A Novel Method to Create Synthetic Samples with Autoencoder Multi-layer Extreme Learning Machine

Yulin He[1,2(*)], Qihang Huang[1], Shengsheng Xu[1], Joshua Zhexue Huang[1,2]

[1]College of Computer Science & Software Engineering, Shenzhen University, Shenzhen 518060, China
[2]National Engineering Laboratory for Big Data System Computing Technology, Shenzhen University, Shenzhen 518060, China
yulinhe@szu.edu.cn,1900271056@email.szu.edu.cn,2070276067@email.szu.edu.cn,zx.huang@szu.edu.cn

Abstract. The imbalanced classification is an important branch of supervised learning and plays the important roles in many application fields. Compared with the sophisticated improvements on classification algorithms, it is easier to obtain the good performance by synthesizing the minority class samples so that the classification algorithms can be trained based on the balanced data sets. In consideration of the strong representation ability of multi-layer extreme learning machine (MLELM), this paper proposes a new method to create the synthetic minority class samples based on auto-encoder ML-ELM (simplified as AE-MLELM-SynMin). Firstly, an AE-MLELM is trained to obtain the deep feature encodings of original minority class samples. Secondly, the crossover and mutation operations are preformed on the original deep feature encodings and a number of new deep feature encodings are generated. Thirdly, the synthetic minority class samples are created by transforming the new deep feature encodings with AE-MLELM. Finally, the persuasive experiments are conducted to demonstrate the effectiveness of AE-MLELM-SynMin method. The experimental results show that our method can obtain the better imbalanced classification performance than SMOTE, Borderline-SMOTE, Random-SMOTE, and SMOTE-IPF methods.

Keywords: Imbalanced classification, minority class, synthetic samples, SMOTE, Autoencoder MLELM

1 Introduction

The imbalanced classification or imbalanced learning is an important branch of data mining and machine learning, which has the broad applications in the actual fields [1], e.g., defect detection, fault detection, medical diagnosis, fraud detection. The imbalanced classification problem means that a classifier is constructed based on a labeled data set, where there are the obvious differences among the numbers of samples belonging to different classes. For example, the defective samples in defect detection application always account for a small part of the total sample, while the qualified samples account for the majority.

The traditional classification algorithms are designed for the balanced data sets. For the imbalanced data sets, the traditional classification methods can not effectively learn from the minority class samples. Although their classification accuracies are still relatively high for imbalanced classification problems, the higher classification accuracies usually cannot effectively reflect the effectiveness of classification algorithms. Because for an imbalanced classification data set, if the classification method judges all samples as normal ones, that is to say, all the abnormal samples are also judged as normal ones, then the final classification accuracy will be high, but such results are meaningless for many practical applications.

At present, the studies on how to solve the imbalanced classification problems mainly focus on the algorithm-driven methods [2–6] and data-driven methods [7, 9, 8, 10–12]. The algorithm-driven method is to construct an algorithm which places more emphasis on the minority class samples. The data-driven method is mainly to make the number of different labeled training samples more balanced. The two simplest methods are random over-sampling and random under-sampling. The former supplements the minority class samples by randomly selecting the minority class samples repeatedly, while the latter randomly selects a small number of majority class samples. In most cases, the over-sampling method is better than the under-sampling method. This is because the under-sampling method eliminates the valuable data. In the actual scenarios, the data are scarce and valuable, so the over-sampling method is often a better choice. Compared with the method based on the algorithm level, the method based on the data level has the characteristics of simple implementation and low computational complexity, so the focus of this research is mainly on the data level method.

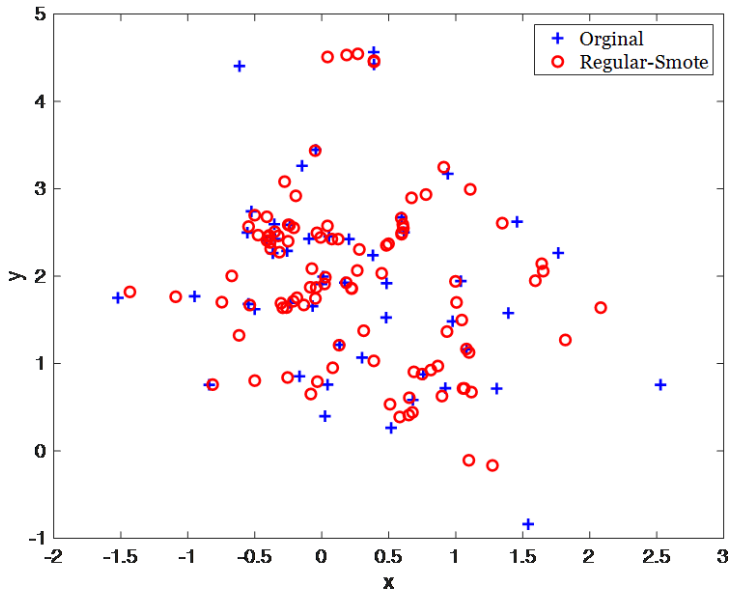
The classic data-level method is the synthetic minority over-sampling technology (SMOTE) [7]. SMOTE does not simply copy the samples. Its principle is as follows. For each minority sample, a sample is randomly selected from its k nearest neighbors and then a sample point is randomly selected on the line of two samples. Because of SMOTE's well imbalanced learning performances, it attracts the widespread attention from scholars. Han et al. [8] proposed the Borderline-SMOTE method. The main idea is to oversample the minority class samples on the boundary between the majority class and minority class samples. Dong et al. [9] proposed a more general Random-SMOTE method to generate new sample points among three sample points. Compared with the classic SMOTE method, the distribution of sample points generated by this method is more uniform and this method can greatly improve the sparseness of sample space. Sáezet al. [10] introduced an Iterative-Partitioning Filter (IPF) to extend the SMOTE algorithm, the SMOTE-IPF algorithm, which overcomes the problems caused by noisy and boundary samples in imbalanced data sets. Lee et al. [13] proposed a Guassian-SMOTE algorithm that combined Gaussian probability distribution with SMOTE algorithm. This method obeies Gaussian probability distribution when SMOTE method is used to create minority class samples. Douzas et al. [14] proposed the k-means-SMOTE algorithm combining k-means and SMOTE algorithm. Based on the k-means algorithm, the method generates more samples in the sparse area of minority class samples than in the dense area of minority class samples.

