

Received December 17, 2020, accepted December 26, 2020, date of publication December 30, 2020, date of current version January 11, 2021.

Digital Object Identifier 10.1109/ACCESS.2020.3047923

An Improving Majority Weighted Minority Oversampling Technique for Imbalanced Classification Problem

CHAO-RAN WANG AND XIN-HUI SHAO 

College of Sciences, Northeastern University, Shenyang 110819, China

Corresponding author: Xin-Hui Shao (xinhui1002@126.com)


This work was supported by the Central University Basic Scientific Research Business Expenses Special Funds (N2005013).

ABSTRACT Minority oversampling techniques have played a pivotal role in the field of imbalanced learning. While traditional oversampling algorithms can cause problems such as intra-class imbalance of samples, ignoring important information of boundary samples, and high similarity between new and old samples. Based on the situation, we proposed a new type of over-sampling method, BIRCH and Boundary Midpoint Centroid Synthetic Minority Over-Sampling Technique (BI-BMCSMOTE). First of all, the algorithm used the BIRCH clustering method to achieve quick cluster of the minority samples. After identifying and removing the noise, it marked the boundary minority samples in the label by probability. Secondly, it generated a density function for each sample cluster, calculated its density and sampling weight, performed midpoint composite sampling among the minority samples marked by probability and other minority samples in each cluster, and then calculated and analyzed the specific value of composite sampling to improve the accuracy of the model. According to the experimental results, the algorithm was proved to be valid.

INDEX TERMS Oversampling, boundary, minority sample, SMOTE, BIRCH, imbalanced learning.

I. INTRODUCTION

The imbalanced data [1] refers to the amount of one class or several classes of data in a dataset is far larger than that of the other classes. In two classes of the imbalanced data, the class with a larger amount is referred to as the majority class and the class with a smaller amount as the minority class. The data mining approaches have been used to establish the models and make decisions. But when it comes to the classification of the imbalanced data, the traditional classification model is not efficient. This is because the classification models drawn from the standard classifiers, such as logistic regression, support vector machine and decision tree, are not productive and distort some minority samples [2]; or because some exceptions are mistaken as noise, vice versa [3]. In addition, some imbalanced data has small samples and lacks density. The strong feature dimensions will limit the training on some classes by the learning models. The issues brought about by the imbalanced data can be found in many areas of data mining, such as credit card fraud [4], medical diagnosis [5], network intrusion [6], oil leakage [7], etc.

The associate editor coordinating the review of this manuscript and approving it for publication was Ali Salehzadeh-Yazdi .

The oversampling technique of synthesizing minority samples is a very popular technique to improve the performance of minority classes in imbalanced datasets.

Synthetic Minority Oversampling Technique (SMOTE) is the beginning of oversampling. After this, more and more experts joined in the team to improve the oversampling algorithm. In the past decade, the processing of imbalanced data has been carried out mainly in three directions, cost-sensitive learning [8], [9], algorithm modification [10] and data preprocessing [11]. Cost-sensitive learning assumes that the misclassified minority samples have a higher cost than the majority samples. This method can be implemented at both data level and algorithm level [2]. Algorithm modification is to improve the existing algorithm or classification paradigm to adapt to the learning of minority class. There are three classic techniques for data preprocessing, namely under-sampling [12], over-sampling [10], [13], [14], [15] and mixed sampling, which are also common methods. Under-sampling is a reasonable censorship of the majority class, while over-sampling is an effective supplement to the minority class data. Chawla *et al.* [16] proposed the Synthetic Minority Oversampling Technique (SMOTE), which synthesizes new samples by performing linear interpolation between minority

samples. However, simply copying a new sample can easily lead to overfitting. As a result, researchers have proposed many new SMOTE improved algorithms. Han *et al.* [17] put forward borderline-SMOTE algorithm, which emphasizes the classified boundary sample but overlooks the non-boundary sample interaction information. Adaptive Synthetic Sampling Technique (ADASYN) [18] can adaptively change the weights of minority samples according to the nearest neighbor ratio of majority and minority samples, and synthesize new samples to balance the skewed distribution of samples. Majority Weighted Minority Oversampling Technique (MWMOTE) [19] sets its importance according to the density of a minority samples in the cluster, calculates the weight to extract samples based on probability, but its interpolation method is still lagging. Density-Based Synthetic Minority Oversampling Technique (DBMOTE) [20] verifies the feasibility of using DBSCAN clustering and achieves good results in combination with SMOTE. DBSCAN and Midpoint Centroid Synthetic Minority Oversampling Technique (DB-MCSMOTE) [21] algorithm first performs DBSCAN clustering on minority samples, and the sampling weight of each cluster was calculated. Then a new sample is synthesized from two points in the cluster which are far apart. This method preserves the key information of each cluster and eliminates noise, but does not consider the influence of minority samples at the boundary.

This paper mainly studies the oversampling techniques for synthesizing minority samples. In order to better verify the validity of the new oversampling algorithm, this paper generates a scatter diagram of two-dimensional imbalanced data, and uses a visualization method to verify the accuracy of the membership degree of the synthetic data, which intuitively demonstrates the contribution of synthetic data to the expression of minority characteristics. Meanwhile, dataset of credit card default is added on the basis of multiple original datasets, on which the oversampling algorithm also presents better performance. On this basis, this paper adds a ratio analysis of the boundary samples and the normal samples, aiming to find out the optimal ratio to improve the performance of the new algorithm.

The main content of this paper is as follows. In section 2, we review SMOTE and describe the definitions, theorems, and other relevant knowledge that the BI-BMCSMOTE algorithm needs to master. In section 3, a new algorithm is proposed. In section 4, This paper conducts empirical research through experimental design. In section 5, The experimental results are obtained and analyzed. At the same time, the ratio between the marked boundary minority samples and the intra-cluster minority samples is analyzed.

II. REVIEW OF THEORY

A. SMOTE ALGORITHM

SMOTE method is a classical oversampling algorithm applied in synthetic minority samples of imbalanced learning, which is an improved algorithm based on random

oversampling, but not just simple replication of random oversampling. SMOTE method is to synthesize new samples manually by conducting linear interpolation among minority samples, and add synthetic minority samples to balance the dataset, thus alleviating the over fitting problem easily arising in random oversampling [16]. The algorithm flow is as follows:

- 1) According to the Euclidian distance, for each sample x_i of a minority class, the KNN algorithm is used to get its k nearest neighbors.
- 2) Sampling ratio $N\%$ was set according to sample imbalance rate. For each sample x_i of minority class samples, a few samples were randomly selected from k minority class neighbors, and the selected neighbor samples were assumed to be x'_i .
- 3) For each randomly selected neighbor sample x'_i , a new sample is constructed according to the following equation:

$$x_{new} = x_i + rand(0, 1) * (x'_i - x_i) \quad (1)$$

- 4) Add the synthesized new sample to the original data set to form a balanced dataset.

B. BIRCH ALGORITHM

Balanced Iterative Reducing and Clustering Using Hierarchies (BIRCH) is an incremental clustering method applying tree structure for fast clustering. This algorithm is suitable for the case of large samples, and introduces data points measured from multiple dimensions in an incremental and dynamic way, trying to produce the cluster with optimum quality within available resources (memory and time constraints). It runs so fast that it can achieve clustering with only one single scan of the dataset.

The BIRCH Algorithm's core concepts are defined as follows:

CF(Clustering Feature) [22]: Given N d-dimensional data points $\{x_i\}(i = 1, 2, \dots, N)$ in a cluster, the Clustering Feature (CF) vector of the cluster is defined as a triple: $CF = \langle N, \Sigma_l, \Sigma_s \rangle$, where N is the number of data points in the cluster, Σ_l is the linear sum of the N data points, i.e., $\sum_{n=1}^N x_n$, and Σ_s is the square sum of the N data points, i.e., $\sum_{n=1}^N x_n^2$.

