Applied Soft Computing Journal 79 (2019) 439–451

Contents lists available at ScienceDirect

Applied Soft Computing Journal

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

Generative learning for imbalanced data using the Gaussian mixed model

Yuxi Xie [a], Lizhi Peng [a,*], Zhenxiang Chen [a], Bo Yang [a], Hongli Zhang [b], Haibo Zhang [c]

[a] Shandong Provincial Key Laboratory of Network Based Intelligent Computing, University of Jinan, Jinan, 250022, PR China
[b] School of Computer Science and Technology, Harbin Institute of Technology, Harbin, 150002, PR China
[c] Department of Compute Science, University of Otago, Dunedin, New Zealand

ARTICLE INFO

Article history:
Received 27 November 2018
Received in revised form 5 March 2019
Accepted 27 March 2019
Available online 12 April 2019

Keywords:
Imbalanced learning
Gaussian mixed model
Sample generation

ABSTRACT

Imbalanced data classification, an important type of classification task, is challenging for standard learning algorithms. There are different strategies to handle the problem, as popular imbalanced learning technologies, data level imbalanced learning methods have elicited ample attention from researchers in recent years. However, most data level approaches linearly generate new instances by using local neighbor information rather than based on overall data distribution. Differing from these algorithms, in this study, we develop a new data level method, namely, generative learning (GL), to deal with imbalanced problems. In GL, we fit the distribution of the original data and generate new data on the basis of the distribution by adopting the Gaussian mixed model. Generated data, including synthetic minority and majority classes, are used to train learning models. The proposed method is validated through experiments performed on real-world data sets. Results show that our approach is competitive and comparable with other methods, such as SMOTE, SMOTE-ENN, SMOTE-TomekLinks, Borderline-SMOTE, and safe-level-SMOTE. Wilcoxon signed rank test is applied, and the testing results show again the significant superiority of our proposal.

1. Introduction

Imbalanced data, wherein the instances of one class outnumber those of another class, are important in machine learning tasks. Classes with many and few instances are called majority and minority classes, respectively. Imbalanced ratio (IR), the key factor to the measurement of the imbalance degree of such data, is the ratio between the number of majority and minority class instances, and it can be presented as follows:

$$IR = \frac{n_{maj}}{n_{min}}, \tag{1}$$

where n_{maj} and n_{min} are the number of majority and minority class instances, respectively. Many real-world tasks, such as medical diagnosis [1], detection of fraudulent telephone calls [2], financial risk management [3], network intrusion detection [4], and satellite image classification [5], can be summarized as imbalanced problems. A key point of such data is that the minority class is typically more important than the majority class from the viewpoint of classification. For example, diagnosing a normal person as a patient is tolerable in medical diagnosis, but diagnosing a patient as a normal person is disastrous. Therefore, the classification accuracy of the minority class is usually of concern in imbalanced problems. Standard metrics, such as accuracy, that measure the performance of standard classifiers are ineffective for imbalanced problems. For example, accuracy focuses on the overall correct classification rate and ignores the importance of different classes, thus, it fails to accurately show the recognition performance of the minority class.

Standard classifiers suffer from imbalanced problems, because their goal is to maximize the overall accuracy. Identifying minority class instances from numerous majority class instances is difficult for standard classifiers, and this difficulty frequently leads to a high misclassification rate for minority class instances, especially for highly imbalanced cases. Therefore, many methodologies have been developed to address imbalanced learning problems. In general, these methods can be classified into three categories: data level strategies, algorithmic level strategies and ensemble learning strategies.

Data level approaches have elicited much research interest because of its independence from classifiers, and many approaches have been presented in recent years. SMOTE, wherein new instances are generated along the lines of a minority class instance to its neighbors, is the most popular among the above-mentioned approaches [6]. Researchers have proposed numerous improved variants of SMOTE, such as SMOTE-ENN [7], SMOTE-TomekLinks [7], borderline-SMOTE [8], and safe-level-SMOTE [9].

* Corresponding author.
E-mail address: plz@ujn.edu.cn (L. Peng).

https://doi.org/10.1016/j.asoc.2019.03.056
1568-4946/

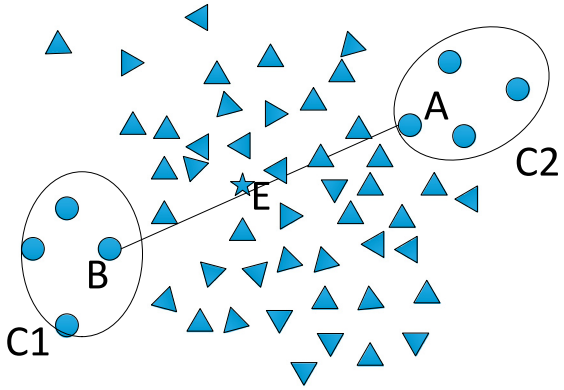


Fig. 1. Several minority clusters in a data set.

Data level approaches are highly effective in certain scenarios, but they may generate incorrect and unnecessary instances [10], which may result in inaccurate learning for classification models. In addition, the instances are linearly generated based on a connection route between two points in these methods and cannot accurately show the distribution of the original data. In Fig. 1, the triangles and circles represent the majority and minority class instances, respectively. Assuming that we generate a synthetic instance from minority instance A and k is 5, B is selected as one of the five-nearest neighbors of A. Then, instance E is generated based on the linear interpolation method. E is a wrong instance because it is located in the majority class area.

To address these problems, we provide three significant contributions to the study of imbalanced problems.

- We propose a completely new concept, namely, generative learning (GL), the main idea of which is that new data are generated, including synthetic minority and majority classes, for training learning models. Different from SMOTE and its variants, in the GL method, new data are completely generated based on the distribution of the original data for imbalanced problems instead of along the lines from a minority instance to its neighbors.
- We apply the Gaussian mixed model (GMM) to perform the data generation process of GL in this study. GMM is used for fitting the distribution of the original data and generating new instances in the minority and majority classes. In the GL method, we train a support vector machine (SVM) with generated instances and keep the optimal generated data set as a training set for testing.
- We conduct a set of empirical studies to test the performance of our proposal by comparing five methods, namely, SMOTE, SMOTE-ENN, SMOTE-TomekLinks, borderline-SMOTE, and safe-level-SMOTE. The experimental results show that our method has obvious advantages over the compared methods. The statistical hypothesis test results illustrate that our method is considerably different from other methods.

The remainder of the paper is organized as follows. Section 2 reviews related work. Section 3 introduces the details of our proposals. The experimental results are obtained and analyzed in Section 4. The conclusion is presented in Section 5.

2. Related work

Various techniques have been proposed to deal with imbalanced learning problems in recent years. López et al. [11] reviewed problems of imbalanced classification. State-of-the-art studies on imbalanced classification techniques can be summarized into three categories, namely, data level, cost-sensitive and ensemble learning approaches. GMM, the linear combination of multiple Gaussian distribution functions, was applied in our study. Notably, it has been successfully used in certain fields. We briefly reviewed its application status.

2.1. Data level approaches

Data level approaches aim to balance data distribution by over-sampling the minority class [12,13] or under-sampling the majority class [14,15]. SMOTE is a widely used over-sampling method [6] in which new minority class instances are generated on the lines that connect a minority class instance and its neighbors. Using SMOTE, researchers have synthesized the same number of instances for each minority class instance but disregarded the distribution characteristics of its neighboring instances. Thus, numerous improved variants of SMOTE have been put forward. He et al. [16] proposed ADASYN and generated a different number of instances for each minority class instance according to the learning difficulty of the instance. Bunkhumpornpat et al. [9] introduced safe-level-SMOTE, wherein instances were generated based on the safety level of the minority class instances. In the minority class, few instances are generated in the boundary region, whereas many instances are generated in the safe region. Ramentol et al. [17] introduced SMOTE-RSB* that improved the quality of synthetic instances by using SMOTE in combination with editing techniques based on rough set theory (RST) and the lower approximation of a subset. Barua et al. [10] proposed majority weighted minority over-sampling TEchnique (MWMOTE) and used it to weigh and cluster minority class instances; then, new instances were generated by SMOTE. Chan et al. [18] divided majority class instances into non-overlapping subsets that were combined with minority class instances to form a new training set. Verbiest et al. [19] removed noisy instances from the original data set and cleaned the data generated by SMOTE based on fuzzy rough prototype selection algorithm. Liu et al. [20] put forward EasyEnsemble and BalanceCascade algorithms to deal with imbalanced problems. They sampled a subset in the majority class to train learners and combined the results of all learners by using EasyEnsemble. In BalanceCascade, learners were trained sequentially, and correctly classified majority class instances were removed in each step. Yu et al. [21] proposed a new under-sampling method based on the ant colony optimization algorithm in 2013 and successfully applied the method to the classification of DNA microarray data.

2.2. Cost-sensitive approaches

Cost-sensitive approaches [22,23] are also effective methods for imbalanced data classification. A cost matrix is used to penalize the instances that are misclassified by models, with an especially strong penalization for the behavior of misclassifying the minority class from the majority class. In this manner, a classification model with minimum cost is created. The method based on cost-proportionate rejection sampling and ensemble aggregation was proposed to rebalance training data [24]. A method has been developed to train artificial neural networks by constructing a cost function [25]. Correspondingly, C4.5 [26] and SVM [27] have been improved based on the approach. Qiu et al. [28] proposed a method that created an optimal example subset with low misclassification costs to obtain majority class instances with useful information. Liu et al. [29] proposed a cost-sensitive sparse representation based classification (CSSRC) method for class-imbalance problems by using probabilistic modeling. Bahnsen et al. [30]

presented an example-dependent cost-sensitive decision tree algorithm. They applied a new cost-based impurity measure and new cost-based pruning criteria to the decision tree. Krawczyk et al. [31] proposed a novel cost-sensitive ensemble based on decision trees for imbalanced classification. The experimental results showed that the method was effective for imbalanced data sets.