The experimental results show that these SMOTE variants create minority class samples to achieve better results on general classifiers. However, the comprehensive analysis shows that there are two major defects in this method. The first defect is the inconsistency of probability distribution between the original samples and the synthetic samples and the second defect is the lack of information amount. This is because the new sample points generated by SMOTE method are randomly selected on the line of two minority class sample points, which causes the new sample points generated to be too fixed and results in insufficient information amount. Although some variants of SMOTE method improves performance to some extent, they does not improve the two shortcomings mentioned earlier.

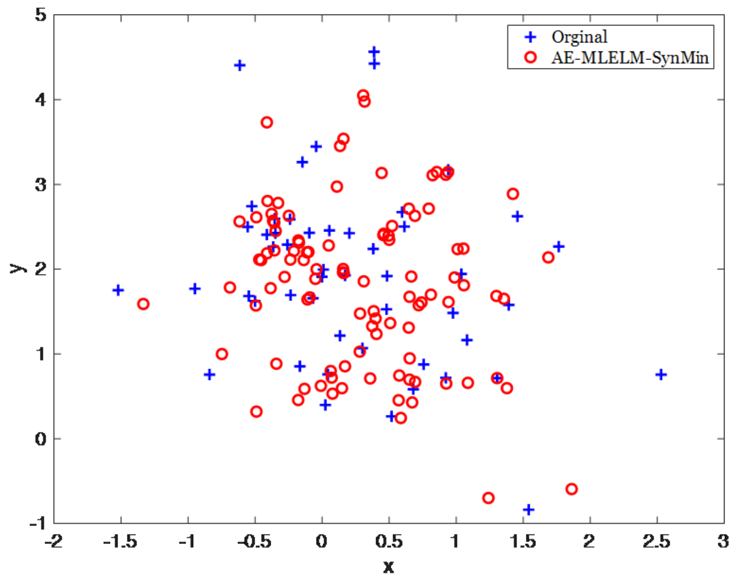
In this paper, we propose a synthetic minority class technology based on auto-encoder multi-layer extreme learning machine (AE-MLELM), abbreviated as AE-MLELM-SynMin. AE-MLELM has a good representation capability. And the intermediate hidden layer is another representation of input layer [15]. The crossover and mutation operations are conducted on the hidden-layer output matrix and then the hidden layer is restored to get new minority class samples. Experimental results show that our method has better performance than SMOTE, Borderline-SMOTE, Random-SMOTE, and SMOTE-IPF. The remainder of this paper is arranged as follows. Section 2 introduces the classic SMOTE method and analyzes its defects. Section 3 presents the proposed AE-MLELM-SynMin method in details. Section 4 shows the experimental results. Section 5 concludes this paper.

2 SMOTE method

SMOTE is a data-driven imbalanced data classification method. It uses the method of synthesizing minority class samples to expand the number of minority class samples until the number of samples in each class is roughly equal. Its basic principle is described below. For each sample $\mathrm{v}_n^{(k)}, n = 1, 2, \cdots, \mathcal{N}_{\min k}$ in the k-th minority class data set $\mathbb{D}_{\min k}$, its $\mathcal{P}$ nearest neighbors are $\mathrm{v}_{n1}^{(k)}, \mathrm{v}_{n2}^{(k)}, \cdots, \mathrm{v}_{n\mathcal{P}}^{(k)}$, a point $\mathrm{v}_{np}^{(k)}$ is randomly selected from these sample points, and the synthesized new minority class sample is denoted as $\overline{\mathrm{v}}^{(k)}$. The new sample point $\bar{v}_d^{(k)}$



(a) Original and SMOTE data



(b) Original and AE-MLELM-SynMin data

Fig. 1. Comparison among Original, SMOTE, and AE-MLELM-SynMin data

is calculated as the following formula.

$$\bar{v}_d^{(k)} = v_{nd}^{(k)} + \lambda \left[v_{npd}^{(k)} - v_{nd}^{(k)} \right], d = 1, 2, \cdots, \mathcal{D}, \tag{1}$$

where $\lambda \in [0, 1]$ is a random number that obeys a uniform distribution. We repeat the above-mentioned process many times until the enough samples are synthesized for the k-th minority class.

