Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

A Synthetic Minority Based on Probabilistic Distribution (SyMProD) Oversampling for Imbalanced Datasets

INTOUCH KUNAKORNTUM[1], WORANICH HINTHONG[2], AND PHOND PHUNCHONGHARN[1]

[1]Computer Engineering Department, King Mongkut's University of Technology Thonburi, Bangkok, Thailand

[2]Faculty of Medicine and Public Health, HRH Princess Chulabhorn College of Medical Science, Chulabhorn Royal Academy, Bangkok, Thailand

Corresponding author: P. Phunchongharn (e-mail: phond.p@mail.kmutt.ac.th).

This research was supported in part by King Mongkut's University of Technology Thonburi through Petchra Pra Jom Klao scholarship program for a master's degree, in part by the Big Data Experience Center , Thailand, and in part by the Faculty of Medicine and Public Health, HRH Princess Chulabhorn College of Medical Science, Chulabhorn Royal Academy, Thailand.

ABSTRACT Handling an imbalanced class problem is a challenging task in real-world applications. This problem affects various prediction models that predict only the majority class and fail to identify the minority class because of the skewed data. The oversampling technique is one of the exciting solutions that handles the imbalanced class problem. However, several existing oversampling methods do not consider the distribution of the target variable and cause an overlapping class problem. Therefore, this study introduces a new oversampling technique, namely Synthetic Minority based on Probabilistic Distribution (SyMProD), to handle skewed datasets. Our technique normalizes data using a Z-score and removes noisy data. Then, the proposed method selects minority samples based on the probability distribution of both classes. The synthetic instances are generated from selected points and several minority nearest neighbors. Our technique aims to create synthetic instances that cover the minority class distribution, avoid the noise generation, and reduce the possibilities of overlapping classes and overgeneralization problems. Our proposed technique is validated using 14 benchmark datasets and three classifiers. Moreover, we compare the performance with seven other conventional oversampling algorithms. The empirical results show that our method achieves better performance compared with other oversampling techniques.

INDEX TERMS Classification, imbalanced dataset, machine learning, oversampling, probabilistic distribution.

I. INTRODUCTION

Nowadays, several researchers and practitioners have implemented prediction models in their applications. In a real-world situation, some datasets contain a skew distribution, which causes an imbalanced dataset problem and decreases model performance. Several applications suffer from the imbalanced dataset problem, e.g., fraud detection [1], [2], churn prediction [3], and medical prediction [4], [5]. The imbalanced problem occurs when the number of instances belongs to one class and is significantly higher than the other classes. A class having the most observation is called the majority class, whereas the other is called the minority class. In this paper, we apply these terms to refer to a group in a dataset.

The imbalanced class problem affects classification models because the objective function is to maximize accuracy [6]. If a dataset has a much different number of instances in each class, the prediction model will predict only the majority class and obtain high accuracy. However, the model cannot classify any minority class instances. This situation represents a low recall metric, and the model cannot be adapted to real-world productions. Therefore, dealing with an imbalanced class dataset is challenging in general [7], [8].

There are three main methods to solve the imbalanced class problem, which are: 1) algorithm level; 2) cost-sensitive; and 3) data level methods [9]. Firstly, the algorithm level method modifies classifiers and changes a decision threshold to handle the imbalanced class. However, the change of decision threshold causes a trade-off between precision and recall metric. This technique could not improve the Area Under the Curve (AUC) metric. Secondly, the cost-sensitive method considers the higher cost for the misclas-