2.3. Ensemble learning approaches

Data level approaches can be ensembled with ensemble learning methods to construct a hybrid model for imbalanced problems. In recent years, ensemble learning approaches have gradually become a hot spot for research on imbalanced classification. Chawla et al. [32] proposed the SMOTEBoost method by combining SMOTE and the boost learning model. Seiffert et al. [33] proposed an improved SMOTEBoost called RUSBoost that is simpler and more efficient than SMOTEBoost. Lim et al. [34] presented a novel method, called cluster-based synthetic data generation, which optimized by an evolutionary algorithm (EA). They generated different samples based on the method to create an ensemble. FOS, a fuzzy-based over-sampling method, was combined with an ensemble learning approach to dispose of imbalanced problems in another study [35]. Wang et al. [36] combined SMOTE with PSO and integrated some well known classifiers for breast cancer data classification. Dez-Pastor et al. [37] analyzed various ensemble-based methods. The results showed that diversity-increasing techniques substantially improve the performance of ensemble methods. A ensemble classifier named EUSBoost was proposed in [38] for balancing training sets by combining a boosting scheme with evolutionary under-sampling. The method outperformed state-of-the-art ensemble classifiers in imbalanced medical problem.

2.4. Application research on GMM

Many scholars have studied GMM and mainly applied it to particular domains, such as image segmentation, object recognition, and video analysis. Ji et al. [39] presented an robust generative asymmetric GMM (RGAGMM) for correcting intensity inhomogeneity and segmenting brain MR images. Niranjil et al. [40] proposed a modified adaptive GMM with three temporal differencing for moving object detection. A naive Bayesian network combined with GMM that grouped high-dimensional features for textural image classification was proposed in Ref. [41]. Xia et al. [42] proposed a modified GMM that used temporal and spatial distribution information for background modeling. However, studies on the application of GMM to imbalanced problems are lacking. Therefore, we initiated preliminary efforts by conducting related explorations and research.

3. Proposed method

3.1. GL

For an n-dimensional binary classification problem, all instances make up an n-dimensional data space $\mathbb{R}^n$. Theoretically, $\mathbb{R}^n$ can be divided into $\mathbb{R}^n_{maj}$ and $\mathbb{R}^n_{min}$, that is, $\mathbb{R}^n = \mathbb{R}^n_{maj} \cup \mathbb{R}^n_{min}$. $\mathbb{R}^n_{maj}$ and $\mathbb{R}^n_{min}$ are the subspaces of majority and minority class instances, respectively. The given training set T, which is the sampling subset of $\mathbb{R}^n$, includes T_{maj} and T_{min}, that is, $T = T_{maj} \cup T_{min}$. T_{maj} and T_{min} are the instance subsets of $\mathbb{R}^n_{maj}$ and $\mathbb{R}^n_{min}$, respectively. Theoretically, in the premise of accurately fitting the real distribution of $\mathbb{R}^n$, the more adequate the sampling for T is, the more adequate the training of learning models is. This context results in enhanced classification performance. However, T is usually an inadequate sampling set for many real classification problems. For imbalanced problems, T_{min} is typically a severely inadequate sampling set. Thus, training learning models well is difficult. To deal with these problems, in GL method, we resample $\mathbb{R}^n$ to obtain new well-scaled training data through the following steps.

(1) Obtain the real distribution of $\mathbb{R}^n_{maj}$ and $\mathbb{R}^n_{min}$ using T_{maj} and T_{min}, respectively, denoted as $N(maj)$ and $N(min)$.
(2) Obtain T'_{maj} and T'_{min} by resampling $\mathbb{R}^n$ according to $N(maj)$ and $N(min)$, respectively. $T' = T'_{maj} \cup T'_{min}$ is the new training set, where $|T'_{maj}| > |T_{maj}|$, $|T'_{min}| \gg |T_{min}|$, $0.9 < \frac{|T'_{maj}|}{|T'_{min}|} < 2$.
(3) For a given learning model, such as SVM, we use the model trained by the processed T' for testing.

The entire process is shown in Fig. 2.

We can sum up two points through the GL method. First, we find that the relatively adequate sampling set T' is generated by large-scale resampling in $\mathbb{R}^n$, especially for an adequate sampling of minority class instances. Thus, a learning model is possible to be well trained. Second, new instances are generated based on the distribution of the original data in GL, which is essentially different from SMOTE and its variants, where new instances are generated linearly.

3.2. Data generation

As described in Section 3.1, the first point of GL is that the distribution of sample space $\mathbb{R}^n$ is obtained based on the original training set T. GMM is an effective method to fit the distribution of the original data approximatively [43]. Therefore, we apply GMM to fit the distribution of the original training set T then resample based on the distribution.

According to GMM, we use K ($1 \le K \le 5$) Gaussian distributions $N(\mu_1, \Sigma_1)$, $N(\mu_2, \Sigma_2)$, $\cdots$, $N(\mu_K, \Sigma_K)$ in our study to fit the distribution of T_{maj} or T_{min}. As shown in Fig. 3, we use two two-dimensional Gaussian distributions, namely, $N(\mu_1, \Sigma_1)$ and $N(\mu_2, \Sigma_2)$, to describe and generate data. Notably, we should respectively obtain a GMM to fit T_{maj} and T_{min}.

We use the following formula to describe the data for a class of a given data set in binary classification problems.

$$p(x) = \sum_{k=1}^{K} \pi_k N(X|\mu_k, \Sigma_k), \tag{2}$$

where X is an instance in T_{maj} or T_{min}, and $N(X|\mu_k, \Sigma_k)$ is the kth component in the mixed model, which is defined as follow:

$$N(X|\mu_k, \Sigma_k) = \frac{1}{\sqrt{2\pi}\,|\Sigma_k|} exp\left[-\frac{1}{2}(X-\mu_k)^T \Sigma_k^{-1}(X-\mu_k)\right]. \tag{3}$$

π_k is the mixture coefficient and satisfies the following equation.

$$\sum_{k=1}^{K} \pi_k = 1, \qquad 0 \le \pi_k \le 1 \tag{4}$$

π_k can be regarded as the weight of each component.

The GMM model consists of the $3K$ unknown parameters, namely, the mean(μ), variance(Σ), and Gaussian component weight(π) of the K Gaussian distributions. These parameters are key factors for distribution fitting. In our study, the EM algorithm is applied to optimize these parameters. The EM algorithm consists of two steps. In the first step, the evaluated parameters are initialized. K clusters are clustered by k-means, the center of each cluster is used to initialize μ, the variance of each cluster is used to initialize Σ, and π is initialized to $\frac{1}{K}$. In the second step, the values calculated by the first step are used to maximize the likelihood function. Therefore, we must first find

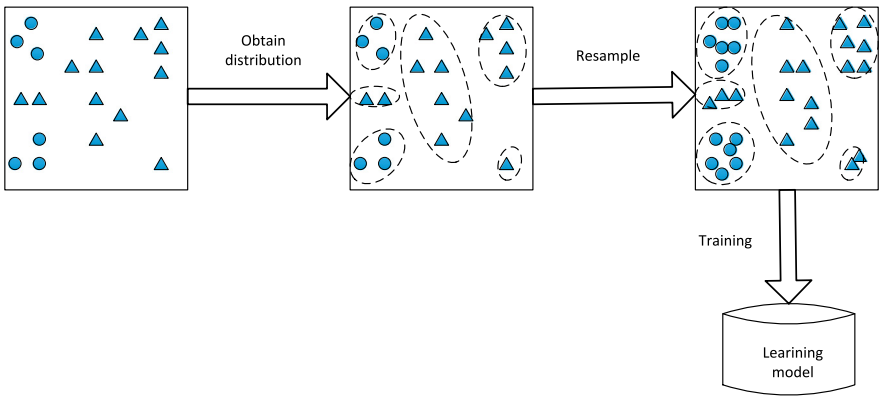


Fig. 2. Schematic of the proposed method.

the likelihood function of GMM. Three parameters are estimated in each Gaussian distribution, namely, π_k, μ_k and Σ_k. To estimate the three parameters, the maximum likelihood functions of the three parameters should be solved separately. By performing logarithmic transformation and derivation on Eq. (2), the solution formulas of π_k, μ_k and Σ_k can be derived as follows:

$$\mu_k = \frac{1}{N_k}\sum_{i=1}^{N}\gamma\left(z_{ik}\right)X_i, \tag{5}$$

$$\Sigma_k = \frac{1}{N_k}\sum_{i=1}^{N}\gamma\left(z_{ik}\right)\left(X_i-\mu_k\right)\left(X_i-\mu_k\right)^T, \tag{6}$$

$$\pi_k = \frac{N_k}{N}, \tag{7}$$

where $\gamma(z_{ik})$ is the posterior probability of X_i belonging to the kth Gaussian distribution. It is introduced to facilitate the estimation of GMM parameters using the EM algorithm. z is the introduced latent variable, where $z_k \in \{0, 1\}$. $z_k = 1$ indicates the probability of selecting the kth Gaussian component, whereas $z_k = 0$ indicates the probability of not selecting the kth Gaussian component. N is the number of instances in T_{maj} or T_{min}, and N_k indicates the number of instances belonging to the kth Gaussian distribution, where

$$N_k = \sum_{i=1}^{N}\gamma\left(z_{ik}\right), \tag{8}$$

$$\gamma\left(z_{ik}\right) = \frac{\pi_k N\left(X_i|\mu_k, \Sigma_k\right)}{\sum_{j=1}^{K}\pi_j N\left(X_i|\mu_j, \Sigma_j\right)}. \tag{9}$$

We generate data based on the parameters of each Gaussian component previously obtained. The number of generated instances is 3000 for the original data set with less than 1000 instances, and 6000 instances are generated for the original data set with more than 1000 instances. The number of generated minority class instances is similar to that of generated majority class instances. The number of instances of each distribution for each class is determined by the weight of the Gaussian component. The instances that exceed the original data definition domain are removed.