The related experiments show that the SMOTE method and its variants have good performance, but the more detailed analysis indicates that these SMOTE-based methods have some inherent defects. SMOTE has the problem of insufficient information amount for the synthesized new samples. In order to explain the shortcoming of SMOTE method, we provide a legend to show the distribution of original and synthesized samples. Fig. 1(a) shows 50 original samples and 100 samples synthesized by SMOTE method. Fig. 1(b) shows 50 original samples and 100 samples synthesized by the AE-MLELM-SynMin method. It can be seen from the figure that the new samples synthesized by SMOTE are mostly distributed near the original sample points. The new sample points synthesized by AE-MLELM-SynMin method are more scattered and have more information amount. The experiment in Section 4 also confirms this observation.

3 Proposed AE-MLELM-SynMin method

Due to the problems of fixed samples and insufficient information amount generated by classical SMOTE algorithm and its variants, we introduce the synthesis minority class technology based on AE-MLELM, namely AE-MLELM-SynMin.

3.1 Training AE-MLELM

For the k-th minority class, an AE-MLELM, denoted as $\mathrm{AE-MLELM}_k$ is firstly trained. AE-MLELM is a special MLELM with the same input and output. Its input and output expressions are shown as:

$$\mathrm{X}_{\min k} = \begin{bmatrix} v_{11}^{(k)} & v_{12}^{(k)} & \cdots & v_{1\mathcal{D}}^{(k)} \\ v_{21}^{(k)} & v_{22}^{(k)} & \cdots & v_{2\mathcal{D}}^{(k)} \\ \vdots & \vdots & \ddots & \vdots \\ v_{\mathcal{N}_{\min k},1}^{(k)} & v_{\mathcal{N}_{\min k},2}^{(k)} & \cdots & v_{\mathcal{N}_{\min k},\mathcal{D}}^{(k)} \end{bmatrix}. \tag{2}$$

For MLELM, each of its hidden layers is an AE-MLELM with the same input and output. We can then calculate the hidden-layer output matrix for each layer. As for the calculation of the first hidden layer output matrix $\mathbf{H}^1$, since its input and output matrices are both $\mathrm{X}_{\min k}$, the first hidden layer output matrix $\mathbf{H}^1$ is obtained by the single hidden layer extreme learning machine with both input and output nodes, then the calculation formula of $\mathbf{H}^1$ is

$$\mathbf{H}^1 = g\left(\mathrm{X}\left(\boldsymbol{\beta}^1\right)^T\right), \tag{3}$$

where $\boldsymbol{\beta}^1$ is the weight of the output layer of the first layer and $g(*)$ is the activation function.

Similarly, for the i-th$(i = 1, 2, \cdots, n)$ hidden layer, its input and output matrices are the output matrix of the hidden layer of the upper layer $\mathbf{H}^{i-1}$, and the output matrix of the hidden layer is $\mathbf{H}^i$. The calculation formula of the output matrix of the hidden layer $\mathbf{H}^i$ is

$$\mathbf{H}^i = g\left(\mathbf{H}^{i-1}\left(\boldsymbol{\beta}^i\right)^T\right), \tag{4}$$

where $\boldsymbol{\beta}^{\mathrm{i}}$ is the output-layer weight. And we derive the calculation formula of the output matrix of the last hidden layer as

$$\mathbf{H}^l = g\left(\mathbf{H}^{l-1}\left(\boldsymbol{\beta}^l\right)^T\right). \tag{5}$$

Then we calculate the output layer weight of the last hidden layer by the following formula

$$\boldsymbol{\beta}^{l+1} = \left(\mathbf{H}^l\right)^\dagger \mathbf{X}, \tag{6}$$

where $\left(\mathbf{H}^l\right)^\dagger$ is the Moore-Penrose inverse [16] of $\mathbf{H}^l$. $\mathrm{AE-MLELM}_k$ uses $\boldsymbol{\beta}^{l+1}$ to decode $\mathrm{X}_{\min k}$.