(CF additivity theorem) [22]: Assume that $CF_1 = \langle N_1, \Sigma_{l_1}, \Sigma_{s_1} \rangle$, and $CF_2 = \langle N_2, \Sigma_{l_2}, \Sigma_{s_2} \rangle$ are the CF vectors of two disjoint clusters. Then the CF vector of the cluster that is formed by merging the two disjoint clusters is as following:

$$CF_1 + CF_2 = \langle N_1 + N_2, \Sigma_{l_1} + \Sigma_{l_2}, \Sigma_{s_1} + \Sigma_{s_2} \rangle \quad (2)$$

CF tree: A height-balanced tree, similar to the β tree, has three parameters: branch factor β , leaf factor λ and threshold τ . Among them, the branch factor β represents the maximum number of non-leaf nodes. Leaf factor λ represents the maximum number of leaf nodes. The threshold τ represents the maximum diameter of the CF stored in the leaf node. As shown in Fig. 1. CF tree consists of root node, branch node, and leaf node, internal nodes contain no more than

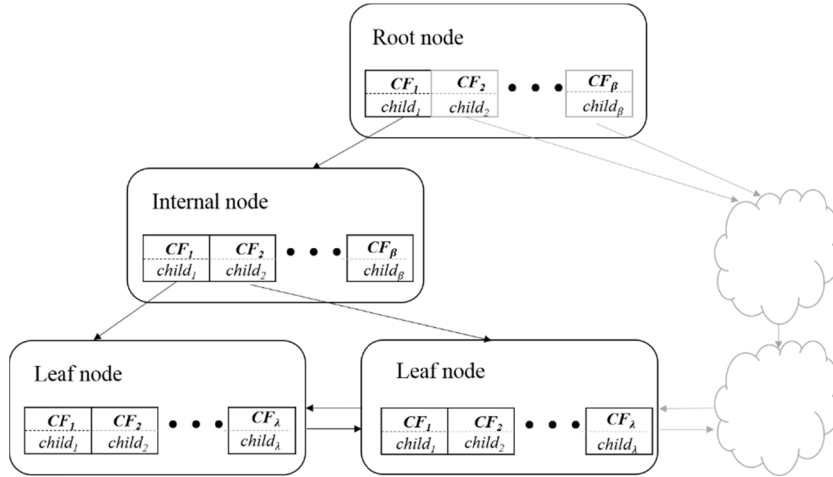


FIGURE 1. Clustering Feature Tree. Describes the establishment of CF tree.

entries in the shape of $[CF_i, child_i]$, where CF_i represents the clustering feature information of the i subcluster on the node, and pointer $child_i$ points to the i child node of the node. The leaf node contains no more than λ entries like an $[CF_i]$. In addition, each leaf node contains pointer $prev$ pointing to the previous leaf node and pointer $next$ pointing to the next leaf node. Each node represents a cluster formed by merging sub-clusters corresponding to clustering features in each entry. CF tree BIRCH Algorithm builds CF trees in memory for clustering by reading data points in turn. The BIRCH algorithm reads data points in turn, builds a CF tree in the memory for clustering, and then uses Agglomerative Clustering to cluster all CF entries globally to obtain a better CF tree to output the result.

III. BI-BMCSMOTE ALGORITHM

When synthesizing new samples, the traditional oversampling algorithm such as SMOTE omits the occurrence of some issues. First, the algorithm does not consider the necessity to filter off noise points, thereby resulting in the inclusion of many noises in the synthesized dataset. Second, the new sample is synthesized by the algorithm on the line connecting two points, so there is a possibility that the new sample may fall into the majority area between the two points. Third, the information of the boundary samples is of great importance, and ignoring the boundary samples will lead to an insignificant improvement outcome in the final classification.

In response to the above problems, this paper proposes a BIRCH and Boundary Midpoint Centroid Synthetic Minority Over-Sampling Technique (BI-BMCSMOTE), which mainly includes four steps: BIRCH clustering, marking boundary minority samples according to probability, calculating cluster density and giving the weight of the sample. Finally, using BI-BMCSMOTE to synthesize new minority samples. Fig. 2 shows the main steps of BI-BMCSMOTE algorithm.

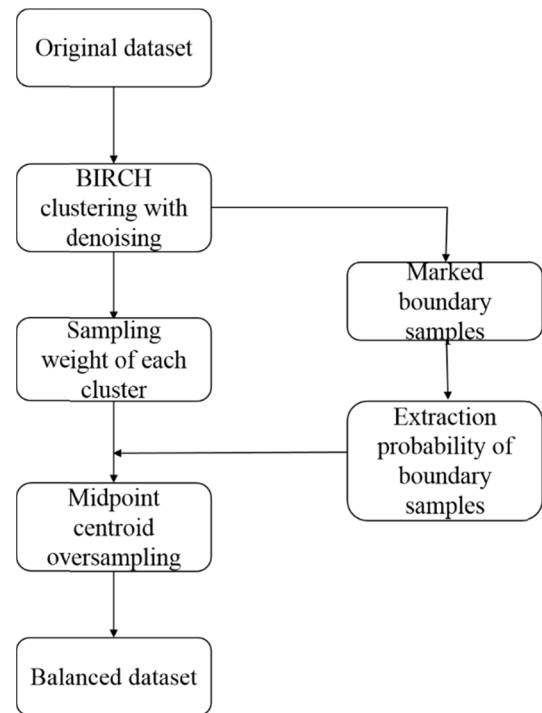


FIGURE 2. The main steps of BI-BMCSMOTE algorithm. Describes the flow of the BI-BMCSMOTE algorithm.

A. CLUSTER DENSITY DISTRIBUTION FUNCTION AND SAMPLING WEIGHT

Definition 1 (Cluster density distribution Function): The density distribution function of cluster C_i is defined as the proportional function of the number of sample points in cluster C_i and the volume of the hypersphere formed by the sample points in cluster C_i , the formula is as follows:

$$density(C_i) = NN_{C_i} / vol_{C_i}(S_d(r_i)) \quad (3)$$

where NN_{C_i} is the number of sample points in cluster C_i ; $vol_{C_i}(S_d(r_i))$ is the volume of the d-dimensional

hypersphere [33] formed by the sample points in the cluster C_i ; r_i is the Euclidean distance from the centroid \mathbf{u}_i to the farthest point in the cluster C_i , where $\mathbf{u}_i = \sum_{\mathbf{x} \in C_i} \mathbf{x} / |C_i|$.

$$vol_{C_i}(S_d(\mathbf{r}_i)) = K_d r_i^d \tag{4}$$

$$\begin{aligned} r_i &= \max_{\mathbf{x}_i \in C_i} \|\mathbf{x}_i - \mathbf{u}_i\|_2 \\ &= \max_{\mathbf{x}_i \in C_i} \sqrt{\sum_{j=1}^d |\mathbf{x}_{ij} - \mathbf{u}_{ij}|^2} \end{aligned} \tag{5}$$

where $K_d = (\frac{\pi^{\frac{d}{2}}}{\Gamma(\frac{d}{2}+1)})$ is the scalar of d ; d is the sample dimension; Γ is the gamma function, defined as $\Gamma(\alpha) = \int_0^\infty x^{\alpha-1} e^{-x} dx$.

Definition 2 (Sampling weight) The sampling weight of cluster C_i is defined as the sum of the reciprocal of its density distribution function divided by the reciprocal of all cluster density distribution functions [21]. The formula is as follows:

$$w_i = \frac{\frac{1}{density(C_i)}}{\sum_{i=1}^m \frac{1}{density(C_i)}} = \frac{\frac{vol_{C_i}(S_d(r_i))}{N_{C_i}}}{\sum_{i=1}^m \frac{vol_{C_i}(S_d(r_i))}{N_{C_i}}} \tag{6}$$

According to the property of $\sum_{i=1}^m w_i = 1$, the number of new samples generated in cluster C_i is $N_{C_i} = w_i * N$.

B. BIRCH CLUSTERING OF MINORITY SAMPLES

First, we use the BIRCH algorithm to cluster the minority samples in the original dataset. The specific theory and characteristics of the BIRCH algorithm have been elaborated in section 2-B. Before performing BIRCH clustering, we also designed a denoising operation to optimize the clustering effect. This is because the BIRCH clustering algorithm is operated by using Sklearn package in Python language., and it does not involve the step of removing noise samples. Therefore, we consider the k the samples of k in the adjacent majority classes of minority samples as noise, and delete them before clustering the minority samples, which proves to be effective after the test.

C. IDENTIFY BOUNDARY MINORITY SAMPLES AND PROPORTIONAL MIDPOINT OVERSAMPLING

Identifying the boundary minority samples can carry out the steps in Section 3-B simultaneously, because the identification method has no correlation with clusters. We can better understand the new algorithm by considering boundary minority sampling together with the proportional midpoint oversampling.

We can lose the key information from the boundary minority samples if the new samples are synthesized by oversampling only. The boundary minority samples are small in amount and can hardly be learned because they are in the overlapping areas with the samples of different classes. We therefore improved the algorithm to identify and label the boundary minority samples, and then decided the sampling probability based on the neighbor density of the majority

class around the minority class, i.e., the greater the density, the higher the probability of sampling.

We then conducted proportional sampling. In each cluster, this method not only obtained the boundary minority samples based on probabilities to synthesize new samples, but also used the minority samples within the cluster to generate new samples. The ratio of synthesizing new samples is set to 1. In Section 5-B, we analyzed and discussed the results under different ratios. This is an innovation of our design. The importance of different data borders varies. Data of less border importance is given a smaller synthesizing ratio for the boundary minority samples.

Finally, the midpoint oversampling [21] was implemented. The BI-BMCSMOTE algorithm synthesizes new samples in a different way from the SMOTE algorithm. SMOTE synthesizes new samples by searching for the closest neighbors of the minority samples, and therefore the new and old samples could be identical. In contrast, the BMCSMOTE algorithm sorts the samples in order in a cluster based on the samples' distance to the centroid, then chooses the samples with furthest distance possible from each other, and synthesizes the new samples on the line that connects the two samples. This approach not only avoids high similarity by increasing the diversity of new samples, but also provides more useful information on classification for classifiers.

