TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 18/22 pp216–231
DOI: 10.26599/TST.2023.9010006
Volume 29, Number 1, February 2024

Joint Sample Position Based Noise Filtering and Mean Shift Clustering for Imbalanced Classification Learning

Lilong Duan, Wei Xue*, Jun Huang, and Xiao Zheng

Abstract: The problem of imbalanced data classification learning has received much attention. Conventional classification algorithms are susceptible to data skew to favor majority samples and ignore minority samples. Majority weighted minority oversampling technique (MWMOTE) is an effective approach to solve this problem, however, it may suffer from the shortcomings of inadequate noise filtering and synthesizing the same samples as the original minority data. To this end, we propose an improved MWMOTE method named joint sample position based noise filtering and mean shift clustering (SPMSC) to solve these problems. Firstly, in order to effectively eliminate the effect of noisy samples, SPMSC uses a new noise filtering mechanism to determine whether a minority sample is noisy or not based on its position and distribution relative to the majority sample. Note that MWMOTE may generate duplicate samples, we then employ the mean shift algorithm to cluster minority samples to reduce synthetic replicate samples. Finally, data cleaning is performed on the processed data to further eliminate class overlap. Experiments on extensive benchmark datasets demonstrate the effectiveness of SPMSC compared with other sampling methods.

Key words: imbalanced data classification; oversampling; noise filtering; clustering

1 Introduction

Imbalanced data classification is an important research topic in machine learning[1]. Here, imbalanced data mean that the number of samples in some classes is far larger than the number of samples in others in a dataset. For example, in two classes of imbalanced data, the class with most samples is called majority class and the class with few samples is called minority class. In many practical applications, the minority class samples have significant research value because they contain important information, such as rare disease diagnosis[2–4], fraudulent transaction detection[5], DNA microarray data analysis[6], text classification[7], network intrusion detection[8, 9], security management[10], etc.

Traditional classification algorithms aim to improve the overall classification accuracy. However, in the imbalanced data scenario, since conventional classification algorithms usually favor majority class samples[11] and ignore minority class samples, directly applying classifiers may result in poor performance[12, 13]. In particular, even a high classification accuracy is obtained, it is not reliable. For example, in disease diagnosis, if the number of diseased samples is only 1% and the number of normal samples is 99% in a dataset, then the accuracy of classifying all samples as normal will be as high as 99%, but this accuracy is not reliable because the minority samples are not correctly identified and the cost of this misclassification is huge. Research has shown that not only between-class imbalance decreases classification performance, but also overlapping between classes[14, 15], noisy samples[16], small disjuncts[17, 18], within-class

• Lilong Duan, Wei Xue, Jun Huang, and Xiao Zheng are with the School of Computer Science and Technology, Anhui University of Technology, Maanshan 243032, China, and also with the Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Hefei 230088, China. E-mail: lilong.duan@foxmail.com; xuewei@ahut.edu.cn; huangjun.cs@ahut.edu.cn; xzheng@ahut.edu.cn.
* To whom correspondence should be addressed.
Manuscript received: 2022-12-03; revised: 2023-01-22; accepted: 2023-01-31

imbalance[19], and duplicate data[20] can also decrease classification performance.

To solve the above problems, many approaches have been proposed, which can be classified into four categories: data level methods[21, 22], algorithm modification[23], cost-sensitive learning[24, 25], and ensemble learning[26]. Specifically, data level methods rebalance the dataset using oversampling, undersampling, or hybrid sampling methods. Algorithm modification is to improve the existing algorithms to recognize minority class samples more accurately. Cost-sensitive learning sets a larger cost for misclassifying minority classes, and ensemble learning improves classification performance by combining several basic classifiers. Among them, the data level method is favored by researchers because it only changes the distribution of the original data, is convenient to use, and can be directly applied to various classifiers.

The most common methods in the data level are random oversampling[17] and random undersampling[27]. The former randomly duplicates the minority class samples and the latter randomly reduces the majority class samples to rebalance the dataset. However, random oversampling is susceptible to overfitting[28] and random undersampling may remove samples containing important information[29]. Although both methods have their advantages, studies have shown that oversampling is preferred to undersampling in many practical applications[30, 31]. To overcome the drawbacks of oversampling methods, many modified oversampling methods have been proposed, such as synthetic minority oversampling technique (SMOTE)[32], adaptive synthetic sampling approach (ADASYN)[33], K-mean-SMOTE[34], majority weighted minority oversampling technique (MWMOTE)[35], borderline-SMOTE (B1-SMOTE and B2-SMOTE)[36], density-based synthetic minority oversampling technique (DBSMOTE)[37], etc. Although these methods improve the recognition accuracy of minority classes, they also have some problems such as insufficient noise filtering and generation of duplicates and outlier samples. To this end, we propose a joint sample position based noise filtering and mean shift clustering (SPMSC) method for imbalanced binary data in this paper. SPMSC can not only adequately filter the noise samples to alleviate their influence in the sample generation process, but also effectively reduce the production of duplicate samples and class overlap, so as to improve the recognition accuracy of samples. The main contributions of this work can be summarized as follows.

- We propose a new noise filtering mechanism that can adequately filter the noise samples in the original dataset to weaken the effect of noise in the sample synthesis phase.
- We cluster minority samples using the mean shift method, which does not need to set the number of clusters in advance. Also, it does not lead to the generation of a large number of duplicate samples at the sample synthesis phase due to inappropriate distance thresholds as in the case of hierarchical clustering.
- We use the Tomek link data cleaning method after sample synthesis to further reduce class overlap in the processed dataset.

The remainder of this paper is organized as follows. We review some popular oversampling methods in Section 2. In Section 3, we describe the SPMSC method in detail. Experimental results and analysis are provided in Section 4. Finally, we conclude this paper in Section 5.

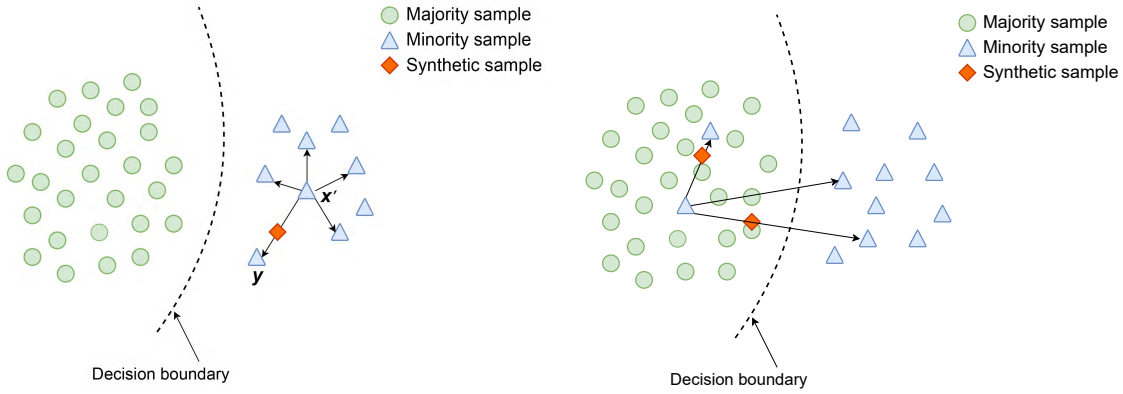
2 Related Work

The essence of sampling is to add or remove resamples to rebalance the originally imbalanced data. In this section, we briefly review some of the popular resampling methods.

SMOTE is one of the most representative oversampling methods. As shown in Fig. 1a, in Ref. [32], Chawla et al. randomly selected a sample y from the k nearest neighbors of minority sample x' to synthesize a new sample by Eq. (1):

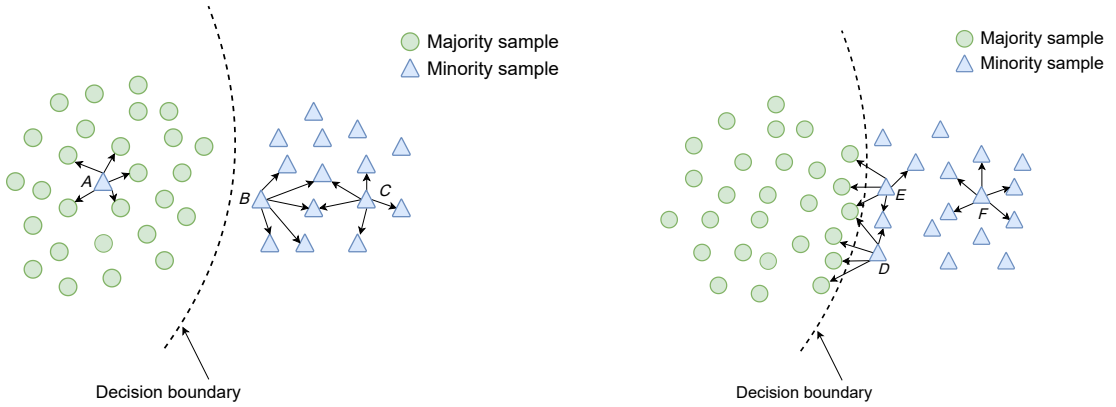
$$\text{synthetic} = x' + \lambda \times (y - x') \tag{1}$$

where λ is a random number between [0, 1]. This method is widely used because it is easy to understand and implement, however, it also has shortcomings. As shown in Fig. 1b, since this method performs sample synthesis for all minority samples without considering their distribution information relative to majority samples, it generates noisy samples and class overlap. Many approaches have been proposed to solve the problems generated by SMOTE. In Ref. [33], He et al. assigned sampling weights to each minority sample based on the number of majority samples in its nearest neighbors. However, it has the shortcomings of ignoring the effects of noisy samples and assigning unreasonable sampling weights. As shown in Fig. 2a, assuming k is 5, the noisy sample A will be assigned a larger weight, and the minority sample B will be assigned the same weight as the minority sample C although it is closer to the



(a) Random linear interpolation by using SMOTE (b) Noise generated by using SMOTE

Fig. 1 Sample synthesis.



(a) An example of the problems caused by ADASYN and borderline-SMOTE (b) An example of the problems caused by safe-level-SMOTE

Fig. 2 Possible scenarios for some oversampling methods.

decision boundary. In Ref. [36], Han et al. proposed a method to divide minority samples into three classes: noise, safe, and danger, and to oversample only the samples in the danger. However, this method may not accurately identify the danger samples in some cases, as shown in Fig. 2a, sample B will be misclassified as safe class although it is a danger sample. Bunkhumpornpat et al.[38] assigned to each minority sample "safe-level" value, and then synthesized samples closer to the largest safe-level. However, as shown in Fig. 2b, in contrast to the minority sample F, the new samples synthesized by the minority samples D and E will be closer to themselves, which may lead to overfitting because these new samples are gathered around minority class samples with a large density and away from the decision boundary. In Ref. [22], Onan proposed a consensus clustering based on undersampling, which utilizes the consensus clustering mechanism to undersample the majority class samples and improves the classification accuracy. In Ref. [25], Jiang et al. proposed to change the class distribution of the training data by cloning minority class samples. Barua et al.[35] proposed a new method to identify the boundary minority samples and assign sampling weights, however, it has the disadvantages of inadequate noise filtering and duplication of generated samples with the original minority data. In Ref. [34], Douzas et al. proposed to firstly cluster the whole dataset by using K-means, then select the appropriate clusters according to the imbalance ratio, and finally use SMOTE to synthesize samples. However, the optimal number of clusters is difficult to find. Nekooeimehr and Lai-Yuen[39] proposed adaptive semi-unsupervised weighted oversampling (A-SUWO) method which uses a semi-unsupervised hierarchical clustering method to cluster minority samples, then uses misclassification errors and cross-validation to determine the number of samples to be synthesized in each subcluster, and finally assigns sampling weights based on the average distance of the minority samples from their nearest majority class neighbors. However, the method is more complex and may not be suitable for larger datasets.