3.2 Conducting crossover and mutation operations

After the training of $\mathrm{AE-MLELM}_k$ is finished, the crossover and mutation operations for the last hidden-layer output matrix $\mathrm{H}^l_{\min k}$ are conducted and the changed hidden-layer output matrix $\overline{\mathrm{H}}^l_{\min k}$ is expressed as

$$\overline{\mathrm{H}}^l_{\min k} = \begin{bmatrix} \bar{h}^{(k)}_{11} & \bar{h}^{(k)}_{12} & \cdots & \bar{h}^{(k)}_{1\mathcal{L}} \\ \bar{h}^{(k)}_{21} & \bar{h}^{(k)}_{22} & \cdots & \bar{h}^{(k)}_{2\mathcal{L}} \\ \vdots & \vdots & \ddots & \vdots \\ \bar{h}^{(k)}_{\mathcal{N}_{\min k},1} & \bar{h}^{(k)}_{\mathcal{N}_{\min k},2} & \cdots & \bar{h}^{(k)}_{\mathcal{N}_{\min k},\mathcal{L}} \end{bmatrix}. \tag{7}$$

- We randomly select two lines $h_m^{(k)}$ and $h_n^{(k)}$ from $\mathrm{H}_{\min k}^{l}$ to perform the crossover operation. The crossover rule is shown in the following formula

$$\begin{cases} \mathrm{h}_m^{(k)} = \left(h_{m1}^{(k)}, \cdots, h_{ml'}^{(k)}, h_{m,l'+1}^{(k)}, \cdots, h_{m\mathcal{L}}^{(k)}\right) \\ \mathrm{h}_n^{(k)} = \left(h_{n1}^{(k)}, \cdots, h_{nl'}^{(k)}, h_{n,l'+1}^{(k)}, \cdots, h_{n\mathcal{L}}^{(k)}\right) \end{cases} \xrightarrow{\text{Crossover}}$$
$$\left(\begin{cases} \overline{\mathrm{h}}_i^{(k)} = (\underbrace{\bar{h}_{i1}^{(k)}, \cdots, \bar{h}_{il'}^{(k)}}_{h_{m1}^{(k)},\cdots,h_{ml'}^{(k)}}, \underbrace{\bar{h}_{i,l'+1}^{(k)}, \cdots, \bar{h}_{i\mathcal{L}}^{(k)}}_{h_{n,l'+1}^{(k)},\cdots,h_{n\mathcal{L}}^{(k)}}) \\ \overline{\mathrm{h}}_j^{(k)} = (\underbrace{\bar{h}_{j1}^{(k)}, \cdots, \bar{h}_{jl'}^{(k)}}_{h_{n1}^{(k)},\cdots,h_{nl'}^{(k)}}, \underbrace{\bar{h}_{j,l'+1}^{(k)}, \cdots, \bar{h}_{j\mathcal{L}}^{(k)}}_{h_{m,l'+1}^{(k)},\cdots,h_{m\mathcal{L}}^{(k)}}) \end{cases}\right), \tag{8}$$

  where $\mathrm{h}_m^{(k)}, \mathrm{h}_n^{(k)} \in \mathrm{H}_{\min k}$, and $\overline{\mathrm{h}}_i^{(k)}, \overline{\mathrm{h}}_j^{(k)} \in \overline{\mathrm{H}}_{\min k}$, and l' is the crossover position. In this paper, l' can be set in the intervals $[0.1\mathcal{L}, 0.4\mathcal{L}]$, $[0.4\mathcal{L}, 0.6\mathcal{L}]$ and $[0.6\mathcal{L}, 0.9\mathcal{L}]$.
- For the mutation operation, the mutation location l' is randomly selected for $\forall h_m^{(k)} \in \mathrm{H}_{\min k}^{l}$. The mutation operation is conducted as the following formula:

$$\mathrm{h}_m^{(k)} = \left(h_{m1}^{(k)}, \cdots, h_{ml'}^{(k)}, h_{m,l'+1}^{(k)}, \cdots, h_{m\mathcal{L}}^{(k)}\right) \xrightarrow{\text{Mutation}}$$
$$\bar{h}_i^{(k)} = \begin{cases} (\underbrace{\bar{h}_{il'}^{(k)}}_{1-h_{ml'}^{(k)}}, \underbrace{\bar{h}_{i,l'+1}^{(k)}, \cdots, \bar{h}_{i\mathcal{L}}^{(k)}}_{h_{m,l'+1}^{(k)},\cdots,h_{m\mathcal{L}}^{(k)}}), \\ \qquad l' = 1 \\ (\underbrace{\bar{h}_{i1}^{(k)}, \cdots, \bar{h}_{il'-1}^{(k)}}_{h_{m1}^{(k)},\cdots,h_{m,l'-1}^{(k)}}, \underbrace{\bar{h}_{il'}^{(k)}}_{1-h_{ml'}^{(k)}}, \underbrace{\bar{h}_{i,l'+1}^{(k)}, \cdots, \bar{h}_{i\mathcal{L}}^{(k)}}_{h_{m,l'+1}^{(k)},\cdots,h_{m\mathcal{L}}^{(k)}}), \\ \qquad l' \in (1, \mathcal{L}) \\ (\underbrace{\bar{h}_{i1}^{(k)}, \cdots, \bar{h}_{il'-1}^{(k)}}_{h_{m1}^{(k)},\cdots,h_{m,l'-1}^{(k)}}, \underbrace{\bar{h}_{il'}^{(k)}}_{1-h_{ml'}^{(k)}}), \\ \qquad l' = \mathcal{L} \end{cases}, \tag{9}$$

  where $\overline{\mathrm{h}}_i^{(k)} \in \overline{\mathrm{H}}_{\min k}$, and $\bar{h}_{il'}^{(k)} = 1 - h_{ml'}^{(k)} \in (0, 1)$.

