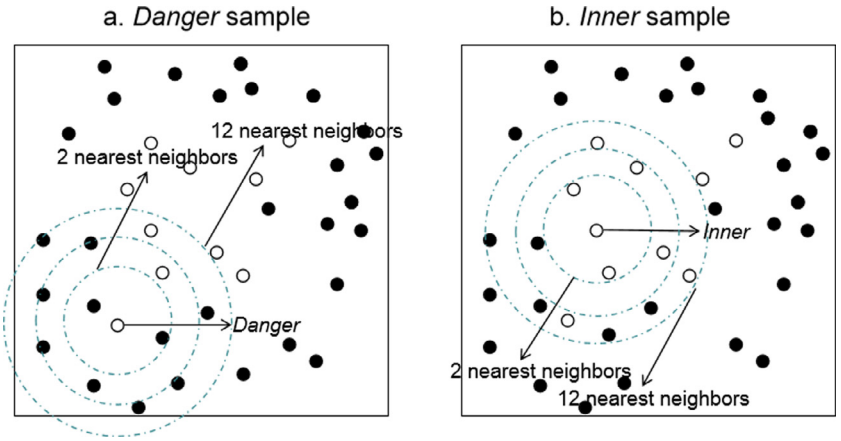

Fig. 1. Dividing P into *Inner* and *Danger* subsets in 2D space ($K = 2, C = 10$). The white points represent positive data, and the black points represent negative data.

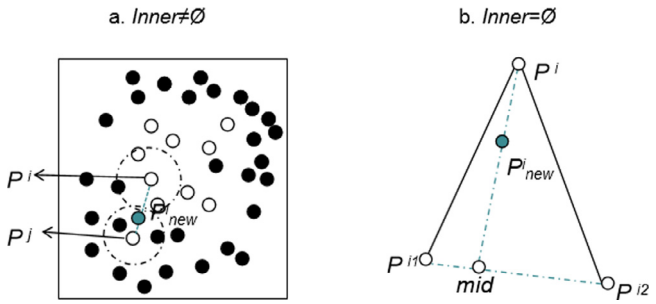

Fig. 2. SMOTE synthesizes new positive data.

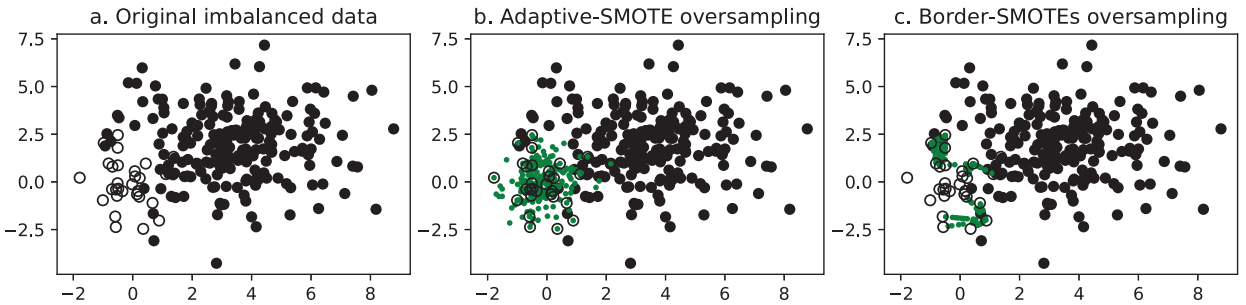

Fig. 3. Oversampling with Border-SMOTEs and Adaptive-SMOTE.

Case 2. If $Inner = \emptyset$, then we randomly select three different samples P^i, P^{i1}, P^{i2} from the *Danger* subset. The SMOTE first synthesizes an intermediate *Midpoint* using P^{i1} and P^{i2} according to Eq. (2.2) firstly and then synthesizes a new positive sample P^i_{new} (shown in Fig. 2b) using *Midpoint* and P^i according to Eq. (2.3):

$$Midpoint = P^{i1} + \delta_1 (P^{i2} - P^{i1}), \tag{2.2}$$

$$P^i_{new} = P^i + \delta_2 (Midpoint - P^i), \tag{2.3}$$

where δ_1 and δ_2 are two random numbers between 0 and 1.

Step 2 is then repeated until $|P^{'}| = |N|$. Fig. 3 shows the oversampling results with Border-SMOTEs and Adaptive-SMOTE. Fig. 3a is the original binary imbalanced dataset in which the white and black circles represent positive and negative data, respectively. As shown in Fig. 3c, the positive data boundary is magnified with Border-SMOTEs after oversampling, and some negative samples are included in the positive domains. Therefore, Border-SMOTEs, on occasion, may sacrifice negative samples for a higher classification accuracy of positive samples. By contrast, Adaptive-SMOTE oversamples using the natural distribution of the positive data and therefore achieves higher classification accuracy and meets human intuition. We intro-

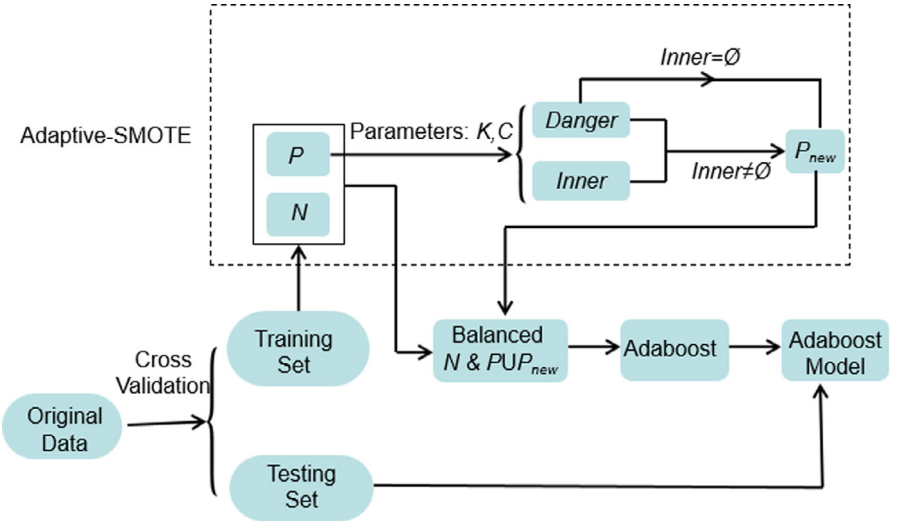

Fig. 4. Flowchart of the Adaptive-SMOTE combined with Adaboost.

duce the Adaptive-SMOTE into Adaboost to improve the test accuracy, and the corresponding flow chart of the framework is shown in Fig. 4.

2.2. Gaussian oversampling

It is crucial to construct a reasonable probability distribution model using the distributional characteristics of the positive dataset and its good properties for the imbalanced problem. The Gaussian (Normal) distribution has been widely used to describe data distributions in statistics and many fields of applied science. The results and methods can be easily derived in an explicit form when the variables are normally distributed. Therefore, the Gaussian distribution model of the positive dataset can be modeled using a probability density function:

$$G(\mathbf{x};\boldsymbol{\Sigma},\mathbf{u}) = \frac{1}{(2\pi)^{m/2}|\boldsymbol{\Sigma}|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\mathbf{u})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\mathbf{u})}, \tag{2.1}$$

where $\mathbf{x} \in \mathbb{R}^M$, $\mathbf{u}$ and $\boldsymbol{\Sigma}$ correspond to the mean and symmetric covariance matrix of the attributes of the positive samples, respectively.

We aim to find a sampling region set $R_{set} = \{R_k | R_k \neq \emptyset\}_{k=1,\ldots,K}$ that is generated by a preset probability set $P_{set} = \{P_k | P_k \in (0,1)\}_{k=1,\ldots,K}$, where $R_i \cap R_j = \emptyset$ if $i \neq j$. Each positive sample is included in one and only one R_k, and the probability over R_k is calculated as

$$\int_{R_k} G(\mathbf{x};\boldsymbol{\Sigma},\mathbf{u})d\mathbf{x} = P_k, k = 1,2,\ldots,K \tag{2.2}$$

but

$$\int_{R_i} d\mathbf{x} \neq \int_{R_j} d\mathbf{x}, i \neq j, 1 \leq i, j \leq K. \tag{2.3}$$

If the number of positive samples that are randomly generated for each R_k is proportional to P_k, then the new positive samples obey the distributional characteristics of the original positive data. For example, the sampling regions ($K = 2$) for a Gaussian distribution are shown in Fig. 5. Typically, if the probabilities of regions $[b, c]$ and $[a, b] \cup [c, d]$ are equal to each other, then random sampling in the two regions does not change the relative density of the data distribution.

Before oversampling, it is necessary to reduce the dimension of the original unbalanced data D. Even if the cardinality of the imbalanced data is not large, high-dimensional features are likely to overwhelm the trainer. Additionally, the interclass data will be more dispersed, and the intraclass data will be more aggregated. For notational convenience, the low-dimensional positive and negative samples after dimension reduction are still respectively denoted by P and N in $\mathbb{R}^m$ space ($m < M$). It can be supposed that the new positive data obey a Gaussian distribution and the Gaussian Oversampling method is performed as follows:

Step 1. The Gaussian distribution model of positive data is formed using two parameters: the mean ($\mathbf{u_P}$) and the covariance matrix ($\boldsymbol{\Sigma_P}$).

Step 2. In this paper, sampling regions are found using preset probabilities. A hierarchical probability set $P_{set} = \{P_k\}_{k=1,2,\ldots,K}$ is fixed firstly. For each P_k, a sampling subregion R_k is found step by step, and the process is shown in Fig. 6.

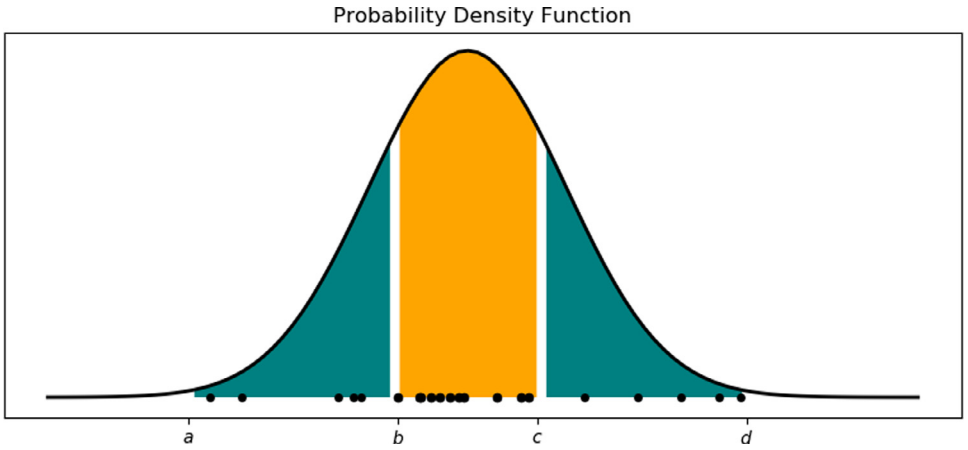

Fig. 5. The corresponding sampling regions given P_{set}.