D. BI-BMCSMOTE ALGORITHM

Compared with traditional algorithms like SMOTE, the BI-BMCSMOTE algorithm not only clusters the samples in the minority class, but also adds boundary minority class samples selected by probability to synthesize new samples. The BI-BMCSMOTE algorithm is also different from the SMOTE algorithm in terms of the way they synthesize new samples. The SMOTE algorithm synthesizes new samples by searching for the nearest neighbors of minority class samples, so that old and new samples are prone to be highly similar. However, the BMCSMOTE algorithm sorts the samples within cluster by their distance to the center of mass, selects the samples in the same cluster that are as far away from each other as possible, and synthesizes new samples on the line connecting the two samples. This method not only avoids high similarity by increasing the diversity of new samples, but also provides the classifier with more effective information about the classification.

IV. EXPERIMENTAL STUDY

The empirical analysis in this section is divided into four components: dataset description, evaluation measures, oversampling algorithm and classifier selection, and method parameter setting of proposed method and comparison method.

A. DATASETS

This article uses eleven actual datasets. Ten actual datasets are selected from UCI database [23] and KEEL database [24], and their characteristic information is shown in Table 1.

Algorithm 1 BI-BMCSMOTE(m, k)

Input: D : Original imbalanced dataset; P : Set of minority class samples; m : The number of categories generated by the BIRCH algorithm; k : The number of minority nearest neighbors.

Output: S : Balanced dataset after Oversampling

Procedure Begin:

1. For the k nearest neighbors of each minority sample, find the minority sample whose majority nearest neighbor is $k - 1$ as noise and remove it. Get the new minority dataset P_1 .
2. Substitute P_1 into the BIRCH clustering algorithm, and output cluster C .
3. For each $p_a \in P_1$, find its k nearest neighbors. Calculate the number of majority neighbors of p_a as maj_a and calculate the number of minority neighbors of p_a as min_a . When $maj_a > min_a$, mark the sample as a boundary minority sample. Finally, the boundary minority sample set is H .
4. Calculate the density function for majority neighbor maj of each boundary minority sample $h_b \in H$: $df_b = \frac{maj_b}{k}$. Normalize to get its density distribution: $dd_a = \frac{df_a}{\sum_{i=1}^b df_i}$.
5. Calculate the extraction probability of each h_b : $pr_b = \frac{dd_b}{\sum_{i=1}^b dd_i}$. The extraction probability of set H is PR .
6. For $C_i \in C$ do:
 - Calculate the density distribution function of cluster C_i according to formula (3): $density(C_i)$.
 - End for
7. Calculate the oversampling weight of each cluster according to formula (6): W
8. For $C_i \in C$ do:
 - 1) According to the oversampling weight w_i , calculate the number of new samples that need to be generated in cluster C_i : $N_{C_i} = w_i * N$.
 - 2) Divide N_{C_i} into $N_{C_{i1}}, N_{C_{i2}}$, make $N_{C_{i1}} + N_{C_{i2}} = N_{C_i}$, the ratio of 1:1. For $N_{C_{i1}}$, sample points in H are extracted with probability. For $N_{C_{i2}}$, extract sample points in P_1 . Get sample sets G_{i1} and G_{i2} respectively.
 - 3) Calculate the Euclidean distance r_j from all sample points in the sample set G_{i1} to the centroid u_i , and sort the sample points in the set G_{i1} according to r_j from small to large: $C_{i1} = \{x_1, \dots, x_{mid}, x_{mid+1}, \dots, x_i\}$.
 - 4) The sorted sample set C_i is divided into the near centroid set $X_{min} = \{x_1, x_2, \dots, x_{mid}\}$ and the far centroid set $X_{max} = \{x_{mid+1}, x_{mid+2}, \dots, x_i\}$ from the middle position.
 - 5) According to the arrangement order of the sets X_{min} and X_{max} , select a sample point from each of the two sets to pair in pairs: $PP = \{x_1 \Delta x_{mid+1}, x_2 \Delta x_{mid+2}, \dots, x_{mid} \Delta x_i\}$, where Δ means pairing the samples, $PP_j = x_j \Delta x_{mid+j}, j = 1, \dots, mid$.

Algorithm 1 (Continued.) BI-BMCSMOTE(m, k)

9. For $j \in \{1, \dots, mid\}$ do:

- 1) Synthesize the first round of new samples l_1^j at the midpoint of the connection between the two sample points after pairing, where $l_1^j = x_j + \frac{x_{mid+j} - x_j}{2}$.
- 2) When $N_i < N_{C_{i1}}$, add l_1^j to the set $S, N_i = N_i + 1$.

End for

10. When $N_i < N_{C_{i1}}$, the new sample l_1^j obtained in the first round is paired with its adjacent parent sample points x_j, x_{mid+j} in pairs, and step 9 is repeated to synthesize the second round new sample l_2^j, l_2^j . Finally, synthesize new sample points in round i until $N_i = N_{C_{i1}}$ is satisfied.
11. The sample set G_{i2} in 8 is operated as in the sample set G_{i1} at steps 8, 9, and 10, and then new synthetic sample is obtained and added to the set S .

Another large-scale actual dataset is selected in the Kaggle database. The detailed description of these eleven actual data sets will be shown below. Before using the oversampling algorithm, all datasets are simply preprocessed, including the removal of duplicate variables and data standardization to increase data validity.

TABLE 1. Description of the datasets for experiment.

Dataset	# instances	# of features	# of minority instances	# of majority instances	Imbalanced ratios
Ecoli	336	7	77	269	3.49
Ionosphere	351	34	126	225	1.79
Pima	768	8	268	500	1.87
Image Segmentation	2310	19	330	1980	6
Yeast	1484	8	429	1055	2.46
MAGIC Gamma Telescope	19020	10	6688	12332	1.84
Yeast3	1484	8	163	1321	8.1
Glass0	214	9	70	144	2.06
Haberman	306	3	81	225	2.78
Breast Tissue	106	9	35	71	2.03

TABLE 2. Confusion matrix.

confusion matrix		Actual	
		Positive	Negative
Predicted	Positive	TP	FP
	Negative	FN	TN

Table 1 shows ten actual datasets. The reason for choosing these datasets is that their sample size and the number of variables have a large span. This shows that the BI-BMCSMOTE oversampling algorithm can be applied to datasets of different dimensions. However, with the development of the society, the sample size of the dataset that needs to be tested keeps increasing, and the variables also become a lot. In order to show that the BI-BMCSMOTE oversampling algorithm also has a good effect on large-scale data, this paper also uses the credit card customer default dataset in the Kaggle database for the analysis of Taiwan's credit card customer delinquency

TABLE 3. Results on KNN.

	SMOTE	Borderline-smote	ADASYN	MWMOTE	DBSMOTE	BI-BMCSMOTE
ecoli(F)	0.9402	0.9324	0.929	0.9271	0.941	0.9486
ionosphere(F)	0.9087	0.9262	0.9453	0.887	0.883	0.8927
pima(F)	0.7977	0.8044	0.8016	0.7821	0.795	0.8155
segment(F)	0.9956	0.995	0.9953	0.9944	0.9955	0.9961
yeast(F)	0.8225	0.8304	0.8266	0.7611	0.8173	0.8254
magic06(F)	0.8684	0.8777	0.8867	0.825	0.8631	0.8746
yeast5(F)	0.9626	0.9621	0.9601	0.9587	0.9691	0.9711
glass2(F)	0.8836	0.8683	0.8876	0.8598	0.8463	0.8626
haberman(F)	0.7467	0.7285	0.7552	0.706	0.7477	0.762
BreastTissue(F)	0.8913	0.9308	0.9185	0.8918	0.893	0.933
dofccc(F)	0.8176	0.8229	0.8125	0.7996	0.8364	0.8584
ecoli(A)	0.9546	0.9521	0.9561	0.9523	0.9617	0.9619
ionosphere(A)	0.9588	0.9711	0.9889	0.9652	0.9346	0.94
pima(A)	0.8499	0.8483	0.8403	0.8277	0.8608	0.862
segment(A)	0.9983	0.9986	0.9986	0.9972	0.9986	0.9986
yeast(A)	0.8725	0.8698	0.8548	0.8132	0.8677	0.8541
magic06(A)	0.9227	0.9186	0.9214	0.8876	0.9143	0.9163
yeast5(A)	0.9813	0.9755	0.975	0.9805	0.9836	0.9836
glass2(A)	0.9184	0.9139	0.9184	0.9079	0.8886	0.9178
haberman(A)	0.7795	0.7758	0.7888	0.7307	0.8004	0.7758
BreastTissue(A)	0.9473	0.9311	0.9206	0.9571	0.9369	0.9624
dofccc(A)	0.8629	0.8628	0.8488	0.8652	0.8856	0.9053

since 2005, which is about 30,000. This dataset contains customer credit card information for five months in 2005. The dataset contains 25 variables, with an imbalance ratio of 4.52, in which the ID variable is meaningless and needs to be deleted. This dataset also uses simple data preprocessing.

In order to visually demonstrate the superiority of the BI-BMCSMOTE oversampling algorithm, a synthetic two-dimensional imbalanced dataset is used. As shown in Fig. 3, the blue triangle is the majority sample, accounting for 80%, and the black circle is the minority sample, accounting for 20%. X1 and X2 represent two features. There are noise samples in this data set and the boundary of the samples is not clearly demarcated. Direct classification without measures will lead to average classification results.