3.3. Scale analysis of generated data

Classification performance is related to the number of training instances to a certain extent. Theoretically, the more the training instances are, the better the classification performance is. However, a large training set means lengthy training time and overfitting. Hence, after considering the time cost and overfitting problems, the amount of generated data should be controlled.

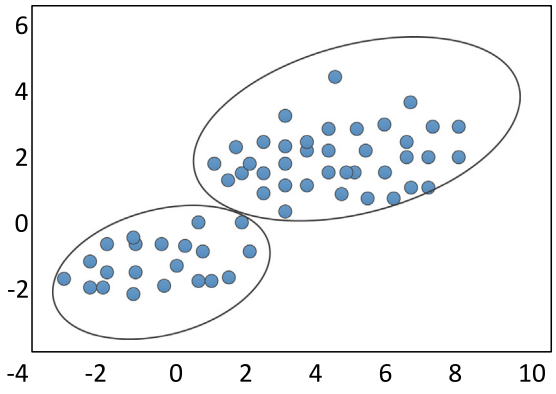


Fig. 3. Sample observation data.

However, if the generated data are too few, then the model may obtain insufficient training or underfitting. Therefore, we should control the scale of the generated data. In this study, the total size of the generated data is empirically set to 3000 or 6000 according to the scale of the original data sets.

To balance the class distribution, we should change the original data class distribution. Hence, the number of generated minority class instances is consistent with the generated majority class instances, which will better adapt to the mechanism of standard learning models to achieve effective identification.

3.4. Analysis of the number of Gaussian components

As depicted in Section 3.2, we use K Gaussian components to fit the distribution of the original data and generate new data. A consequent problem is the number of Gaussian components to be used. That is, how to obtain the best performance by setting parameter K.

Obviously, using extremely few Gaussian components to fit the original data cannot accurately fit the distribution of the actual data. On the contrary, if too many Gaussian components are used, then the time cost will increase. In many real cases, collecting data is difficult, especially for positive data instances. This context is an important reason for imbalanced data sets. To fit the distribution of such data, the number of Gaussian components should be set to a low level. For example, fitting the data in Fig. 3 with only one Gaussian component is obviously unreasonable and leads to inconsistent data distribution between new data and raw data. Therefore, dividing the data into two clusters is obviously reasonable. Hence, two Gaussian components can be used to accurately fit the distributions of such data in Fig. 3. We

should select the appropriate number of Gaussian components to fit the data. Real-world data can be typically divided into K clusters and appropriately described by K Gaussian components. Usually, K is set neither too large nor too small. In our study, we set K in [1, 5].

3.5. Algorithm and algorithm complexity analysis

Algorithm 1 describes the complete method, in which the original data set T pertains to the input data and the generated data set T' is the output data.

Notably, several instances in T' may outrange T. Thus, we post-process T' by removing the instances that outrange T.

Algorithm 1 GL

Input: Raw data set T;
Output: Generated data set $T^{'}$;
1: Initialize $T^{'}$ and K;
2: **for** each Gaussian component parameters μ_k, Σ_k and π_k in T_{maj} and T_{min} **do**
3: Initialize μ_k, Σ_k and π_k;
4: **do**
5: Calculate Eq.(9) based on μ_k, Σ_k and π_k;
6: Update μ_k, Σ_k and π_k according to Eqs.(5), (6), and (7);
7: Calculate the log-likelihood function of Eq.(2);
8: **while** iters
9: Generate data according to μ_k, Σ_k and π_k and add the data to $T^{'}$;
10: **end for**
11: Post-process $T^{'}$ to obtain $T_p^{'}$(remove samples that outrange T);
12: Use $T_p^{'}$ to train SVM;

In the data generation stage, we use the K Gaussian components in T_{maj} and T_{min}, respectively. Thus, we should calculate $2K$ times. In each iteration, the three parameters are updated and iterated *iters* times. In addition, each update of the parameters should be calculated $\left|T_{maj}\right|$ or $|T_{min}|$ times. Therefore, the time complexity is $O(2K * iters * \left|T_{maj}\right| / |T_{min}|)$. In the model training stage, we use processed T' to train SVM, in which the time complexity is $O(\left|T_p'\right|^2)$. Therefore, the total time complexity of the algorithm is $O(2K * iters * \left|T_{maj}\right| / |T_{min}| + \left|T_p'\right|^2)$.

4. Experimental study

In this section, a set of empirical studies are designed to evaluate the performance of our proposal. We present the performance metrics used in the study (Section 4.1). Then, we describe the details of data sets used and the parameter configurations of the learning algorithms selected for this study (Section 4.2). Next, we introduce a statistical test method that can be used to compare the experimental results (Section 4.3). The results, including experimental and hypothesis test results, are then presented (Section 4.4). Then, we introduce empirical studies and analyses that are divided into two parts. First, we carry out an analysis of the number of Gaussian components selected for our model (Section 4.5). Second, we conduct an analysis of the impact of generated data scale on classification performance (Section 4.6). Finally, we compare the runtime of all compared algorithms on large-scale data sets (Section 4.7) and analyze the performance of GL on multi-class imbalanced data sets (Section 4.8).

Table 1
Confusion matrix.

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

4.1. Performance measures

Confusion matrix, a visualization tool for classification results, is the basis for evaluating classifier performance [44]. Each column of the confusion matrix represents the predicted class, and each row represents the actual class. Table 1 is a typical binary confusion matrix. TP (true positive) is the number of correctly classified positive instances, FN (false negative) is the number of the positive instances that are incorrectly classified as negative. FP (false positive) is the number of the negative instances that are incorrectly classified as positive, and TN (true negative) is the number of the correctly classified negative instances. Many evaluation metrics are calculated by the confusion matrix. We use F-measure and area under the curve (AUC) as the performance measures in our empirical evaluations.

F-measure, which combines precision and recall, shows the prediction performance of a learning model for each class. Hence, F-measure is frequently used as an important metric in imbalanced classification problems. Precision p is the ratio between the number of correctly classified positive instances and the number of all predicted positive instances. Recall r is the ratio between the number of correctly classified positive instances and the number of all positive instances. They are defined as follows:

$$p = \frac{TP}{TP + FP}, \tag{10}$$

$$r = \frac{TP}{TP + FN}. \tag{11}$$

F-measure, the harmonic mean of precision and recall, is defined as follows:

$$F - measure = \frac{2 * r * p}{r + p}. \tag{12}$$

Receiver operating characteristic curve (ROC) is a two-dimensional graphic [45] in which the X-axis and Y-axis represent the false positive rate (FPR) and the true positive rate (TPR), respectively. The diagonal line represents the interaction between FPR and TPR of a completely random model. The point (0, 1) shows a perfect classification case in which all instances are classified correctly. AUC is the area under the ROC curve. The closer the curve is to the upper left corner, the better the performance of the classifier is. In other words, AUC [46] will be as large as possible. TPR and FPR are defined as follows:

$$TPR = \frac{TP}{TP + FN}, \tag{13}$$

$$FPR = \frac{FP}{FP + TN}. \tag{14}$$

AUC is another important performance metric for imbalanced classification, and it is defined as follows [47]:

$$AUC = \frac{1 + TPR - FPR}{2}. \tag{15}$$

It assesses the number of positive and negative instances that are correctly classified.

MAUC [48] which is used to evaluate the multi-class imbalanced data is the average of AUC over all pairs of classes, which

Table 2
Description of the real-world data sets.

Data set	# instances	# attributes	IR
yeast-2_vs_4	514	8	9.08
ecoli4	336	7	15.8
glass-0-1-6_vs_2	192	9	10.29
abalone9-18_vs_2	731	8	16.4
ecoli2_vs_2	336	7	5.46
glass4	214	9	15.47
kr-vs-k-zero–one_vs_draw	2901	6	26.63
new-thyroid1	215	5	5.14
segment0	2308	19	6.02
shuttle-c0-vs-c4	1829	9	13.87
winequality-white-3_vs_7	900	11	44
glass2	214	9	11.59
newthyroid2	215	5	5.14
vehicle0	846	18	3.25
winequality-red-8_vs_6	656	11	35.44
led7digit-0-2-4-5-6-7-8-9_vs_1	443	7	10.97
abalone19	4174	8	129.44
glass0	214	9	2.06
haberman2	306	3	2.78
iris0	150	4	2
vehicle2	846	18	2.88
yeast4	1484	8	28.1
kr-vs-k-one_vs_fifteen	2244	6	27.77
ecoli1	336	7	3.36
ecoli3	336	7	8.6
yeast-0-5-6-7-9_vs_4	528	8	9.35
yeast-2_vs_8	482	8	23.1
ecoli-0-1_vs_5	240	6	11
ecoli-0_vs_1	220	7	1.86
car-vgood	1728	6	25.58

is defined as follows:

$$MAUC = \frac{2}{c(c-1)} \sum_{i<j} \frac{A(i|j) + A(j|i)}{2}, \tag{16}$$

where, c is the number of classes, $A(i|j)$ is the probability that an instance of class j has a lower probability to predict as class i than an instance of class i.

4.2. Data sets and compared algorithms

Three types of imbalanced data sets are used for evaluating our method. First, thirty real-world imbalanced data sets from the KEEL data repository [49] are employed. Second, ten large-scale data sets from UCI are used for measuring the time performance of GL and compared methods. Third, five multi-class imbalanced KEEL data sets are used for the experiments.

Table 2 provides the details of KEEL data sets. In the table, the numbers under the second column represent the number of instances in each data set, those under the third column denote the number of attributes in each data set, and IR indicates the imbalance rate. A ten-fold cross-validation is applied in the experiment.

The details of the selected large-scale data sets are shown in Table 3. For the multi-class data sets, one of the classes is as the minority class, and the remaining classes are merged as the majority class.

Table 4 presents the details of the five multi-class data sets. IR is the ratio of the number of the class with the most instances and the class with minimal instances. #class represents the number of classes.