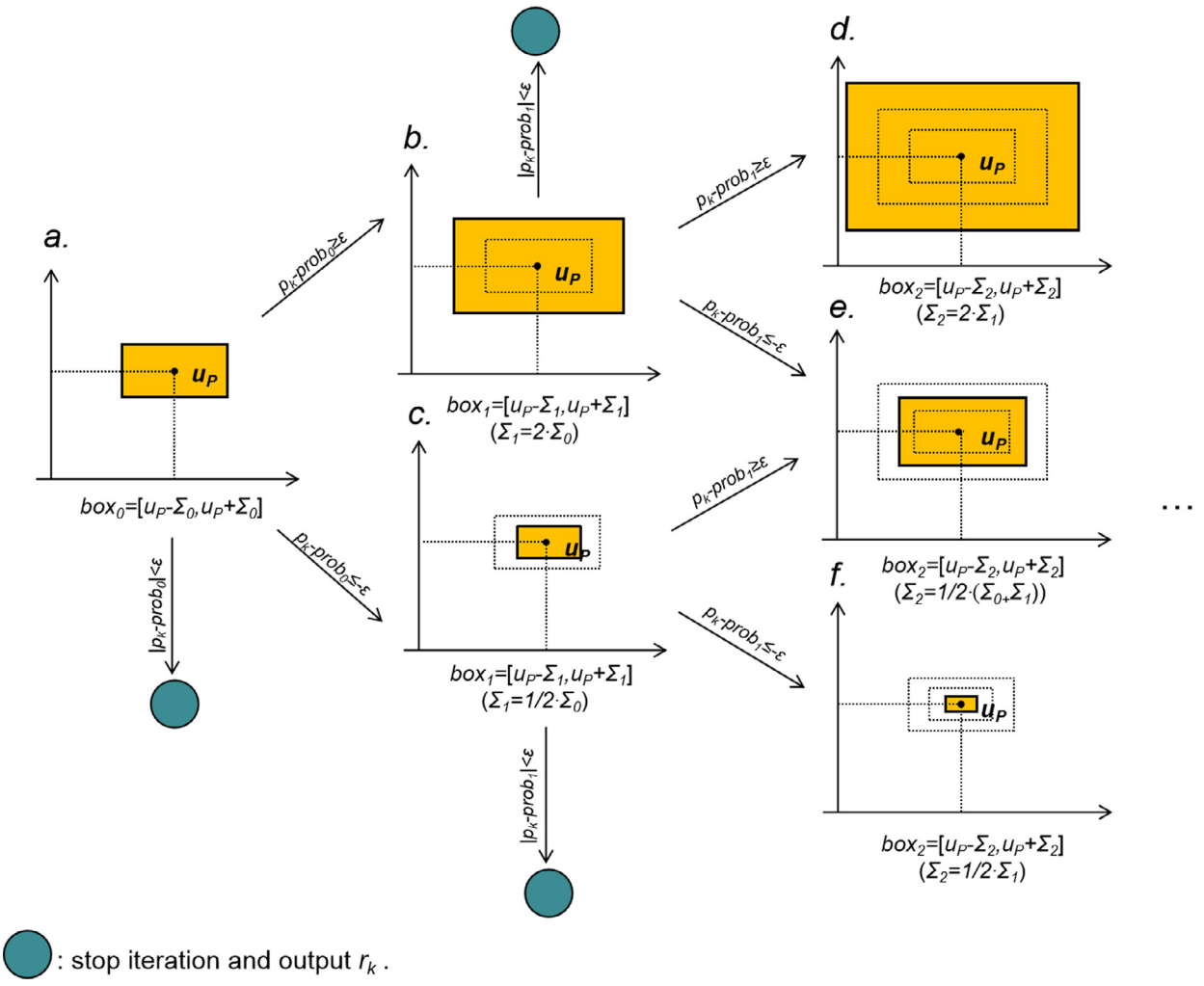

Fig. 6. The main process to find R_k given P_k in two-dimensional space.

Table 1
The fluctuation of box_2.

	$p_k - prob_0 \geq \varepsilon$	$p_k - prob_0 < \varepsilon$
$p_k - prob_1 \geq \varepsilon$	$\mathbf{\Sigma}_2 = \frac{1}{2} \cdot \mathbf{\Sigma}_1$	$\mathbf{\Sigma}_2 = \frac{1}{2} \cdot (\mathbf{\Sigma}_1 + \mathbf{\Sigma}_0)$
$p_k - prob_1 < \varepsilon$	$\mathbf{\Sigma}_2 = \frac{1}{2} \cdot (\mathbf{\Sigma}_1 + \mathbf{\Sigma}_0)$	$\mathbf{\Sigma}_2 = 2 \cdot \mathbf{\Sigma}_1$

For the convenience of description, we take the process of finding R_k in a two-dimensional space as an example. The fluctuation $\mathbf{\Sigma_0}$ is a vector of diagonal elements of matrix $\mathbf{\Sigma_p}$. $[\mathbf{u_p} - \mathbf{\Sigma}_0, \mathbf{u_p} + \mathbf{\Sigma}_0] = \{\mathbf{x} = (x^1, x^2)^T | u_P^i - \Sigma_0^i \leq x^i \leq u_P^i + \Sigma_0^i, i = 1, 2\}$. Let $p_k = \sum_{i=1}^{k} P_i$. Then, the region corresponding to p_k is denoted as $r_k = \bigcup_{i=1}^{k} R_i,\ k = 1, 2, \ldots, K$.

First, we need to find the region r_k, and a relaxed rectangle $box_0 = [\mathbf{u_p} - \mathbf{\Sigma}_0, \mathbf{u_p} + \mathbf{\Sigma}_0]$ is initialized (shown in Fig. 6a), which is a rectangular region centered on $\mathbf{u_p}$, fluctuating with $\mathbf{\Sigma}_0$. The probability is the integral of the Gaussian distribution function over box_0:

$$prob_0 = \int_{box_0} G(\mathbf{x}; \mathbf{\Sigma_p}, \mathbf{u_p}) d\mathbf{x}. \tag{2.4}$$

If the difference between p_k and $prob_0$ is less than a positive real number, i.e., $|p_k - prob_0| < \varepsilon$, then the sampling region r_k is box_0. Otherwise, another relaxed rectangle $box_1 = [\mathbf{u_p} - \mathbf{\Sigma}_1, \mathbf{u_p} + \mathbf{\Sigma}_1]$ is formed to make the probability over r_k more closer to p_k, where

$$\mathbf{\Sigma}_1 = \begin{cases} 2 \cdot \mathbf{\Sigma}_0, & if\ p_k - prob_0 \geq \varepsilon \\ \frac{1}{2} \cdot \mathbf{\Sigma}_0, & if\ p_k - prob_0 < \varepsilon. \end{cases} \tag{2.5}$$

The probability $prob_1$ over box_1 is obtained by a method similar to Eq. (2.4). If $p_k - prob_0 \geq \varepsilon$, then $\mathbf{\Sigma}_1 = 2 \cdot \mathbf{\Sigma}_0$ makes more samples fall into box_1, and the probability over box_1 increases (shown in Fig. 6b). Otherwise, shrinking the sampling region makes the training model close to the natural one (shown in Fig. 6c).

Second, in a similar way, if $|p_k - prob_1| < \varepsilon$, then the sampling region of p_k is $r_k = box_1$. Otherwise, a relaxed rectangle $box_2 = [\mathbf{u_p} - \mathbf{\Sigma}_2, \mathbf{u_p} + \mathbf{\Sigma}_2]$ is used to approach r_k (shown in Fig. 6d, f, and g), where an improved fluctuation $\mathbf{\Sigma}_2$ satisfies Table 1. It should be noted that $\mathbf{\Sigma}_2$ contains the information not only of $\mathbf{\Sigma}_1$ but also of $\mathbf{\Sigma}_0$, which makes box_2 more reasonable. Then, the probability $prob_2$ over box_2 is compared with p_k again. If $|p_k - prob_2| < \varepsilon$, then the sampling region of p_k is $r_k = box_2$. Otherwise, r_k is approximated according to the above method. Since the region of the data distribution is bounded, r_k can be found after a finite number of cycles.

If all regions $\{r_k | k = 1, 2, \ldots, K\}$ are found, then we obtain each sampling region as

$$R_k = \begin{cases} r_k / r_{k-1}, & if\ k \geq 2 \\ r_k, & if\ k = 1. \end{cases} \tag{2.6}$$

Step 3. Randomly sampling the rectangle R_k using the number $(|N| - |P|) \cdot P_k,\ k = 1, 2, \ldots, K$ makes the dataset balanced.

The setting of the parameters P_{set} is an important issue for training the classifier. To achieve a high confidence level from an infinite number of sampling regions, the probability over those sampling regions should be sufficiently large, which makes the sampling model more likely to be representative. Therefore, if the regional division is finer, the probability distribution over the subregions will be closer to the true distribution, and the probability value over those regions will increase. In addition, the sum of the preset probabilities $\sum_{k=1}^{K} P_k$ is not necessarily 1; otherwise, sampling fewer points over a large region is worthless. To quickly find R_{set}, the process of finding r_{k+1} can be initialized using r_k but not box_0 in step 2.

3. Experiments

3.1. Datasets and experimental evaluation

The performance of the two sampling methods is examined using 15 benchmark imbalanced datasets that are available in the KEEL tool [1]. Detailed information on the datasets is shown in Table 2, where **Attr.** represents the number of attributes in the dataset and **Imb.** is the imbalance ratio, i.e., the ratio between negative and positive examples. To explore the robustness of the two proposed sampling methods, the value of **Imb.** ranges from 1.82 to 82.00.

Although *Acc* (shown in Eq. (3.1)) [12] can be used to evaluate the performance of the classification for balanced datasets, it is no longer a proper measure, since it does not distinguish between the number of correctly classified examples of positive or negative classes. Therefore, the *F*-measure [12] is used to complement the evaluation criteria. As illustrated in Table 3, *F*-measure and *Acc* are defined using a confusion matrix. Here, *FN* is the number of positive samples that are predicted to be negative, and the rest can be understood similarly. The *Acc* of the classifier is defined as follows:

$$Acc = \frac{TP + TN}{TP + FN + FP + TN}. \tag{3.1}$$

Table 2
Characteristics of the datasets that are used in the experiments.

ID	Dataset	$\|D\|$	$\|P\|$	$\|N\|$	Attr.	Imb.
1	*glass1*	214	76	138	9	1.82
2	*vehicle1*	846	217	629	18	2.90
3	*vehicle3*	846	212	634	18	2.99
4	*ecoli1*	336	77	259	7	3.36
5	*new-thyroid1*	215	35	180	5	5.14
6	*newthyroid2*	215	35	180	5	5.14
7	*shuttle-c0-vs-c4*	1829	123	1706	9	13.87
8	*glass4*	214	13	201	9	15.46
9	*yeast-2-vs-8*	482	20	462	8	23.10
10	*flare-F*	1066	43	1023	11	23.79
11	*abalone-3-vs-11*	502	15	487	8	32.47
12	*yeast5*	1484	44	1440	8	32.73
13	*ecoli-0-1-3-7-vs-2-6*	281	7	274	7	39.14
14	*shuttle-2-vs-5*	3316	49	3267	9	66.67
15	*poker-8-9-vs-5*	2075	25	2050	10	82.00

Table 3
Confusion matrix.

	Predicted Positive	Predicted Negative
Actual Positive	*TP*	*FN*
Actual Negative	*FP*	*TN*

The *F*-measure measures both the precision (P_m) and the recall (R_m) as

$$F - measure = 2 \cdot \frac{P_m \times R_m}{P_m + R_m}, \tag{3.2}$$

where $P_m = \frac{TP}{TP+FP}$ and $R_m = \frac{TP}{TP+FN}$.

Furthermore, the *ROC* [12] curves show the relationship between $TPR = \frac{TP}{TP+FN}$ and $FPR = \frac{FP}{FP+TN}$ with different classification thresholds. The *ROC* curve gives us a qualitative measure where the closer the curve is to the upper left corner, the better the model built for the unbalanced data. The *AUC*, the area under the *ROC* curve, gives us a quantitative measure of the imbalanced techniques.

3.2. Experimental methods

Five sampling methods (Non-Sampling, SMOTE, Borderline-SMOTE [13], Adaptive-SMOTE, and Gaussian Oversampling) are combined with a common classifier to verify the effectiveness of the two proposed oversampling methods. In addition, each sampling method is combined with three usual classifiers (SVM, Adaboost [22] and RandomForest [19]) to explore their performance and robustness. The details are shown as follows.