B. EVALUATION MEASURES

Accuracy is an important measurement standard in traditional classification evaluation measures, but in imbalanced data classification, accuracy is not applicable. This is because of the rare proportion of the minority samples, even if the prediction is wrong will have a high accuracy. Therefore, researchers have put forward some powerful schemes for evaluating imbalanced dataset indicators [25], [26]. In this paper, *F-measure* and AUC are adopted as evaluation

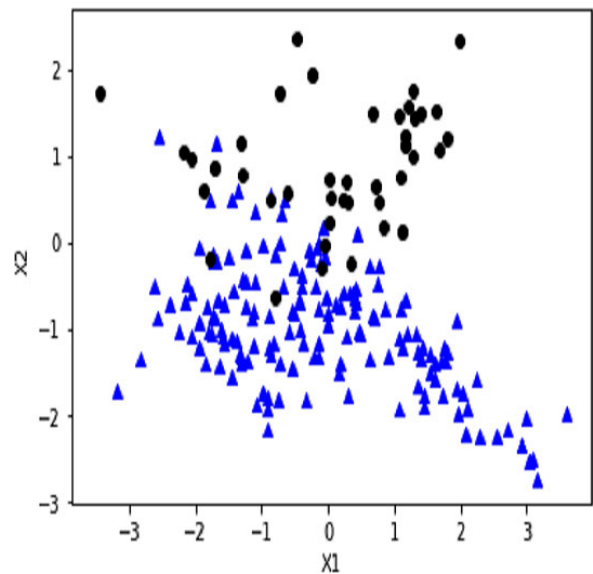


FIGURE 3. Two-dimensional imbalanced dataset. A synthetic two-dimensional imbalanced dataset.

measures to apply to the classification effect detection of imbalanced datasets [25]. F-measure and AUC are based on the confusion matrix (see Table 2).

TABLE 4. Results on SVM.

	SMOTE	Borderline-smote	ADASYN	MWMOTE	DBSMOTE	BI-BMCSMOTE
ecoli(F)	0.9191	0.9145	0.9169	0.9102	0.9172	0.9363
ionosphere(F)	0.9351	0.945	0.9403	0.9329	0.9496	0.9556
pima(F)	0.7563	0.7876	0.7367	0.7713	0.8107	0.8064
segment(F)	0.9955	0.9865	0.9871	0.9969	0.9926	0.998
yeast(F)	0.7424	0.7554	0.7614	0.7617	0.7866	0.8147
magic07(F)	0.8377	0.7995	0.7948	0.8462	0.8499	0.8917
yeast6(F)	0.9471	0.95	0.9397	0.9494	0.9725	0.9603
glass3(F)	0.8052	0.7994	0.7962	0.7962	0.8145	0.807
haberman(F)	0.6416	0.6457	0.584	0.5737	0.6455	0.7461
BreastTissue(F)	0.9182	0.9297	0.8676	0.9133	0.8925	0.9404
dofcc(F)	0.6541	0.6452	0.5861	0.7463	0.7555	0.8777
ecoli(A)	0.9631	0.9383	0.9477	0.9578	0.9786	0.9794
ionosphere(A)	0.9779	0.9857	0.9858	0.9838	0.9873	0.9897
pima(A)	0.8511	0.8414	0.825	0.8582	0.8944	0.8833
segment(A)	0.9995	0.9993	0.9993	0.9995	0.9995	0.9995
yeast(A)	0.8158	0.7863	0.7887	0.8339	0.8327	0.8665
magic07(A)	0.9169	0.8808	0.8712	0.9224	0.9285	0.9481
yeast6(A)	0.9838	0.977	0.9757	0.9849	0.994	0.988
glass3(A)	0.8601	0.8363	0.8302	0.8663	0.8597	0.8707
haberman(A)	0.7342	0.7159	0.7032	0.7169	0.7155	0.7983
BreastTissue(A)	0.9743	0.9671	0.9517	0.9773	0.9773	0.9823
dofcc(A)	0.7595	0.7336	0.7116	0.8552	0.8273	0.9204

According to the predicted and actual results of the classifier, it is divided into four combinations: True Positive (TP), False Negative (FN), False Positive (FP), True Negative (TN). Therefore, the definition of F-measure can be deduced:

$$F - measure = \frac{(1 + \beta^2) * precision * recall}{\beta^2 * recall + precision} \quad (7)$$

where $recall = \frac{TP}{TP+FN}$, $precision = \frac{TP}{TP+FP}$, and $\beta > 0$ is the relative importance of $recall$ to $precision$. F -measure is the harmonic average of $precision$ and $recall$, and is the result of weighing the importance of the two indicators. AUC is the probability that the positive sample is greater than the negative sample in the random test. It is the sum of the area under the ROC curve, and the value is less than 1.

C. OVER-SAMPLING ALGORITHM AND SELECTION OF CLASSIFIER

This paper uses five oversampling algorithms, SMOTE, Borderline-Smote, ADASYN, MWMOTE, and DBSMOTE, to compare with the BI-BMCSMOTE oversampling algorithm, and substitutes them into ten actual datasets and a larger credit default dataset for sample synthesis. After obtaining the balanced data set, the new dataset is classified using five mainstream machine learning models: K nearest

neighbors (KNN) [26], random forest (RF) [28], support vector machine (SVM) [29], eXtremeGradient Boosting (XGBoost) [30], and Light Gradient Boosting Machine (LGBM) [30]. Some of these classification models are classic and well known, while others are very popular and frequently used in academia and industry. This paper selects these classification models for testing in order to show that the balanced dataset synthesized by the BI-BMCSMOTE oversampling algorithm has a wide range of validity and stability.

D. EXPERIMENTAL ENVIRONMENT SETTING

In this paper, the number of neighbors of SMOTE, Borderline-SMOTE, ADASYN is 5. The values for the parameters of MWMOTE are $k1 = 5$, $k2 = 3$, $k3 = 5$. The parameter ϵ of the DBSCAN clustering algorithm in the DBSMOTE algorithm is set to range (0.001, 2) which the interval is 0.05. The range of $MinPts$ parameters to be set is (2,10). The optimal parameters ϵ and $MinPts$ are selected by manual screening. KNN algorithm's nearest neighbor number is $K \in [3, 5]$.

In our proposed method, the number of categories generated by BIRCH clustering algorithm is $m \in [2, 6]$. The number of minority nearest neighbors of BI-BMCSMOTE algorithm is $k \in [3, 12]$. In the process of

TABLE 5. Results on RF.

	SMOTE	Borderline-smote	ADASYN	MWMOTE	DBSMOTE	BI-BMCSMOTE
ecoli(F)	0.9401	0.9346	0.9253	0.9337	0.9395	0.9504
ionosphere(F)	0.9404	0.9518	0.9417	0.9329	0.9533	0.9596
pima(F)	0.8037	0.8185	0.8111	0.8079	0.8108	0.8251
segment(F)	0.9975	0.9944	0.9981	0.9966	0.9983	0.9981
yeast(F)	0.8415	0.8396	0.8343	0.8034	0.8403	0.8506
magic08(F)	0.8912	0.8948	0.8993	0.8783	0.8919	0.9072
yeast7(F)	0.9719	0.974	0.9695	0.9687	0.9713	0.974
glass4(F)	0.9084	0.9102	0.9176	0.8932	0.9228	0.895
haberman(F)	0.7651	0.772	0.7639	0.7382	0.7571	0.7823
BreastTissue(F)	0.9133	0.9312	0.9292	0.9073	0.898	0.9318
dofccc(F)	0.8515	0.8598	0.8477	0.8721	0.8798	0.8817
ecoli(A)	0.9836	0.9745	0.9755	0.9763	0.9885	0.9826
ionosphere(A)	0.9833	0.9886	0.9881	0.9865	0.9868	0.9887
pima(A)	0.9055	0.8944	0.8928	0.8853	0.9083	0.9094
segment(A)	1	1	1	1	1	1
yeast(A)	0.9198	0.916	0.9121	0.89	0.9218	0.914
magic08(A)	0.9591	0.9572	0.9592	0.9485	0.9593	0.964
yeast7(A)	0.9954	0.9957	0.9953	0.9945	0.9952	0.996
glass4(A)	0.9644	0.9618	0.9553	0.954	0.9757	0.9651
haberman(A)	0.8353	0.8113	0.8294	0.804	0.8658	0.8256
BreastTissue(A)	0.9784	0.9706	0.9698	0.9871	0.983	0.9684
dofccc(A)	0.9228	0.9279	0.9209	0.9328	0.937	0.9368

clustering, Agglomerative Clustering is used to cluster all CF tuples, which can eliminate the unreasonable tree structure caused by the sample reading order, and some tree structure splits caused by the limitation of the number of CF nodes.

The performance results of each data set are obtained by hierarchical five-fold cross validation [26]. When a new sample is synthesized, the ratio of the selected boundary minority samples to the intra-cluster minority samples is 1:1. In the final synthetic balanced dataset, the ratio of the majority class to the minority class is 1:1 [31], which has a more accurate test effect.