4.2.1. Compared algorithms

To evaluate the performance of the proposed method, we compared our proposal with five over-sampling algorithms, namely, SMOTE [6], SMOTE-ENN [7], SMOTE-TomekLinks [7], borderline-SMOTE [8], and safe-level-SMOTE [9]. In addition, two recent

Table 3
Details of the large-scale data sets.

Data set	# instances	# attributes	IR
Chesscking-Rook vs. King	28056	6	113.05
HTRU2	17898	9	67.44
letter-recognition	20000	16	24.35
musk	6598	168	5.49
nursery	12960	8	2
page-block0	5472	10	8.79
parkinsons telemonitoring	5875	26	41.57
sat	6435	36	9.28
waveform	5000	21	2.04
isolet	7797	617	24.99

Table 4
Details of the multi-class data sets.

Data set	# instances	# attributes	#class	IR
Balance	625	4	3	5.88
Ecoli	336	7	5	7.15
Glass	205	9	5	5.85
Hayes-roth	132	4	3	1.7
New-thyroid	215	5	3	5

Table 5
Parameter settings of the compared algorithms.

Method	Parameter
GL	Number of Gaussian components=5, Number of iterations=10000, Experiment times=30
SMOTE	Neighbors=5, Type of SMOTE=both, Balancing=Y, Quality of generated examples=1, Type of Interpolation =standard, $\mu = 0.5$, $\alpha = 0.5$
SMOTE-ENN	Neighbors ENN=3, Neighbors SMOTE =5, Type of SMOTE=both, Balancing=Y, Quality of generated examples=1
SMOTE-TL	Neighbors=5, Type of SMOTE=both, Balancing=Y, Quality of generated examples=1
B1-SMOTE	Number of neighbors for SMOTE=5, Type of Borderline SMOTE=1, Type of SMOTE=both, Balancing=Y, Quantity of generated examples=1
SL-SOMTE	Number of neighbors=5, Type of SMOTE =both, Balancing=Y, Quantity of generated examples=1
k-means SMOTE	kϵ{2,20,50,100,250,500},knnϵ{3,5,20, ∞}, irtϵ{1, ∞ }, de ϵ{0,2,number of features}
MWMOTE	kNoisy = 5, kMajority = 3, kMinority, threshold = 5, cmax = 2, cclustering = 3, classAttr = "Class"

SMOTE variants were introduced to compare time performance with GL. Table 5 presents the specific parameter configurations of the compared algorithms. SVM was used as the base learning model for GL and the compared algorithms.

4.3. Statistical tests

Hypothesis testing is a widely used technique in statistics. In experimental studies, it can be used to analyze whether or not significant differences exist in the experimental results of various methods and how different they are [50,51]. We used the non-parametric Wilcoxon signed rank hypothesis test [52,53] due to the unknown overall distribution of the experimental results. In the Wilcoxon test, we calculated the D-value of the experimental results of the two comparative methods and ranked the D-value in ascending order of the absolute value. All positive and negative D-values were summed separately and recorded as R^+ and R^-,

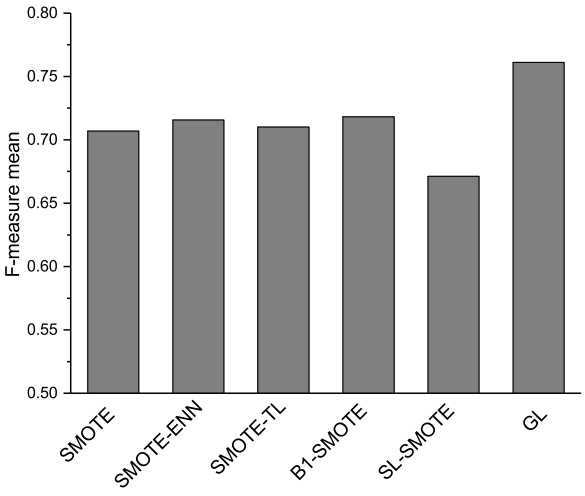


Fig. 4. Average results (F-measure) obtained in Table 6.

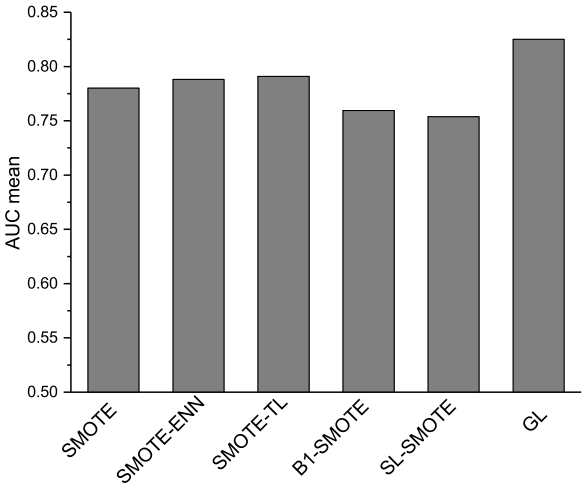


Fig. 5. Average results (AUC) obtained in Table 7.

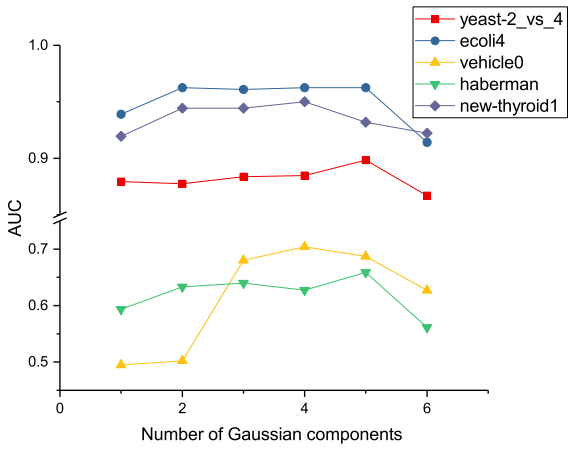


Fig. 6. Relationship between the number of Gaussian components and AUC.

respectively. If the difference between R^+ and R^- is sufficiently large, the null hypothesis that no differences exist between the two methods is rejected. The Wilcoxon test is also based on a comparison of the calculated p-value with a given significance level α to determine whether or not the null hypothesis should be rejected.

4.4. Comparison results

In this section, we show the experimental results of the F-measure and AUC by comparing the GL method with five typical imbalanced preprocessing methods, namely, SMOTE, SMOTE-ENN, SMOTE-TomekLinks, borderline-SMOTE, and safe-level-SMOTE, on real-world data sets. We implemented a Wilcoxon signed rank hypothesis test and performed an analysis based on these results.

4.4.1. Experimental results

Table 6 shows the F-measure results of our proposed method and other methods on 30 data sets. Table 7 presents the AUC results of the experimental study. The best method for each data set is highlighted in bold. 4 shows the average F-measure results of the approaches, and 5 provides the average AUC results.

Tables 6 and 7 show that the GL method outputs higher values for most of the data sets whether F-measure or AUC. The good performance of our method can be observed based on results compared with the other methods. For F-measure and AUC, the result values the proposed method outputs are significantly higher than those of other methods for certain data sets, such as “glass-0-1-6 vs 2”, “segment0”, and “new-thyroid1”. Furthermore, the average result value obtained by GL on 30 real-world data sets is the highest among all compared methods, as shown in . The comparison results strongly suggest that our method is more effective than the other algorithms in imbalanced problems.

4.4.2. Hypothesis testing result

To provide statistical support to the results previously obtained, the Wilcoxon signed rank test was carried out in this experiment to analyze whether significant differences exist in the experimental results between the proposed and five compared methods. Table 8 shows the Wilcoxon test results on F-measure. Table 9 shows the Wilcoxon test results on AUC.

All R^+ values are significantly higher than R^- values in all competition cases for F-measure and AUC. All R^+ are over 300, whereas all R^- are less than 100, which suggests that GL gains advantages in all comparisons. At the same time, all p-values are far less than the given significance level α, that is, 0.05. Therefore, GL is obviously and substantially different from the five compared methods. According to these observations, the performance of our method is higher than that of the five methods in general. Therefore, we deduce that the proposed method is efficient for imbalanced learning.

4.5. Empirical study of the number of Gaussian components

As depicted in Section 3.4, the number of Gaussian components K is a key parameter of our proposal. To study the relationship between the number of Gaussian components and performance, we conducted an experimental study. We varied K from 1 to 6. For each K value, a complete GL learning process was executed on each data set. We used line charts to show the results, as shown in Figs. 6 and 7. We selected five data sets for this study.

Figs. 6 and 7 show that when the numbers of Gaussian components are 1 and 6, the values of AUC and F-measure are relatively low. The performance of the proposed method can be improved when the number of Gaussian components ranges from 1 to 2, especially for “ecoli4” and “new-thyroid1”. When the number of Gaussian components ranges from 5 to 6, a decrease in the performance level is apparent. When the number of Gaussian

Table 6
F-measure results on the real-world data sets.