TABLE 1. Description of datasets

Dataset	Minority Class	Majority Class	# of variables	# of records	# of minority	# of majority	IR
Balance	Class "B"	All other	4	625	49	576	11.76
Ecoli	Class "pp"	All other	7	336	52	284	5.46
Glass	Class "1"	All other	9	214	70	144	2.06
Haberman	Class "2"	Class "1"	3	306	81	225	2.78
Heart	Class "0"	Class "1"	13	303	138	165	1.20
Ionosphere	Class "b"	Class "g"	34	351	126	225	1.79
Libra	Class "1", "2", and "3"	All other	90	360	72	288	4.00
Pima	Class "1"	Class "0"	8	768	268	500	1.87
Segment	Class "WINDOW"	All other	19	2310	330	1980	6.00
Vehicle	Class "van"	All other	18	846	199	647	3.25
Simulated1	Class "1"	Class "0"	2	1000	205	795	3.87
Simulated2	Class "1"	Class "0"	2	1000	304	696	2.28
Simulated3	Class "1"	Class "0"	10	1000	203	797	3.92
Simulated4	Class "1"	Class "0"	10	1000	300	700	2.33

sification. This method requires the domain knowledge of a dataset, which is hard to estimate an appropriate value [9], [10]. Lastly, the data level approach balances the number of proportions between majority and minority classes. Any algorithm can be trained with data after balancing, which is more versatile than other methods [11]. The techniques in this method are undersampling and oversampling. The undersampling removes the majority class instances, whereas the oversampling creates minority class samples. Both approaches aim to balance the class of the dataset. In this paper, we focus on the data level method with an oversampling technique because the undersampling can eliminate insight data. Moreover, if the size of the minority class is small, which is the main problem of imbalanced class, the technique will remove most of the majority class instances that will affect the prediction performance.

This paper presents a novel oversampling technique called Synthetic Minority based on Probabilistic Distribution (SyMProD) Oversampling. Our approach generates minority class instances using three major steps. Firstly, SyMProD reduces the possibility of noise creation by applying Z-score normalization and excluding the noisy data based on the Z-score value. Secondly, the method determines a probability distribution and assigns the probability to each minority instance. Then, the referenced data are selected from the minority class instances based on the probability distribution to generate new data points. We aim to eliminate the overgeneralization problem by assigning the different probability and solve the overlapping class problem by considering the majority and minority class distribution. In the last step, SyMProD generates synthetic instances from minority referenced data and multiple neighbors instead of creating along a line between two points.

We focus on the binary class problem and evaluate our approach using ten public UCI datasets and four scikit-learn simulated datasets. The proposed method is compared with the seven other oversampling techniques. We employ three classifiers to train the prediction model and measure the Area Under the Curve (AUC), F-Measure, and G-Mean as performance metrics.

The remainder of this paper is organized as follows: Section II summarizes related works of existing data level algorithms to handle the imbalanced class problem. Section III explains the detail of our proposed algorithm, its characteristics, and evaluation methods. Section IV shows analytical results, the performance of SyMProD, and discussion. Last, we conclude the paper in Section V.

II. LITERATURE REVIEW

Several researchers have shown that the imbalanced class dataset can decrease the prediction model performance [12], [13]. Multiple techniques in the data level intended to handle data distribution before creating a prediction model to solve the problem [7]. The resampling technique rebalanced data and changed the class distribution, which included random undersampling and random oversampling. Random undersampling removed majority class instances until the proportion of majority class was equal to the minority class. However, data for training the prediction model might not be enough because the removal of samples may eliminate valuable insight. The random oversampling method duplicated minority class instances until artificial data reached the desired size. Nevertheless, this technique had a main drawback to the overfitting problem [14]. The classifier only learned on the specific decision region of replicated data [15].

To overcome this problem, Chawla *et al.* [16] proposed the oversampling technique called Synthetic Minority Oversampling Techniques (SMOTE). The k nearest neighbors were searched based on the feature space similarities, and one neighbor was selected in each minority data point. SMOTE created the synthetic instances along the line between point and its neighbor to balance the dataset, where the position was randomly chosen between zero and one.

Compared to the random oversampling, SMOTE solved the overfitting problem and expanded the classifiers' decision region because this technique did not apply the replication method [17]. However, SMOTE assigned equal weight to every instances that caused an overgeneralization problem [7]. Moreover, the method could generate noise data, which decreased model performance, and may lead to an overlapping problem [18]. The new instances also did not provide the distribution of data because new samples were created in the line between data points [15].