- *Non − Sampling + SVM* (***SVM***): SVM method that trains by directly using the original imbalanced samples *P* and *N*.
- *SMOTE + SVM* (***SS***): A modified SVM, where the SMOTE balances *P* and *N* before classification.
- *Borderline − SMOTE + SVM* (***BSS***): An extension of the SVM, where the Borderline-SMOTE oversamples on *P* and *N* before classification.
- *Adaptive − SMOTE + SVM* (***ASS***): A modified SVM, where the Adaptive-SMOTE balances *P* and *N* before classification.
- *Gaussian Oversampling + SVM* (***GOS***): An improved SVM, in which Gaussian Oversampling balances *P* and *N* before classification.
- *Non − Sampling + Adaboost* (***Ada***): An ensemble classifier that trains using the entire data (*P* and *N*).
- *SMOTE + Adaboost* (***SA***): An extension of Adaboost, where the SMOTE synthesizes new positive examples before classification.
- *Random − SMOTE + Adaboost* (***RSA***): An improved Adaboost, where the Random-SMOTE oversamples on *P* and *N* and Adaboost trains a classifier based on balanced samples.
- *Borderline − SMOTE + Adaboost* (***BSA***): An extension of the Adaboost approach that uses the Borderline-SMOTE to balance *P* and *N* before the Adaboost trains using balanced samples.
- *Gaussian Oversampling + Adaboost* (***GOA***): A modified Adaboost method, where Gaussian Oversampling balances *P* and *N* before training a classifier.
- *Non − Sampling + RandomForest* (***RF***): An ensemble classifier that trains by directly using *P* and *N*.
- *SMOTE + RandomForest* (***SRF***): A modified RandomForest, where the SMOTE balances *P* and *N* before classification.
- *Borderline − SMOTE +RandomForest* (***BSRF***): An improved version of RandomForest, where Borderline-SMOTE oversamples on *P* and *N* before classification.

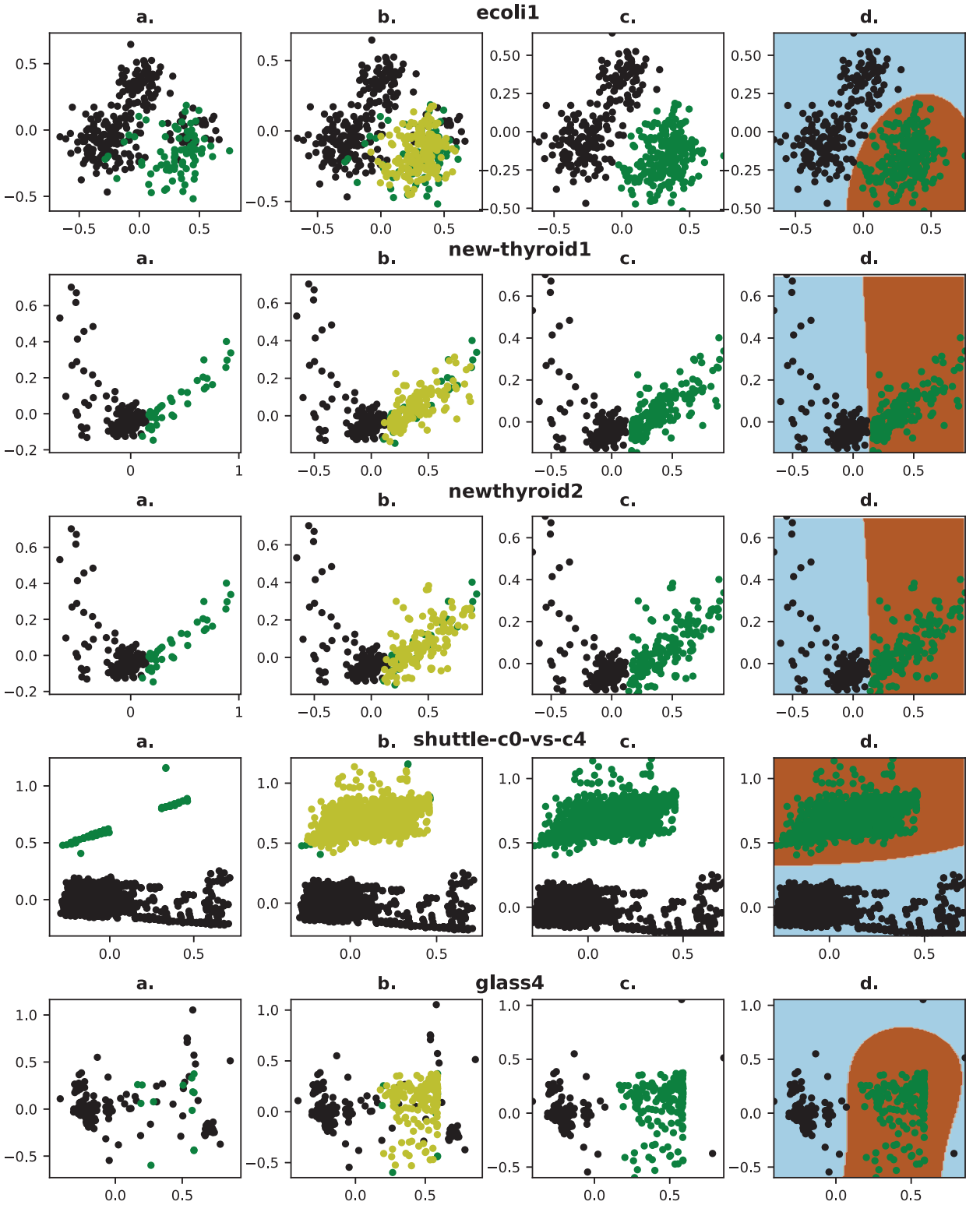

Fig. 7. Building the classification models if the datasets are balanced using Adaptive-SMOTE in 2D space.

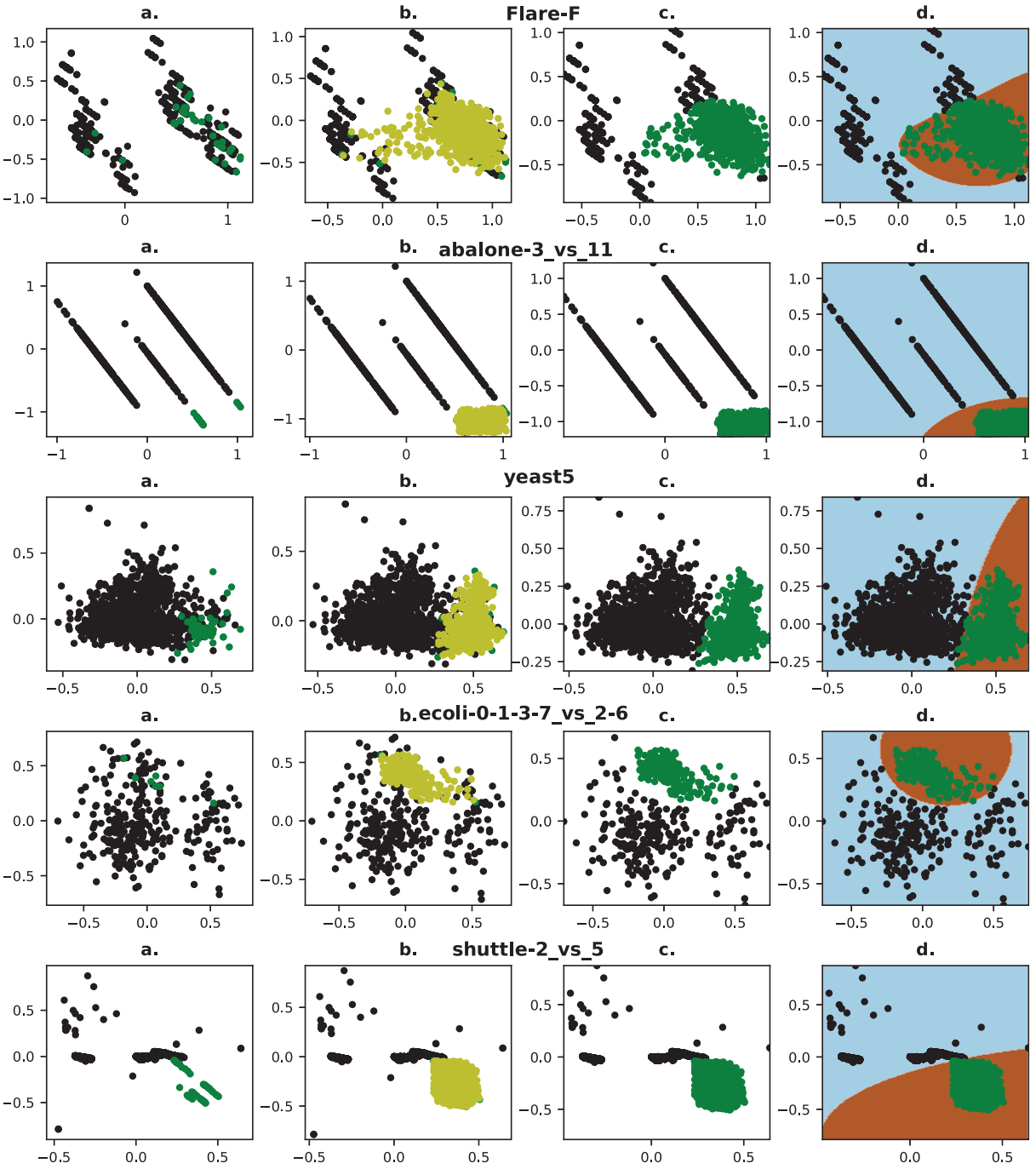

Fig. 8. Building the classification models if the datasets are balanced using Adaptive-SMOTE in 2D space.

- *Adaptive − SMOTE + RandomForest* (***ASRF***): An extension of the RandomForest approach that uses Adaptive-SMOTE to balance P and N before the RandomForest trains using the balanced samples.
- *Gaussian Oversampling + RandomForest* (***GORF***): A modified RandomForest method, where Gaussian Oversampling balances P and N before training the classifier.

The kernel function of the ***SVM*** is linear. A Decision Tree is used to train weak classifiers in ***Ada*** and ***RF***. In Adaptive - SMOTE, the constants are defined as $K = C = 5$. In Gaussian Oversampling, $P_{set} = \{P_k | P_k = 0.015, k = 1, 2, \ldots, 66\}$. During the

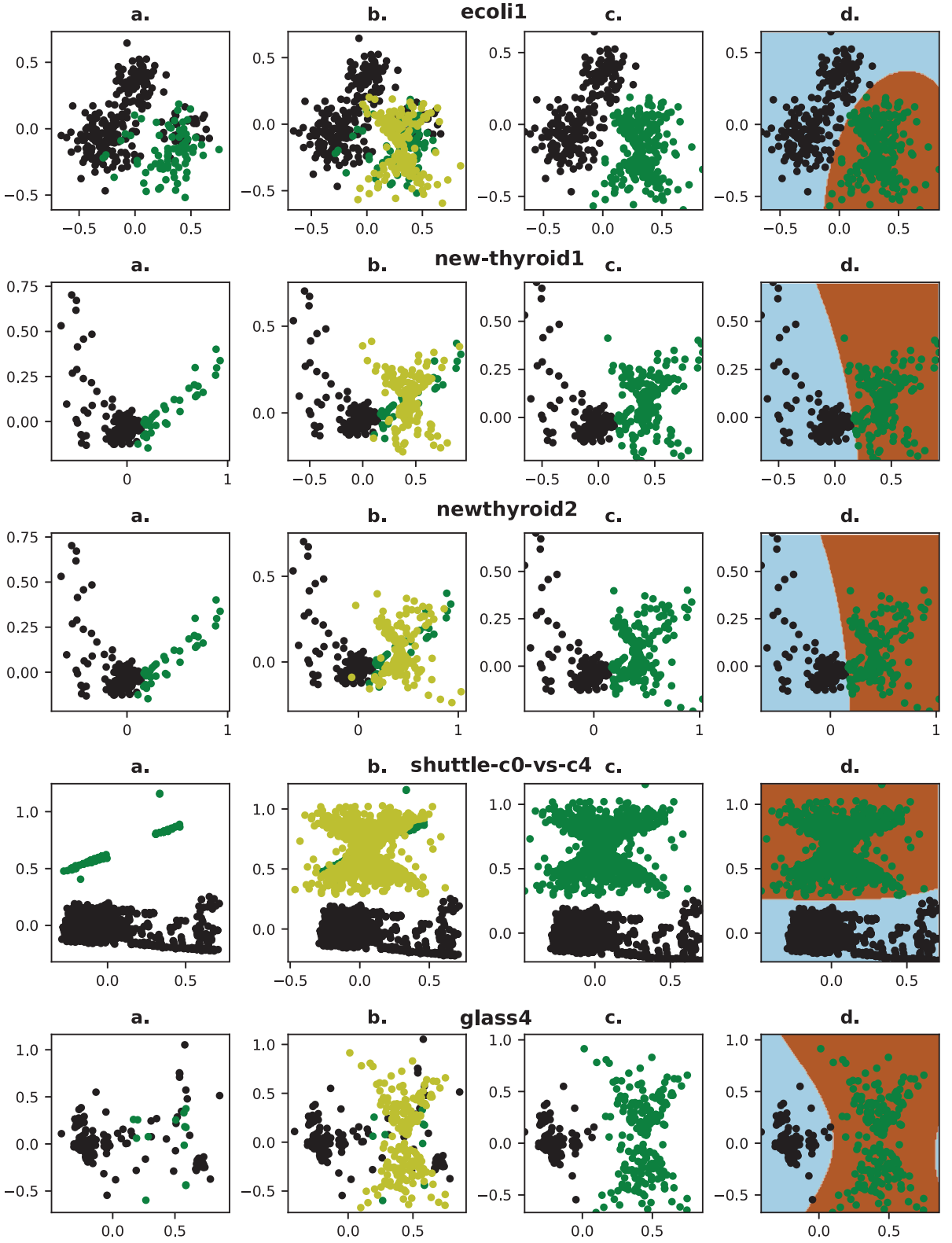

Fig. 9. Building the classification models if the datasets are balanced with Gaussian Oversampling in 2D space.