Data set	SMOTE	SMOTE-ENN	SMOTE-TL	B1-SMOTE	SL-SMOTE	GL
yeast-2_vs_4	0.7968	0.8032	0.7845	0.8228	0.7666	**0.8571**
ecoli4	0.8755	0.8718	0.8659	**0.9053**	0.7734	0.8940
glass-0-1-6_vs_2	0.5516	0.5558	0.5378	0.5468	0.5527	**0.6549**
abalone9-18_vs_2	0.5673	0.5716	0.5519	**0.6384**	0.5171	0.6101
ecoli2_vs_2	0.8383	0.8342	0.8169	0.8385	0.8376	**0.8449**
glass4	0.8506	0.8330	0.8314	0.8160	0.7609	**0.9353**
kr-vs-k-zero–one_vs_draw	0.9388	0.9347	0.9355	**0.9629**	0.9120	0.8888
new-thyroid1	0.8186	0.8553	0.8674	0.8214	0.7245	**0.9438**
segment0	0.8855	0.9176	0.9199	0.9184	0.7750	**0.9784**
shuttle-c0-vs-c4	0.9076	0.9352	0.9382	0.8877	0.6795	**0.9705**
winequality-white-3_vs_7	0.5038	0.5091	0.5158	0.5168	0.4853	**0.5293**
glass2	0.5812	0.5904	0.5601	0.5709	0.5788	**0.6904**
newthyroid2	0.7850	0.8237	0.8314	0.7591	0.6790	**0.9196**
vehicle0	0.4759	0.5071	0.5129	0.4811	0.4714	**0.7193**
winequality-red-8_vs_6	0.5271	0.5315	0.5218	0.5245	0.5041	**0.5434**
led7digit-0-2-4-5-6-7-8-9_vs_1	0.7379	0.7423	0.7337	0.7707	0.7250	**0.8102**
abalone19	0.4569	0.4567	0.4548	**0.5057**	0.4438	0.4577
glass0	0.7554	0.7565	0.7355	0.7389	0.7451	**0.7701**
haberman2	0.4762	0.5066	0.5273	0.5082	0.4810	**0.6400**
iris0	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
vehicle2	0.4720	0.5568	0.5492	0.4809	0.4308	**0.6741**
yeast4	0.6120	0.6093	0.6066	**0.6428**	0.5995	0.6190
kr-vs-k-one_vs_fifteen	0.9643	0.9912	0.9874	**1.0000**	0.8914	**1.0000**
ecoli1	0.8395	0.8242	0.8152	0.8349	0.8300	**0.8464**
ecoli3	0.7653	0.7566	0.7340	**0.7718**	0.7564	0.7535
yeast-0-5-6-7-9_vs_4	0.6665	0.6665	0.6520	**0.6863**	0.6746	0.6744
yeast-2_vs_8	0.4205	**0.4221**	0.4100	0.4107	0.4157	0.3872
ecoli-0-1_vs_5	0.5114	0.4782	0.4782	0.5126	0.5126	**0.5127**
ecoli-0_vs_1	0.9686	0.9682	0.9693	0.9693	0.9690	**0.9707**
car-vgood	0.6584	0.6588	0.6582	0.7023	0.6445	**0.7576**

Table 7
AUC results on the real-world data sets.

Data set	SMOTE	SMOTE-ENN	SMOTE-TL	B1-SMOTE	SL-SMOTE	GL
yeast-2_vs_4	0.8824	0.8846	0.8770	0.8911	0.8695	**0.8984**
ecoli4	0.9547	0.9531	0.9500	0.9406	0.9311	**0.9625**
glass-0-1-6_vs_2	0.6176	0.6126	0.6147	0.5657	0.6181	**0.7528**
abalone9-18_vs_2	0.7648	0.7606	0.7699	0.7505	0.7054	**0.7751**
ecoli2_vs_2	**0.9071**	0.9053	0.8965	0.9006	**0.9071**	0.8977
glass4	0.9230	0.9231	0.8426	0.8986	0.9226	**0.9279**
kr-vs-k-zero–one_vs_draw	0.9900	0.9898	**0.9946**	0.9649	0.9827	0.9693
new-thyroid1	0.7810	0.8194	0.8319	0.7764	0.6964	**0.9319**
segment0	0.8338	0.8760	0.8790	0.8774	0.7164	**0.9725**
shuttle-c0-vs-c4	0.8551	0.8946	0.8987	0.8304	0.6250	**0.9500**
winequality-white-3_vs_7	0.5102	0.5045	0.5301	0.5193	0.4824	**0.6898**
glass2	0.7919	0.7969	0.7722	0.6540	0.7890	**0.7975**
newthyroid2	0.7671	0.8186	0.8299	0.7373	0.6699	**0.9069**
vehicle0	0.5214	0.5392	0.5419	0.5246	0.5196	**0.6871**
winequality-red-8_vs_6	0.5867	0.5906	0.5875	0.5844	0.5344	**0.6094**
led7digit-0-2-4-5-6-7-8-9_vs_1	0.8570	0.8702	**0.8747**	0.8316	0.8671	0.8616
abalone19	0.7167	0.7430	0.7405	0.5929	0.7272	**0.7432**
glass0	0.7907	0.7945	0.7829	0.7731	0.7802	**0.8052**
haberman	0.4786	0.5141	0.6032	0.5166	0.4828	**0.6587**
iris0	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
vehicle2	0.5229	0.5708	0.5661	0.5275	0.5016	**0.6515**
yeast4	0.8395	0.8381	0.8498	0.8244	0.8328	**0.8576**
kr-vs-k-one_vs_fifteen	0.9972	0.9993	0.9991	**1.0000**	0.9901	**1.0000**
ecoli1	**0.8858**	0.8762	0.8705	0.8786	0.8799	0.8856
ecoli3	**0.8983**	0.8933	0.8817	0.8925	0.8933	0.8927
yeast-0-5-6-7-9_vs_4	0.7995	0.7995	0.7979	0.7890	**0.8112**	0.7927
yeast-2_vs_8	0.5012	0.5023	0.4885	0.4337	0.4960	**0.5202**
ecoli-0-1_vs_5	0.5227	0.5000	0.5000	**0.5250**	**0.5250**	**0.5250**
ecoli-0_vs_1	0.9654	0.9625	0.9690	**0.9721**	0.9692	0.9683
car-vgood	0.9083	0.9098	0.9083	0.8675	0.9100	**0.9207**

components ranges between 3 and 5, the obtained performance achieves a high level and tends to be stable.

For "vehicle0", a distinct leap occurs in the number of Gaussian components from 2 to 3. Therefore, we conclude that data generated by 1 or 2 Gaussian components do not accurately fit the distribution of the original data in "vehicle0". Fig. 8(d) shows that the distribution of data that are generated by three Gaussian components is consistent with the distribution of the original data in Fig. 8(a). The data generated by one Gaussian component substantially deviates from the distribution of the original data, which results in low performance. When the number of Gaussian components ranges between 3 and 5, AUC and F-measure fluctuate within a small range. However, when the number increases to 6, the two values decrease.

These observations indicate that the number of Gaussian components directly affects classification performance. Using too

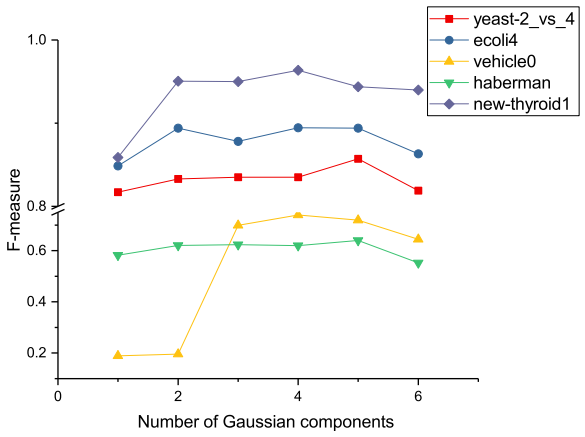


Fig. 7. Relationship between the number of Gaussian components and F-measure.

Table 8
Wilcoxon test results of GL versus five Methods on F-measure.

Method	R^+	R^-	*p*-value
GL vs. SMOTE	398	37	0.000095
GL vs. SMOTE-ENN	401	34	0.000073
GL vs. SMOTE-TL	410	25	0.000031
GL vs. B1-SOMOTE	318	88	0.008826
GL vs. SL-SMOTE	409	26	0.000035

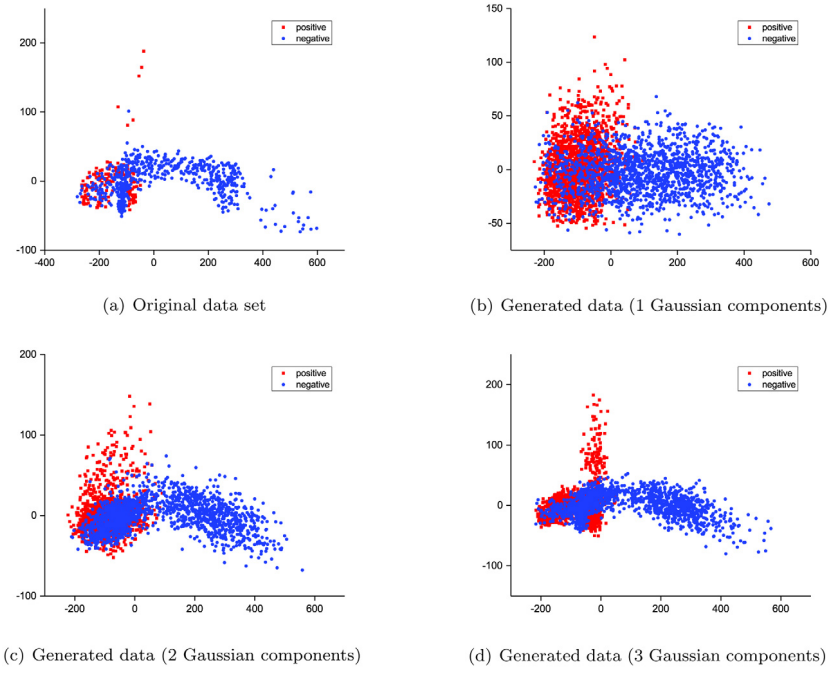
Table 9
Wilcoxon test results of GL versus five Methods on AUC.

Method	R^+	R^-	*p*-value
GL vs. SMOTE	362	73	0.001781
GL vs. SMOTE-ENN	372	63	0.000835
GL vs. SMOTE-TL	380	55	0.000442
GL vs. B1-SOMOTE	372	6	0.000011
GL vs. SL-SMOTE	365	41	0.000225

many or too few Gaussian components to fit and generate data will decrease the classification performance because the generated data do not accurately fit the distribution of the original data. Using only one Gaussian component for fitting the data is extremely rough, whereas using too many Gaussian components to fit the data may aggregate the data that should not be clustered into a cluster. Using three to five Gaussian components is the best choice for obtaining increased classification performance.