V. EXPERIMENTAL RESULTS AND ANALYSIS

This section first presents the visual effects of the SMOTE and BI-BMCSMOTE algorithms by using a synthetic two-dimensional imbalanced dataset, so as to demonstrate the superiority of the BI-BMCSMOTE algorithm. Next, the test effectiveness of each oversampling algorithm in different classifiers is compared by tabulation. The evaluation indexes include F-Measure and AUC. To verify the ratio of new samples synthesized by extracted boundary minority class samples to those synthesized by extracted intra-cluster minority class samples, 10 actual and credit datasets are further tested, followed by an interpretation of results.

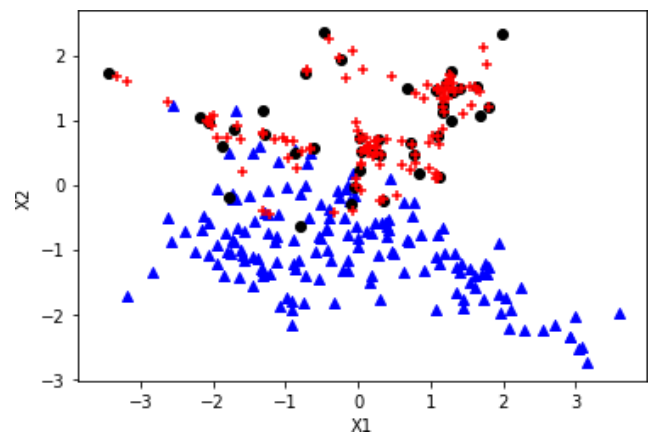


FIGURE 4. SMOTE oversampling result. Visualization of SMOTE oversampling.

A. BI-BMCSMOTE ALGORITHM VISUALIZATION RESULTS AND ANALYSIS

Fig. 4 and Fig. 5 visualize the over-sampled results of the SMOTE and BI-BMCSMOTE algorithms on the synthetic datasets, respectively. Fig. 4 shows that due to the intrinsic noise in dataset, the samples synthesized by the SMOTE algorithm invade the majority class samples and further amplify

TABLE 6. Results on XGB.

	SMOTE	Borderline-smote	ADASYN	MWMOTE	DBSMOTE	BI-BMCSMOTE
ecoli(F)	0.9373	0.9393	0.9384	0.9225	0.9425	0.9432
ionosphere(F)	0.9321	0.9473	0.9332	0.9379	0.9416	0.9368
pima(F)	0.8163	0.803	0.7863	0.7982	0.7984	0.818
segment(F)	0.9967	0.9966	0.9966	0.9964	0.9969	0.9975
yeast(F)	0.81	0.8081	0.8205	0.8048	0.8146	0.8484
magic05(F)	0.8589	0.8316	0.8297	0.8671	0.8898	0.9031
yeast4(F)	0.9684	0.971	0.9677	0.968	0.9688	0.9755
glass1(F)	0.9009	0.8751	0.8996	0.89	0.9183	0.9189
haberman(F)	0.7437	0.7566	0.7579	0.7134	0.7525	0.781
BreastTissue(F)	0.8972	0.9099	0.9444	0.9008	0.9008	0.9285
dofccc(F)	0.811	0.8051	0.8032	0.8729	0.8137	0.8843
ecoli(A)	0.9818	0.9762	0.9781	0.9781	0.9817	0.9837
ionosphere(A)	0.9843	0.9864	0.9854	0.9849	0.9782	0.9814
pima(A)	0.8766	0.868	0.8597	0.8778	0.8911	0.8927
segment(A)	0.9998	1	1	0.9998	0.9999	0.9999
yeast(A)	0.889	0.8881	0.8902	0.8894	0.894	0.9075
magic05(A)	0.936	0.9147	0.9085	0.9411	0.952	0.9598
yeast4(A)	0.9929	0.9915	0.9912	0.9927	0.9957	0.9937
glass1(A)	0.9428	0.9359	0.9405	0.9499	0.9528	0.9611
haberman(A)	0.8032	0.7913	0.8165	0.801	0.8216	0.8446
BreastTissue(A)	0.9639	0.9507	0.9657	0.9557	0.9567	0.968
dofccc(A)	0.8994	0.8942	0.8943	0.936	0.895	0.9411

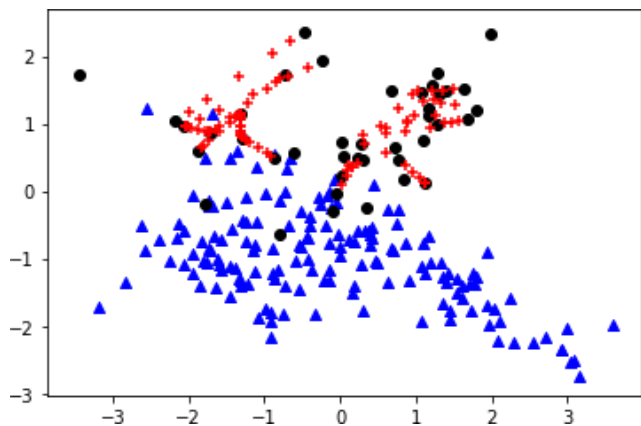


FIGURE 5. BI-BMCSMOTE oversampling result. Visualization of BI-BMCSMOTE oversampling.

the noise. Moreover, it is also found that the minority class samples with more minority class neighbors are involved in synthesizing more samples, while the boundary minority class samples have less chance to participate in synthesizing samples. As a result, the learning of intra-cluster samples is wasted. Due to insufficient learning capacity of boundary samples, some important information about the boundary

is also ignored. Fig. 5 shows that the BI-BMCSMOTE algorithm avoids the noise interference from samples. The BI-BMCSMOTE algorithm does not neglect the learning of intra-cluster samples while focusing on learning boundary samples. Another advantage is its ability to freely adjust the ratio of boundary samples to intra-cluster samples during synthesis (the ratio is set to 1:1 in the test).

B. ANALYSIS OF EXPERIMENTAL RESULTS OF BI-BMCSMOTE ALGORITHM

The test results of the optimal oversampling algorithm are bolded to facilitate comparison. F-Measure is denoted by “F” at the end of datasets, while AUC is denoted by “A” at the end of datasets. As shown in Table 3-7, the BI-BMCSMOTE algorithm has an average F-Measure score of 0.897 on the five classifiers, which improves by 2.5% than average F-Measure score of other algorithms. It gets the highest F-Measure score in 76.36% of all 55 tests. The new algorithm demonstrates the largest F-Measure improvement over the MWMOTE algorithm (an average improvement of 3.05%). In view of classifiers, the F-Measure of new algorithm performs the best on the LGBM classifier (an average score of 0.9067). The new algorithm demonstrates the largest F-Measure improvement over other algorithms

TABLE 7. Results on LGBM.

	SMOTE	Borderline-smote	ADASYN	MWMOTE	DBSMOTE	BI-BMCSMOTE
ecoli(F)	0.944	0.9248	0.9228	0.9241	0.9359	0.9439
ionosphere(F)	0.917	0.9334	0.9442	0.9336	0.9422	0.9527
pima(F)	0.7994	0.8106	0.7975	0.8006	0.7972	0.8109
segment(F)	0.9958	0.9944	0.9952	0.9958	0.9967	0.9959
yeast(F)	0.8152	0.8276	0.832	0.7987	0.8283	0.8439
magic09(F)	0.8635	0.8456	0.8425	0.8676	0.8906	0.9035
yeast8(F)	0.9663	0.973	0.9707	0.9655	0.9717	0.9732
glass5(F)	0.8854	0.8777	0.9053	0.8889	0.8815	0.9288
haberman(F)	0.7554	0.7308	0.763	0.7002	0.7765	0.8026
BreastTissue(F)	0.8727	0.9116	0.896	0.9203	0.9024	0.9333
dofccc(F)	0.8403	0.8335	0.8355	0.8704	0.8423	0.8845
ecoli(A)	0.9757	0.9669	0.9697	0.9763	0.9844	0.9779
ionosphere(A)	0.9817	0.9863	0.9836	0.9847	0.9796	0.9797
pima(A)	0.8517	0.8621	0.8568	0.8539	0.8806	0.8855
segment(A)	0.9984	0.9999	0.9994	0.9982	0.9999	0.9999
yeast(A)	0.8959	0.899	0.8979	0.8735	0.8978	0.9059
magic09(A)	0.9397	0.9234	0.9175	0.941	0.9544	0.959
yeast8(A)	0.9918	0.9896	0.9909	0.9898	0.9923	0.9931
glass5(A)	0.9499	0.9465	0.9406	0.9333	0.9406	0.9526
haberman(A)	0.8017	0.7899	0.8175	0.7824	0.8289	0.8501
BreastTissue(A)	0.9733	0.9568	0.9605	0.9748	0.9619	0.9806
dofccc(A)	0.917	0.9129	0.9143	0.9343	0.9127	0.9396