3 Proposed Method

In this section, we propose a new method for imbalanced data classification. Our method consists of three main steps: (1) noisy sample filtering, (2) mean shift clustering and sample synthesis, and (3) data cleaning.

3.1 Preliminary

Mean shift is a center-based nonparametric clustering method. The method works by calculating the average value of the distance between a candidate point x_i and the points within a given radius r and then updating the position of x_i, which forms a cluster with the points within its radius when x_i is not moving. The shift vector m is calculated by Eq. (2):

$$m(x_i) = \frac{\sum_{x_j \in N(x_i)} K(x_j - x_i) x_j}{\sum_{x_j \in N(x_i)} K(x_j - x_i)} - x_i \qquad (2)$$

where m points to the region with the largest density increase, and $N(x_i)$ represents the neighborhood of x_i in a given r range. According to the shift vector m, the update process of a candidate point x_i is shown in Eq. (3).

$$x_i^{t+1} = x_i^t + m(x_i^t) \qquad (3)$$

where t is the number of iterations needed. The algorithm does not require setting the number of clusters beforehand, which can be set automatically without relying on the parameter bandwidth that indicates the range of the region to be searched, and it can handle clusters of arbitrary shape. Therefore, the mean shift algorithm is suitable for the segmentation of data. For more information about the use of this algorithm, please refer to sklearn (https://scikit-learn.org/stable/).

3.2 Our method

MWMOTE[35] is a popular oversampling method for processing imbalanced data problems. Although it is specific for some problems such as within-class imbalance and class overlap, it also has the shortcomings of inadequate noise filtering and generating duplicate samples. Motivated by this, we propose a new approach to cope with these problems.

3.2.1 Noise sample filtering

At present, many algorithms remove noise based on K-NearestNeighbor (KNN) noise filtering criterion, i.e., if all k nearest neighbors of a minority sample are other classes, then the minority sample is considered as a noisy sample. However, this noise removal method is difficult to eliminate the most noisy samples. As shown in Fig. 3, only noisy sample L is removed using this method, while noisy samples M and N will still be retained. In order to

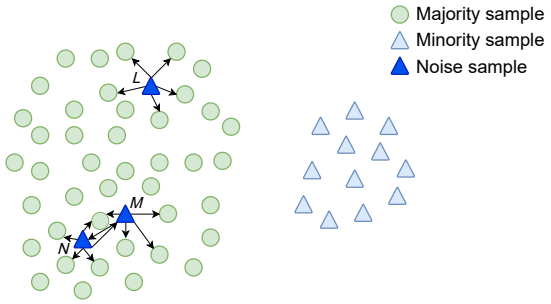


Fig. 3 Disadvantages of KNN-based noise filtering.

filter noise adequately, we propose a new noise filtering method. There are three main steps: Firstly, an input dataset Q is further divided into minority sample set $S_{\min}$ and majority sample set S_{maj} according to the labels, and then the Euclidean distance between each sample in $S_{\min}$ and each sample in Q is calculated to form a distance matrix.

$$\text{dis} = \text{Euclidean}_{x_i \in S_{\min}, y_i \in Q}(x_i, y_j) \qquad (4)$$

Specifically, because an sample has zero Euclidean distance from itself, we set it to a large constant in order to avoid this situation influencing the next judgments. Then, for each minority sample, we find the nearest one and three instances to it in dis using Eqs. (5) and (6), respectively.

$$\text{index}_1 = \text{smallest}_{x_i \in S_{\min}}(1, \text{dis}(x_i)) \qquad (5)$$

$$\text{index}_3 = \text{smallest}_{x_i \in S_{\min}}(3, \text{dis}(x_i)) \qquad (6)$$

Finally, two conditional judgments are made, where if index_1 is not a minority class, then the number of count belonging to the majority class in index_3 is judged in turn, and if count is greater than or equal to 2, then the minority sample is considered as noise. To further compare our denoising method with the KNN-based denoising method above, we use the two-dimensional dataset paw02a-600-5-70-BI in keel (https://sci2s.ugr.es/keel/datasets.php#sub1) to visualize the denoising results. As shown in Figs. 4 and 5 compared to MWMOTE's KNN-based denoising method, our method can more adequately remove noise even though some of the noise samples are specially distributed.

3.2.2 Mean shift clustering and sampling weighting

The use of average-linkage agglomerative clustering to divide the minority class samples may result in a large number of class clusters or only one or a few minority samples in a class cluster because an optimal distance threshold cannot be found, which is likely

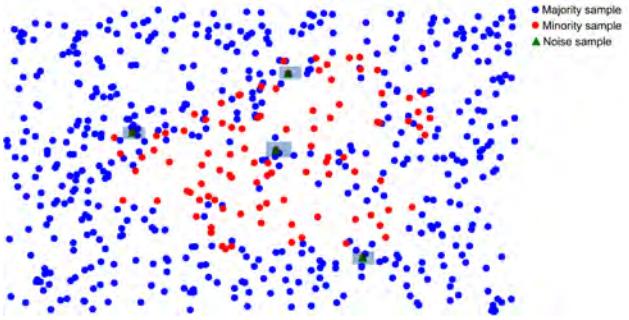


Fig. 4 KNN-based noise filtering.

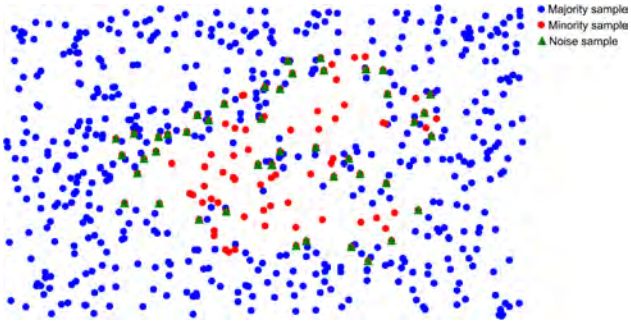


Fig. 5 Proposed denoising method.

to generate many duplicate instances in the sample synthesis step. Motivated by this, we use a parameter-free mean shift clustering algorithm to divide minority samples. This algorithm does not require setting the number of clusters beforehand and does not generate too many class clusters, thus reducing the generation of duplicate samples in the sample synthesis step. We will give an experimental comparison in Section 4. Next, we assign sampling weights to the minority class samples that satisfy the requirements according to Eqs. (7)–(9):

$$I_w(y_j, x_i) = \underset{x_i \in S_{\text{imin}}, y_j \in S_{\text{bmaj}}}{C_f(y_j, x_i) \times D_f(y_j, x_i)} \tag{7}$$

$$S_w(x_i) = \underset{x_i \in S_{\text{imin}}}{\sum_{y_j \in S_{\text{bmaj}}} I_w(y_j, x_i)} \tag{8}$$

$$S_p(x_i) = \frac{S_w(x_i)}{\underset{z_i \in S_{\text{imin}}}{\sum} S_w(z_i)} \tag{9}$$

Specifically, please refer to Ref. [35] for a detailed explanation of the boundary minority class samples S_{imin}, boundary majority class samples S_{bmaj}, closeness factor C_f, and density factor D_f.

3.2.3 Data cleaning

After synthesizing instances for each boundary minority class sample in the cluster they belong to by using Eq. (1) according to the selection probability S_p, we use the Tomek link method to clean the processed dataset with the aim of further removing class overlap. Tomek link is a data cleaning method defined as follows: For two different classes of samples x and y, $d(x, y)$ denotes the distance between them, and if there is no sample z such that $d(x, y) < d(x, z)$ or $d(x, y) < d(y, z)$, then (x, y) is a Tomek link and is removed.

In summary, the full steps of the proposed method are presented in Algorithm 1. For Algorithm 1, we have two remarks.

Remark 1. The purpose of first judging index_1 in the noise denoising process is to avoid misclassifying some minority samples as noisy samples because of their particular distribution.

Remark 2. When clustering minority class samples

Algorithm 1 SPMSC

Input
Q: dataset, S_{min}: minority class sample set, S_{maj}: majority class sample set, denoise: storing noise samples, and num: the number of samples to be generated.
K_1: nearest neighbors for finding the boundary majority class and K_2: nearest neighbors for finding the boundary minority class.
Procedure begin
Calculate the distance matrix between the sample in S_{min} and the sample in Q using Eq. (4).
for each $x_i \in S_{\text{min}}$, obtain index_1 and index_3 by using Eqs. (5) and (6).
if the index_1 label is not a minority class, initialize $m = 0$, **for** each index $\in \text{index}_3$, **if** the label of the index is the majority class, $m + 1$, if $m \geqslant 2$, **then** add that minority class to denoise.
Denoised minority class sample set $T_{\text{min}} = S_{\text{min}} - \text{denoise}$.
for each $x_i \in T_{\text{min}}$, find its K_1 nearest majority class samples $N_{\text{maj}}(x_i)$ to form boundary majority sample set $S_{\text{bmaj}} = \underset{x_i \in T_{\text{min}}}{\cup} N_{\text{maj}}(x_i)$.
for each $y_i \in S_{\text{bmaj}}$, find its K_2 nearest minority class samples $N_{\text{min}}(y_j)$ to form boundary minority sample set $S_{\text{imin}} = \underset{y_j \in S_{\text{bmaj}}}{\cup} N_{\text{min}}(y_i)$.
Clustering of S_{min} using the mean shift algorithm.
for each $x_i \in S_{\text{imin}}$ and $y_j \in S_{\text{bmaj}}$, sampling weights are calculated by using Eq. (7).
for each $x_i \in S_{\text{imin}}$, the selection probability is calculated using Eq. (9).
Initialize the set $S = S_{\text{min}}$.
do for $i = 1, 2, \ldots, \text{num}$
  Select sample x according to S_p and find the cluster_x
  where x is located.
  Select sample y randomly in cluster_x.
  A synthetic sample syn is generated by using Eq. (1) and
  adds syn to S: $S = S \cup \text{syn}$.
end loop
Obtain new dataset new: $\text{new} = S \cup S_{\text{maj}}$
The dataset new is cleaned using Tomek link.
End

using the mean shift algorithm, no parameters need to be set, where the bandwidth is estimated by the provided estimate_bandwidth function.

4 Result and Discussion

In this section, we conduct experiments and analyze the experimental results from multiple perspectives to verify the effectiveness of the SPMSC method.

4.1 Dataset description and comparison methods

We conduct experiments by using 21 datasets from KEEL, UCI[§], UCI_extended[¶], and RCSMOTE[✠]. These datasets have different sample sizes, feature attributes, and degrees of imbalance, and the details are shown in Table 1. Among them, wdbc is breast cancer dataset and its labels "M" and "B" denote malignant and benign, respectively. To be consistent with the other 20 datasets, we consider malignant as the minority sample assigned label "1" and benign as the majority sample assigned label "0".