4.6. Relationship between generated data scale and performance

As depicted in Section 3.3, the number of training instances is important for algorithm performance. To study the impact of data scale on algorithm performance, we designed a set of experiments. We set the number of generated data to 1 and 10 times the number of the original data. For each scale, a complete GL learning process was executed on each binary KEEL data set. Figs. 9 and 10 depict the result of the generated data scale on classification performance. The values of the measure metrics change in the process of the generated data that are 1 to 10 times the amount of the original data. From the results in Figs. 9 and 10, some interesting pattern can be extracted. For AUC and F-measure, the values increase continuously at the initial stage then tend to be stable with the growth of data size. When the amount of the generated data is relatively small, the values of AUC and F-measure are low because a small amount of generated data results in inadequate training. When the amount of generated data increases at the later stage, the values of the two metrics fluctuate within a small range. A significant leap occurs when the amount of generated data increases from 1 to 3 times the number of the original data for "vehicle0". Hence, we generated 3000 or 6000 new instances in the experiment based on the amount of the original data. In this manner, classification performance is not influenced by a small amount of generated data, and the training time is not extremely long because of the appropriate amount of generated data.



(a) Original data set

(b) Generated data (1 Gaussian components)

(c) Generated data (2 Gaussian components)

(d) Generated data (3 Gaussian components)

Fig. 8. Case study of vehicle0 data set.

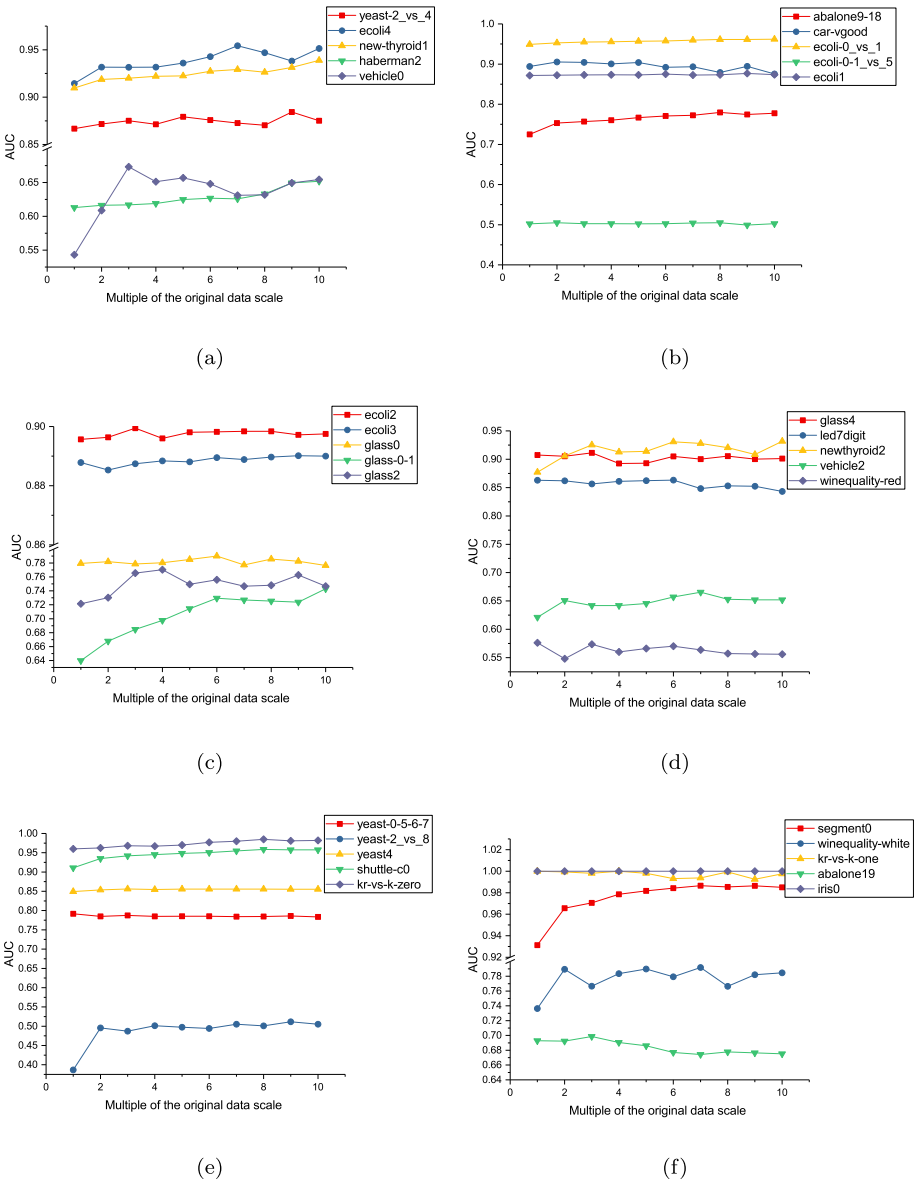


Fig. 9. Relationship between the generated data scale and AUC.

4.7. Time comparison of algorithms on large-scale data

In [54], imbalanced big data classification problem is referred to a challenge that forces us to develop computationally effective solutions for processing them. Therefore, we recorded runtime in seconds of our proposal and compared algorithms on large-scale data sets. We also analyzed the computational complexity of the algorithms in Table 10. Where, $|T_{bor}|$ is the number of borderline instances and $|T_{safe}|$ is the number of safe instances. c and f represent the number of clusters of k-means and filtered clusters, respectively. And t is the number of iterations. $|T_{minf}|$ is the set in which the noise minority instances are removed. $|T_{bmaj}|$ is a borderline majority set which includes the nearest majority instances derived for each instance in $|T_{minf}|$. $|T_{imin}|$ is the informative minority set which includes the nearest minority instances derived for each instance in $|T_{bmaj}|$.

The results of runtime are presented in Table 11. GL runs fastest on some data sets. GL defeats several of the compared algorithms on half of the data sets. For "nursery", the runtime of our method is the shortest. For "letter-recognition" and "waveform", our method defeats 4 of the 7 methods. We can conclude that GL used for dealing with large-scale imbalanced data in some cases is computationally effective. However, in the case of the data with a large number of features, our method runs longer. The reason is that GMM used to fit and generate high-dimensional feature data is difficult. In addition, B1-SMOTE uses the shortest time on most data sets, because it only generates new data on the boundary. The runtime of SMOTE-ENN and SMOTE-TL are similar because of

Table 10
Computational complexity of the compared algorithms.

Method	Complexity
SMOTE	$O(\|T_{min}\| * \frac{\|T_{maj}\|}{\|T_{min}\|})$
SMOTE-ENN	$O(\|T_{min}\| * \frac{\|T_{maj}\|}{\|T_{min}\|} + \|T_{min}\|)$
SMOTE-TL	$O(\|T_{min}\| * \frac{\|T_{maj}\|}{\|T_{min}\|} + \|T_{min}\|)$
B1-SMOTE	$O(\|T_{min}\| + \|T_{bor}\| * \frac{\|T_{maj}\| - \|T_{safe}\|}{\|T_{bor}\|})$
k-means SMOTE	$O(\|T\| * c * t + c + f + f)$
MWMOTE	$O(\|T_{min}\| + \|T_{minf}\| + \|T_{bmaj}\| + \|T_{bmaj} + T_{imin}\| + \|T_{imin}\| + \|T_{maj} - T_{min}\|)$

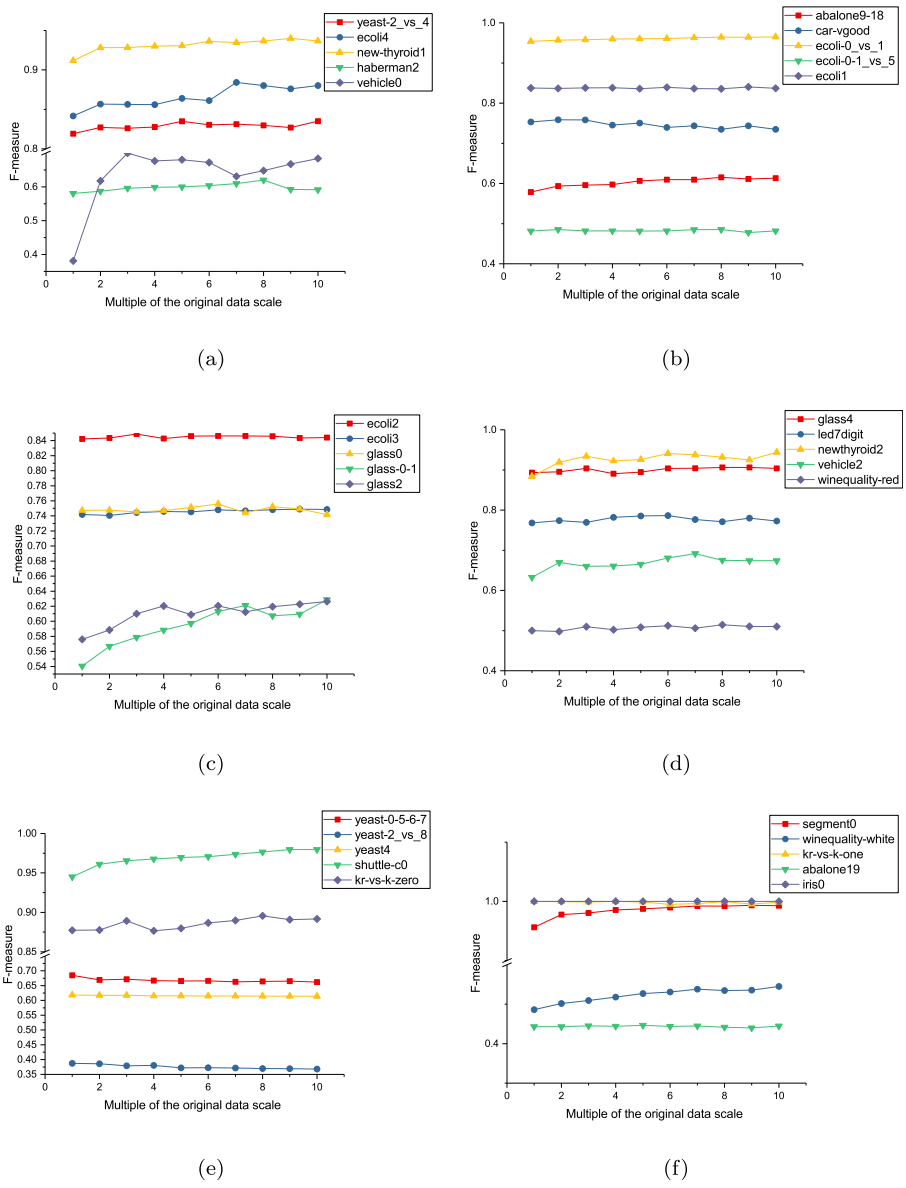


Fig. 10. Relationship between the generated data scale and F-measure.