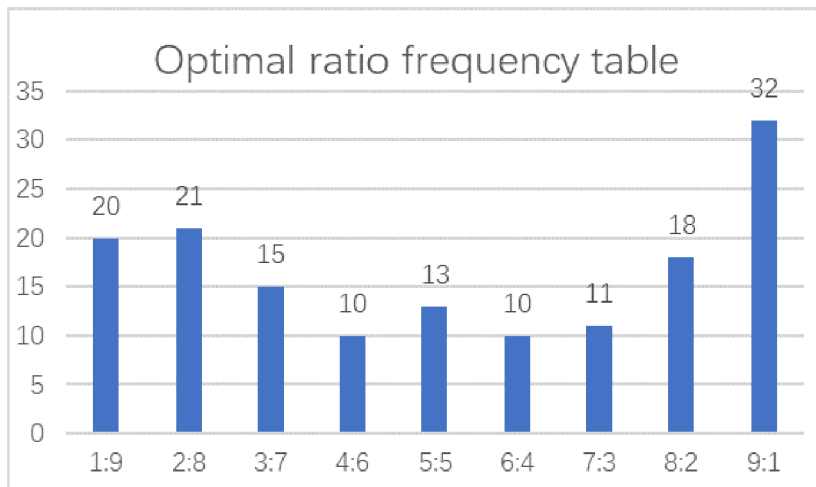


FIGURE 6. Optimal ratio frequency table. The ratio of new samples synthesized by extracted boundary minority class samples to those synthesized by extracted intra-cluster minority class samples, where the optimal value is the frequency.

in the SVM classifier (an average improvement of 5.21%). In addition, the BI-BMCSMOTE algorithm has an average AUC score of 0.9384 on the five classifiers, which improves by 1.54% than average AUC score of other algorithms. It gets

the highest AUC score in 69.1% of all 55 tests. The new algorithm demonstrates the largest AUC improvement over the ADASYN algorithm (an average improvement of 2.06%). In view of classifiers, the AUC of new algorithm performs

TABLE 8. Ratio analysis.

		1:9	2:8	3:7	4:6	5:5	6:4	7:3	8:2	9:1
ecoli(F)	KNN	0.9498	0.9466	0.945	0.9484	0.9486	0.9517	0.9479	0.9495	0.9461
ecoli(F)	SVM	0.9306	0.9383	0.9366	0.9346	0.9363	0.9331	0.9363	0.9379	0.9366
ecoli(F)	RF	0.9524	0.9482	0.9462	0.9488	0.9504	0.9365	0.9401	0.946	0.9439
ecoli(F)	XGB	0.9421	0.9489	0.9475	0.948	0.9432	0.9387	0.944	0.9435	0.9453
ecoli(F)	LGBM	0.9423	0.9352	0.9373	0.9364	0.9439	0.9421	0.9488	0.9395	0.9398
ecoli(A)	KNN	0.96	0.9591	0.9602	0.9612	0.9619	0.9606	0.964	0.9608	0.9632
ecoli(A)	SVM	0.9814	0.9795	0.98	0.9797	0.9794	0.9781	0.9778	0.9769	0.9762
ecoli(A)	RF	0.9746	0.9799	0.9816	0.9802	0.9826	0.9838	0.9826	0.9868	0.9874
ecoli(A)	XGB	0.9768	0.9826	0.982	0.9796	0.9837	0.9849	0.9796	0.9795	0.9876
ecoli(A)	LGBM	0.9794	0.9776	0.9839	0.9818	0.9779	0.982	0.9792	0.9852	0.9844
ionosphere(F)	KNN	0.8934	0.8934	0.8934	0.8927	0.8927	0.8901	0.893	0.8956	0.8956
ionosphere(F)	SVM	0.9559	0.9559	0.9559	0.9556	0.9556	0.9556	0.9571	0.9571	0.9571
ionosphere(F)	RF	0.9576	0.9617	0.9553	0.957	0.9596	0.9556	0.957	0.9611	0.9512
ionosphere(F)	XGB	0.9329	0.9391	0.937	0.9449	0.9368	0.9347	0.9426	0.9426	0.9473
ionosphere(F)	LGBM	0.9413	0.9466	0.9475	0.949	0.9527	0.9511	0.9445	0.9545	0.9445
ionosphere(A)	KNN	0.9402	0.9402	0.9402	0.94	0.94	0.9399	0.9425	0.9426	0.9425
ionosphere(A)	SVM	0.9896	0.9897	0.9897	0.9897	0.9897	0.9897	0.9891	0.9891	0.9892
ionosphere(A)	RF	0.9882	0.9886	0.9871	0.9885	0.9887	0.9883	0.9879	0.9892	0.9893
ionosphere(A)	XGB	0.9821	0.9834	0.9824	0.9819	0.9814	0.9818	0.9825	0.9816	0.9818
ionosphere(A)	LGBM	0.9799	0.9825	0.9824	0.9847	0.9797	0.9787	0.9793	0.9836	0.9791
pima(F)	KNN	0.8074	0.8157	0.8078	0.8172	0.8155	0.8041	0.8037	0.8038	0.7995
pima(F)	SVM	0.806	0.8052	0.8117	0.7996	0.8064	0.7991	0.8028	0.7981	0.7971
pima(F)	RF	0.818	0.8265	0.8164	0.8229	0.8251	0.8245	0.8142	0.8219	0.8164
pima(F)	XGB	0.8231	0.8212	0.8172	0.8129	0.818	0.8179	0.8163	0.8225	0.8144
pima(F)	LGBM	0.8129	0.8139	0.8019	0.8198	0.8109	0.7948	0.8135	0.8071	0.8063
pima(A)	KNN	0.8532	0.8615	0.8566	0.8602	0.862	0.8578	0.8591	0.8596	0.8627
pima(A)	SVM	0.8784	0.8852	0.8822	0.8839	0.8833	0.8804	0.8795	0.8799	0.8774
pima(A)	RF	0.9056	0.9071	0.9002	0.9051	0.9094	0.9044	0.9005	0.9023	0.9048
pima(A)	XGB	0.8933	0.8907	0.8881	0.8942	0.8927	0.894	0.896	0.8986	0.8985
pima(A)	LGBM	0.8678	0.8725	0.8739	0.8786	0.8855	0.8674	0.8729	0.8739	0.8755
segment(F)	KNN	0.9964	0.997	0.9962	0.9961	0.9961	0.9964	0.9964	0.9958	0.9969
segment(F)	SVM	0.9969	0.9972	0.9975	0.9977	0.998	0.998	0.9983	0.9986	0.9989
segment(F)	RF	0.9978	0.9986	0.9983	0.9972	0.9981	0.9969	0.9975	0.9972	0.9972
segment(F)	XGB	0.9958	0.997	0.9973	0.9964	0.9975	0.997	0.997	0.9975	0.997
segment(F)	LGBM	0.9961	0.9969	0.9958	0.9945	0.9959	0.9955	0.9972	0.9956	0.9958
segment(A)	KNN	0.9981	0.9989	0.9989	0.9989	0.9986	0.9989	0.9986	1	0.9997
segment(A)	SVM	0.9995	0.9995	0.9995	0.9995	0.9995	0.9995	0.9995	0.9995	0.9995
segment(A)	RF	1	1	1	1	1	1	1	1	1
segment(A)	XGB	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
segment(A)	LGBM	0.9995	0.9999	0.9994	0.9994	0.9999	0.9999	0.9999	0.9999	0.9994
yeast(F)	KNN	0.8308	0.8198	0.8228	0.8227	0.8254	0.8226	0.8165	0.8194	0.8232
yeast(F)	SVM	0.8125	0.8102	0.8103	0.8133	0.8147	0.8099	0.8101	0.8106	0.8087
yeast(F)	RF	0.8438	0.8464	0.8433	0.8444	0.8506	0.848	0.8517	0.8532	0.853
yeast(F)	XGB	0.8409	0.8413	0.8383	0.8447	0.8484	0.8487	0.8469	0.8471	0.8461
yeast(F)	LGBM	0.8392	0.8379	0.8417	0.836	0.8439	0.8398	0.8407	0.8491	0.8362

TABLE 8. (Continued.) Ratio analysis.