To verify the effectiveness of the proposed method, we compare SPMSC with eight popular sampling methods, namely random oversampling (ROS)[17], SMOTE[32], SMOTE-Tomeklinks (STL)[40], ADASYN[33], B1-SMOTE[36], B2-SMOTE[36], safe-level-SMOTE (SLS)[38], and MWMOTE[35] on three classifiers, KNN[41], GaussianNB[42], and SVM[43].

[§] https://archive.ics.uci.edu/ml/index.php.

[¶] https://github.com/felix-last/evaluate-kmeans-smote/releases/download/v0.0.1/uci_extended.tar.gz.

[✠] https://raw.githubusercontent.com/M-Hashemzadeh/RCSMOTE/master/ImplementationSourceCodes.zip.

4.2 Evaluation measures

The method for evaluating classifier performance in machine learning is based on confusion matrix. As shown in Fig. 6, where *TN* represents the number of negative (majority) class samples correctly classified, *FP* represents the number of negative (majority) class samples misclassified as positive (minority) class samples, *FN* represents the number of positive (minority) class samples misclassified as negative (majority) class samples, and *TP* represents the number of positive (minority) class samples correctly classified. The traditional evaluation method accuracy is not applicable in imbalance scenarios because it only takes into account the overall accuracy and ignores the importance of minority class samples. Therefore, some evaluation methods for imbalance scenarios[44, 45] are proposed, and the specific definitions are shown as follows:

$$F\text{-measure} = \frac{2 \times \text{Sensitivity} \times \text{Precision}}{\text{Sensitivity} + \text{Precision}} \tag{10}$$

$$G\text{-mean} = \sqrt{\text{Sensitivity} \times \text{Specificity}} \tag{11}$$

$$AUC = \frac{1 + TPR - FPR}{2} \tag{12}$$

Table 1 Description of the imbalanced datasets.

Dataset	Number of minority samples	Number of majority samples	Number of samples	Attribute	Degree of imbalance
Yeast1	429	1055	1484	8	1:2.46
Yeast3	163	1321	1484	8	1:8.10
Yeast5	44	1440	1484	8	1:32.73
Pima	268	500	768	8	1:1.87
Glass0	70	144	214	9	1:2.06
Haberman	81	225	306	3	1:2.78
Vehicle1	217	629	846	18	1:2.90
Vehicle3	212	634	846	18	1:2.99
Glass-0-1-2-3_vs_4-5-6	51	163	214	9	1:3.20
Vehicle0	199	647	846	18	1:3.25
Ecoli1	77	259	336	7	1:3.36
Ecoli3	35	301	336	7	1:8.60
Ilpd	165	414	579	10	1:2.51
Heart	120	150	270	13	1:1.25
Liver_disorders2	72	200	272	6	1:2.78
Liver_disorders4	36	200	236	6	1:5.56
Pima2	134	500	634	8	1:3.73
Segment	330	1980	2310	16	1:6.00
Tic-tac-toe	332	626	958	9	1:1.89
Winequality-red4	53	1546	1599	11	1:29.17
Wdbc	212	357	569	31	1:1.68

	Predicted negative	Predicted positive
Actual negative	*TN*	*FP*
Actual positive	*FN*	*TP*

Fig. 6 Confusion matrix.

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

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

4.3 Experimental setting

The three classifiers and eight sampling algorithms are used in the experimental comparison, where, for the SVM classifier we used the hinge loss function as the loss term, $C = 0.1$, max_ite $= 10\,000$, and penalty is L_2. MWMOTE uses the parameters suggested in the original paper, i.e., $K_1 = 5$, $K_2 = 3$, $K_3 = |S_{\min}|/2$, $C_p = 3$, $C_f(\text{th}) = 5$, and CMAX $= 2$. The rest of classifier and comparison algorithm parameters used are set to default values.

4.4 Result comparison and analysis

4.4.1 Duplicate data comparison

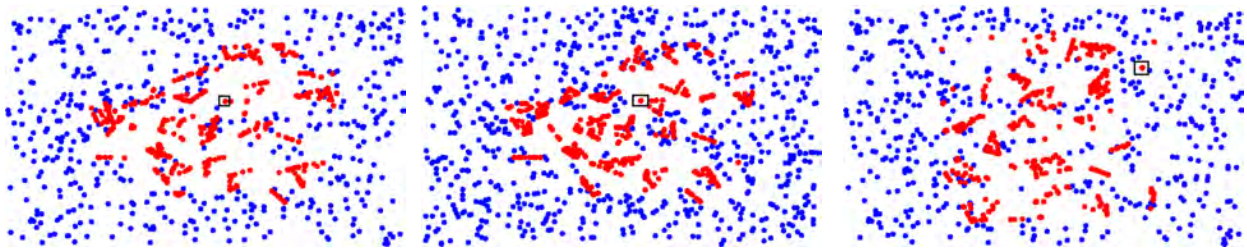
To visualize the duplicate points in the processed dataset, three 2-dimensional datasets are selected from KEEL. Figure 7 shows the data distribution after using the MWMOTE method. The blue point represents the majority class, the red point represents the minority class, and the point in the black rectangular box is the synthetic duplicate sample. It can be observed that the edges of these duplicate sample points show a sawtooth shape. Table 2 shows the number of duplicates of the original data in the synthesized samples using our method compared to the MWMOTE method, from which it can be concluded that our method can effectively reduce the generation of duplicate samples in the sample synthesis phase compared to the MWMOTE method.

4.4.2 Contrastive analysis of sampling performance

In this subsection, we compare the performance of the SPMSC method with the comparison method.

Tables 3–5 show the results for SPMSC and other eight sampling methods obtained using three classifiers on 21 datasets, with the best measures in bold. Additionally, 4-fold stratified cross validation was used to maintain the proportion of classes in the original data, and each experiment was repeated three times in order to eliminate the effect of randomness. As shown in Table 3, SPMSC obtains the best results at least on one measure in 19 out of 21 datasets when using the KNN classifier. As shown in Table 4, SPMSC obtains the best results at least on one measure in 11 out of 21 datasets when using the NB classifier. As shown in Table 5, SPMSC obtains the best results at least on one measure in 13 out of 21 datasets when using the SVM classifier. In comparison with the other eight sampling methods, the SPMSC method obtains the highest number of best results.

Figure 8 shows the average of the results of SPMSC and the other eight sampling methods. For KNN and SVM classifiers, SPMSC outperforms the other methods in all three measures. For the NB classifier, SPMSC



(a) paw02a-600-5-70-BI (b) paw02a-800-7-60-BI (c) 03subcl5-600-5-30-BI-full

Fig. 7 Visualization of duplicate data generated using the MWMOTE method.

Table 2 Comparison of duplicate data generated by the SPMSC method and the MWMOTE method.

Dataset	Number of synthetic samples	Number of duplicate samples	
		MWMOTE	SPMSC
paw02a-600-5-70-BI	400	90	5
paw02a-800-7-60-BI	700	159	14
03subcl5-600-5-30-BI-full	400	104	11

Table 3 Experimental results obtained on 21 datasets by using KNN classifier (*F-M* is short for *F*-measure and *G-M* is short for *G*-mean).