Several modification methods have been adapted from SMOTE and proposed to handle the imbalanced problem. Han *et al.* [17] presented the Borderline-SMOTE technique to create synthetic instances on the border of the decision region based on the proportion of neighbor classes. The authors claimed that this method removed the noise creation problem because they did not select the minority points if all nearest neighbors were the majority class. However, there were some cases where borderline-SMOTE incorrectly detected the decision boundary [19]. On the contrary, Bunkhumpornpat *et al.* [20] proposed the Safe-level SMOTE to generate new points that were farther away from the borderline. This technique could solve the overlapping problem because new instances were created in a safe area based on the number of minority neighbors.

Cluster-SMOTE [21] utilized the clustering to group the minority class and created synthetic instances within the cluster. However, Douza *et al.* [9] reported that this method did not propose the number of synthetic samples in each group and the optimal number of clustering groups. Then, they developed the k-means SMOTE. The method clustered both classes and only generated in the safe area to avoid the overlapping class. The number of samples in each group was calculated to improve the performance from the Cluster-SMOTE algorithm. Santos *et al.* [4] proposed the clustering technique to group data and balance the class in each group. After that, they sampled the data in each group and merged them to create a prediction model. However, this technique may not solve the problem because the data used for training the model may not include the original data, which were the practical insight, and the sampled data may contain an imbalanced dataset. Another approach in the clustering technique, Nekooeimehr and Lai-Yuen [18] presented Adaptive Semi-Unsupervised Weighted Oversampling (A-SUWO). They applied the clustering technique to group the minority class, determined the number of synthetic samples, and assigned weights to generate samples. This method avoided the overlapping problem and handled the sub-clusters of the dataset.

He *et al.* [22] presented the Adaptive Synthesis (ADASYN) sampling approach that applied the concept of weighting minority class instances based on the ratio of majority neighbors. This method solved an overgeneralization problem, which SMOTE faced, but it still contained the overlapping problem. Some points on the boundary decision may not be generated if all neighbors were minority class. [19]. Moreover, other oversampling techniques were proposed as solutions to the imbalanced problem such as SIMO [11], which applied the Support Vector Machine to detect valuable points, RACOG, which utilized the joint distribution and Gibbs sampling [10], SMOTEBoost [23], and CURE-SMOTE [24].

In summary, most of the methods handled the imbalanced dataset in the oversampling technique by producing new instances to improve the prediction model performance with several designs. For example, they discovered the hard-to-learn region, removed the noise, applied the clustering algorithm, and detected the boundary to rebalance the skewed dataset. However, some methods were sensitive to outliers, caused overlapping problems, and only generated samples along the line of two points that did not reproduce the distribution of original data. [3], [15].

To overcome these problems, we propose a Synthetic Minority based on the Probabilistic Distribution (SyMProD) oversampling technique. Our approach applies the Z-score normalization to remove noisy data. Then, the method assigns a probability to each minority instance and selects data points into the synthesizing process. Synthetic instances are created within the minority class region from several minority instances. The method determines the probability distribution, avoids the overlapping region, removes the overfitting and overgeneralization problem, and probably increases prediction model performance.

III. METHODOLOGY

A. SYNTHETIC MINORITY BASED ON PROBABILISTIC DISTRIBUTION (SyMProD) OVERSAMPLING

In this study, we implemented an oversampling technique called Synthetic Minority based on Probabilistic Distribution (SyMProD) to balance skewed data. Our proposed method applied a concept of the probability distribution to assign a probability to each minority instance. SyMProD consisted of three main steps: i) Noise Removal; ii) Minority Point Selection; and iii) Instance Synthesis. The framework of SyMProD is presented in Algorithm (1). Firstly, the method calculated the difference between a majority and a minority class as the number of generated instances n_{gen} before employing the oversampling technique. The n_{gen} minority instances were created to balance the class.