Algorithm 1 AE-MLELM-SynMin algorithm

```
Input  The original minority class data sets $X_{\min 1}, X_{\min 2}, \cdots, X_{\min \mathcal{K}}$;
Output  The synthetic minority class data sets $\overline{X}_{\min 1}, \overline{X}_{\min 2}, \cdots, \overline{X}_{\min \mathcal{K}}$;
 1: for $k = 1; k <= \mathcal{K}; k++$ do
 2:   Training AE-MLELM$_k$:
 3:   Calculating $H^i$ for each layer according to Eq. (5);
 4:   Calculating the output layer weights matrix $\beta^{l+1}_{\min k}$;
 5:   Conducting crossover and mutation operations based on the last hidden-layer matrix $H^l_{\min k}$;
 6:   while The number of synthetic hidden-layer output vectors does not reach $\overline{\mathcal{N}}_{\min k}$ do
 7:     Crossover operation as shown in Eq. (8);
 8:     Mutation operation as shown in Eq. (9);
 9:   end while
10:   Creating synthetic data set $\overline{X}_{\min k}$ based on synthetic hidden-layer output matrix $\overline{H}^l_{\min k}$
      according to Eq. (10);
11: end for
```

3.3 Creating synthetic samples

Finally, the minority class samples are synthesized. After obtaining the hidden layer output matrix of the mutation, the generation formula of synthesized minority class samples is

$$\overline{X}_{\min k} = \overline{H}^l_{\min k} \beta^{l+1}_{\min k}, \tag{10}$$

where $\beta^{l+1}_{\min k}$ and $\overline{H}^l_{\min k}$ are obtained from the first and second steps of the algorithm. Algorithm 1 gives the pseudo-code of AE-MLELM-SynMin. For each of minority classes, the above-mentioned steps are repeated until the sufficient samples are synthesized.

4 Experiments

In this section, we mainly use two experiments to verify the feasibility and effectiveness of the AE-MLELM-SynMin algorithm. These two experiments involve the comparative analysis of information amounts between SMOTE and AE-MLELM-SynMin data and the comparison among SMOTE [7], Borderline-SMOTE [8], Random-SMOTE [9] and SMOTE-IPF [10] algorithms based on decision tree classifier.

4.1 Experiment setting

AE-MLELM-SynMin is implemented by the Python programming language and SMOTE, Borderline-SMOTE, Random-SMOTE and SMOTE-IPF algorithms are downloaded from the Python Package Index (PyPI) [1], which is a

[1] https://pypi.org/

Table 1. Details of 10 binary classification data sets

Data set	Attribute	Sample	Majority class	Minority class	Imbalance ratio
ecoli1	7	336	259	77	3.36
ecoli2	7	336	284	52	5.46
yeast1	8	1484	1055	429	2.46
yeast5	8	1484	1440	44	32.73
glass0	9	214	144	70	2.06
glass4	9	214	201	13	15.47
vowel0	13	988	898	90	9.98
segment0	19	2308	1979	329	6.02
vehicle0	18	846	647	199	3.25
new-thyroid1	5	215	180	35	5.14

Table 2. Information amount comparison among original data, SMOTE data, and AE-MLELM-SynMin data

Data set size	1000	2000	3000	4000	5000	6000	7000	8000	9000	10000
Original data	1.765	1.772	1.756	1.762	1.760	1.762	1.764	1.764	1.767	1.760
SMOTE data	1.763	1.771	1.754	1.761	1.759	1.761	1.763	1.764	1.767	1.760
AE-MLELM-SynMin data	2.783	2.485	2.381	2.285	2.427	2.293	2.270	2.303	2.228	2.270

software library for the Python programming language. In the experiments, the number of nearest neighbors are all set to 5 for these SMOTE algorithm variants. The number of iterations and percentage of samples removed for SMOTE-IPF are set at 3 and 1%, respectively.

For AE-MLELM-SynMin, we set (1) the crossover-mutation factor $\xi = 2$, that is, the ratio of crossover and mutation in the synthesized samples is 2:1; (2) The position intervals of crossover operation are $[0.4\mathcal{L}, 0.6\mathcal{L}]$, $[0.1\mathcal{L}, 0.4\mathcal{L}]$ and $[0.6\mathcal{L}, 0.9\mathcal{L}]$; and (3) the number of positions to be changed in a mutation operation is generated randomly on the interval $[0.001\mathcal{L}, 0.1\mathcal{L}]$ and the interval of mutation point is $[1, \mathcal{L}]$, where $\mathcal{L}$ is the number of hidden-layer nodes of AE-MLELM. We get the real data sets from KEEL-dataset repository [17] including 10 binary classification data sets in Table 1.