Table 11
Runtime of GL and compared algorithms.

Data set	SMOTE	SMOTE-ENN	SMOTE-TL	B1-SMOTE	k-means SMOTE	MWMOTE	GL
Chesscking-Rook vs. King	17.7631	35.5580	39.5259	16.6767	228.1749	28.5200	884.5468
HTRU2	17.6971	16.5834	19.7735	14.4864	165.7968	234.9600	831.4628
letter-recognition	27.1347	84.6436	68.8150	22.6048	229.2436	71.7600	58.8416
musk	59.0028	165.4555	129.3441	57.4616	404.5417	209.2700	6000.2846
nursery	9.9001	14.3004	13.4830	10.2073	175.7179	3500.3800	7.5683
page-block0	4.2649	5.1133	6.0213	4.1457	137.1886	17.9800	6.1004
parkinsons telemonitoring	9.1021	11.5375	9.7347	8.1210	146.1518	14.8100	3118.9562
sat	13.3062	36.7572	27.8205	12.0689	206.4954	38.7700	32.2026
waveform	10.2665	29.7320	19.4344	9.8245	165.4301	203.2100	16.4958
isolet	204.4203	1989.4092	1319.2137	227.0167	1458.9056	694.4100	2718.3578

the same computational complexity. In most cases, SMOTE runs shorter time than SMOTE-ENN and SMOTE-TL because it has less computational complexity than them.

4.8. Experimental results on multi-class imbalanced data sets

For multi-class imbalanced data sets, the class with the largest number of instances is considered as the majority class. Each class except the majority class is oversampled as equally as the size of the majority class in all methods. The experimental results on multi-class data sets are shown in Table 12. Our method outputs the highest value both AUC and F-measure on 4 of 5 data sets. We can draw that GL is effective compared to other methods in dealing with multi-class imbalanced data.

Previous imbalanced methods designed to handle binary problems have limitations for classification of multi-class imbalanced

Table 12
Experimental results on multi-class imbalanced data sets.

Data set	SMOTE		SMOTE-ENN		SMOTE-TL		B1-SMOTE		GL	
	MAUC	F-measure	MAUC	F-measure	MAUC	F-measure	MAUC	F-measure	MAUC	F-measure
Balance	0.5429	0.5516	0.5612	0.5489	0.5360	0.5476	0.5488	0.5558	**0.6476**	**0.6165**
Ecoli	0.7414	0.7333	0.7678	0.7517	0.7356	0.7241	0.7483	0.7393	**0.7827**	**0.7543**
Glass	0.6730	0.6768	0.6523	0.5982	0.6672	0.6674	**0.6769**	**0.6799**	0.6065	0.6037
Hayes-roth	0.8222	0.8214	0.5144	0.4720	0.8222	0.8217	0.8222	0.8214	**0.8232**	**0.8224**
New-thyroid	0.8953	0.8985	0.8842	0.8778	0.8939	0.8950	0.8860	0.8939	**0.9205**	**0.9153**

data. As depicted in Section 1, SMOTE generates wrong instances in the majority class region which results in over generalization. Borderline-SMOTE needs to find the borderline instances and then generate new data. For multi-class data, the borderline instances are difficult to determine, which may generate overlapping data and cause over-fitting. However, our proposal avoids these problems and generates new data based on the distribution of original data. Therefore, generated data satisfy adequately the distribution of the whole multi-class data.

5. Conclusion

We develop a new method called GL to address imbalanced problems. In the GL method, new instances are generated based on the distribution of the original data by GMM. The generated data, including synthetic minority and majority class instances, are then used to train the learning model for testing. The most distinct characteristic of GL is that it synthesizes instances by fitting the original data distributions rather than local neighbor information. This setup enables GL to generate new instances in a global and distributional manner. Compared with other oversampling algorithms, our proposal can obtain more accurate discrimination information for imbalanced data and is thus more effective for imbalanced learning. Furthermore, our method is validated on real-world data sets. The experimental results prove that GL exhibits the highest performance among the compared methods. Furthermore, the statistical hypothesis test results indicate that GL is considerably different from other methods. We recommend several directions for future work. GL can be applied to ensemble learning for imbalanced problems and even to the classification of standard data.

Acknowledgments

This research was partially supported by the National Natural Science Foundation of China under grant No. 61472164, No. 61573166, No. 61572230, and No. 61672262, the Shandong Provincial Key R&D Program, China under Grants No. 2017CXZC1206, No. 2016GGX101001, and No. 2018CXGC0706, the Doctoral Fund of University of Jinan, China under grant No. XBS1623, and No. XBS1523.

References

[1] M.A. Mazurowski, P.A. Habas, J.M. Zurada, J.Y. Lo, J.A. Baker, G.D. Tourassi, Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance, Neural Netw. 21 (2–3) (2008) 427–436, http://dx.doi.org/10.1016/j.neunet.2007.12.031.

[2] T. Fawcett, F. Provost, Adaptive fraud detection, Data Min. Knowl. Discovery 1 (3) (1997) 291–316, http://dx.doi.org/10.1023/A:1009700419189.

[3] Y.-M. Huang, C.-M. Hung, H.C. Jiau, Evaluation of neural networks and data mining methods on a credit assessment task for class imbalance problem, Nonlinear Anal. RWA 7 (4) (2006) 720–747, http://dx.doi.org/10.1016/j.nonrwa.2005.04.006.

[4] I. Chaïri, S. Alaoui, A. Lyhyaoui, Intrusion detection based sample selection for imbalanced data distribution, in: Innovative Computing Technology (INTECH), 2012 Second International Conference on, IEEE, 2012, pp. 259–264, http://dx.doi.org/10.1109/intech.2012.6457778.

[5] S. Suresh, N. Sundararajan, P. Saratchandran, Risk-sensitive loss functions for sparse multi-category classification problems, Inform. Sci. 178 (12) (2008) 2621–2638, http://dx.doi.org/10.1016/j.ins.2008.02.009.

[6] 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, http://dx.doi.org/10.1613/jair.953.

[7] G.E. Batista, R.C. Prati, M.C. Monard, A study of the behavior of several methods for balancing machine learning training data, ACM SIGKDD Explor. News Lett. 6 (1) (2004) 20–29, http://dx.doi.org/10.1145/1007730.1007735.

[8] H. Han, W.Y. Wang, B.H. Mao, Borderline-smote: A new over-sampling method in imbalanced data sets learning, Lecture Notes in Comput. Sci. 3644 (5) (2005) 878–887, http://dx.doi.org/10.1007/11538059_91.

[9] C. Bunkhumpornpat, K. Sinapiromsaran, C. Lursinsap, Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, 2009, pp. 475–482, http://dx.doi.org/10.1007/978-3-642-01307-2_43.

[10] S. Barua, M.M. Islam, X. Yao, K. Murase, Mwmote–majority weighted minority oversampling technique for imbalanced data set learning, IEEE Trans. Knowl. Data Eng. 26 (2) (2014) 405–425, http://dx.doi.org/10.1109/tkde.2012.232.

[11] V. López, A. Fernández, S. García, V. Palade, F. Herrera, An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics, Inform. Sci. 250 (2013) 113–141, http://dx.doi.org/10.1016/j.ins.2013.07.007.

[12] J.A. Sáez, B. Krawczyk, M. Woźniak, Analyzing the oversampling of different classes and types of examples in multi-class imbalanced datasets, Pattern Recognit. 57 (2016) 164–178, http://dx.doi.org/10.1016/j.patcog.2016.03.012.

[13] G. Douzas, F. Bacao, Self-organizing map oversampling (somo) for imbalanced data set learning, Exp. Syst. Appl. 82 (2017) 40–52, http://dx.doi.org/10.1016/j.eswa.2017.03.073.

[14] W.-C. Lin, C.-F. Tsai, Y.-H. Hu, J.-S. Jhang, Clustering-based undersampling in class-imbalanced data, Inform. Sci. 409 (2017) 17–26, http://dx.doi.org/10.1016/j.ins.2017.05.008.

[15] I. Triguero, M. Galar, D. Merino, J. Maillo, H. Bustince, F. Herrera, Evolutionary undersampling for extremely imbalanced big data classification under apache spark, in: Evolutionary Computation (CEC), 2016 IEEE Congress on, IEEE, 2016, pp. 640–647, http://dx.doi.org/10.1109/cec.2016.7743853.

[16] H. He, Y. Bai, E.A. Garcia, S. Li, Adasyn: Adaptive synthetic sampling approach for imbalanced learning, in: Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on, IEEE, 2008, pp. 1322–1328, http://dx.doi.org/10.1109/ijcnn.2008.4633969.