1) Noise Removal

The first step of our approach was to remove outliers from data because the synthetic instance could decrease the model performance if it was generated from noisy data. This step tried to reduce the chance of noise generation. Our method applied Z-score normalization to both classes from an original dataset $\boldsymbol{I}$ and indicated a noise by comparing it with a noise threshold NT. The method selected a point if the absolute value of normalization was lower than NT, as shown in Fig. 1(b). The non-outlier data were added into a set of noise filtered dataset $\boldsymbol{X}$.

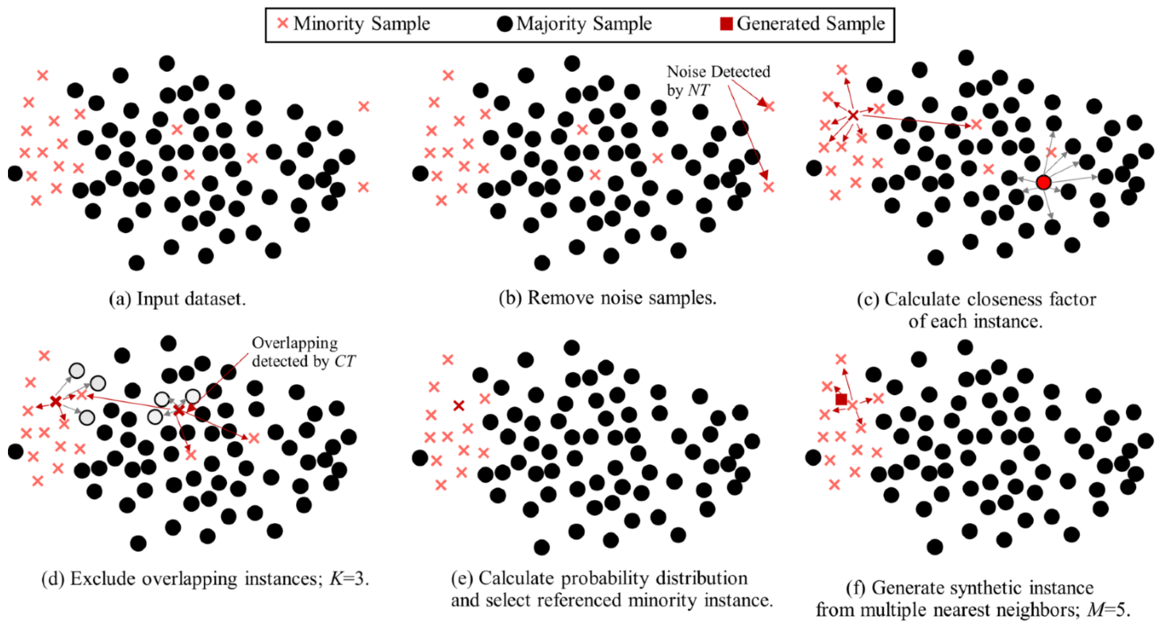

× Minority Sample
● Majority Sample
■ Generated Sample
Noise Detected by NT
Overlapping detected by CT
(a) Input dataset.
(b) Remove noise samples.
(c) Calculate closeness factor of each instance.
(d) Exclude overlapping instances; K=3.
(e) Calculate probability distribution and select referenced minority instance.
(f) Generate synthetic instance from multiple nearest neighbors; M=5.

FIGURE 1. SyMProD generates synthetic instances based on probability distribution and handles noise and overlapping problems.

2) Minority Point Selection

This step selected the referenced minority instances to generate new samples in the next step based on a probability distribution. The number of selected instances was the difference between the number of majority and minority classes from the original dataset, n_{gen}. The method considered the region of both classes to prevent the overlapping problem. In the conventional oversampling technique, SMOTE, minority instances were randomly selected to produce data points along the line that might cause the overlapping problem between two classes.

Firstly, our method took the noise filtered dataset $\boldsymbol{X}$ from Noise Removal step and split into a minority class group $\boldsymbol{X}_{\text{min}}$ and a majority class group $\boldsymbol{X}_{\text{maj}}$ based on a label.