For the Settings of hidden layers and node number of AE-MLELM, we set the depth as 5 and node number of each layer as 1000. In this experiment, we choose the decision tree algorithm as the classifier. The reason is that decision tree algorithm generally works well on class-imbalanced data.

4.2 Information amount analysis of SMOTE and AE-MLELM-SynMin

In order to explain the effectiveness of AE-MLELM-SynMin method, we use the resubstitution entropy to measure the information amount of dataset [18]. In the experiment, We randomly create 10 different original data sets with sizes ranging from 1000 to 10000 in step of 1000 and synthesize the corresponding 10 SMOTE data sets and 10 AE-MLELM-SynMin data sets, respectively. Then, we calculate the information amounts of the original data, SMOTE data and AE-MLELM-SynMin data. We repeat the synthetic sample 10 times for each original data set and calculate the average value of the information amounts. For a given data set

$$\begin{aligned}\mathbb{A} = \{\mathrm{a}_n \mid \mathrm{a}_n = \left(a_{n1}, a_{n2}, \cdots, a_{n\mathcal{D}}\right), a_{md} \in \Re, \\ n = 1, 2, \cdots, \mathcal{N}, d = 1, 2, \cdots, \mathcal{D}\}\text{'}\end{aligned} \tag{11}$$

its entropy is calculated as

$$\operatorname{Info}(\mathbb{A}) = \frac{1}{\mathcal{D}} \sum_{d=1}^{\mathcal{D}} \operatorname{Ent}\left(\mathrm{A}_d\right) \tag{12}$$

where A_d is the d-th attribute of data set $\mathbb{A}$. The re-substitution entropy of data set $\{a_{1d}, a_{2d}, \cdots, a_{\mathcal{N}d}\}$ is calculated as

$$\operatorname{Ent}\left(\mathrm{A}_d\right) = -\sum_{n=1}^{\mathcal{N}} \operatorname{In}\left[\hat{p}_{-n}\left(a_{nd}\right)\right], \tag{13}$$

where

$$\hat{p}_{-n}\left(a_{nd}\right) = \frac{1}{\mathcal{N}-1} \sum_{\substack{m=1 \\ m \neq n}}^{\mathcal{N}} \frac{1}{\sqrt{2\pi} h} \exp\left[-\frac{1}{2}\left(\frac{a_{nd}-a_{md}}{h}\right)^2\right] \tag{14}$$

is the leave-one-out cross-validation kernel density estimator, $h > 0$ is the bandwidth parameter. And we set h as 0.2 in this experiment.

Table 2 shows the information amount comparison results of original data, SMOTE data and AE-MLELM-SynMin data. In table 2, we can draw the

Table 3. Comparative results for binary classification problems based on decision tree classifier

Algorithm / Data Set	**AE-MLELM-SynMin(2021)**				**SMOTE(2002)**			
	Auc	G-mean	F1	Accuracy	Auc	G-mean	F1	Accuracy
ecoli1	0.881±0.037	0.879±0.039	0.768±0.050	0.877±0.031	0.884±0.042	0.883±0.044	0.773±0.058	0.880±0.033
ecoli2	**0.880±0.047**	**0.875±0.052**	**0.780±0.067**	**0.928±0.023**	:0.870±0.051	0.865±0.058	0.753±0.075	0.916±0.029
yeast1	0.687±0.027	0.676±0.039	0.557±0.036	0.708±0.052	**0.698±0.020**	**0.694±0.021**	**0.571±0.026**	**0.728±0.022**
yeast5	**0.959±0.029**	**0.958±0.030**	0.570±0.071	0.956±0.013	0.948±0.046	0.946±0.050	0.695±0.068	0.976±0.008
glass0	**0.796±0.050**	**0.786±0.058**	**0.707±0.063**	0.762±0.049	0.783±0.053	0.779±0.053	0.696±0.061	0.766±0.050
glass4	**0.861±0.103**	**0.843±0.143**	0.618±0.159	0.939±0.032	0.831±0.112	0.804±0.158	0.620±0.172	0.948±0.025
vowel0	**0.966±0.017**	**0.966±0.017**	**0.837±0.048**	**0.965±0.013**	0.947±0.025	0.946±0.026	0.813±0.051	0.960±0.013
segment0	**0.987±0.009**	**0.987±0.009**	0.970±0.013	0.991±0.004	0.986±0.008	0.986±0.008	0.969±0.012	**0.991±0.003**
vehicle0	0.915±0.030	0.913±0.031	0.806±0.047	0.889±0.036	0.921±0.023	0.921±0.023	0.850±0.032	0.923±0.017
new-thyroid1	**0.966±0.033**	**0.966±0.035**	0.914±0.060	0.969±0.023	0.946±0.047	0.944±0.050	0.908±0.068	0.969±0.024