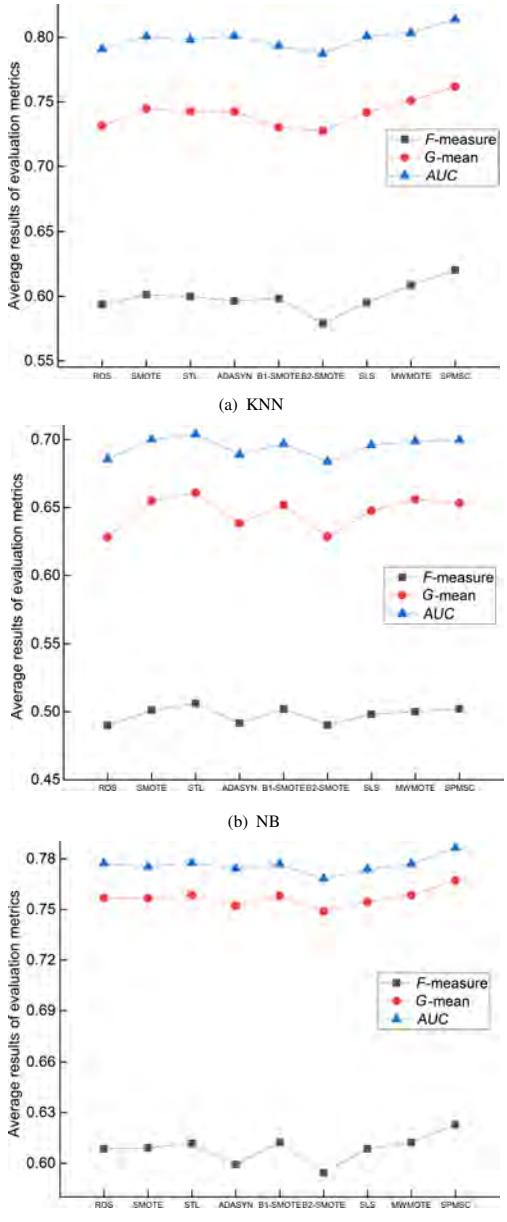
Dataset	Measure	ROS	SMOTE	STL	ADASYN	B1-SMOTE	B2-SMOTE	SLS	MWMOTE	SPMSC
Yeast1	*F-M*	0.522 743	0.536 692	0.530 222	0.531 450	0.526 050	0.528 153	0.535 802	0.547 124	**0.553 061**
	G-M	0.655 485	0.666 418	0.661 095	0.662 405	0.657 654	0.657 407	0.665 758	0.675 997	**0.681 259**
	AUC	0.717 969	0.722 232	0.720 031	0.721 376	0.709 195	0.709 174	0.727 138	0.739 181	**0.746 506**
Yeast3	*F-M*	0.676 411	0.660 401	0.681 125	0.644 022	0.681 272	0.587 982	0.654 146	0.676 332	**0.693 541**
	G-M	0.891 676	0.883 111	0.890 240	0.886 152	0.869 750	0.863 613	0.883 747	0.898 834	**0.902 797**
	AUC	0.915 162	0.920 008	0.921 669	0.914 774	0.917 576	0.918 808	0.933 057	0.928 967	**0.933 179**
Yeast5	*F-M*	**0.617 09**	0.581 992	0.586 894	0.576 608	0.594 626	0.531 439	0.573 519	0.584 642	0.586 229
	G-M	0.901 115	0.920 686	0.921 039	0.920 406	0.921 821	0.940 573	**0.955 442**	0.933 357	0.933 667
	AUC	0.943 845	0.954 293	0.954 104	0.954 451	0.942 487	0.960 038	0.963 699	0.965 878	**0.966 162**
Pima	*F-M*	0.630 745	0.636 319	0.632 69	0.631 125	0.635 532	0.638 106	0.648 711	0.639 191	**0.666 279**
	G-M	0.709 839	0.713 319	0.711 135	0.706 979	0.713 088	0.711 956	0.724 938	0.716 678	**0.737 819**
	AUC	0.759 433	0.767 970	0.771 224	0.767 731	0.764 239	0.761 179	0.777 657	0.778 224	**0.783 507**
Glass0	*F-M*	0.684 341	0.649 884	0.666 602	0.653 333	**0.700 074**	0.675 992	0.677 002	0.691 898	0.687 228
	G-M	0.760 103	0.724 730	0.743 947	0.721 762	**0.770 171**	0.739 385	0.753 216	0.762 917	0.758 680
	AUC	0.854 666	0.842 320	0.843 387	0.848 051	**0.859 307**	0.843 273	0.843 035	0.842 218	0.851 841
Haberman	*F-M*	0.421 019	0.440 88	0.357 196	0.405 857	0.374 330	0.407 483	0.387 785	**0.459 331**	0.440 081
	G-M	0.581 717	0.600 795	0.525 155	0.568 410	0.542 283	0.569 476	0.553 118	**0.615 404**	0.597 123
	AUC	0.629 310	0.633 590	0.605 722	0.623 992	0.616 953	0.604 969	0.635 230	0.639 898	**0.657 138**
Vehicle1	*F-M*	0.607 019	0.640 211	0.618 508	0.621 596	0.613 285	0.611 922	0.636 870	0.623 796	**0.641 375**
	G-M	0.662 310	0.670 863	0.667 071	0.665 134	0.654 432	0.658 299	0.675 294	0.658 596	**0.690 055**
	AUC	0.787 609	**0.795 734**	0.788 960	0.788 365	0.785 827	0.788 863	0.783 594	0.791 720	0.794 212
Vehicle3	*F-M*	0.605 194	0.637 168	0.623 961	0.620 988	0.617 169	0.617 063	0.614 278	0.623 436	**0.651 422**
	G-M	0.687 533	0.710 690	0.692 633	0.677 971	0.694 288	0.695 193	0.693 086	0.680 048	**0.712 928**
	AUC	0.797 793	0.805 608	0.806 233	0.812 216	0.807 651	0.797 736	0.810 702	0.792 435	**0.812 696**
Glass-0-1-2-3_vs_4-5-6	*F-M*	0.747 578	0.712 082	0.748 193	**0.764 255**	0.749 501	0.715 580	0.741 925	0.731 710	0.738 032
	G-M	0.823 241	0.789 808	0.818 214	**0.837 035**	0.823 587	0.814 925	0.821 308	0.811 880	0.814 975
	AUC	0.874 293	0.901 798	0.881 623	0.914 888	0.883 417	0.909 783	0.874 070	0.880 470	**0.916 891**
Vehicle0	*F-M*	0.815 186	0.823 788	0.841 176	0.831 928	0.819 156	0.782 420	0.825 707	0.823 652	**0.853 790**
	G-M	0.910 515	0.917 008	0.925 746	0.928 395	0.916 256	0.899 755	0.917 837	0.921 975	**0.942 815**
	AUC	0.967 894	0.974 264	0.975 171	0.971 552	0.970 341	0.957 626	0.974 061	0.977 874	**0.980 050**
Ecoli1	*F-M*	0.725 720	0.727 888	0.711 919	0.724 080	0.730 829	0.741 560	0.717 686	0.748 132	**0.752 647**
	G-M	0.817 409	0.818 596	0.809 286	0.823 722	0.814 030	0.820 512	0.820 526	0.824 980	**0.831 529**
	AUC	0.849 427	0.875 194	0.862 338	0.874 160	0.873 270	0.868 328	0.859 187	0.872 312	**0.880 111**
Ecoli3	*F-M*	0.540 513	0.594 937	0.599 679	0.583 611	**0.649 275**	0.617 845	0.549 430	0.582 168	0.624 126
	G-M	0.821 258	0.869 171	0.870 967	0.866 037	0.870 285	0.872 538	0.864 788	0.854 959	**0.887 046**
	AUC	0.894 766	0.903 738	0.912 889	0.897 445	0.913 961	0.907 463	**0.914 497**	0.904 661	0.905 135
Ilpd	*F-M*	0.448 776	0.447 730	0.455 276	0.459 848	0.417 290	0.451 186	**0.465 025**	0.455 049	0.464 537
	G-M	0.573 177	0.561 525	0.574 443	0.579 325	0.548 303	0.571 692	0.582 387	0.579 452	**0.583 680**
	AUC	0.634 529	0.620 863	0.635 553	0.620 397	0.620 137	0.627 583	0.626 710	0.622 023	**0.636 269**
Heart	*F-M*	0.797 115	0.819 588	0.815 289	0.796 385	0.829 265	0.809 292	0.824 590	0.828 488	**0.832 421**
	G-M	0.816 010	0.834 591	0.833 414	0.810 798	0.841 005	0.820 960	0.841 850	0.843 583	**0.847 947**
	AUC	0.867 028	0.872 804	0.878 781	0.862 011	0.869 965	0.865 040	**0.885 165**	0.882 278	0.878 924
Liver_disorders2	*F-M*	0.290 462	0.278 902	0.330 081	0.313 657	0.294 858	0.314 726	0.319 391	0.314 614	**0.350 284**
	G-M	0.441 491	0.437 244	0.484 934	0.465 420	0.449 435	0.460 712	0.467 958	0.458 493	**0.504 684**
	AUC	0.489 444	0.487 083	0.539 028	**0.543 472**	0.472 778	0.452 361	0.526 389	0.490 972	0.528 194
Liver_disorders4	*F-M*	0.292 618	0.337 410	0.293 651	0.299 566	0.281 588	0.247 524	0.272 866	0.302 781	**0.342 319**
	G-M	0.565 799	**0.614 234**	0.560 365	0.577 422	0.559 526	0.518 399	0.539 649	0.575 478	0.611 144
	AUC	0.604 167	0.624 444	0.559 444	0.621 667	0.610 278	0.552 778	0.580 278	0.593 333	**0.628 333**
Pima2	*F-M*	0.447 081	0.473 759	0.471 146	0.466 204	0.476 020	0.436 463	0.463 285	0.495 543	**0.504 809**
	G-M	0.650 070	0.676 277	0.672 752	0.671 170	0.669 098	0.639 291	0.663 252	0.694 574	**0.701 012**
	AUC	0.702 607	0.719 742	0.717 297	0.714 878	0.734 442	0.732 310	0.735 352	0.746 740	**0.748 690**
Segment	*F-M*	0.880 019	**0.884 240**	0.883 388	0.869 285	0.870 928	0.789 679	0.861 353	0.880 296	0.871 311
	G-M	0.955 697	0.954 073	0.955 136	**0.958 285**	0.951 425	0.936 239	0.949 397	0.956 944	0.957 501
	AUC	0.976 831	0.978 627	0.977 052	0.978 785	0.977 318	0.978 783	0.979 045	0.979 799	**0.983 245**
Tic-tac-toe	*F-M*	0.627 109	0.655 931	0.658 492	0.663 588	0.615 011	0.582 630	0.646 761	**0.687 021**	0.680 496
	G-M	0.690 542	0.715 622	0.716 701	0.720 644	0.673 680	0.651 727	0.707 028	**0.739 496**	0.728 968
	AUC	0.751 769	0.768 579	0.764 450	0.759 428	0.734 572	0.701 264	0.750 909	**0.798 718**	0.796 982
Winequality-red-4	*F-M*	0.139 106	0.136 370	0.137 932	0.141 528	0.148 952	**0.170 618**	0.124 273	0.136 538	0.137 645
	G-M	0.485 385	0.597 635	0.597 514	0.599 987	0.441 653	0.507 551	0.533 908	0.603 335	**0.605 299**
	AUC	0.597 243	0.650 406	0.658 263	0.647 844	0.613 032	0.620 332	0.643 371	0.652 611	**0.673 694**
Wdbc	*F-M*	0.950 643	0.949 163	0.950 982	0.923 582	0.936 692	0.900 051	0.953 471	0.949 109	**0.956 013**
	G-M	0.960 807	0.961 342	0.961 747	0.944 237	0.952 627	0.928 613	0.963 038	0.961 277	**0.966 552**
	AUC	0.990 767	**0.991 799**	0.990 021	0.984 179	0.985 202	0.975 830	0.990 580	0.989 200	0.990 290

Table 4 Experimental results obtained on 21 datasets by using NB classifier.