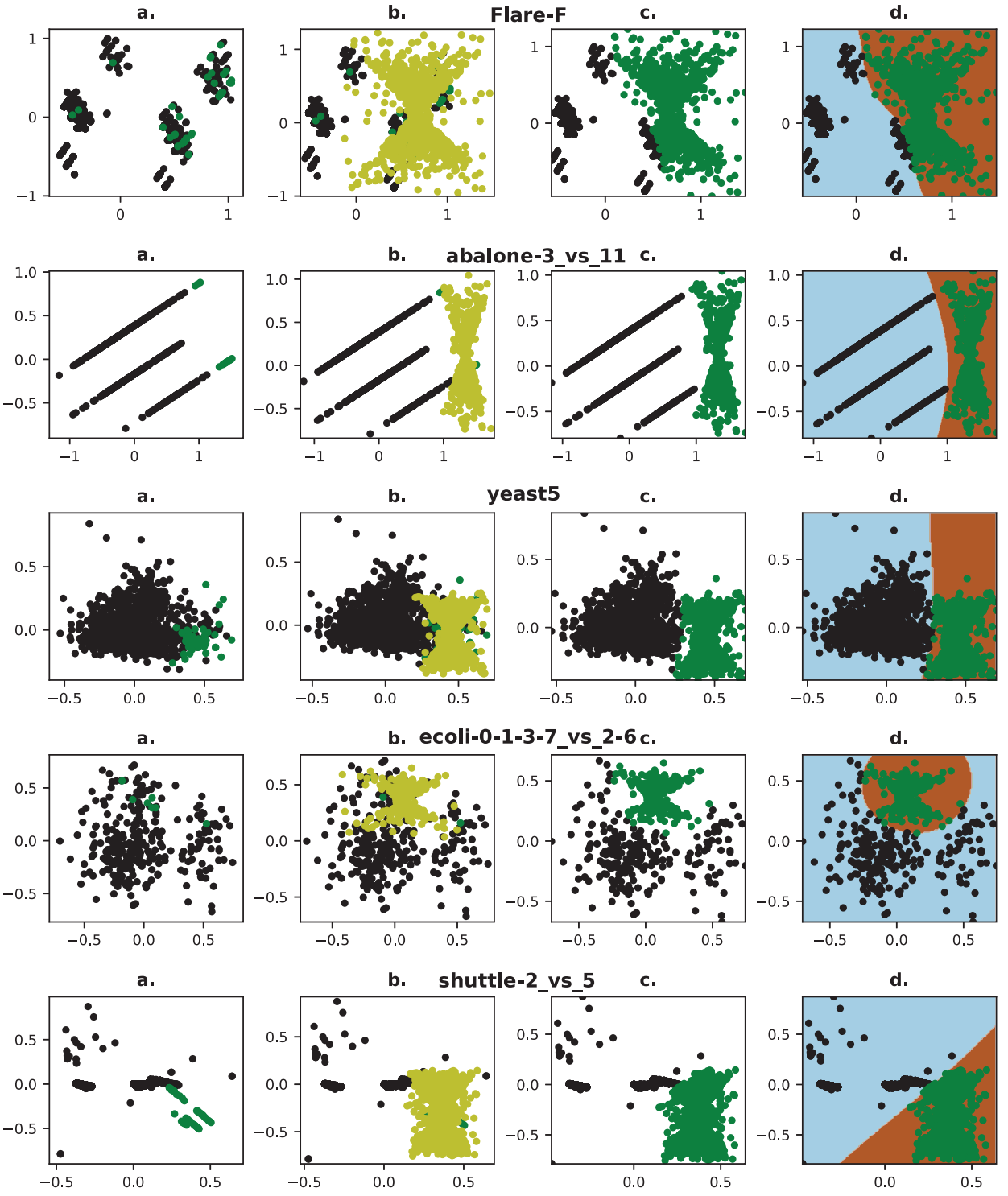

Fig. 10. Building the classification models if datasets are balanced with Gaussian Oversampling in 2D space.

process of finding rectangles, $\varepsilon = 0.005$, and principal component analysis (PCA) [15] is used to reduce the dimensionality if the feature dimension is more than 3 before oversampling.

In these experiments, the performance of each method is based on the average and standard deviation of 5 runs, and in each run, a 5-fold cross-validation technique is used. More precisely, the training dataset is randomly partitioned into 5 subsamples of equal size. In each round, 1 subsample out of 5 is used for testing. The accuracy rates from the 5 testing

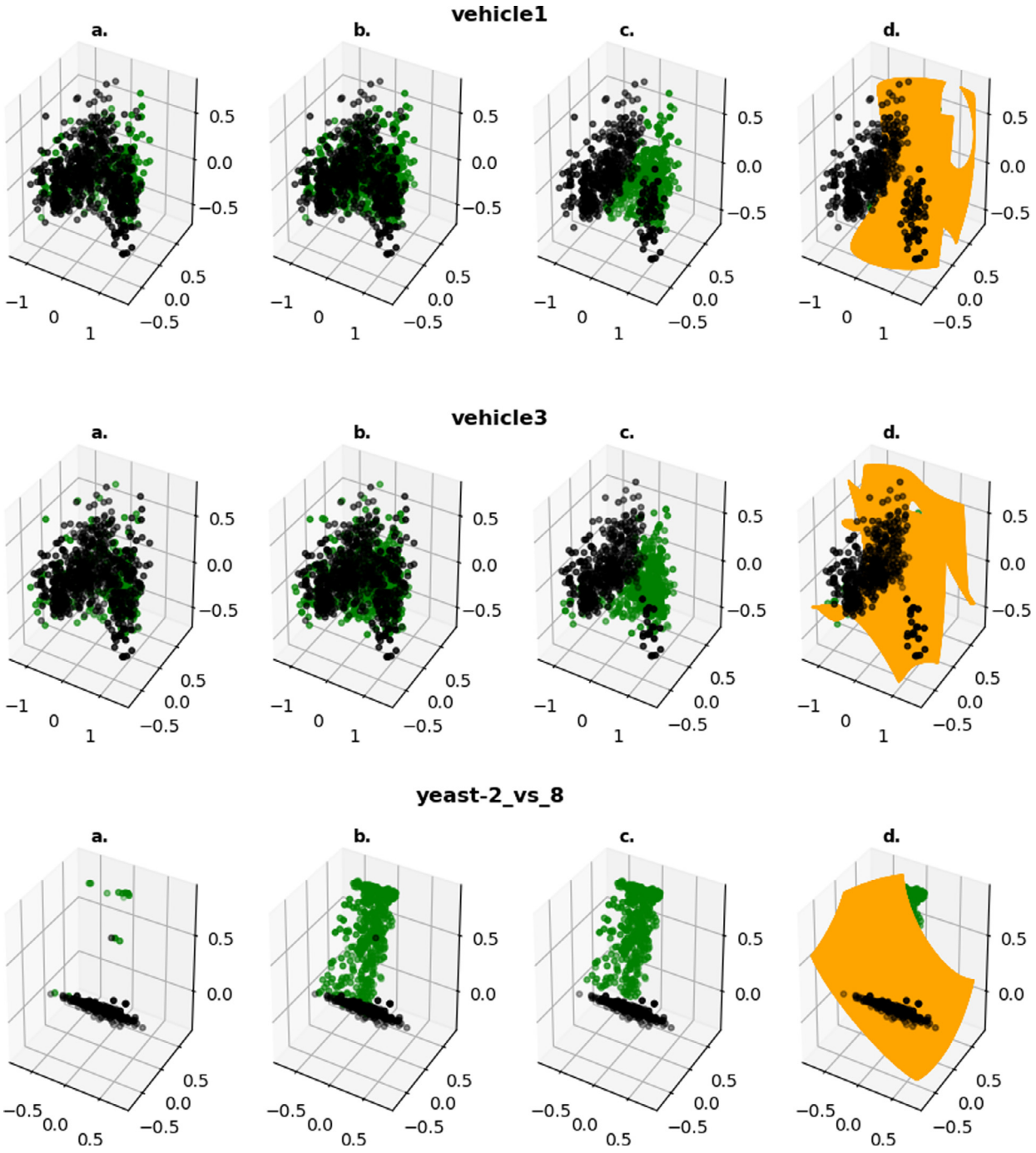

Fig. 11. Building the classification models if datasets are balanced using Adaptive-SMOTE in 3D space.

datasets are averaged in each run. All experiments are conducted using Python 3.6 software and run on a computer with an i7-7700 CPU and 32 GB of RAM.

3.3. Experimental results and analysis

3.3.1. Visualization of the proposed sampling methods

To clearly describe the proposed sampling methods, the four steps of operations are performed over datasets a, b, c, and d in Figs. 7–11, where the black scatters are negative, the green scatters are positive, and the yellow scatters are new synthetic positive samples.