[17] E. Ramentol, Y. Caballero, R. Bello, F. Herrera, Smote-rsb*: a hybrid preprocessing approach based on oversampling and undersampling for high imbalanced data-sets using smote and rough sets theory, Knowl. Inform. Syst.ems 33 (2) (2012) 245–265, http://dx.doi.org/10.1007/s10115-011-0465-6.

[18] P.K. Chan, S.J. Stolfo, Toward Scalable Learning with Non-Uniform Class and Cost Distributions: A Case Study in Credit Card Fraud Detection, vol. 98, KDD, 1998, pp. 164–168, http://dx.doi.org/10.1.1.49.5098.

[19] N. Verbiest, E. Ramentol, C. Cornelis, F. Herrera, Preprocessing noisy imbalanced datasets using smote enhanced with fuzzy rough prototype selection, Appl. Soft Comput. 22 (5) (2014) 511–517, http://dx.doi.org/10.1016/j.asoc.2014.05.023.

[20] X.-Y. Liu, J. Wu, Z.-H. Zhou, Exploratory undersampling for class-imbalance learning, IEEE Trans. Syst. Man Cybern. B 39 (2) (2009) 539–550, http://dx.doi.org/10.1109/tsmcb.2008.2007853.

[21] H. Yu, J. Ni, J. Zhao, Acosampling: An ant colony optimization-based undersampling method for classifying imbalanced dna microarray data, Neurocomputing 101 (2013) 309–318, http://dx.doi.org/10.1016/j.neucom.2012.08.018.

[22] F. Cheng, J. Zhang, C. Wen, Cost-sensitive large margin distribution machine for classification of imbalanced data, Pattern Recognit. Lett. 80 (2016) 107–112, http://dx.doi.org/10.1016/j.patrec.2016.06.009.

[23] S. Shinde, S. Sayyad, Cost sensitive improved levenberg marquardt algorithm for imbalanced data, in: Computational Intelligence and Computing Research (ICCIC), 2016 IEEE International Conference on, IEEE, 2016, pp. 1–4, http://dx.doi.org/10.1109/iccic.2016.7919598.
[24] B. Zadrozny, J. Langford, N. Abe, Cost-sensitive learning by cost-proportionate example weighting, in: Data Mining, 2003. ICDM 2003. Third IEEE International Conference on, IEEE, 2003, pp. 435–442, http://dx.doi.org/10.1109/icdm.2003.1250950.
[25] Z.-H. Zhou, X.-Y. Liu, Training cost-sensitive neural networks with methods addressing the class imbalance problem, IEEE Trans. Knowl. Data Eng. 18 (1) (2006) 63–77, http://dx.doi.org/10.1109/tkde.2006.17.
[26] K.M. Ting, An instance-weighting method to induce cost-sensitive trees, IEEE Trans. Knowl. Data Eng. 14 (3) (2002) 659–665, http://dx.doi.org/10.1109/tkde.2002.1000348.
[27] K. Veropoulos, C. Campbell, N. Cristianini, et al., Controlling the sensitivity of support vector machines, in: Proceedings of the International Joint Conference on AI, vol. 55, 1999, p. 60, http://dx.doi.org/10.1.1.42.7895.
[28] C. Qiu, L. Jiang, G. Kong, A differential evolution-based method for class-imbalanced cost-sensitive learning, in: Neural Networks (IJCNN), 2015 International Joint Conference on, IEEE, 2015, pp. 1–8, http://dx.doi.org/10.1109/ijcnn.2015.7280419.
[29] Z. Liu, C. Gao, H. Yang, Q. He, A cost-sensitive sparse representation based classification for class-imbalance problem, Sci. Program. 2016 (12) (2016) 8, http://dx.doi.org/10.1155/2016/8035089.
[30] A.C. Bahnsen, D. Aouada, B. Ottersten, Example-dependent cost-sensitive decision trees, Expert Syst. Appl. 42 (19) (2015) 6609–6619, http://dx.doi.org/10.1016/j.eswa.2015.04.042.
[31] B. Krawczyk, M. Woźniak, G. Schaefer, Cost-sensitive decision tree ensembles for effective imbalanced classification, Appl. Soft Comput. 14 (1) (2014) 554–562, http://dx.doi.org/10.1016/j.asoc.2013.08.014.
[32] N.V. Chawla, A. Lazarevic, L.O. Hall, K.W. Bowyer, Smoteboost: Improving prediction of the minority class in boosting, in: European Conference on Principles of Data Mining and Knowledge Discovery, Springer, 2003, pp. 107–119, http://dx.doi.org/10.1007/978-3-540-39804-2_12.
[33] C. Seiffert, T.M. Khoshgoftaar, J. Van Hulse, A. Napolitano, Rusboost: A hybrid approach to alleviating class imbalance, IEEE Trans. Syst., Man, Cyber.-A: Syst. Humans 40 (1) (2010) 185–197, http://dx.doi.org/10.1109/tsmca.2009.2029559.
[34] P. Lim, C.K. Goh, K.C. Tan, Evolutionary cluster-based synthetic oversampling ensemble (eco-ensemble) for imbalance learning, IEEE Trans. Cyber. 47 (9) (2017) 2850–2861, http://dx.doi.org/10.1109/tcyb.2016.2579658.
[35] S. Liu, Y. Wang, J. Zhang, C. Chen, Y. Xiang, Addressing the class imbalance problem in twitter spam detection using ensemble learning, Comput. Secur. 69 (2017) 35–49, http://dx.doi.org/10.1016/j.cose.2016.12.004.
[36] K.-J. Wang, B. Makond, K.-H. Chen, K.-M. Wang, A hybrid classifier combining smote with pso to estimate 5-year survivability of breast cancer patients, Appl. Soft Comput. 20 (7) (2014) 15–24, http://dx.doi.org/10.1016/j.asoc.2013.09.014.
[37] J.F. Díez-Pastor, J.J. Rodríguez, C.I. García-Osorio, L.I. Kuncheva, Diversity techniques improve the performance of the best imbalance learning ensembles, Inform. Sci. 325 (2015) 98–117, http://dx.doi.org/10.1016/j.ins.2015.07.025.
[38] B. Krawczyk, M. Galar, Ł. Jeleń, F. Herrera, Evolutionary undersampling boosting for imbalanced classification of breast cancer malignancy, Appl. Soft Comput. 38 (C) (2016) 714–726, http://dx.doi.org/10.1016/j.asoc.2015.08.060.
[39] Z. Ji, Y. Xia, Y. Zheng, Robust generative asymmetric gmm for brain mr image segmentation, Comput. Methods Progr. Biomed. 151 (2017) 123–138, http://dx.doi.org/10.1016/j.cmpb.2017.08.017.
[40] A. Niranjil Kumar, C. Sureshkumar, Background subtraction in dynamic environment based on modified adaptive gmm with ttd for moving object detection, J. Electr. Eng. Technol. 10 (1) (2015) 372–378, http://dx.doi.org/10.5370/jeet.2015.10.1.372.
[41] J. Tao, Q. Li, C. Zhu, J. Li, A hierarchical naive bayesian network classifier embedded gmm for textural image, Int. J. Appl. Earth Obs. Geoinf. 14 (1) (2012) 139–148, http://dx.doi.org/10.1016/j.jag.2011.08.012.
[42] H. Xia, S. Song, L. He, A modified gaussian mixture background model via spatiotemporal distribution with shadow detection, Signal, Image Video Proc. 10 (2) (2016) 343–350, http://dx.doi.org/10.1007/s11760-014-0747-z.
[43] C. Stauffer, W.L. Grimson, Adaptive background mixture models for real-time tracking, in: Cvpr, IEEE, 1999, p. 2246, http://dx.doi.org/10.1109/cvpr.1999.784637.
[44] D.M. Powers, Evaluation: From precision, recall and f-factor to roc, informedness, markedness & correlation, J. Mach. Lear. Technol. 2 (2011) 2229–3981, http://dx.doi.org/10.1.1.214.9232.
[45] A.P. Bradley, The use of the area under the roc curve in the evaluation of machine learning algorithms, Pattern Recog. 30 (7) (1997) 1145–1159, http://dx.doi.org/10.1016/s0031-3203(96)00142-2.
[46] J. Huang, C.X. Ling, Using auc and accuracy in evaluating learning algorithms, IEEE Trans. Knowl. Data Eng. 17 (3) (2005) 299–310, http://dx.doi.org/10.1109/tkde.2005.50.
[47] A. Cano, A. Zafra, S. Ventura, Weighted data gravitation classification for standard and imbalanced data, IEEE Trans. Cybern. 43 (6) (2013) 1672–1687, http://dx.doi.org/10.1109/TSMCB.2012.2227470.
[48] T. Zhu, Y. Lin, Y. Liu, Synthetic minority oversampling technique for multiclass imbalance problems, Pattern Recognit. 72 (2017) 327–340, http://dx.doi.org/10.1016/j.patcog.2017.07.024.
[49] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera, Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework., J. Mult.-Valued Logic Soft Comput. 17 (2011) http://dx.doi.org/10.1.1.294.9855.
[50] S. García, A. Fernández, J. Luengo, F. Herrera, A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability, Soft Comput. 13 (10) (2009) 959, http://dx.doi.org/10.1007/s00500-008-0392-y.
[51] S. García, A. Fernández, J. Luengo, F. Herrera, Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power, Inform. Sci. 180 (10) (2010) 2044–2064, http://dx.doi.org/10.1016/j.ins.2009.12.010.
[52] D.J. Sheskin, Handbook of Parametric and Nonparametric Statistical Procedures, 2nd ed., CRC Press, 1997, http://dx.doi.org/10.4324/9780203489536, 382–382.
[53] W. Gregory, D. Foreman, Nonparametric Statistics for Non-Statisticians, Hoboken: John Wiley and Sons, 2009, http://dx.doi.org/10.1002/9781118165881.
[54] B. Krawczyk, Learning from imbalanced data: open challenges and future directions, Progr. Artif. Intell. 5 (4) (2016) 221–232, http://dx.doi.org/10.1007/s13748-016-0094-0.