Dataset	Measure	ROS	SMOTE	STL	ADASYN	B1-SMOTE	B2-SMOTE	SLS	MWMOTE	SPMSC
Yeast1	*F-M*	0.456 125	0.458 645	0.457 923	0.458 730	0.457 897	0.455 935	0.457 501	0.460 728	**0.462 404**
	G-M	0.196 169	0.233 243	0.225 275	0.227 735	0.227 596	0.209 021	0.216 401	0.243 092	**0.261 612**
	AUC	0.516 403	0.522 134	0.520 714	0.521 882	0.520 712	0.516 920	0.519 509	0.525 670	**0.529 242**
Yeast3	*F-M*	0.219 601	0.233 448	**0.236 745**	0.218 333	0.218 486	0.206 591	0.227 033	0.231 906	0.228 198
	G-M	0.346 482	0.437 491	**0.452 808**	0.338 814	0.354 067	0.237 156	0.399 952	0.427 586	0.412 470
	AUC	0.560 114	0.591 879	**0.598 693**	0.557 084	0.558 186	0.526 028	0.577 891	0.588 095	0.580 502
Yeast5	*F-M*	0.114 764	0.169 266	0.175 865	0.164 341	**0.185 158**	0.124 036	0.141 530	0.174 980	0.178 614
	G-M	0.725 108	0.822 667	0.831 483	0.815 966	**0.841 108**	0.709 541	0.789 541	0.830 068	0.833 602
	AUC	0.763 194	0.832 828	0.840 120	0.826 926	**0.851 136**	0.752 525	0.807 734	0.838 731	0.841 509
Pima	*F-M*	0.645 945	0.646 374	0.637 224	0.640 895	0.632 705	0.634 009	0.646 661	**0.655 554**	0.647 528
	G-M	0.721 980	0.723 317	0.715 240	0.718 069	0.711 247	0.712 415	0.723 570	**0.731 123**	0.724 135
	AUC	0.725 552	0.726 687	0.717 552	0.720 149	0.713 418	0.714 284	0.726 687	**0.734 284**	0.726 418
Glass0	*F-M*	0.648 094	**0.652 034**	0.649 575	0.626 997	0.612 366	0.607 527	0.648 094	0.622 321	0.651 941
	G-M	0.651 987	0.658 523	0.652 378	0.634 395	0.619 109	0.606 533	0.651 987	0.637 977	**0.664 927**
	AUC	0.705 882	0.709 355	0.705 882	0.687 296	0.672 998	0.666 054	0.705 882	0.686 887	**0.712 418**
Haberman	*F-M*	0.397 980	0.409 007	0.431 200	0.395 654	0.406 893	0.423 564	0.410 009	0.414 651	**0.446 276**
	G-M	0.538 918	0.549 014	0.568 243	0.549 944	0.551 443	0.567 449	0.549 591	0.570 110	**0.591 058**
	AUC	0.607 151	0.613 518	0.623 410	0.596 742	0.608 647	0.616 267	0.613 142	0.603 791	**0.628 180**
Vehicle1	*F-M*	0.548 641	0.549 270	0.549 440	0.544 596	0.520 670	0.516 226	0.549 500	0.547 146	**0.552 504**
	G-M	0.658 387	0.655 214	0.659 961	0.622 060	0.624 599	0.593 398	0.659 831	0.652 197	**0.664 494**
	AUC	0.680 977	0.678 799	0.680 387	0.675 831	0.662 731	0.656 929	0.681 183	0.678 045	**0.683 493**
Vehicle3	*F-M*	0.538 009	0.540 603	0.543 196	0.541 562	**0.556 529**	0.546 873	0.546 298	0.537 867	0.539 489
	G-M	0.660 356	0.662 850	0.669 427	0.657 586	0.660 412	0.652 023	**0.670 367**	0.663 493	0.664 351
	AUC	0.671 981	0.675 110	0.679 031	0.680 509	**0.690 769**	0.684 460	0.680 609	0.674 314	0.675 110
Glass-0-1-2-3_vs_4-5-6	*F-M*	0.704 351	0.697 455	0.708 851	0.731 197	0.712 044	**0.757 099**	0.715 747	0.697 455	0.714 076
	G-M	0.786 217	0.783 029	0.793 611	0.810 372	0.800 491	**0.839 872**	0.796 798	0.783 029	0.801 492
	AUC	0.813 759	0.810 634	0.821 050	0.829 864	0.828 418	**0.854 216**	0.824 175	0.810 634	0.827 617
Vehicle0	*F-M*	0.550 449	0.556 710	0.555 883	0.495 115	0.491 435	0.488 522	0.551 346	0.555 603	**0.559 914**
	G-M	0.719 673	0.726 740	0.725 349	0.672 182	0.670 231	0.666 693	0.720 795	0.725 440	**0.729 619**
	AUC	0.738 867	0.745 127	0.744 286	0.677 976	0.674 297	0.670 531	0.739 582	0.743 334	**0.747 437**
Ecoli1	*F-M*	0.500 329	0.560 281	0.592 680	0.503 932	**0.644 887**	0.512 575	0.538 384	0.533 907	0.425 556
	G-M	0.630 874	0.711 160	0.740 656	0.608 356	**0.772 606**	0.612 207	0.685 275	0.674 823	0.417 278
	AUC	0.691 833	0.744 060	0.766 958	0.683 812	**0.800 580**	0.697 389	0.725 070	0.721 943	0.594 925
Ecoli3	*F-M*	0.436 652	0.505 780	0.538 696	0.474 945	**0.575 198**	0.519 686	0.513 131	0.552 585	0.534 887
	G-M	0.778 663	0.846 429	0.868 974	0.820 141	**0.886 917**	0.865 521	0.836 960	0.876 219	0.852 733
	AUC	0.799 525	0.852 683	0.872 639	0.831 038	**0.889 262**	0.869 306	0.843 947	0.879 284	0.857 105
Ilpd	*F-M*	0.556 796	**0.568 248**	0.566 032	0.560 267	0.565 164	0.553 269	0.558 290	0.560 685	0.564 265
	G-M	0.614 951	0.634 785	0.631 428	0.623 446	0.635 256	0.613 727	0.619 612	0.622 717	**0.637 195**
	AUC	0.679 356	**0.692 635**	0.690 220	0.684 164	0.688 953	0.676 882	0.681 748	0.684 175	0.688 930
Heart	*F-M*	0.820 148	0.811 612	0.820 031	0.819 730	0.821 077	0.820 968	0.816 641	0.818 033	**0.824 975**
	G-M	0.837 840	0.830 104	0.837 897	0.836 446	0.837 334	0.837 387	0.834 707	0.835 739	**0.842 185**
	AUC	0.838 276	0.830 731	0.838 276	0.836 652	0.837 530	0.837 530	0.834 987	0.835 864	**0.842 443**
Liver_disorders2	*F-M*	0.409 303	0.404 563	0.424 769	0.415 193	0.401 602	0.400 243	**0.426 514**	0.397 700	0.413 093
	G-M	0.427 289	0.453 305	**0.489 620**	0.450 340	0.444 083	0.449 047	0.458 026	0.449 125	0.469 329
	AUC	0.520 278	0.519 444	**0.548 333**	0.530 278	0.514 444	0.515 000	0.544 167	0.512 500	0.531 389
Liver_disorders4	*F-M*	0.240 762	0.278 479	0.260 661	0.264 386	0.273 366	0.264 478	0.244 441	0.269 838	**0.283 549**
	G-M	0.407 270	0.484 687	0.470 993	0.473 079	**0.523 571**	0.497 473	0.442 441	0.487 275	0.519 918
	AUC	0.477 222	0.535 000	0.512 222	0.512 222	0.536 944	0.520 833	0.485 833	0.526 111	**0.548 611**
Pima2	*F-M*	0.547 570	0.550 409	0.556 164	0.539 910	0.550 126	0.541 306	0.555 083	0.551 332	**0.563 640**
	G-M	0.729 104	0.732 563	0.734 107	0.730 484	0.734 527	0.733 970	0.733 161	0.732 823	**0.742 681**
	AUC	0.732 461	0.735 360	0.737 684	0.732 724	0.736 824	0.735 188	0.736 572	0.735 360	**0.745 936**
Segment	*F-M*	0.481 369	0.482 577	**0.482 894**	0.472 092	0.436 648	0.435 348	0.479 383	0.480 721	0.480 994
	G-M	0.794 299	**0.795 966**	0.795 843	0.783 563	0.742 439	0.744 212	0.793 147	0.791 630	0.791 519
	AUC	0.805 256	**0.807 510**	0.807 014	0.792 216	0.744 712	0.747 931	0.804 760	0.800 494	0.799 998
Tic-tac-toe	*F-M*	0.437 575	0.435 789	0.431 762	0.434 024	0.432 149	**0.450 205**	0.430 968	0.420 254	0.445 793
	G-M	0.416 158	0.465 035	0.468 068	0.472 304	0.517 304	0.500 811	0.476 358	0.487 673	**0.520 611**
	AUC	0.498 893	0.520 742	0.521 133	0.523 720	0.529 920	0.529 363	0.530 252	0.529 676	**0.549 589**
Winequality-red-4	*F-M*	**0.131 406**	0.104 350	0.104 415	0.102 687	0.121 150	0.115 523	0.102 375	0.106 528	0.115 048
	G-M	0.627 736	0.621 106	0.621 522	0.620 205	0.585 697	0.607 544	0.617 265	0.627 608	**0.645 387**
	AUC	0.649 211	0.631 720	0.632 042	0.629 782	0.629 076	0.633 591	0.629 038	0.637 041	**0.654 174**
Wdbc	*F-M*	0.902 657	0.907 750	0.904 793	0.917 330	**0.926 669**	0.920 888	0.902 657	0.911 713	0.911 947
	G-M	0.919 784	0.924 505	0.921 029	0.936 901	**0.945 502**	0.942 120	0.919 784	0.927 363	0.927 287
	AUC	0.920 599	0.925 316	0.922 003	0.937 161	**0.945 625**	0.942 382	0.920 599	0.928 109	0.928 125

Table 5 Experimental results obtained on 21 datasets by using SVM classifier.