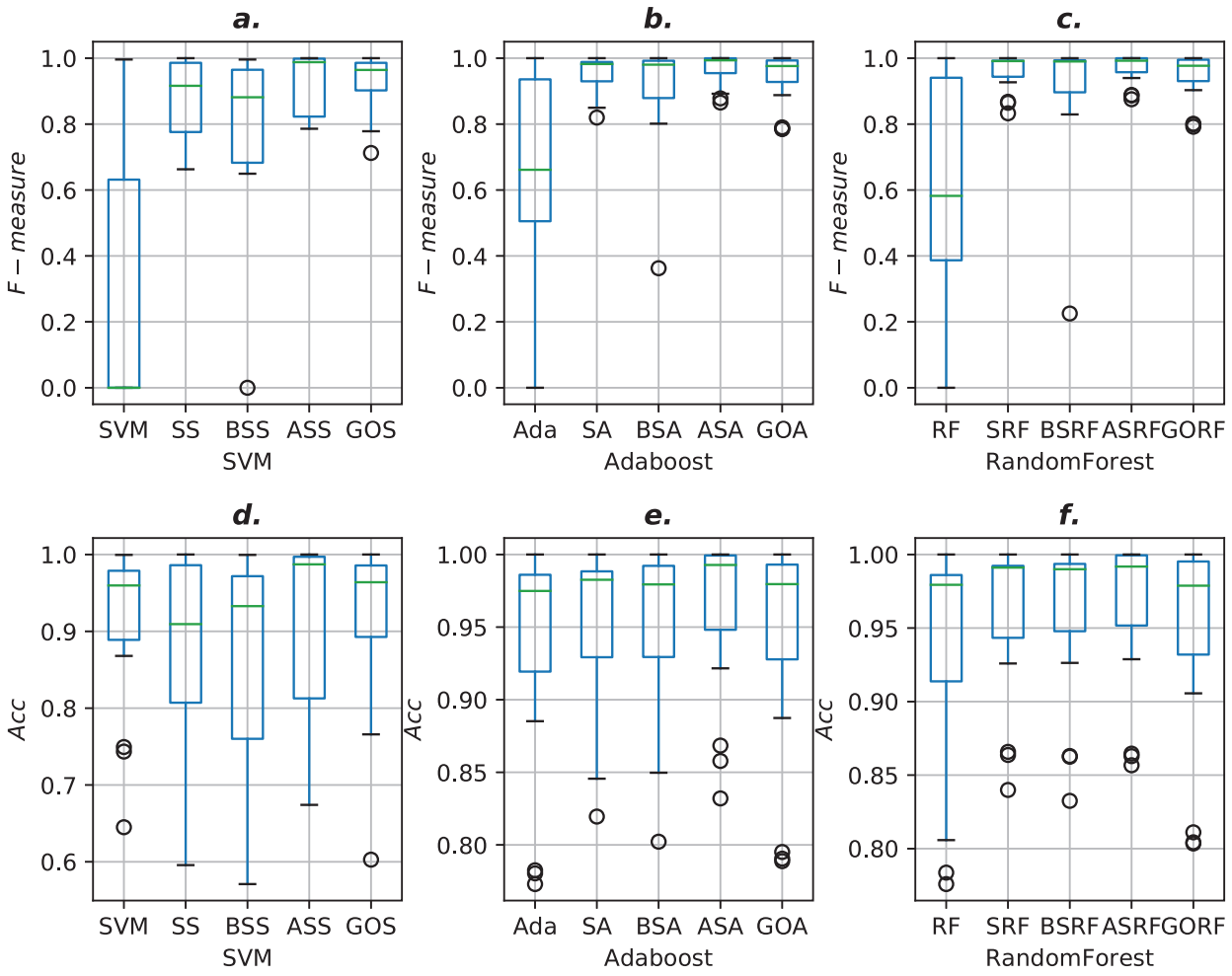

Fig. 12. Boxplot for the results of the experiment on the set of benchmarks according to the *F*-measure and *Acc* criterion.

a. Mapping the datasets into a low-dimensional space: This mapping clearly and directly illustrates the original distribution characteristics of the datasets.
b. Balanced datasets: Most of the new positive points retain the nature of figure a. few samples overlap, which does not influence the overall distribution.
c. Denoise: The noisy points are the points that are close to the opposite class. We remove the points that have less of an effect on the boundary and make the boundary clearer.
d. Show the boundary: The SVM with the Gaussian kernel is used as a classifier.

Figs. 7–10 show the construction of the classification models for the 10 datasets from Table 2 when the sampling method is Adaptive-SMOTE (and Gaussian Oversampling). In the case of Adaptive-SMOTE sampling (Figs. 7–8), the new positive samples (Figs. 7–8b.) and the original positive samples (Figs. 7–8a.) have the same distribution areas and visual density, especially for *ecoli1, new-thyroid1, newthyroid2, glass4, Flare-F, yeast5*, and *ecoli-0-1-3-7-vs-2-6*. However, the new positive samples of *shuttle-c0-vs-c4, abalone-3-vs-11*, and *shuttle-2-vs-5* do not have the density characteristics of the original datasets, but are located in the original distribution areas, as the banded distribution reveals the datasets' distribution areas only. Furthermore, after denoising (Figs. 7–8c.) all boundaries (Figs. 7–8b.) are reasonable visually, referring to Figs. 7–8a. In the case of Gaussian Oversampling (Figs. 9–10), starting from the center (mean) of the original positive samples, the newly generated positive samples spread out in a fan shape, and the farther away from the center, the more sparse of the data, which reflects the distribution of the original positive samples to some extent. Overall, all boundaries satisfy our expectations after denoising.

In addition, there must be a high-dimensional space that maps different classes of samples to different regions. Therefore, we can always find a proper map for the other 5 datasets. Taking *vehicle1, vehicle3*, and *yeast-2-vs-8* as examples, they are mapped to 3D space, as shown in Fig. 11.

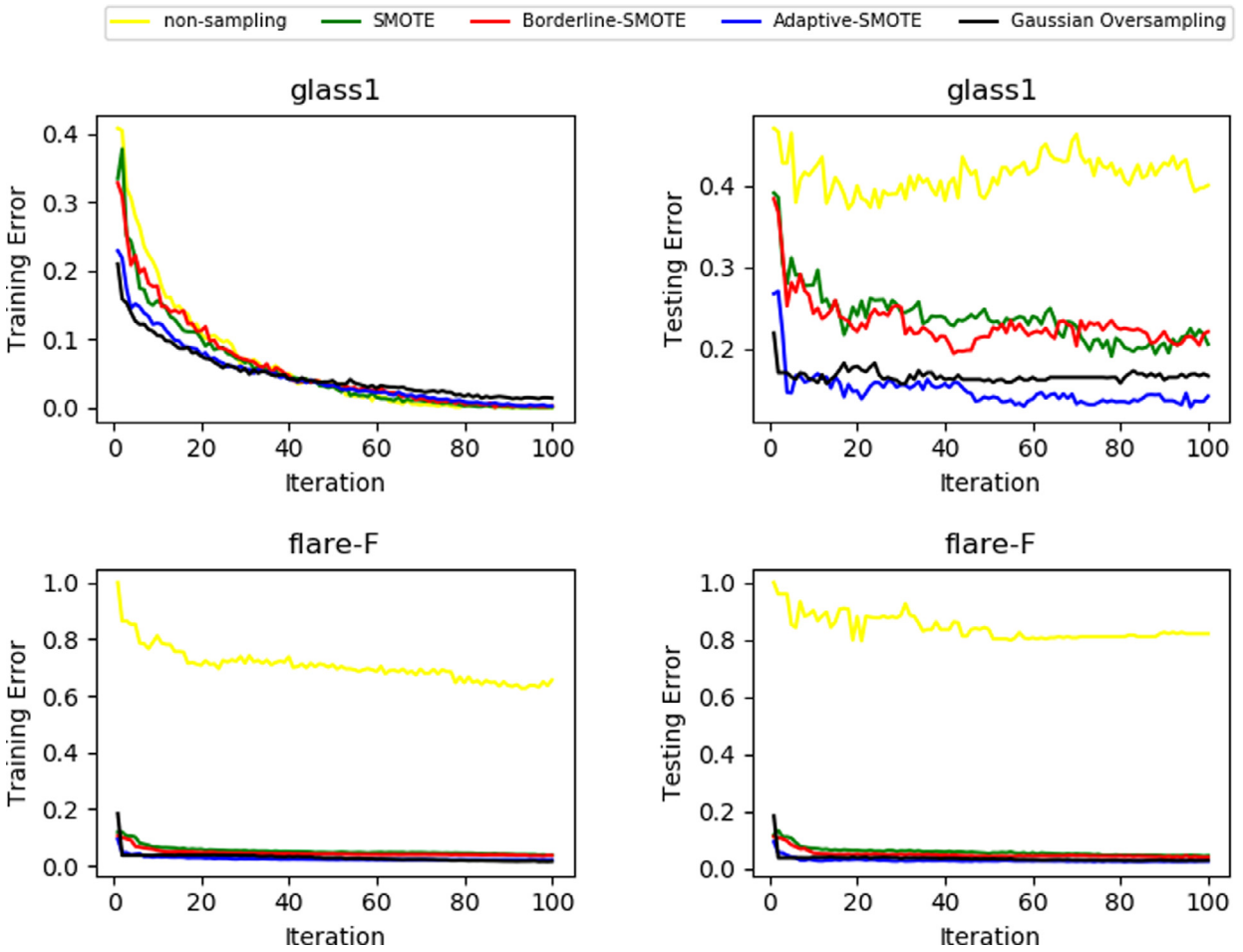

Fig. 13. The training and testing error of *F*-measure if the classifier is Adaboost.

3.3.2. Performance of imbalanced datasets

The average $F-measures$ of the 3 classifiers with the 5 different sampling methods are summarized in Table 5. The best performance of the sampling methods with a fixed classifier is in bold. For example, SVM performs better when using Adaptive-SMOTE or Gaussian Oversampling than with other sampling methods for each dataset. The same is true for most datasets when the Adaboost and RandomForest classifiers are used, except for the 6th dataset, where the best value is 0.9933, which is close to the 0.9928 of Gaussian Oversampling. In addition, to assess the robustness of the sampling methods, Table 6 reports the average standard deviation of the *F*-measure. Because there is no random number in the Non-Sampling method, the average standard deviations of the ***SVM, Ada***, and ***RF*** columns are replaced with "–". As shown in the table, Adaptive-SMOTE or Gaussian Oversampling achieves the lowest standard deviations for most datasets, except for only two cases combined with different classifiers. The reasons for the bad performance are due to the classifier and the datasets. For example, when combined with the SVM, the *ecoli-0-1-3-7-vs-2-6* dataset, which is balanced by the proposed sampling methods, has unstable *Acc* performance. First, unlike the ensemble classifiers (Adaboost and RandomForest), SVM is somewhat arbitrary. Second, compared with the other sampling methods shown in Fig. 8, the denoising of *ecoli-0-1-3-7-vs-2-6* sacrifices more negative samples, which does not show the characteristics of Adaptive-SMOTE very well. In addition, *ecoli-0-1-3-7-vs-2-6* in Fig. 10 disobeys a normal distribution, which is the condition of Gaussian Oversampling.

Similarly, the average *Accs* of the 3 classifiers with the 5 different sampling methods are reported in Table 7. Although there are 4 best results that are not achieved by Adaptive-SMOTE or Gaussian Oversampling, the results that are achieved by them are not the worst. For example, the RandomForest classifier obtains an average *Acc* of 0.8658 for the second dataset with ***SMRF***, but the value of 0.8567 that is produced using ***ASRF*** is still higher than the lowest value of 0.7759. As mentioned before, Borderline-SMOTE sacrifices some negative samples for a higher classification accuracy of positive samples, but Adaptive-SMOTE and Gaussian Oversampling follow the natural distribution of positive samples. Therefore, most values of the ***BSS, BSA***, and ***BSRF*** columns are lower than those of the ***ASS, GOS, ASA, GOA, ASRF***, and ***GORF*** columns. In addition,