yeast(A)	KNN	0.8553	0.8472	0.8447	0.8516	0.8541	0.8565	0.8517	0.8555	0.8609
yeast(A)	SVM	0.8687	0.8672	0.8684	0.8668	0.8665	0.8631	0.8588	0.8557	0.8531
yeast(A)	RF	0.9108	0.9059	0.9079	0.9062	0.914	0.9143	0.9157	0.9174	0.9224
yeast(A)	XGB	0.9025	0.9005	0.9008	0.904	0.9075	0.91	0.908	0.9082	0.9132
yeast(A)	LGBM	0.9021	0.9005	0.8974	0.8977	0.9059	0.9029	0.9015	0.9078	0.9043
magic04(F)	KNN	0.8706	0.8712	0.8718	0.8723	0.8746	0.8732	0.8741	0.8757	0.8758
magic04(F)	SVM	0.8936	0.8937	0.893	0.8922	0.8917	0.8911	0.8914	0.89	0.8895
magic04(F)	RF	0.9066	0.9081	0.9068	0.907	0.9072	0.9057	0.9082	0.9081	0.9087
magic04(F)	XGB	0.9055	0.9047	0.9042	0.9033	0.9031	0.9019	0.9009	0.9011	0.8995
magic04(F)	LGBM	0.9046	0.9037	0.9045	0.9031	0.9035	0.9022	0.9022	0.9009	0.9013
magic04(A)	KNN	0.9168	0.9163	0.9165	0.9163	0.9163	0.9161	0.917	0.9172	0.9175
magic04(A)	SVM	0.9505	0.95	0.9493	0.9488	0.9481	0.9474	0.9466	0.9458	0.9448
magic04(A)	RF	0.9638	0.9643	0.9644	0.964	0.964	0.9641	0.9644	0.965	0.9649
magic04(A)	XGB	0.9609	0.9606	0.9605	0.9602	0.9598	0.9594	0.9592	0.9591	0.9589
magic04(A)	LGBM	0.9604	0.9598	0.9596	0.9601	0.959	0.9584	0.9584	0.959	0.9588
yeast3(F)	KNN	0.9689	0.9696	0.97	0.9707	0.9711	0.9725	0.9731	0.973	0.973
yeast3(F)	SVM	0.9604	0.9598	0.9597	0.9612	0.9603	0.9619	0.9604	0.9595	0.9597
yeast3(F)	RF	0.9752	0.9766	0.9762	0.9748	0.974	0.972	0.9727	0.9708	0.9735
yeast3(F)	XGB	0.975	0.9746	0.9764	0.9738	0.9755	0.9726	0.9736	0.9703	0.9714
yeast3(F)	LGBM	0.972	0.9738	0.9735	0.975	0.9732	0.9722	0.973	0.9718	0.973
yeast3(A)	KNN	0.979	0.9781	0.9797	0.9806	0.9836	0.9846	0.9836	0.9841	0.9843
yeast3(A)	SVM	0.9877	0.9874	0.9874	0.9877	0.988	0.9878	0.9879	0.9876	0.9871
yeast3(A)	RF	0.9966	0.9967	0.9968	0.9958	0.996	0.996	0.996	0.9956	0.996
yeast3(A)	XGB	0.9933	0.9931	0.9933	0.9934	0.9937	0.9936	0.9938	0.9941	0.9945
yeast3(A)	LGBM	0.9908	0.9925	0.9934	0.9914	0.9931	0.9915	0.9927	0.99	0.9919
glass0(F)	KNN	0.8646	0.8636	0.8684	0.8641	0.8626	0.8681	0.8789	0.8858	0.8858
glass0(F)	SVM	0.8074	0.8044	0.8076	0.8005	0.807	0.8045	0.8004	0.8041	0.7995
glass3(F)	RF	0.8923	0.8974	0.8949	0.9085	0.895	0.907	0.9156	0.9218	0.9159
glass0(F)	XGB	0.8922	0.8921	0.9188	0.9063	0.9189	0.9193	0.9266	0.9246	0.9257
glass0(F)	LGBM	0.9118	0.9171	0.911	0.9226	0.9288	0.8976	0.919	0.9127	0.9319
glass0(A)	KNN	0.9137	0.9128	0.9111	0.9138	0.9178	0.9045	0.9107	0.9143	0.9186
glass0(A)	SVM	0.8746	0.8733	0.872	0.87	0.8707	0.8616	0.8573	0.8604	0.8552
glass0(A)	RF	0.9525	0.955	0.9595	0.956	0.9651	0.9639	0.9644	0.9695	0.9702
glass0(A)	XGB	0.9457	0.9489	0.9544	0.9573	0.9611	0.9507	0.9589	0.9612	0.9626
glass0(A)	LGBM	0.9527	0.9546	0.9564	0.9538	0.9526	0.9556	0.9573	0.9531	0.9679
haberman (F)	KNN	0.7192	0.7327	0.7399	0.7585	0.762	0.7362	0.7738	0.7811	0.7975
haberman (F)	SVM	0.7199	0.7372	0.7314	0.7424	0.7461	0.7429	0.761	0.769	0.7783
haberman (F)	RF	0.7703	0.7506	0.7763	0.7856	0.7823	0.7706	0.8011	0.7972	0.801
haberman (F)	XGB	0.772	0.7977	0.7823	0.7916	0.781	0.787	0.7964	0.7943	0.8096
haberman (F)	LGBM	0.7963	0.8064	0.8087	0.7974	0.8026	0.8106	0.8174	0.8066	0.822
haberman(A)	KNN	0.7393	0.7372	0.7481	0.7562	0.7758	0.7642	0.7809	0.8012	0.8155
haberman(A)	SVM	0.7922	0.7986	0.7979	0.8037	0.7983	0.806	0.8091	0.8142	0.8184
haberman(A)	RF	0.8057	0.8093	0.8062	0.8245	0.8256	0.8375	0.8443	0.8556	0.868
haberman(A)	XGB	0.8128	0.8248	0.8189	0.8176	0.8446	0.834	0.8401	0.8385	0.8545
haberman(A)	LGBM	0.8405	0.855	0.8524	0.8488	0.8501	0.8525	0.8584	0.8691	0.8849
BreastTissue(F)	KNN	0.919	0.9325	0.945	0.9424	0.933	0.9527	0.9447	0.9424	0.9527

TABLE 8. (Continued.) Ratio analysis.

BreastTissue(F)	SVM	0.9441	0.9441	0.9376	0.94	0.9404	0.9341	0.9281	0.9261	0.9309
BreastTissue(F)	RF	0.9336	0.92	0.9204	0.9198	0.9318	0.9084	0.929	0.9204	0.9204
BreastTissue(F)	XGB	0.9133	0.9197	0.9011	0.9083	0.9285	0.9143	0.9065	0.9093	0.9204
BreastTissue(F)	LGBM	0.8915	0.9342	0.9134	0.926	0.9333	0.9208	0.9333	0.9257	0.9094
BreastTissue(A)	KNN	0.9445	0.9456	0.9525	0.9513	0.9624	0.9525	0.955	0.9533	0.9505
BreastTissue(A)	SVM	0.9828	0.9771	0.9771	0.9758	0.9823	0.9748	0.9797	0.9793	0.9703
BreastTissue(A)	RF	0.9811	0.9704	0.9731	0.9692	0.9684	0.968	0.9768	0.9771	0.9626
BreastTissue(A)	XGB	0.9496	0.9574	0.9557	0.9602	0.968	0.9505	0.9577	0.9537	0.9497
BreastTissue(A)	LGBM	0.982	0.9796	0.9765	0.9592	0.9806	0.9709	0.9762	0.9711	0.9644
dofccc(F)	KNN	0.8702	0.8703	0.8702	0.8704	0.8584	0.8469	0.8317	0.8123	0.7881
dofccc(F)	SVM	0.8879	0.8879	0.888	0.8875	0.8777	0.8662	0.8525	0.8349	0.8124
dofccc(F)	RF	0.8922	0.8916	0.8909	0.8917	0.8817	0.8707	0.8576	0.8399	0.8188
dofccc(F)	XGB	0.894	0.8942	0.894	0.8939	0.8843	0.8731	0.8598	0.8434	0.8224
dofccc(F)	LGBM	0.8928	0.8929	0.8935	0.8925	0.8845	0.8724	0.8589	0.8421	0.8209
dofccc(A)	KNN	0.9121	0.9122	0.9122	0.9122	0.9053	0.8989	0.8902	0.8791	0.8641
dofccc(A)	SVM	0.9261	0.9262	0.9264	0.9262	0.9204	0.9136	0.9054	0.8959	0.8836
dofccc(A)	RF	0.9414	0.9416	0.9418	0.9411	0.9368	0.9311	0.9247	0.9161	0.9058
dofccc(A)	XGB	0.9458	0.9459	0.9458	0.9456	0.9411	0.9361	0.9297	0.9221	0.9128
dofccc(A)	LGBM	0.9442	0.9444	0.9444	0.9442	0.9396	0.935	0.9277	0.9203	0.9109
count:		20	21	15	10	13	10	11	18	32

the best on the RF classifier (an average score of 0.9501). The new algorithm demonstrates the largest AUC improvement over other algorithms in the SVM classifier (an average improvement of 3.8%). The results above indicate that the BI-BMCSMOTE algorithm has some advantages over other oversampling algorithms. Moreover, it has excellent stability when running on the SVM classifier, compared with some other algorithms.

C. CHOOSING APPROPRIATE VALUES FOR BI-BMCSMOTE PARAMETERS

The BI-BMCSMOTE algorithm has two parameters to be selected: the number of clusters m generated by BIRCH clustering and the number of neighbors k of the minority samples. For m , if m is set to a large value, more clusters with smaller size will be generated. If m is set to a small value, fewer clusters with larger will be generated. Therefore, the choice of m depends on the size of the dataset. Of the datasets tested on 106 to nearly 30,000 samples, the m range was set best between 2 and 6. For K , if we set a larger value, it will make the nearest neighbor number of a minority sample larger. If there are not many minority samples in the dataset, and a higher k is set, it will cause more nearest neighbors of the majority class to join the nearest neighbors of the minority class, affecting the experimental results. In addition, if K is set at a non-extreme value, it has little influence on the experimental results. Therefore, the choice of k value depends on the number of minority samples or even the size of the dataset. Of the datasets tested on 106 to nearly 30,000 samples, the k range was set best between 3 and 12.