Dataset	Measure	ROS	SMOTE	STL	ADASYN	B1-SMOTE	B2-SMOTE	SLS	MWMOTE	SPMSC
Yeast1	*F-M*	0.592 368	0.592 131	0.581 181	0.588 900	0.584 872	0.576 488	0.593 156	0.590 803	**0.596 054**
	G-M	0.713 346	0.713 843	0.705 003	0.709 820	0.701 247	0.692 685	0.714 714	0.712 832	**0.717 396**
	AUC	0.714 461	0.715 402	0.705 799	0.713 628	0.710 064	0.701 789	0.716 163	0.714 406	**0.718 483**
Yeast3	*F-M*	0.671 487	0.661 126	0.671 183	0.614 135	0.626 888	0.575 565	0.665 148	0.668 794	**0.677 518**
	G-M	**0.906 631**	0.890 026	0.899 324	0.905 097	0.900 683	0.893 496	0.897 850	0.901 446	0.898 952
	AUC	**0.907 212**	0.890 506	0.899 807	0.906 600	0.901 438	0.896 096	0.898 138	0.901 794	0.899 401
Yeast5	*F-M*	0.487 323	0.495 966	0.493 059	0.492 150	0.501 350	0.433 877	0.467 972	0.498 103	**0.503 425**
	G-M	0.967 164	0.968 239	0.967 881	0.967 885	0.968 957	0.957 730	0.964 281	0.968 600	**0.969 319**
	AUC	0.967 708	0.968 750	0.968 403	0.968 403	0.969 444	0.958 681	0.964 931	0.969 097	**0.969 792**
Pima	*F-M*	**0.676 214**	0.661 902	0.663 558	0.654 307	0.671 824	0.658 444	0.663 527	0.672 284	0.671 133
	G-M	**0.748 418**	0.736 703	0.738 310	0.730 347	0.744 003	0.732 103	0.738 494	0.745 400	0.744 304
	AUC	**0.749 940**	0.738 612	0.739 478	0.731 075	0.744 866	0.733 134	0.739 478	0.746 403	0.745 672
Glass0	*F-M*	0.649 787	0.662 953	0.662 596	0.654 503	0.650 612	0.658 080	0.654 341	0.662 953	**0.668 139**
	G-M	0.636 695	0.649 333	0.648 893	0.617 743	0.615 245	0.622 282	0.647 167	0.649 333	**0.660 019**
	AUC	0.714 461	0.721 814	0.725 286	0.711 397	0.707 925	0.715 278	0.718 342	0.721 814	**0.732 230**
Haberman	*F-M*	0.441 240	0.433 730	0.430 352	0.416 802	0.458 551	0.459 494	0.435 317	0.457 352	**0.468 581**
	G-M	0.579 542	0.573 780	0.572 188	0.559 309	0.599 428	0.596 414	0.574 643	0.597 680	**0.610 503**
	AUC	0.626 982	0.623 042	0.620 849	0.614 482	0.636 842	0.639 482	0.624 530	0.636 803	**0.642 685**
Vehicle1	*F-M*	0.692 512	0.696 358	0.696 701	0.688 481	**0.707 012**	0.705 141	0.698 989	0.696 667	0.695 970
	G-M	0.745 186	0.745 669	0.760 113	0.753 700	**0.773 932**	0.779 834	0.746 416	0.752 276	0.754 094
	AUC	0.777 648	0.782 176	0.786 267	0.781 495	0.801 638	**0.805 376**	0.779 998	0.783 062	0.784 554
Vehicle3	*F-M*	0.645 242	0.645 182	0.656 978	0.651 763	0.661 518	**0.681 340**	0.669 321	0.645 753	0.655 155
	G-M	0.713 410	0.719 744	0.723 238	0.716 800	0.720 371	0.729 039	**0.733 066**	0.712 103	0.727 605
	AUC	0.751 473	0.756 941	0.762 479	0.760 101	0.760 917	**0.775 859**	0.770 351	0.753 811	0.764 818
Glass-0-1-2-3_vs_4-5-6	*F-M*	0.739 797	0.781 917	0.782 399	0.744 398	0.775 010	**0.821 882**	0.768 846	0.743 954	0.799 921
	G-M	0.821 315	0.851 598	0.851 892	0.830 478	0.848 660	**0.896 149**	0.839 201	0.823 035	0.863 703
	AUC	0.834 909	0.860 313	0.860 313	0.841 551	0.857 264	**0.899 167**	0.850 698	0.838 034	0.873 053
Vehicle0	*F-M*	0.907 931	0.911 645	0.911 645	0.903 730	0.908 021	0.848 079	0.894 209	0.911 437	**0.914 718**
	G-M	0.963 438	0.963 495	0.963 495	0.962 004	0.963 574	0.941 985	0.955 549	0.963 497	**0.967 576**
	AUC	0.963 801	0.963 671	0.963 671	0.962 308	0.963 852	0.943 401	0.955 765	0.963 671	**0.967 900**
Ecoli1	*F-M*	0.771 700	0.747 271	0.755 952	0.761 996	**0.790 773**	0.784 675	0.765 418	0.772 630	0.769 159
	G-M	0.839 339	0.822 935	0.832 183	0.842 877	**0.869 705**	0.864 350	0.835 094	0.840 279	0.841 206
	AUC	0.853 680	0.837 304	0.845 477	0.857 193	**0.879 936**	0.872 243	0.849 353	0.854 338	0.854 308
Ecoli3	*F-M*	0.561 027	0.603 521	0.599 024	0.553 674	0.605 588	0.589 680	0.586 079	0.614 486	**0.616 453**
	G-M	0.875 045	0.889 053	0.889 106	0.881 495	0.888 942	0.894 011	0.885 688	0.892 399	**0.894 259**
	AUC	0.877 018	0.890 263	0.890 285	0.884 262	0.890 285	**0.895 885**	0.886 974	0.893 618	0.895 263
Ilpd	*F-M*	0.562 560	0.566 082	**0.576 328**	0.571 439	0.558 329	0.553 462	0.572 568	0.562 131	0.574 120
	G-M	0.639 392	0.647 703	0.657 334	0.647 780	0.634 284	0.628 180	0.649 217	0.635 958	**0.659 075**
	AUC	0.688 177	0.690 763	**0.700 966**	0.696 696	0.681 482	0.677 249	0.697 970	0.685 248	0.699 845
Heart	*F-M*	0.818 424	0.811 556	0.813 251	0.799 889	0.808 144	**0.827 710**	0.803 223	0.808 167	0.821 585
	G-M	0.836 829	0.830 491	0.831 536	0.819 464	0.825 719	**0.844 010**	0.822 917	0.827 216	0.838 914
	AUC	0.837 577	0.830 909	0.831 697	0.819 986	0.825 818	**0.844 109**	0.823 364	0.827 531	0.839 065
Liver_disorders2	*F-M*	0.450 104	0.446 553	0.431 227	0.421 169	0.449 695	0.398 329	0.404 643	0.430 548	**0.467 003**
	G-M	**0.602 826**	0.589 149	0.551 726	0.560 439	0.593 400	0.531 377	0.533 357	0.569 691	0.594 609
	AUC	0.607 500	0.595 833	0.572 222	0.569 444	0.598 333	0.545 000	0.549 444	0.576 944	**0.611 111**
Liver_disorders4	*F-M*	0.368 846	0.357 452	0.381 826	0.378 776	0.333 571	0.242 191	0.299 029	0.351 667	**0.387 227**
	G-M	0.615 953	0.610 874	0.646 806	0.639 850	0.566 946	0.465 268	0.521 900	0.617 224	**0.650 434**
	AUC	0.634 722	0.627 222	0.656 111	0.658 889	0.598 056	0.501 389	0.559 444	0.629 722	**0.670 278**
Pima2	*F-M*	0.550 264	0.518 469	0.549 494	0.523 274	0.527 983	0.535 752	**0.557 247**	0.542 289	0.548 858
	G-M	0.736 702	0.708 609	0.736 243	0.722 478	0.719 777	0.730 171	**0.738 545**	0.732 339	0.736 923
	AUC	0.738 389	0.712 784	0.738 289	0.723 642	0.721 814	0.731 642	**0.741 490**	0.734 289	0.738 177
Segment	*F-M*	0.641 972	0.642 479	0.641 955	0.631 502	0.632 087	0.612 918	0.642 782	0.646 316	**0.647 629**
	G-M	0.899 813	0.898 413	0.898 123	0.895 871	0.895 336	0.886 025	0.899 237	0.900 075	**0.900 622**
	AUC	0.904 031	0.902 263	0.902 011	0.900 739	0.899 991	0.891 657	0.903 283	0.903 778	**0.904 283**
Tic-tac-toe	*F-M*	0.447 417	0.450 643	0.441 412	0.451 107	0.450 734	0.441 919	**0.467 160**	0.454 993	0.462 661
	G-M	0.375 374	0.416 403	0.388 171	0.386 118	**0.484 603**	0.469 514	0.466 263	0.406 388	0.388 969
	AUC	0.491 412	0.502 660	0.486 441	0.500 281	0.513 803	0.496 946	**0.522 584**	0.499 097	0.507 446
Winequality-red-4	*F-M*	0.139 225	0.134 963	0.141 861	0.132 388	0.204 639	0.184 780	**0.205 321**	0.155 597	0.159 273
	G-M	0.705 717	0.687 058	0.696 039	0.685 211	0.641 563	0.652 654	0.711 038	0.702 645	**0.713 418**
	AUC	0.710 055	0.695 016	0.703 105	0.692 754	0.684 233	0.683 974	**0.728 409**	0.708 100	0.717 992
Wdbc	*F-M*	0.962 548	0.969 261	0.964 581	0.949 967	0.951 637	0.892 277	0.964 397	0.969 297	**0.973 980**
	G-M	0.970 696	0.974 925	0.971 117	0.961 968	0.963 518	0.919 298	0.970 061	0.974 870	**0.978 706**
	AUC	0.970 881	0.975 095	0.971 332	0.962 454	0.963 859	0.921 833	0.970 362	0.975 079	**0.978 842**



(a) KNN

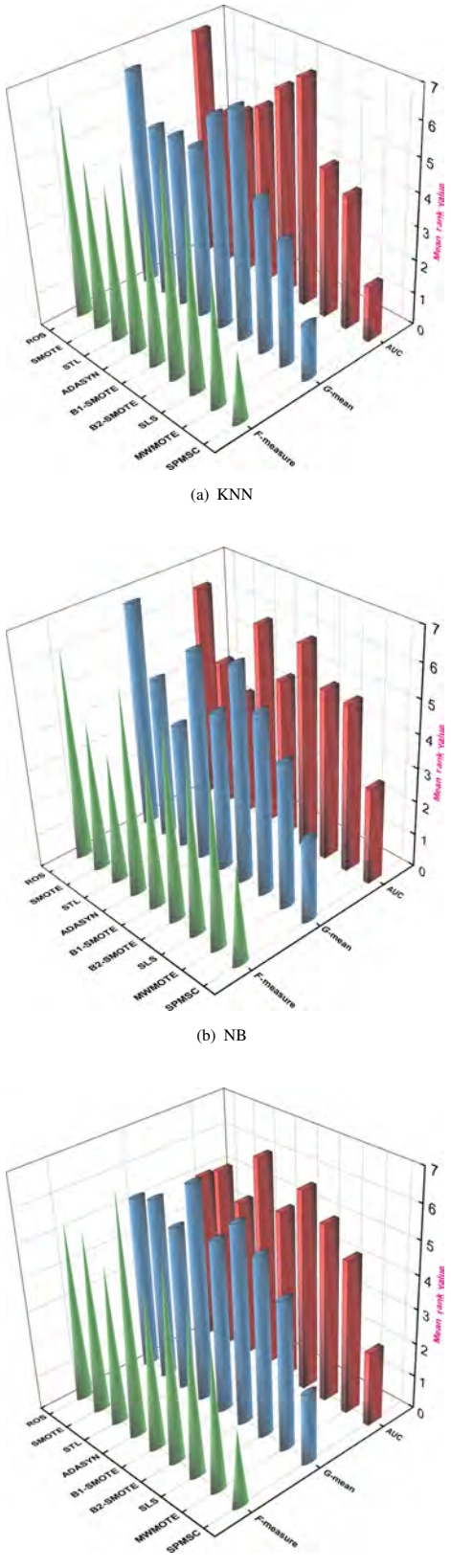
(b) NB

(c) SVM

Fig. 8 Average results of nine sampling methods on 21 datasets achieved by (a) KNN classifier, (b) NB classifier, and (c) SVM classifier.

is only lower than the STL method in the *F*-measure metric, lower than the SMOTE, STL, and MWMOTE methods in the *G*-mean metric, and lower than the SMOTE and STL methods in the *AUC* metric, but still ranked in the top from an overall perspective.

To more intuitively compare the performance of SPMSC with the other eight sampling methods, the average rank of each method on 21 datasets is calculated. As shown in Fig. 9, the lower average rank values represent a higher rank, and from Fig. 9, we can get that SPMSC method ranks the highest on three classifiers.



(a) KNN

(b) NB

(c) SVM

Fig. 9 Average rank results of nine sampling methods on 21 datasets achieved by (a) KNN classifier, (b) NB classifier, and (c) SVM classifier.

4.4.3 Contrastive analysis of performance improvement

Further, to compare the classification performance improvement of the data processed using the sampling method compared to the original data, we select the *AUC* metric from the results obtained on KNN, NB, and SVM classifiers for comparison. Table 6 shows the *AUC* results obtained by directly classifying 21 datasets with the three classifiers. Figure 10 describes the difference between the *AUC* values of each dataset in Tables 3–5 and the *AUC* values of each dataset in Table 6. As shown in Fig. 10, a positive value indicates that the data processed by the sampling method on the same classifier have better classification performance than the original data, and a negative value indicates that the data processed by the sampling method on the same classifier have lower classification performance than the original data. It is not difficult to find that the SPMSC method obtains the highest number of maximum differences, which indicates that the SPMSC outperforms the other eight comparison sampling methods. It is also observed that when KNN and NB classifiers are used, the *AUC* differences between the data processed using sampling methods and the original data are positive and negative. Yet when SVM classifier is used, almost all the differences are positive. It shows that the performance of the same sampling method varies for different classifiers.