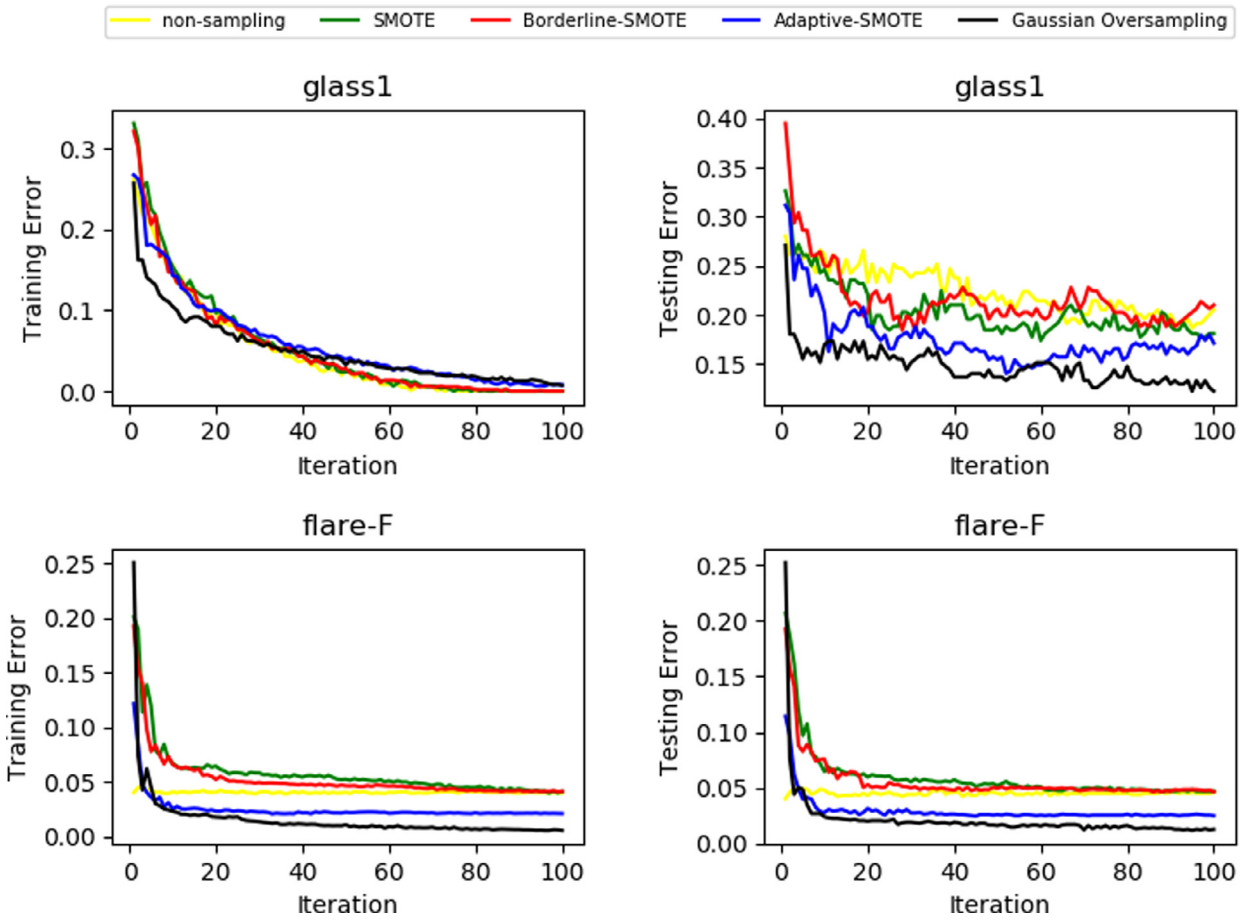

Fig. 14. The training and testing error of *Acc* if the classifier is Adaboost.

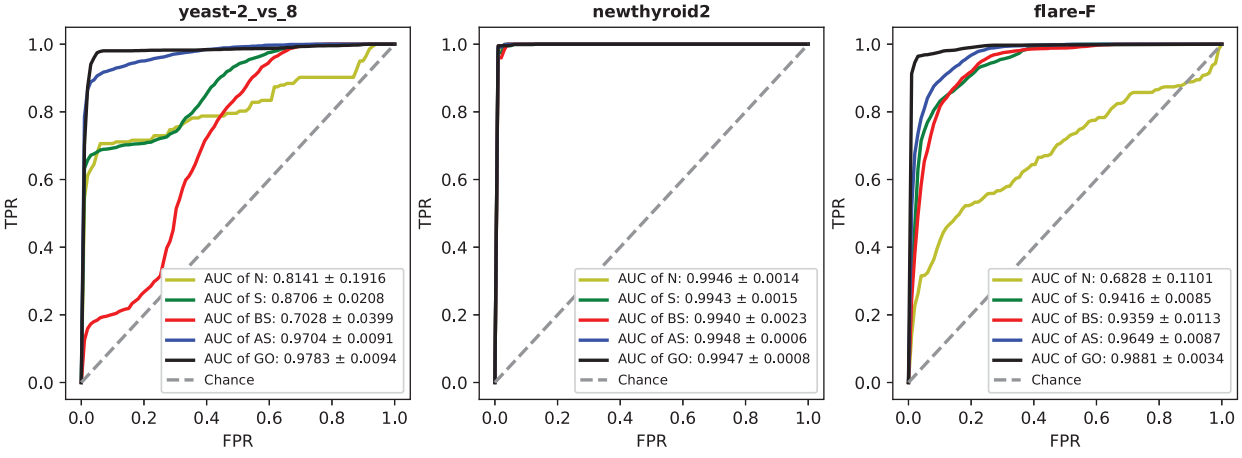

Fig. 15. Average *ROC* curves showing the effectiveness of the proposed sampling methods.

Adaptive-SMOTE and Gaussian Oversampling perform better than Non-Sampling with respect to the *Acc* on most datasets, which indicates that the two sampling methods improve the classification precisions of positive datasets and whole datasets. As before, Table 8 shows the average *Acc* standard deviation. Adaptive-SMOTE and Gaussian Oversampling achieve the lowest standard deviations in most cases, which indicate that the classifiers with these two sampling methods are robust.

In addition, the above experimental results (Tables 5–8) show that were appropriately classified by Gaussian Oversampling generally have a large number of positive samples ($|P| > 30$), which meets the conditions of the Central Limit Theo-

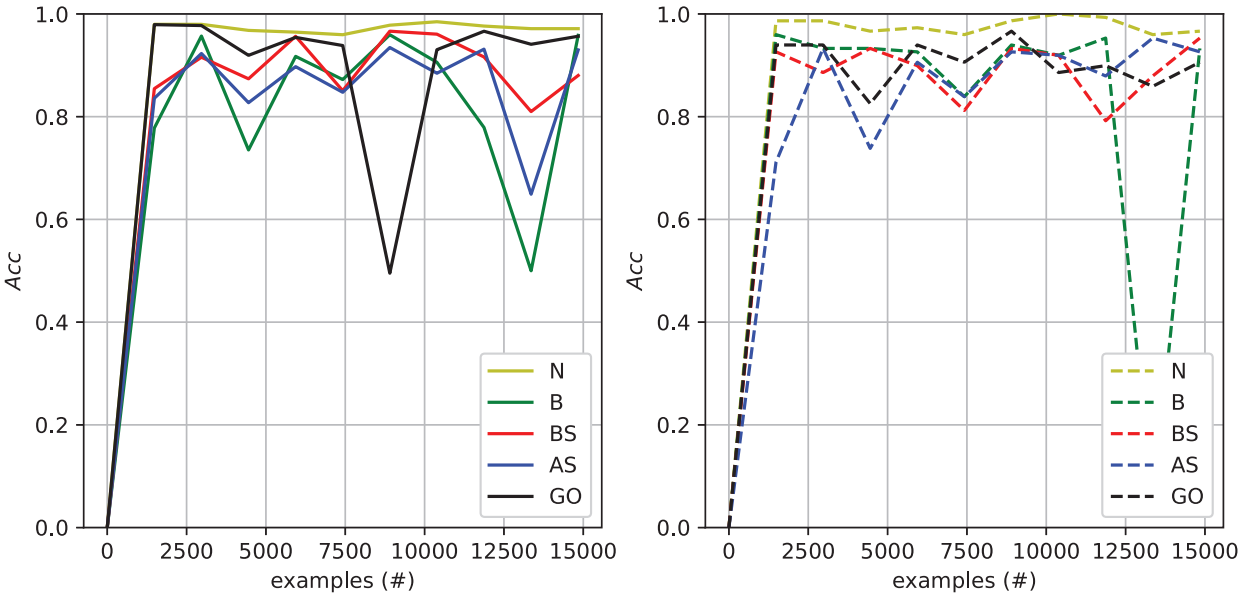

Fig. 16. Real-time study for *ecoli2*.

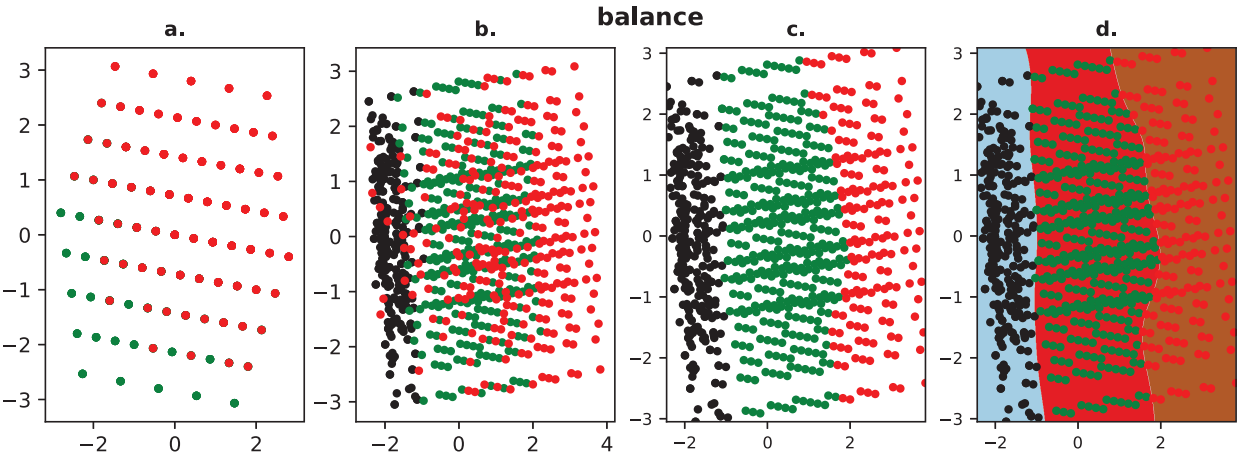

Fig. 17. Adaptive-SMOTE performance for the multiclass problem in 2D space.

rem [3]. However, Adaptive-SMOTE is unconstrained by this condition. Therefore, if the dataset includes a mass of positive samples, then both Gaussian Oversampling and Adaptive-SMOTE can be applied. Otherwise, Adaptive-SMOTE is the better choice.

To obtain better insights into the results mentioned above, Fig. 12 shows the boxplots of the *F*-measure and *Acc* for the 15 methods. Most boxplots show that Adaptive-SMOTE outperforms the other three sampling methods (Non-Sampling, SMOTE, and Borderline-SMOTE), except in Fig. 12d. Here, although the *Acc* of ***ASS*** is scattered, the upper quartile and the middle are considerable. All oversampling methods do not perform better than Non-Sampling with respect to the *Acc* only if the classifier is SVM. First, without the self-correcting (Adaboost) and majority voting (RandomForest) mechanisms, SVM is more arbitrary for balanced datasets. Second, compared with balanced datasets, imbalanced datasets have more advantages with respect to the *Acc*, which measures the overall accuracy, especially for extremely unbalanced datasets. Similarly, Gaussian Oversampling achieves good performances with respect to the *F*-measure (shown in Fig. 12a–c), but it is related to the classifier selection under *Acc*. For example, Gaussian Oversampling achieves a representative result if the classifier is SVM (Fig. 12d). However, Gaussian Oversampling obtains slightly lower values of *Acc*s if the classifiers are Adaboost and RandomForest (Fig. 12e and f). In summary, Adaptive-SMOTE is more precise (*F*-measure and *Acc*) compared with Gaussian Oversampling method and is more robust to the classifiers and datasets.

To better understand the convergence of the iterative process, Figs. 13 and 14 present the training and testing errors of the *F*-measure and *Acc* for the 5 sampling methods when the classifier is Adaboost. The vertical and horizontal axes

Table 4
The classification report based on the metrics that were used with imbalanced datasets.

class	Non-Sampling							Sampling						
	P_m	R_m	S_m	*F*-measure	*G*-mean	iba.	num.	P_m	R_m	S_m	*F*-measure	*G*-mean	iba.	num.
0	0.00	0.00	1.00	0.00	0.00	0.00	49	0.98	1.00	0.99	0.99	0.99	0.99	288
1	0.92	1.00	0.93	0.96	0.96	0.94	288	0.84	0.99	0.91	0.91	0.95	0.91	288
2	0.92	1.00	0.93	0.96	0.96	0.93	288	0.99	0.79	1.00	0.88	0.89	0.77	288
avg.	0.85	0.92	0.93	0.88	0.89	0.86	625	0.94	0.93	0.96	0.93	0.94	0.89	864

show the error and the number of iterations, respectively. It can be seen from the two upper subfigures of Fig. 13 that the initial *F*-measure training errors of the two proposed methods for the *glass1* dataset are lower than the others, but all training errors decrease to the same value as the number of iterations increases, which means that all classification models can appropriately fit the training datasets after going though iterations. However, the testing errors of Adaptive-SMOTE and Gaussian Oversampling always remain lower than those of the other methods. For the *flare-F* dataset, all training and testing errors converge to low values if the sampling is applied, but the non-sampling errors remain high. The same is true when the evaluation criterion is the *Acc*. It should be noted that the results of Gaussian Oversampling are better than those of Adaptive-SMOTE when the number of positive samples in the *glass1* and *flare-F* datasets is 76 and 43, respectively, which are larger than 30 and can be viewed as big-sample datasets.