The structure of $\boldsymbol{X}_{\text{min}}$ was $\{\boldsymbol{X}_{\text{min}}(i), \boldsymbol{Y}_{\text{min}}(i)\}$, $i = 1, 2, 3, \ldots, n_{\text{min}}$ where n_{min} was the number of minority instances, $\boldsymbol{X}_{\text{min}}(i)$ was feature spaces, and $\boldsymbol{Y}_{\text{min}}(i)$ was a label. The majority class group $\boldsymbol{X}_{\text{maj}}$ had n_{maj} samples. The structure of $\boldsymbol{X}_{\text{maj}}$ was the same as the $\boldsymbol{X}_{\text{min}}$ structure. The method determined the minority class distribution by applying the Euclidean distance to measure the distance between two points, as in (1). Assume p and q were points with N dimensions.

$$\mathrm{d}(p, q) = \sqrt{\sum_{i=1}^{N} (p(i) - q(i))^2} \tag{1}$$

In each minority instance, $\boldsymbol{X}_{\text{min}}(i)$, $i = 1, 2, 3, \ldots, n_{\text{min}}$, the method calculated the distance from each point to other points in the same class to return the total distance among minority instances as follows:

$$\mathrm{Dist}\left(\boldsymbol{X}_{\text{min}}(i)\right) = \sum_{j=1}^{n_{\text{min}}} \mathrm{d}\left(\boldsymbol{X}_{\text{min}}(i), \boldsymbol{X}_{\text{min}}(j)\right) \tag{2}$$

We defined the closeness factor, C, in each minority instance using (3). The closeness factor assigned a higher value to the point that was close to other points in the same class. Then, we applied the min-max normalization to scale the data. For the majority group, we also assigned the distance and closeness factor among the majority instances to determine the majority distribution. Fig. 1(c) illustrates the sample connection among points. In our algorithm, the closeness factor was calculated from an instance to all points in the same class.

$$\mathrm{C}\left(\boldsymbol{X}_{\text{min}}(i)\right) = \frac{1}{\mathrm{Dist}\left(\boldsymbol{X}_{\text{min}}(i)\right)} \tag{3}$$

The method avoided the overlapping problem by deciding whether or not to choose minority instances based on the minority and majority class distribution. For each minority instance $\boldsymbol{X}_{\text{min}}(i)$, the method found the K nearest neighbors of each class. The K majority and minority class nearest neighbors of each minority instance $\boldsymbol{X}_{\text{min}}(i)$ were collected in a set of $\boldsymbol{S}_{\text{maj}}(i)$ and $\boldsymbol{S}_{\text{min}}(i)$, respectively.

The set of $\boldsymbol{S}_{\text{maj}}(i)$ contained $\{\boldsymbol{S}_{\text{maj}}(i,1), \boldsymbol{S}_{\text{maj}}(i,2), \ldots, \boldsymbol{S}_{\text{maj}}(i,k)\}$, whereas $\boldsymbol{S}_{\text{min}}(i)$ contained $\{\boldsymbol{S}_{\text{min}}(i,1), \boldsymbol{S}_{\text{min}}(i,2), \ldots, \boldsymbol{S}_{\text{min}}(i,k)\}$. Thus, $\boldsymbol{S}_{\text{maj}}(i,1)$ and $\boldsymbol{S}_{\text{min}}(i,1)$ were the first majority class instance and the first minority class instance that were the nearest neighbors of i^{th} minority instance.