D. RATIO ANALYSIS OF SYNTHETIC NEW SAMPLES

The new samples were generated from the previously defined boundary minority samples and the minority samples in the cluster. Therefore, in order to verify whether the ratio of new samples generated by the two types of minority samples has an impact on final result, Table 8 showed the test results of the new algorithm on 11 data sets. The ratio is successively set to 1:9, 2:8, 3:7...9:1. The test results in Table 8 show that the new algorithm cannot perform the best when the ratio is set to 1:1. At last, a frequency table of optimal ratio is made by recording the number of the highest F-Measure and AUC scores in each ratio. As shown in Fig. 6, the frequency table of optimal ratio has high values on both sides and low values in the middle, because some datasets have important information about boundary points and more boundary samples need to be synthesized for learning. If some datasets have less information about boundary points, the learning process should focus on the synthesis of intra-cluster samples.

VI. CONCLUSION

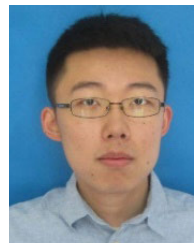
This paper proposes a new BI-BMCSMOTE oversampling algorithm, which considers the minority sample boundaries with sample cluster density functions. It provides a new method for imbalanced datasets. The BI-BMCSMOTE algorithm is executed in four steps: conduct BIRCH clustering through a single scan of dataset by applying a tree structure; calculate the number of samples in each cluster according to the cluster density; identify the boundary minority samples and mark them according to probability; synthesize new samples proportionally from the marked boundary

minority sample and the normal sample. This method uses BIRCH clustering with fast running speed and good stability, enhanced boundary learning, midpoint centroid oversampling also avoids overfitting. Therefore, it can be concluded that this algorithm can synthesize diversified new samples and balance the synthetic data.

The novelty of the BI-BMCSMOTE algorithm lies in that this method not only considers the important information of the boundary samples, but also retains the normal sample information and the boundary sample information selected based on importance degree on the premise of identifying and removing noise. Our future research will focus on better improving the stability of BIRCH clustering algorithm to deal with data of different scales, and improving the synthesizing way of samples to further prevent over fitting.

REFERENCES

- [1] Y. Sun, A. K. Wong, and M. S. Kamel, "Classification of imbalanced data: A review," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 23, no. 4, pp. 687–719, 2009.
- [2] G. Haixiang, L. Yijing, J. Shang, G. Mingyun, H. Yuanyue, and G. Bing, "Learning from class-imbalanced data: Review of methods and applications," *Expert Syst. Appl.*, vol. 73, pp. 220–239, May 2017.
- [3] C. Beyan and R. Fisher, "Classifying imbalanced data sets using similarity based hierarchical decomposition," *Pattern Recognit.*, vol. 48, no. 5, pp. 1653–1672, May 2015.
- [4] W. Wei, J. Li, L. Cao, Y. Ou, and J. Chen, "Effective detection of sophisticated online banking fraud on extremely imbalanced data," *World Wide Web*, vol. 16, no. 4, pp. 449–475, Jul. 2013.
- [5] M. M. Rahman and D. N. Davis, "Addressing the class imbalance problem in medical datasets," *Int. J. Mach. Learn. Comput.*, vol. 3, no. 2, p. 224, 2013.
- [6] P. Bedi, N. Gupta, and V. Jindal, "Siam-IDS: Handling class imbalance problem in intrusion detection systems using siamese neural network," *Procedia Comput. Sci.*, vol. 171, pp. 780–789, Jan. 2020.
- [7] S. T. Yekeen, A. Balogun, and K. B. W. Yusof, "A novel deep learning instance segmentation model for automated marine oil spill detection," *ISPRS J. Photogramm. Remote Sens.*, vol. 167, pp. 190–200, Sep. 2020.
- [8] O. Loyola-Gonzalez, J. F. C. O. Martinez-Trinidad, J. A. Carrasco-Ochoa, and M. Garcia-Borroto, "Cost-sensitive pattern-based classification for class imbalance problems," *IEEE Access*, vol. 7, pp. 60411–60427, 2019.
- [9] Y. Sun, M. S. Kamel, A. K. C. Wong, and Y. Wang, "Cost-sensitive boosting for classification of imbalanced data," *Pattern Recognit.*, vol. 40, no. 12, pp. 3358–3378, Dec. 2007.
- [10] J. Wei, H. Huang, L. Yao, Y. Hu, Q. Fan, and D. Huang, "NI-MWMOTE: An improving noise-immunity majority weighted minority oversampling technique for imbalanced classification problems," *Expert Syst. Appl.*, vol. 158, Nov. 2020, Art. no. 113504.
- [11] T. Zhu, Y. Lin, and Y. Liu, "Synthetic minority oversampling technique for multiclass imbalance problems," *Pattern Recognit.*, vol. 72, pp. 327–340, Dec. 2017.
- [12] S.-J. Yen and Y.-S. Lee, "Cluster-based under-sampling approaches for imbalanced data distributions," *Expert Syst. Appl.*, vol. 36, no. 3, pp. 5718–5727, Apr. 2009.
- [13] P. Sadhukhan and S. Palit, "Adaptive learning of minority class prior to minority oversampling," *Pattern Recognit. Lett.*, vol. 136, pp. 16–24, Aug. 2020.
- [14] T. Zhu, Y. Lin, and Y. Liu, "Improving interpolation-based oversampling for imbalanced data learning," *Knowl.-Based Syst.*, vol. 187, Jan. 2020, Art. no. 104826.
- [15] J. Wei, H. Huang, L. Yao, Y. Hu, Q. Fan, and D. Huang, "IA-SUWO: An improving Adaptive semi-supervised weighted oversampling for imbalanced classification problems," *Knowl.-Based Syst.*, vol. 203, pp. 106–116, Jun. 2020.
- [16] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, Jun. 2002.
- [17] H. Han, W. Wang, and B. Mao, "Borderline-SMOTE: A new oversampling method in imbalanced data sets learning," in *Proc. Int. Conf. Intell. Comput.*, Berlin, Germany: Springer, 2005, pp. 878–887.
- [18] H. He, Y. Bai, E. A. Garcia, and S. Li, "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IEEE World Congr. Comput. Intell.)*, Jun. 2008, pp. 1322–1328.
- [19] S. Barua, M. M. Islam, X. Yao, and K. Murase, "MWMOTE-majority weighted minority oversampling technique for imbalanced data set learning," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 2, pp. 405–425, Feb. 2014.
- [20] C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap, "DBSMOTE: Density-based synthetic minority over-sampling Echnique," *Appl. Intell.*, vol. 36, no. 3, pp. 664–684, 2012.
- [21] L. Wang and J. Y. Hybrid, "Algorithm of DBSCAN and improved SMOTE for oversampling," *Comput. Eng. Appl.*, vol. 56, no. 18, pp. 111–118, 2020.
- [22] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: An efficient data clustering method for very large," *ACM SIGMOD Rec.*, vol. 25, no. 2, pp. 103–114, 1996.
- [23] D. Dua and C. Graff. (2017). *UCI Machine Learning Repository*. [Online]. Available: <http://archive.ics.uci.edu/ml>.
- [24] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, and S. García, "KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework," *J. Multiple-Valued Logic Soft Comput.*, vol. 17, nos. 2–3, pp. 255–287, 2011.
- [25] A. Maratea, A. Petrosino, and M. Manzo, "Adjusted F-measure and kernel scaling for imbalanced data learning," *Inf. Sci.*, vol. 257, pp. 331–341, Feb. 2014.
- [26] G. M. Weiss and F. Provost, "Learning when training data are costly: The effect of class distribution on tree induction," *J. Artif. Intell. Res.*, vol. 19, pp. 315–354, Oct. 2003.
- [27] G. Guo, H. Wang, D. Bell, Y. Bi, and K. Greer, "KNN model-based approach in classification," in *On The Move to Meaningful Internet Systems: CoopIS, DOA, and ODBASE (Lecture Notes in Computer Science)*, vol. 2888. Berlin, Germany: Springer, 2003, pp. 986–996.
- [28] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.
- [29] J. A. K. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Process. Lett.*, vol. 9, no. 3, pp. 293–300, Jun. 1999.
- [30] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 785–794.
- [31] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T. Y. Liu, "LightGBM: A highly efficient gradient boosting decision tree," in *Proc. Adv. Neural Inf. Process. Syst. Syst.*, 2017, pp. 3149–3157.
- [32] M. Ester, H. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. 2nd Int. Conf. Knowl. Discov. Data Mining*, 1996, pp. 226–231.
- [33] R. K. Pathria, "Volume' and 'surface area' of an n-dimensional sphere of radius R," in *Statistical Mechanics*. London, U.K.: Elsevier, 1996, pp. 504–505.



CHAO-RAN WANG received the B.S. degree from Heilongjiang University, Harbin, China, in 2018. He is currently pursuing the master's degree with Northeastern University, Shenyang, China. His research interest includes data mining.



XIN-HUI SHAO received the B.S. and M.S. degrees in mathematics from Northeastern Normal University, China, in 1994 and 1997, respectively, and the D.S. degree in mathematics from Northeastern University, China, in 2006.

•••