Moreover, as an important judgment for imbalanced techniques, the *ROC* is used to illustrate the effectiveness of the two proposed sampling methods. As shown in Fig. 15, the three datasets, *yeast-2-vs-8, newthyroid2*, and *flare-F*, from Table 2 are balanced with five sampling methods. The classifier used in this experiment is SVM, where $gamma = auto$. In the legend, N, S, BS, AS, and GO are abbreviations of Non-Sampling, SMOTE, Borderline-SMOTE, Adaptive-SMOTE, and Gaussian Oversampling, respectively. Each value shown in figure is $AUC \pm \delta$, where δ is the standard deviation of the *AUC* showing the fluctuation of the sampling methods. As shown in the figure, the *ROC* curves of AS and GO are the closest to the upper left, and their *AUC* values are the highest, especially for *yeast-2-vs-8* and *flare-F*. In addition, the sampling methods with the smallest standard deviations are AS and GO, which reveals that the proposed sampling methods are more stable compared with the other sampling methods.

To investigate the application of the proposed sampling methods using actual data, we employed the *ecoli2* dataset consisting of 336 samples, 7 features, 52 positive samples, and 284 negative samples. Online or real-time training was conducted, the models were updated based on old models without retraining, and this process was repeated. *ecoli2* was repeatedly expanded by fifty times and divided into ten successive inputs, so the mini-batch size was 1680. Each batch was divided into a training set and testing set at a ratio of 4:1. The *SGDClassifier* was used for training, and $partial\_fit$ was used to fit the training set and testing set based on the old model, i.e., new data were fed into old models, and this process updated the models. The results for the *Acc* are shown in Fig. 16. The left figure (and the right figure) has five solid lines (five dotted lines) corresponding to the five sampling methods, which is the training process (testing process). As shown in the two figures, Non-Sampling has the best *Acc* for both training and testing. In addition, Adaptive-SMOTE and Gaussian Oversampling obtain the best *Acc*s except for Non-sampling, which could be understandable. Based on the results for *Acc*, unbalanced datasets have more advantages compared with balanced datasets, as *Acc* measures the overall accuracy.

3.3.3. Multiclass problem

In summary, Adaptive-SMOTE possesses good properties for the binary classification of imbalanced datasets. Next, we study the performance of Adaptive-SMOTE for multiclass problems. The *balance* multiclass dataset comes from KEEL. In this dataset, there are 625 samples belonging to 3 classes, and each sample has 3 features. The default classifier is the SVM with the Gaussian kernel in the experiments. Table. 4 presents some imbalanced classification metrics used to determine the performance of the proposed sampling method. P_m and R_m were introduced in Section 3.1, $S_m = \frac{TN}{TN+FP}$, *G*-mean [8] is the root of the product of the classwise R_ms; and we use **iba.** to balance any scoring functions. Obviously, all metrics are measured in ranges of [0,1], and the better the model is, the closer to 1 the metrics will be.

As shown in Table 4, after sampling, the dataset is balanced and the class number varies from {0: 49; 1: 288; 2: 288} to {0: 288; 1: 288; 2: 288}. Compared with Non-Sampling, the metrics P_m and S_m of class 0 are significantly improved while R_m slightly decreases, since the oversampling process weakens the expansion of the positive boundary and enhances the distributional characteristics of the positive samples. In addition, *F*-measure, $G-means$, and **iba.** of class 0 are greatly improved after sampling. Most of the performances for class 1 and class 2 worsen after sampling, but each of the average **avg.** metrics for sampling is better than those of Non-Sampling, which illustrates that Adaptive-SMOTE contributes to the good performance of multiclass problems overall. In addition, to better understand the process of building a model, sampling and classification are visualized in 2D space (Fig. 17), which is similar to the process of Figs. 7–11. However, it is in higher-dimensional space instead of in 2D space that balanced datasets cooperate and are mapped into 2D space, as shown in Fig. 17b.

Table 5
Average *F*-measure value of all datasets with different methods. The best results are in bold.

ID	SVM					Adaboost					RandomForest				
	SVM	*SMS*	*BSS*	*ASS*	*GOS*	*Ada*	*SMA*	*BSA*	*ASA*	*GOA*	*RF*	*SMRF*	*BSRF*	*ASRF*	*GORF*
1	0.0000	0.6775	0.6495	**0.7885**	0.7124	0.6614	0.8198	0.8013	**0.8654**	0.7857	0.6793	0.8328	0.8293	**0.8868**	0.8017
2	0.0000	0.6629	0.6698	0.7859	**0.7993**	0.5335	0.8496	0.8553	**0.8780**	0.7900	0.4687	0.8677	0.8658	**0.8754**	0.7920
3	0.0000	0.6769	0.6873	**0.8177**	0.7783	0.5200	0.8503	0.8530	**0.8921**	0.7849	0.4389	0.8646	0.8639	**0.8885**	0.7992
4	0.6821	0.8878	0.9021	**0.9329**	0.9080	0.7510	0.9091	0.9027	**0.9340**	0.8879	0.7165	0.9269	0.9271	**0.9398**	0.9030
5	0.5889	0.9876	0.9712	**0.9920**	0.9709	0.9526	0.9892	0.9911	**0.9935**	0.9932	0.9351	0.9913	0.9931	0.9923	**0.9951**
6	0.6179	0.9842	0.9658	**0.9879**	0.9713	0.9190	0.9860	**0.9933**	0.9892	0.9928	0.9461	0.9913	0.9938	0.9922	**0.9955**
7	0.9960	0.9998	0.9951	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	1**1.0000**	**1.0000**	0.9991	**1.0000**	**1.0000**
8	0.0000	0.9162	0.9361	**0.9935**	0.8979	0.5961	0.9823	0.9841	**0.9984**	0.9678	0.3347	0.9921	0.9900	**0.9988**	0.9578
9	0.6451	0.6945	0.6781	0.8278	**0.9643**	0.4906	0.9506	0.9725	**0.9799**	0.9762	0.5053	0.9701	0.9743	**0.9772**	0.9770
10	0.0000	0.8640	0.8814	**0.9164**	0.9141	0.2245	0.9545	0.9585	**0.9752**	0.9718	0.1132	0.9603	0.9690	**0.9750**	0.9694
11	0.5173	0.9958	0.9959	0.9998	**1.0000**	0.9818	0.9994	**1.0000**	**1.0000**	0.9971	1.0000	0.9998	**1.0000**	**1.0000**	0.9971
12	0.0000	0.9631	0.9639	**0.9988**	0.9779	0.6852	0.9875	0.9871	**0.9996**	0.9822	0.5822	0.9927	0.9904	**0.9997**	0.9835
13	0.0000	0.9485	0.0000	**0.9981**	0.9067	0.2311	0.9860	0.3627	**0.9997**	0.9791	0.2200	0.9912	0.2253	**0.9998**	0.9785
14	0.7932	**1.0000**	0.7738	**1.0000**	0.9976	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9998	0.9984	1.0000	1.0000	**1.0000**	0.9998
15	0.0000	0.8574	0.8325	0.8164	**0.9938**	0.0000	0.9540	0.9803	**0.9944**	0.9760	0.0000	0.9887	0.9919	**0.9944**	0.9765
Average	0.3227	0.8744	0.7935	**0.9237**	0.9195	0.6364	0.9479	0.9095	**0.9666**	0.9390	0.5959	0.9580	0.9075	**0.9680**	0.9417

Table 6
Average *F*-measure standard deviation of all datasets with different methods. The best results are in bold.

ID	SVM					Adaboost					RandomForest				
	SVM	*SMS*	*BSS*	*ASS*	*GOS*	*Ada*	*SMA*	*BSA*	*ASA*	*GOA*	*RF*	*SMRF*	*BSRF*	*ASRF*	*GORF*
1	-	0.1460	0.1052	**0.0423**	0.0387	-	0.0483	0.0604	**0.0322**	0.0511	-	0.0509	0.0563	**0.0351**	0.0548
2	-	0.0371	0.0273	**0.0217**	0.0256	-	0.0155	0.0266	**0.0153**	0.0271	-	0.0212	0.0168	**0.0166**	0.0278
3	-	0.0282	0.0286	**0.0209**	0.0227	-	0.0213	0.0225	**0.0154**	0.0218	-	0.0208	0.0207	**0.0157**	0.0221
4	-	0.0326	0.0343	**0.0213**	0.0251	-	0.0296	0.0355	**0.0213**	0.0241	-	**0.0191**	0.0340	0.0205	0.0281
5	-	0.0109	0.0204	**0.0093**	0.0249	-	0.0141	0.0125	0.0088	**0.0082**	-	0.0102	0.0097	**0.0074**	0.0085
6	-	0.0124	0.0257	**0.0150**	**0.0150**	-	0.0190	0.0093	**0.0087**	0.0095	-	0.0102	0.0093	**0.0082**	0.0092
7	-	0.0009	0.0104	**0.0000**	**0.0000**	-	**0.0000**	**0.0000**	**0.0000**	**0.0000**	-	**0.0000**	0.0042	**0.0000**	0.0170
8	-	0.0253	0.0228	**0.0019**	0.0276	-	0.0170	0.0146	**0.0009**	0.0178	-	0.0083	0.0108	**0.0009**	0.0204
9	-	0.0426	0.0708	0.0261	**0.0131**	-	0.0122	0.0130	0.0132	**0.0101**	-	0.0156	0.0113	0.0111	**0.0093**
10	-	0.0108	0.0217	0.0122	**0.0103**	-	0.0096	0.0096	**0.0067**	0.0083	-	0.0093	0.0068	0.0063	**0.0059**
11	-	0.0041	0.0043	0.0001	**0.0000**	-	0.0015	0.0034	**0.0001**	0.0050	-	0.0009	**0.0000**	**0.0000**	0.0042
12	-	0.0078	0.0056	**0.0002**	0.0062	-	0.0045	0.0034	**0.0002**	0.0045	-	0.0035	0.0045	**0.0002**	0.0045
13	-	0.0217	**0.0000**	0.0006	0.0274	-	0.0095	0.3602	**0.0003**	0.0108	-	0.0083	0.3372	**0.0002**	0.0142
14	-	0.0000	0.1693	**0.0000**	0.0014	-	**0.0000**	**0.0000**	**0.0000**	0.0003	-	**0.0000**	**0.0000**	**0.0000**	0.0005
15	-	0.0109	0.0189	0.0128	**0.0025**	-	0.0063	0.0053	**0.0028**	0.0029	-	0.0031	0.0028	**0.0026**	0.0028

Table 7
Average *Acc* value of all datasets with different methods. The best results are in bold.