Table 6 ***AUC*** **results obtained for the original dataset on KNN, NB, and SVM classifiers.**

Dataset	*AUC* result		
	KNN classifer	NB classifer	SVM classifer
Yeast1	0.735 060	0.517 142	0.594 049
Yeast3	0.922 960	0.583 555	0.809 665
Yeast5	0.947 159	0.804 167	0.500 000
Pima	0.767 299	0.709 642	0.713 388
Glass0	0.829 668	0.705 882	0.601 307
Haberman	0.630 394	0.566 447	0.504 762
Vehicle1	0.787 479	0.687 085	0.575 065
Vehicle3	0.811 017	0.679 902	0.516 509
Glass-0-1-2-3_vs_4-5-6	0.876 073	0.824 175	0.834 199
Vehicle0	0.981 349	0.723 781	0.948 378
Ecoli1	0.863 790	0.725 909	0.808 614
Ecoli3	0.921 444	0.774 415	0.498 333
Ilpd	0.626 836	0.680 558	0.500 000
Heart	0.877 359	0.845 733	0.836 000
Liver_disorders2	0.521 250	0.508 056	0.500 000
Liver_disorders4	0.585 833	0.511 389	0.500 000
Pima2	0.730 111	0.686 971	0.611 125
Segment	0.979 645	0.804 507	0.743 940
Tic-tac-toe	0.717 654	0.525 865	0.500 000
Winequality-red-4	0.600 756	0.527 612	0.500 000
Wdbc	0.991 241	0.920 599	0.970 813

4.4.4 Wilcoxon signed rank test

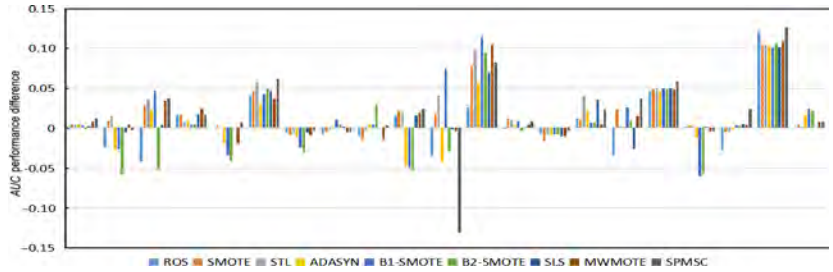
In this subsection, from the perspective of statistical analysis, we use a nonparametric test called the Wilcoxon signed rank test[46] to verify the statistical significance of the proposed method with the other eight sampling methods. The results are shown in Table 7. When using the KNN classifier, the p-values of all three measures are below the significance level $\alpha = 0.05$. Therefore, all null hypotheses are rejected, which indicates a significant improvement of SPMSC compared to the other eight sampling methods. When using the NB classifier, the null hypothesis cannot be rejected as follows: STL and B1-SMOTE under F-measure, STL under G-mean, and STL, B1-SMOTE, and MWMOTE under *AUC*, indicating that SPMSC does not have a significant improvement in comparison with these methods. Except for these cases, all null hypotheses are rejected, indicating that SPMSC has a significant improvement in comparison with other methods. When using the SVM classifier, SPMSC has significant improvement compared to other methods except for B2-SMOTE under G-mean measure.

4.4.5 Running time comparison

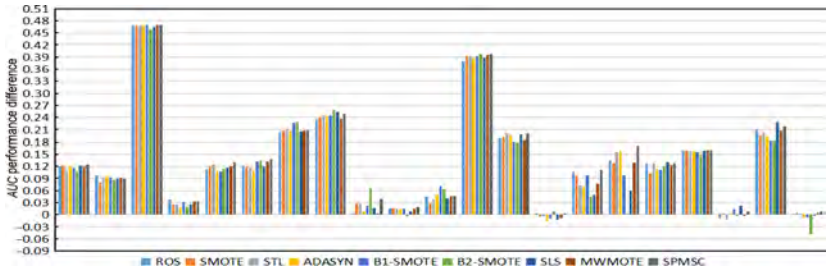
In this subsection, from the perspective of running time, we compare the time cost of the SPMSC method with the comparison method, and the results are shown in Table 8. From the results, it can be found that random oversampling has the shortest running time and SMOTE ranks the second due to their simple implementation mechanism. Compared to the original method SMOTE, some SMOTE variants such as ADASYN, B1-SMOTE, B2-SMOTE, etc., require some additional time for weighting the samples and deciding the boundary samples. Specifically, the SPMSC method requires some additional running time compared to the comparison method due to the computation of the distance matrix and the discrimination of the boundary minority samples. But, this extra computing time of a few or tens of seconds is acceptable, especially for offline computing.

In summary, the comparison and analysis of the experimental results can be concluded that the SPMSC method outperforms the comparison method on most datasets. Furthermore, a statistical analysis method called the Wilcoxon sign rank test is used to further demonstrate a significant difference between the SPMSC method and the comparison method.

(a) KNN



(b) NB



(c) SVM

Fig. 10 Performance difference between the dataset processed using the sampling method and the original dataset on the three classifiers.

5 Conclusion

In this paper, we propose a joint sample position based noise filtering and mean shift clustering (SPMSC) method to deal with imbalanced data. The advantages of SPMSC are that it can adequately filter noisy samples by utilizing information about the position and distribution of minority samples relative to the majority; it uses a mean shift algorithm to cluster minority samples to prevent duplicate data from being generated at the sample synthesis stage due to the creation of inappropriate class clusters; and it uses a data cleaning method to further eliminate class overlap in the processed dataset. For evaluating the proposed method, 21 datasets with different imbalance ratios and eight popular sampling algorithms are used, and the experimental results show the effectiveness of SPMSC.

Acknowledgment

This work was supported in part by the Anhui Provincial Natural Science Foundation (No. 2208085MF168) and the Program for Synergy Innovation in the Anhui Higher Education Institutions of China (Nos. GXXT-2019-025 and GXXT-2022-052).

References

[1] P. Branco, L. Torgo, and R. P. Ribeiro, A survey of predictive modeling on imbalanced domains, *ACM Comput. Surv.*, vol. 49, no. 2, pp. 1–50, 2016.

[2] S. Fotouhi, S. Asadi, and M. W. Kattan, A comprehensive data level analysis for cancer diagnosis on imbalanced data, *J. Biomed. Inform.*, vol. 90, p. 103089, 2019.

[3] J. Yang, X. Wu, J. Liang, X. Sun, M. -M. Cheng, P. L. Rosin, and L. Wang, Self-paced balance learning for clinical skin disease recognition, *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 8, pp. 2832–2846, 2019.

[4] K. Luo, G. Wang, Q. Li, and J. Tao, An improved SVM-RFE based on F-statistic and mPDC for gene selection in cancer classification, *IEEE Access*, vol. 7, pp. 147617–147628, 2019.

[5] W. W. Soh and R. M. Yusuf, Predicting credit card fraud on a imbalanced data, *Int. J. Data Sci. Adv. Anal.*, vol. 1, no. 1, pp. 12–17, 2019.

Table 7 Wilcoxon signed rank test between the SPMSC method and the comparison method.

Classifer	*F*-measure			*G*-mean			*AUC*		
	Algorithm	*p*-value	$\alpha = 0.05$	Algorithm	*p*-value	$\alpha = 0.05$	Algorithm	*p*-value	$\alpha = 0.05$
KNN	ROS	0.001 021	Rejected	ROS	0.000 123	Rejected	ROS	0.000 106	Rejected
	SMOTE	0.000 214	Rejected	SMOTE	0.000 141	Rejected	SMOTE	0.000 123	Rejected
	STL	0.000 419	Rejected	STL	0.000 080	Rejected	STL	0.000 702	Rejected
	ADASYN	0.000 281	Rejected	ADASYN	0.000 321	Rejected	ADASYN	0.000 321	Rejected
	B1-SMOTE	0.002 642	Rejected	B1-SMOTE	0.000 162	Rejected	B1-SMOTE	0.000 321	Rejected
	B2-SMOTE	0.000 281	Rejected	B2-SMOTE	0.000 080	Rejected	B2-SMOTE	0.000 069	Rejected
	SLS	0.000 106	Rejected	SLS	0.000 702	Rejected	SLS	0.001 155	Rejected
	MWMOTE	0.007 066	Rejected	MWMOTE	0.004 615	Rejected	MWMOTE	0.000 321	Rejected
NB	ROS	0.006 363	Rejected	ROS	0.001 304	Rejected	ROS	0.002 098	Rejected
	SMOTE	0.017 270	Rejected	SMOTE	0.011 738	Rejected	SMOTE	0.030 365	Rejected
	STL	0.098 741	Not rejected	STL	0.169 775	Not rejected	STL	0.305 198	Not rejected
	ADASYN	0.012 949	Rejected	ADASYN	0.003 705	Rejected	ADASYN	0.006 363	Rejected
	B1-SMOTE	0.180 841	Not rejected	B1-SMOTE	0.035 480	Rejected	B1-SMOTE	0.091 845	Not rejected
	B2-SMOTE	0.027 306	Rejected	B2-SMOTE	0.014 269	Rejected	B2-SMOTE	0.029 829	Rejected
	SLS	0.020 812	Rejected	SLS	0.002 356	Rejected	SLS	0.017 270	Rejected
	MWMOTE	0.038 632	Rejected	MWMOTE	0.049 552	Rejected	MWMOTE	0.058 186	Not rejected
SVM	ROS	0.000 214	Rejected	ROS	0.002 356	Rejected	ROS	0.001 021	Rejected
	SMOTE	0.000 069	Rejected	SMOTE	0.000 702	Rejected	SMOTE	0.000 060	Rejected
	STL	0.000 245	Rejected	STL	0.000 367	Rejected	STL	0.000 281	Rejected
	ADASYN	0.000 060	Rejected	ADASYN	0.000 245	Rejected	ADASYN	0.000 321	Rejected
	B1-SMOTE	0.015 707	Rejected	B1-SMOTE	0.042 020	Rejected	B1-SMOTE	0.017 270	Rejected
	B2-SMOTE	0.017 270	Rejected	B2-SMOTE	0.068 035	Not rejected	B2-SMOTE	0.038 632	Rejected
	SLS	0.017 270	Rejected	SLS	0.002 961	Rejected	SLS	0.011 738	Rejected
	MWMOTE	0.000 214	Rejected	MWMOTE	0.002 356	Rejected	MWMOTE	0.000 321	Rejected

Table 8 Runtime of the SPMSC method and the comparison method.

Dataset	Runtime (s)								
	ROS	SMOTE	STL	ADASYN	B1-SMOTE	B2-SMOTE	SLS	MWMOTE	SPMSC
Yeast1	0.0008	0.0041	0.0309	0.0135	0.0206	0.0226	0.0405	23.1532	11.7985
Yeast3	0.0009	0.0035	0.0368	0.0075	0.0102	0.0117	0.0575	3.6535	3.3682
Yeast5	0.0009	0.0034	0.0328	0.0051	0.0059	0.0069	0.0656	1.4152	1.6774
Pima	0.0006	0.0025	0.0088	0.0071	0.0120	0.0132	0.0139	8.0749	4.1992
Glass0	0.0005	0.0012	0.0028	0.0023	0.0038	0.0043	0.0040	0.5362	0.3733
Haberman	0.0005	0.0014	0.0028	0.0025	0.0040	0.0040	0.0054	0.8721	0.6220
Vehicle1	0.0008	0.0042	0.0411	0.0126	0.0168	0.0181	0.0473	6.5939	4.3879
Vehicle3	0.0008	0.0043	0.0400	0.0125	0.0168	0.0175	0.0496	5.9462	3.9708
Glass-0-1-2-3_vs_4-5-6	0.0005	0.0012	0.0028	0.0021	0.0078	0.0080	0.0091	0.3630	0.3203
Vehicle0	0.0008	0.0043	0.0427	0.0122	0.0160	0.0159	0.0489	4.8383	3.6301
Ecoli1	0.0005	0.0014	0.0043	0.0028	0.0043	0.0050	0.0080	0.7334	0.5603
Ecoli3	0.0006	0.0015	0.0049	0.0023	0.0030	0.0031	0.0107	0.3465	0.3728
Ilpd	0.0006	0.0020	0.0073	0.0050	0.0080	0.0088	0.0139	3.6298	2.2631
Heart	0.0005	0.0014	0.0035	0.0041	0.0056	0.0059	0.0035	1.4638	0.8093
Liver_disorders2	0.0006	0.0015	0.0034	0.0026	0.0038	0.0039	0.0062	0.7293	0.4516
Liver_disorders4	0.0006	0.0014	0.0030	0.0020	0.0025	0.0027	0.0070	0.3012	0.2630
Pima2	0.0007	0.0021	0.0076	0.0044	0.0063	0.0075	0.0163	2.0576	1.5813
Segment	0.0013	0.0093	0.3011	0.0372	0.0425	0.0430	0.1790	13.4986	11.6032
Tic-tac-toe	0.0007	0.0030	0.0144	0.0090	0.0133	0.0136	0.0169	11.6969	7.2688
Winequality-red-4	0.0010	0.0035	0.0355	0.0060	0.0069	0.0076	0.0746	1.3710	1.9317
Wdbc	0.0007	0.0034	0.0175	0.0105	0.0152	0.0153	0.0208	4.1194	2.7673

[6] H. Yu, J. Ni, and J. Zhao, ACOSampling: An ant colony optimization-based undersampling method for classifying imbalanced DNA microarray data, *Neurocomputing*, vol. 101, pp. 309–318, 2013.