Algorithm / Data Set	**Borderline-SMOTE(2005)**				**Random-SMOTE(2011)**			
	Auc	G-mean	F1	Accuracy	Auc	G-mean	F1	Accuracy
ecoli1	**0.893±0.035**	**0.891±0.035**	**0.779±0.053**	0.88±0.034	0.883±0.041	0.881±0.042	0.771±0.053	0.879±0.030
ecoli2	0.852±0.053	0.844±0.061	0.740±0.080	0.915±0.030	0.875±0.05	0.871±0.054	0.753±0.073	0.914±0.031
yeast1	0.697±0.022	0.692±0.025	0.571±0.027	0.723±0.036	**0.698±0.020**	0.693±0.021	**0.571±0.026**	0.726±0.021
yeast5	0.949±0.049	0.947±0.055	0.683±0.076	0.975±0.008	0.945±0.044	0.943±0.047	**0.704±0.069**	**0.977±0.007**
glass0	0.761±0.051	0.756±0.052	0.670±0.060	0.746±0.048	0.786±0.049	0.782±0.050	0.700±0.057	**0.768±0.049**
glass4	0.851±0.134	0.815±0.211	**0.667±0.227**	**0.957±0.027**	0.831±0.114	0.801±0.173	0.618±0.186	0.947±0.026
vowel0	0.948±0.029	0.948±0.030	0.813±0.055	0.96±0.014	0.941±0.030	0.940±0.031	0.801±0.052	0.958±0.013
segment0	0.956±0.019	0.955±0.020	0.929±0.027	0.980±0.008	**0.986±0.007**	**0.986±0.007**	0.969±0.012	0.991±0.004
vehicle0	**0.924±0.022**	**0.924±0.023**	0.857±0.031	0.927±0.017	0.921±0.022	0.92±0.023	**0.859±0.03**	**0.93±0.015**
new-thyroid1	0.952±0.052	0.950±0.057	0.921±0.072	0.974±0.023	0.950±0.045	0.948±0.049	**0.921±0.061**	**0.974±0.020**

Algorithm / Data Set	**SMOTE-IPF(2015)**				/			
	Auc	G-mean	F1	Accuracy	/	/	/	/
ecoli1	0.886±0.040	0.884±0.041	0.778±0.058	**0.884±0.033**	/	/	/	/
ecoli2	0.865±0.050	0.860±0.055	0.746±0.077	0.913±0.031	/	/	/	/
yeast1	0.698±0.021	0.693±0.023	0.571±0.027	0.727±0.023	/	/	/	/
yeast5	0.946±0.041	0.944±0.044	0.700±0.068	**0.977±0.007**	/	/	/	/
glass0	0.782±0.047	0.778±0.048	0.695±0.055	0.765±0.046	/	/	/	/
glass4	0.834±0.120	0.802±0.187	0.614±0.182	0.947±0.025	/	/	/	/
vowel0	0.946±0.027	0.946±0.029	0.813±0.050	0.960±0.013	/	/	/	/
segment0	0.986±0.008	0.986±0.008	**0.970±0.012**	0.991±0.004	/	/	/	/
vehicle0	0.919±0.023	0.918±0.024	0.856±0.029	0.928±0.015	/	/	/	/
new-thyroid1	0.952±0.046	0.950±0.049	0.921±0.063	**0.974±0.020**	/	/	/	/

following conclusions. First, AE-MLELM-SynMin data has the highest information amount. Second, original data and SMOTE data have the same information amount, and AE-MLELM-SynMin data has 1.2 to 1.6 times more information amount than original data or SMOTE data. So the data synthesized by AE-MLELM-SynMin has a large increase in the amount of information, and finally can achieve a better effect on the general classifier.

4.3 Comparison among AE-MLELM-SynMin, SMOTE, Borderline-SMOTE, Random-SMOTE, and SMOTE–IPF

In this experiment, we use the decision tree classifier downloaded from the scikit-learn Python machine learning library [2] to classify the datasets created by AE-MLELM-SynMin, SMOTE, Borderline-SMOTE, Random-SMOTE and SMOTE-IPF, and compare their imbalanced classification performances. Table 1 shows the 10 binary data sets used in this experiment. The specific process of this experiment is that we first randomly divide each original data set into 70% training set and 30% testing set and then fill the minority samples created by AE-MLELM-SynMin and SMOTE variants for the training set. Finally we use the trained classifier to predict the testing set and evaluate the experimental results with the four indicators of AUC [19], G-mean [20], F1 [21] and accuracy. The above process is repeated 400 times and take the average of 400 AUCs, G-means, F1s and accuracies as the final experimental result.

The classification results of the decision tree are shown in Table 3, which lists the AUC, G-mean, F1 and accuracy of decision tree classifiers under AE-MLELM-SynMin, SMOTE, Borderline-SMOTE, Random-SMOTE and SMOTE-IPF methods. We can draw several conclusions from the comparison of bold markers in Table 3. Firstly, AE-MLELM-SynMin method has the better AUCs and G-means compared with other SMOTE-based methods on 7 data sets. Secondly, the AE-MLELM-SynMin method achieves the better F1s on 3 data sets. In general, the performance of AE-MLELM-SynMin method is better than other SMOTE-based methods when dealing with the imbalanced classification problems.