ID	SVM					Adaboost					RandomForest				
	SVM	*SMS*	*BSS*	*ASS*	*GOS*	*Ada*	*SMA*	*BSA*	*ASA*	*GOA*	*RF*	*SMRF*	*BSRF*	*ASRF*	*GORF*
1	0.6450	0.5957	0.5710	**0.6741**	0.6028	0.7730	0.8195	0.8022	**0.8320**	0.7905	0.8058	0.8399	0.8325	**0.8629**	0.8043
2	0.7435	0.6652	0.6560	0.7291	**0.7931**	0.7825	0.8456	0.8513	**0.8578**	0.7951	0.7759	**0.8658**	0.8630	0.8567	0.8035
3	0.7494	0.6779	0.6752	0.7535	**0.7659**	0.7804	0.8479	0.8497	**0.8684**	0.7888	0.7837	0.8637	0.8625	**0.8646**	0.8111
4	0.8680	0.8837	0.8965	**0.9169**	0.9029	0.8852	0.9081	0.9008	**0.9216**	0.8874	0.8780	0.9259	0.9263	**0.9288**	0.9056
5	0.9098	0.9878	0.9706	**0.9913**	0.9722	0.9842	0.9894	0.9911	0.9928	**0.9933**	0.9805	0.9917	0.9933	0.9918	**0.9950**
6	0.9116	0.9844	0.9656	**0.9872**	0.9722	0.9749	0.9861	**0.9933**	0.9887	0.9928	0.9842	0.9911	0.9939	0.9918	**0.9955**
7	0.9995	0.9998	0.9995	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9999	**1.0000**	**1.0000**
8	0.9393	0.9094	0.9328	**0.9876**	0.8873	0.9608	0.9826	0.9841	**0.9970**	0.9682	0.9495	0.9920	0.9900	**0.9978**	0.9583
9	**0.9772**	0.7663	0.5831	0.8461	0.9639	0.9668	0.9504	0.9727	**0.9792**	0.9763	0.9718	0.9706	0.9749	0.9765	**0.9772**
10	**0.9597**	0.8536	0.8721	0.9097	0.9079	0.9535	0.9538	0.9580	**0.9747**	0.9720	0.9495	0.9608	0.9694	**0.9746**	0.9698
11	0.9805	0.9957	0.9959	0.9997	**1.0000**	0.9988	0.9994	**1.0000**	**1.0000**	0.9971	**1.0000**	0.9998	**1.0000**	**1.0000**	0.9971
12	0.9703	0.9618	0.9626	**0.9977**	0.9776	0.9813	0.9874	0.9869	**0.9992**	0.9823	0.9807	0.9926	0.9904	**0.9993**	0.9837
13	0.9751	0.9504	0.9730	**0.9964**	0.8981	0.9758	0.9861	0.9794	**0.9995**	0.9796	0.9794	0.9912	0.9779	**0.9996**	0.9788
14	0.9949	**1.0000**	0.9949	**1.0000**	0.9977	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9998	0.9999	**1.0000**	**1.0000**	**1.0000**	0.9998
15	0.9880	0.8479	0.8450	0.7793	**0.9939**	0.9879	0.9522	0.9793	**0.9939**	0.9926	0.9879	0.9902	0.9922	**0.9943**	0.9932
Average	0.9075	0.8720	0.8596	0.9046	**0.9090**	0.9337	0.9472	0.9499	**0.9603**	0.9411	0.9351	0.9583	0.9577	**0.9626**	0.9449

Table 8
Average *Acc* standard deviation of all datasets with different methods. The best results are in bold.

ID	SVM					Adaboost					RandomForest				
	SVM	*SMS*	*BSS*	*ASS*	*GOS*	*Ada*	*SMA*	*BSA*	*ASA*	*GOA*	*RF*	*SMRF*	*BSRF*	*ASRF*	*GORF*
1	-	0.0892	0.0530	0.0562	**0.0446**	-	0.0380	0.0583	0.0384	**0.0378**	-	0.0490	0.0578	**0.0414**	0.0511
2	-	0.0325	0.0284	0.0267	**0.0237**	-	0.0190	0.0269	**0.0175**	0.0237	-	0.0210	**0.0163**	0.0196	0.0217
3	-	0.0308	0.0288	0.0249	**0.0196**	-	0.0212	0.0201	**0.0177**	0.0218	-	0.0196	0.0192	**0.0183**	0.0228
4	-	0.0320	0.0361	0.0266	**0.0249**	-	0.0284	0.0363	0.0243	**0.0226**	-	**0.0205**	0.0337	0.0225	0.0239
5	-	0.0106	0.0205	**0.0101**	0.0242	-	0.0138	0.0123	0.0096	**0.0080**	-	0.0096	0.0097	**0.0080**	0.0087
6	-	**0.0120**	0.0252	0.0158	0.0152	-	0.0180	**0.0089**	0.0091	0.0097	-	0.0103	0.0089	**0.0088**	0.0094
7	-	0.0009	0.0011	**0.0000**	**0.0000**	-	**0.0000**	**0.0000**	**0.0000**	**0.0000**	-	**0.0000**	0.0005	**0.0000**	**0.0000**
8	-	0.0280	0.0238	**0.0036**	0.0289	-	0.0161	0.0148	**0.0018**	0.0168	-	0.0085	0.0106	**0.0017**	0.0198
9	-	0.0306	0.0511	0.0211	**0.0134**	-	0.0124	0.0120	0.0134	**0.0103**	-	0.0155	0.0111	0.0113	**0.0095**
10	-	0.0121	0.0224	0.0111	**0.0106**	-	0.0094	0.0087	**0.0068**	0.0078	-	0.0092	0.0060	0.0060	**0.0056**
11	-	0.0043	0.0044	0.0003	**0.0000**	-	0.0017	**0.0000**	0.0001	0.0051	-	0.0010	**0.0000**	0.0001	0.0044
12	-	0.0076	0.0059	**0.0004**	0.0060	-	0.0043	0.0036	**0.0003**	0.0042	-	0.0036	0.0045	**0.0003**	0.0045
13	-	0.0208	0.0188	**0.0012**	0.0284	-	0.0093	0.0177	**0.0005**	0.0098	-	0.0084	0.0176	**0.0003**	0.0136
14	-	**0.0000**	0.0034	**0.0000**	0.0013	-	**0.0000**	**0.0000**	**0.0000**	0.0003	-	**0.0000**	**0.0000**	**0.0000**	0.0005
15	-	0.0113	0.0170	0.0150	**0.0025**	-	0.0064	0.0052	0.0029	**0.0028**	-	0.0030	**0.0026**	0.0027	0.0028

4. Conclusion

In this paper, we proposed two oversampling methods (Adaptive-SMOTE and Gaussian Oversampling) to deal with imbalanced problems. Based on the original data distribution features, Adaptive-SMOTE can adaptively divide the positive dataset into *Danger* and *Inner* and oversample the data using SMOTE. This approach prevents the expansion of positive samples and strengthens the distributional characteristics of the original dataset. Gaussian Oversampling provides a novel division strategy for sampling regions, which makes sampling more reasonable. The two proposed methods could achieve high *F*-measure and *Acc* values and are robust to datasets and classifiers according to numerical experimental results. As an extension, Adaptive-SMOTE is used to balance the dataset for multiclass problems and achieves good results.

Declaration of Competing Interest

The authors declared that they have no conflicts of interest to this work.

Acknowlgedgments

The authors would like to thank the editor and anonymous reviewers for their constructive comments that greatly improved the quality of this paper. This work was supported by Shanghai Municipal Science and Technology Major Project (No.2018SHZDZX01) and ZJLab, the National Natural Science Foundation of China (No.11201051), and Fundamental Research Funds for the Central Universities (No.DUT18JC02).

References

[1] J. Alcala-Fdez, A. Fernandez, J. Luengo, J. Derrac, S. Garcia, Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework, Multiple-Valued Logic Soft Comput. 17 (2–3) (2011) 255–287.
[2] G. Batista, M.C. Monard, A study of the behavior of several methods for balancing machine learning training data, ACM SIGKDD Explorations Newsl. (2004) 20–29.
[3] P. Billingsley, Probability and Measure (3rd Edition), John Wiley and Sons, 1995.
[4] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, SMOTE: synthetic minority over-sampling technique, J. Artif. Intell. Res. 16 (2002) 321–357.
[5] N.V. Chawla, A. Lazarevic, L.O. Hall, K.W. Bowyer, SMOTEBoost: improving prediction of the minority class in boosting, Knowl. Discov. Databases 2838 (2003) 107–119.
[6] S. Chen, S. Chang, Q. Huang, J. He, H. Wang, Q. Huang, SVM-based synthetic fingerprint discrimination algorithm and quantitative optimization strategy, PLoS ONE 9 (10) (2014) e111099.
[7] Y. Dong, X. Wang, A new over-sampling approach: random-SMOTE for learning from imbalanced data sets, in: Proceedings of the 5th International Conference on Knowledge Science, Engineering and Management, 2011, pp. 343–352.
[8] R.P. Espindola, N.F.F. Ebecken, On extending f-measure and g-mean metrics to multi-class problems, Data Min. 25 (2005).
[9] T. Fawcett, F. Provost, Adaptive fraud detection, Data Min. Knowl. Discov. 1 (3) (1997) 291–316.
[10] C. George, B. Roger, Statistical inference (2nd ed.). duxbury. ISBN 0-534-24312-6, 2001.
[11] J. Gong, H. Kim, RHSBoost: improving classification performance in imbalance data, Comput. Stat. Data Anal. 111 (2017) 1–13.
[12] H. Guo, H.L. Viktor, Learning from imbalanced data sets with boosting and data generation: the databoost-IM approach, ACM SIGKDD Explorations Newsl. 6 (1) (2004) 30–39.
[13] H. Han, W.Y. Wang, B.H. Mao, Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning, Adv. Intell. Comput. 3644 (2005) 878–887.
[14] H. He, E. Garcia, Learning from imbalanced data, IEEE Trans. Knowl. Data Eng. 21 (9) (2009) 1263–1284.
[15] Z. He, J. Mao, X. Han, Non-parametric estimation of particle size distribution from spectral extinction data with PCA approach, Powder Technol. 325 (2018) 510–518.
[16] H. He, B. Yang, A.G. Edwardo, S. Li, ADASYN: adaptive synthetic sampling approach for imbalanced learning, in: 2008 International Joint Conference on Neural Networks, 2008, pp. 1322–1328.

[17] B. Krawczyk, M. Galar, L. Jele, F. Herrera, Evolutionary undersampling boosting for imbalanced classification of breast cancer malignancy, Appl. Soft. Comput. 38 (2016) 714–726.
[18] J. Li, S. Fong, S. Mohammed, J. Fiaidhi, Improving the classification performance of biological imbalanced datasets by swarm optimization algorithms, J. Supercomput. 72 (10) (2016) 3708–3728.
[19] A. Liaw, M. Wiener, Classification and regression by randomforest, R News 2 (3) (2002) 18–22.
[20] X.Y. Liu, J. Wu, Z.H. Zhou, Exploratory undersampling for class-imbalance learning, IEEE Trans. Syst. 39 (2) (2009).
[21] G. Menardi, N. Torelli, Training and assessing classification rules with imbalanced data, Data Min. Knowl. Discov. 28 (1) (2014) 92C122.
[22] E. Owusu, Y. Zhan, Q. Mao, An SVM-adaboost facial expression recognition system, Appl. Intell. 40 (3) (2014) 536–545.
[23] F. Provost, G.M. Weiss, Learning when training data are costly: the effect of class distribution on tree induction, J. Artif. Intell. Res. 19 (2003) 315–354.
[24] Y. Sun, M.S. Kamel, A.K.C. Wong, Y. Wand, Cost sensitive boosting for classification of imbalanced data, Pattern Recognit. 40 (12) (2007) 3358–3378.
[25] Y. Tang, Y.Q. Zhang, N.V. Chawla, S. Krasser, SVMs modeling for highly imbalanced classification, IEEE Trans. Syst. Man Cybern. 39 (1) (2009) 281–288.
[26] I. Triguero, R. del S., V. Lopez, J. Bacardit, J.M. Benitez, F. Herrera, ROSEFW-RF: the winner algorithm for the ECBDL' 14 big data competition: an extremely imbalanced big data bioinformatics problem, Knowl.-Based Syst. 87 (2015) 69–79.
[27] Q. Wang, Z.H. Luo, J.C. Huang, Y.H. Feng, Z. Liu, A novel ensemble method for imbalanced data learning: bagging of extrapolation-SMOTE SVM, Comput. Intell. Neurosci. 2017 (2017).
[28] K. Zahirnia, M. Teimouri, R. Rahmani, A. Salaq, Diagnosis of type 2 diabetes using cost-sensitive learning, in: Proceedings of the 5th International Conference on Computer and Knowledge Engineering, 2015, pp. 158–163. October.
[29] Z.H. Zhou, Ensemble learning, in: Encyclopedia of Biometrics, 2009, pp. 270–273.