[7] Y. Li, H. Guo, Q. Zhang, M. Gu, and J. Yang, Imbalanced text sentiment classification using universal and domain-specific knowledge, *Knowl.-Based Syst.*, vol. 160, pp. 1–15, 2018.

[8] V. Engen, J. Vincent, and K. Phalp, Enhancing network based intrusion detection for imbalanced data, *Int. J. Knowl.-Based Intell. Eng. Syst.*, vol. 12, nos. 5&6, pp. 357–367, 2008.

[9] R. Abdulhammed, M. Faezipour, A. Abuzneid, and A. AbuMallouh, Deep and machine learning approaches for anomaly-based intrusion detection of imbalanced network traffic, *IEEE Sensors Lett.*, vol. 3, no. 1, p. 7101404, 2018.

[10] A. Azaria, A. Richardson, S. Kraus, and V. S. Subrahmanian, Behavioral analysis of insider threat: A

survey and bootstrapped prediction in imbalanced data, *IEEE Trans. Comput. Social Syst.*, vol. 1, no. 2, pp. 135–155, 2014.

[11] H. He and E. A. Garcia, Learning from imbalanced data, *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, 2009.

[12] S. Maldonado and J. López, Imbalanced data classification using second-order cone programming support vector machines, *Pattern Recognit.*, vol. 47, no. 5, pp. 2070–2079, 2014.

[13] D. J. Yu, J. Hu, Z. M. Tang, H. B. Shen, J. Yang, and J. Y. Yang, Improving protein-ATP binding residues prediction by boosting SVMs with random under-sampling, *Neurocomputing*, vol. 104, pp. 180–190, 2013.

[14] S. Alshomrani, A. Bawakid, S. O. Shim, A. Fernández, and F. Herrera, A proposal for evolutionary fuzzy systems using feature weighting: Dealing with overlapping in imbalanced datasets, *Knowl.-Based Syst.*, vol. 73, pp. 1–17, 2015.

[15] R. C. Prati, G. E. Batista, and M. C. Monard, Class imbalances versus class overlapping: An analysis of a learning system behavior, in *Proc. 3^{rd} Mexican International Conference on Artificial Intelligence*, Mexico City, Mexico, 2004, pp. 312–321.

[16] S. A. Shahee and U. Ananthakumar, An adaptive oversampling technique for imbalanced datasets, in *Proc. 18^{th} Industrial Conference on Data Mining*, New York, NY, USA, 2018, pp. 1–16.

[17] N. Japkowicz and S. Stephen, The class imbalance problem: A systematic study, *Intell. Data Anal.*, vol. 6, no. 5, pp. 429–449, 2002.

[18] T. Jo and N. Japkowicz, Class imbalances versus small disjuncts, *ACM SIGKDD Explor. Newslett.*, vol. 6, no. 1, pp. 40–49, 2004.

[19] N. Japkowicz, Concept-learning in the presence of between-class and within-class imbalances, in *Proc. 14^{th} Biennial Conference of the Canadian Society for Computational Studies of Intelligence*, Ottawa, Canada, 2001, pp. 67–77.

[20] H. A. Majzoub and I. Elgedawy, AB-SMOTE: An affinitive borderline SMOTE approach for imbalanced data binary classification, *Int. J. Mach. Learn. Comput.*, vol. 10, no. 1, pp. 31–37, 2020.

[21] T. Zhu, Y. Lin, and Y. Liu, Synthetic minority oversampling technique for multiclass imbalance problems, *Pattern Recognit.*, vol. 72, pp. 327–340, 2017.

[22] A. Onan, Consensus clustering-based undersampling approach to imbalanced learning, *Sci. Program.*, vol. 2019, p. 5901087, 2019.

[23] 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, p. 113504, 2020.

[24] L. Jiang, C. Li, and S. Wang, Cost-sensitive Bayesian network classifiers, *Pattern Recognition Lett.*, vol. 45, pp. 211–216, 2014.

[25] L. Jiang, C. Qiu, and C. Li, A novel minority cloning technique for cost-sensitive learning, *Int. J. Pattern Recognit. Artif. Intell.*, vol. 29, no. 4, p. 1551004, 2015.

[26] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches, *IEEE Trans. Syst., Man, Cybern. C* (*Appl. Rev.*), vol. 42, no. 4, pp. 463–484, 2012.

[27] I. Mani, I. Zhang, J. Zhang, and K. S. Mani, KNN approach to unbalanced data distributions: A case study involving information extraction, in *Proc. Workshop on Learning from Imbalanced Datasets*, Washington, DC, USA, 2003, pp. 1–7.

[28] R. C. Holte, L. E. Acker, and B. W. Porter, Concept learning and the problem of small disjuncts, in *Proc. 11^{th} International Joint Conference on Artificial Intelligence*, Detroit, MI, USA, 1989, pp. 813–818.

[29] Z. Wang and H. Wang, Global data distribution weighted synthetic oversampling technique for imbalanced learning, *IEEE Access*, vol. 9, pp. 44770–44783, 2021.

[30] H. Yu, C. Mu, C. Sun, W. Yang, X. Yang, and X. Zuo, Support vector machine-based optimized decision threshold adjustment strategy for classifying imbalanced data, *Knowl.-Based Syst.*, vol. 76, pp. 67–78, 2015.

[31] J. V. Hulse, T. M. Khoshgoftaar, and A. Napolitano, Experimental perspectives on learning from imbalanced data, in *Proc. 24^{th} International Conference on Machine Learning*, Corvallis, OR, USA, 2007, pp. 935–942.

[32] 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, no. 1, pp. 321–357, 2002.

[33] H. He, Y. Bai, E. A. Garcia, and S. Li, ADASYN: Adaptive synthetic sampling approach for imbalanced learning, in *Proc. 2008 IEEE International Joint Conference on Neural Networks* (*IEEE World Congress on Computational Intelligence*), Hong Kong, China, 2008, pp. 1322–1328.

[34] G. Douzas, F. Bacao, and F. Last, Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE, *Inf. Sci.*, vol. 465, pp. 1–20, 2018.

[35] S. Barua, M. M. Islam, X. Yao, and K. Murase, MMWOTE—Majority weighted minority oversampling technique for imbalanced data set learning, *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 2, pp. 405–425, 2014.

[36] H. Han, W. Y. Wang, and B. H. Mao, Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning, in *Proc. International Conference on Intelligent Computing*, Hefei, China, 2005, pp. 878–887.

[37] C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap, DBSMOTE: Density-based synthetic minority oversampling technique, *Appl. Intell.*, vol. 36, no. 3, pp. 664–684, 2012.

[38] C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap, Safe-level-SMOTE: Safe-level-synthetic minority oversampling technique for handling the class imbalanced problem, in *Proc. 13^{th} Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Bangkok, Thailand, 2009, pp. 475–482.

[39] I. Nekooeimehr and S. K. Lai-Yuen, Adaptive semi-unsupervised weighted oversampling (A-SUWO) for imbalanced datasets, *Expert Syst. Appl.*, vol. 46, pp. 405–416, 2016.

[40] G. E. Batista, A. L. Bazzan, and M. C. Monard, Balancing

training data for automated annotation of keywords: A case study, in *Proc. 2nd Brazilian Workshop on Bioinformatics*, Macaé, Brazil, 2003, pp. 10–18.

[41] T. Cover and P. Hart, Nearest neighbor pattern classification, *IEEE Trans. Inf. Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[42] T. F. Chan, G. H. Golub, and R. J. LeVeque, Updating formulae and a pairwise algorithm for computing sample variances, in *Proc. COMPSTAT 1982 5th Symposium*, Toulouse, France, 1982, pp. 30–41.

[43] C. Cortes and V. Vapnik, Support-vector networks, *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[44] A. Onan, Bidirectional convolutional recurrent neural network architecture with group-wise enhancement mechanism for text sentiment classification, *J. King Saud Univ. Comput. Inf. Sci.*, vol. 34, no. 5, pp. 2098–2117, 2022.

[45] N. Japkowicz, Assessment metrics for imbalanced learning, in *Imbalanced Learning: Foundations, Algorithms, and Applications*, H. He and Y. Ma, eds. Hoboken, NJ, USA: John Wiley & Sons, 2013, pp. 187–206.

[46] G. W. Corder and D. I. Foreman, *Nonparametric Statistics: A Step-by-Step Approach*. Hoboken, NJ, USA: John Wiley & Sons, 2014.

Lilong Duan received the bachelor degree from Suzhou University, Suzhou, China in 2021. He is currently pursuing the master degree at the School of Computer Science and Technology, Anhui University of Technology, Maanshan, China. His research interests include machine learning and data mining.

Jun Huang received the PhD degree in computer science from University of Chinese Academy of Sciences (UCAS), Beijing, China in 2017. He is an associate professor of Anhui University of Technology, China. From Oct 1st, 2019 to Sep 30, 2020, he was a post doctoral researcher at The University of Tokyo, Japan. His research interest is in machine learning and data mining, particularly in multi-label learning and multi-view learning. He has published more than ten research papers in top conferences and journals in machine learning and data mining and served as the reviewer and technique committee member for many journals and conferences. He won the third prize of Science and Technology Progress award of Anhui Province in 2019, President Award of Chinese Academy of Sciences, China in 2017, and Excellent PhD Dissertation of University of Chinese Academy of Sciences in 2018.

Wei Xue received the PhD degree in computer science and technology from Nanjing University of Science and Technology, Nanjing, China in 2017. He is currently an associate professor at the School of Computer Science and Technology, Anhui University of Technology, Maanshan, China. From July 2014 to April 2017, he was a visiting student with the Institute of Automation, Chinese Academy of Sciences, Beijing, China. From July 2019 to July 2021, he was a postdoctoral fellow with the National Key Laboratory of Science and Technology on Automatic Target Recognition, National University of Defense Technology, Changsha, China. His research interests include machine learning and its applications.

Xiao Zheng received the PhD degree in computer science and technology from Southeast University, Nanjing, China in 2014. He is currently a professor at the School of Computer Science and Technology, Anhui University of Technology, Maanshan, China. His research interests include service computing, social computing, and computer network. He has been a guest editor of *IEICE Transactions on Communications*. He is a senior member of CCF and a member of ACM.