5 Conclusion

The proposed AE-MLELM-SynMin method was constructed based on AE-MLELM, which introduced the ideas of crossover and mutation in evolutionary algorithm. This method had the advantages of easy understanding and simple implementation. The experimental results showed that, compared with other SMOTE-based methods, the AE-MLELM-SynMin method can improve

[2] https://scikit-learn.org/stable/modules/tree.html

the information amount of synthesized samples and thus obtained the better imbalanced classification performance than SMOTE-based methods.

Acknowledgement

The authors would like to thank the chairs and anonymous reviewers whose meticulous readings and valuable suggestions help them to improve this paper significantly. This paper was supported by National Natural Science Foundation of China (61972261) and Basic Research Foundation of Shenzhen (JCYJ 20210324093609026, JCYJ 20200813091134001), and Scientific Research Foundation of Shenzhen University for Newly-introduced Teachers (860/000002110628).

References

1. Japkowicz N., Stephen S.: The class imbalance problem: A systematic study. Intelligent Data Analysis 6(5): 429–449 (2002)
2. Díez-Pastor J. F., Rodríguez J. J., García-Osorio C., Kuncheva L. I.: Random Balance: Ensembles of variable priors classifiers for imbalanced data. Knowledge-Based Systems 85: 96–111 (2015)
3. Liu X., Wu J., Zhou Z.: Exploratory Undersampling for Class-Imbalance Learning. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 39(2): 539–550 (2009)
4. Sun Y., Kamel M. S., Wong A. K.C., Wang Y.: Cost-sensitive boosting for classification of imbalanced data. Pattern Recognition 40(12): 3358-3378 (2007)
5. Tan S.: Neighbor-weighted K-nearest neighbor for unbalanced text corpus. Expert Systems with Applications 28(4): 667–671 (2005)
6. Zong W. W. and Huang G. B. and Chen Y. Q.: Weighted extreme learning machine for imbalance learning. Neurocomputing 101: 229–242 (2013)
7. Chawla N. V., Bowyer K. W., Hall L. O., Kegelmeyer W. P.: SMOTE: Synthetic Minority over-Sampling Technique. Journal of Artificial Intelligence Research 16(1): 321–357 (2002)
8. Han H., Wang W. Y., Mao B. H.: Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning. Lecture Notes in Computer Science 3644: 878–887 (2005)
9. Dong Y. J., Wang X. H.: A New Over-Sampling Approach: Random-SMOTE for Learning from Imbalanced Data Sets. In Proceedings of the 5th International Conference on Knowledge Science, Engineering and Management 10: 343–352 (2011)
10. Sáez J. A., Luengo J., Stefanowski J., Herrera F.: SMOTE–IPF: Addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering. Information Sciences 291: 184–203 (2015)
11. Calleja J. L., Fuentes O.: A Distance-Based Over-Sampling Method for Learning from Imbalanced Data Sets. In Proceedings of the Twentieth International Florida Artificial Intelligence Research Society Conference (2007)
12. Puntumapon, Kamthorn, Waiyamai, Kitsana: A pruning-based approach for searching precise and generalized region for synthetic minority over-sampling. Advances in Knowledge Discovery and Data Mining 371–382 (2012)
13. Lee H., Kim J., Kim S.: Gaussian-Based SMOTE Algorithm for Solving Skewed Class Distributions. International Journal of Fuzzy Logic and Intelligent Systems 17: 229–234 (2017)
14. Douzas G., Bacao F., Last F.: Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE. Information Sciences 465: 1–20 (2018)
15. Kasun L., Zhou H. M., Huang G. B., Vong C. M.: Representational Learning with ELMs for Big Data. IEEE Intelligent Systems 28: 31–34 (2013)

16. Lu. S. X., Wang X., Zhang G. Q., Zhou X.: Effective algorithms of the Moore-Penrose inverse matrices for extreme learning machine.Intelligent Data Analysis 19: 743–760 (2015)
17. Alcala-Fdez J., Fernández A., Luengo J., Derrac J., Garc'ia S., Sanchez L., Herrera, F.: KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework. Journal of Multiple-Valued Logic and Soft Computing 17: 255–287 (2010)
18. He Y. L., Liu J. N.K., Wang X. Z., Hu Y. X.: Optimal bandwidth selection for re-substitution entropy estimation. Applied Mathematics and Computation 219(8): 3425–3460 (2012)
19. Hand D. J., Till R. J.: A Simple Generalisation of the Area Under the ROC Curve for Multiple Class Classification Problems. Machine Learning 45(2): 171–186 (2001)
20. Sun Y., Kamel M. S., Wang Y.: Boosting for Learning Multiple Classes with Imbalanced Class Distribution. In Proceedings of the Sixth International Conference on Data Mining 592–602 (2006)
21. Lipton Z. C., Elkan C., Naryanaswamy B.: Optimal Thresholding of Classifiers to Maximize F1 Measure. In Proceedings of Machine Learning and Knowledge Discovery in Databases 225–239 (2014)