To reduce the possibility of overlapping problem, the method defined the minority and majority closeness factor

TABLE 2. Performance evaluations of the oversampling methods using Random Forest algorithm

Data	Metric	SMOTE	BorSMOTE	KSMOTE	SLSMOTE	CluSMOTE	ADASYN	A-SUWO	SyMProD
balance	AUC	0.3754±0.0060	0.3741±0.0049	0.3959±0.0097	**0.4092±0.0045**	0.3585±0.0060	0.3761±0.0042	0.3467±0.0066	**0.4056±0.0070**
	FM	**0.2980±0.0044**	0.2956±0.0050	0.2536±0.0083	0.2437±0.0070	0.2726±0.0054	**0.2963±0.0057**	0.2845±0.0069	0.2234±0.0044
	GM	0.4262±0.0049	0.4249±0.0057	**0.4338±0.0104**	0.4270±0.0057	0.4155±0.0052	**0.4289±0.0047**	0.4235±0.0063	0.4198±0.0070
ecoli	AUC	0.9649±0.0025	0.9631±0.0027	**0.9680±0.0030**	0.9548±0.0032	0.9622±0.0030	0.9541±0.0029	0.9602±0.0024	**0.9669±0.0021**
	FM	0.8048±0.0108	0.8273±0.0113	**0.8431±0.0090**	0.7485±0.0179	0.7975±0.0136	0.7367±0.0147	**0.8410±0.0089**	0.8331±0.0124
	GM	0.9233±0.0045	0.9218±0.0040	0.9291±0.0045	0.9036±0.0061	0.9194±0.0040	0.9049±0.0058	**0.9295±0.0038**	**0.9311±0.0035**
glass	AUC	0.9255±0.0031	0.9214±0.0029	**0.9326±0.0032**	0.9159±0.0029	0.9238±0.0032	0.9233±0.0029	0.9190±0.0030	**0.9372±0.0038**
	FM	0.7611±0.0079	0.7411±0.0112	**0.7912±0.0091**	0.6951±0.0106	0.7742±0.0090	0.7407±0.0117	0.7442±0.0105	**0.8058±0.0104**
	GM	0.8264±0.0080	0.8050±0.0104	**0.8545±0.0074**	0.7462±0.0150	0.8393±0.0077	0.8055±0.0114	0.8125±0.0103	**0.8664±0.0086**
haberman	AUC	0.6869±0.0076	0.6825±0.0075	0.6973±0.0069	0.6463±0.0051	**0.7045±0.0046**	0.6789±0.0064	0.6883±0.0064	**0.7140±0.0047**
	FM	0.5091±0.0082	0.5130±0.0081	0.5086±0.0084	0.4886±0.0050	**0.5258±0.0067**	0.5144±0.0077	0.5119±0.0075	**0.5327±0.0065**
	GM	0.6245±0.0168	0.6334±0.0129	0.6500±0.0079	0.6007±0.0069	**0.6640±0.0077**	0.6371±0.0120	0.6536±0.0081	**0.6728±0.0059**
heart	AUC	0.9095±0.0028	0.9091±0.0029	**0.9097±0.0023**	0.9078±0.0022	0.9091±0.0026	0.9107±0.0022	0.9063±0.0029	**0.9098±0.0031**
	FM	**0.8199±0.0057**	0.8159±0.0053	0.8176±0.0054	**0.8199±0.0044**	0.8171±0.0061	0.8189±0.0053	0.8128±0.0054	0.8167±0.0074
	GM	**0.8211±0.0069**	0.8106±0.0077	0.8174±0.0079	**0.8210±0.0050**	0.8151±0.0075	0.8146±0.0080	0.8121±0.0080	0.8118±0.0115
ionosphere	AUC	0.9763±0.0010	0.9766±0.0015	0.9757±0.0012	0.9768±0.0010	**0.9770±0.0014**	**0.9768±0.0015**	0.9744±0.0011	0.9765±0.0009
	FM	0.8986±0.0061	0.8956±0.0054	**0.8990±0.0044**	0.8922±0.0057	0.8951±0.0048	0.8937±0.0079	0.8972±0.0044	**0.9009±0.0049**
	GM	0.9278±0.0045	0.9258±0.0037	**0.9286±0.0032**	0.9228±0.0041	0.9254±0.0033	0.9245±0.0060	0.9254±0.0033	**0.9299±0.0035**
libra	AUC	0.9846±0.0021	0.9801±0.0027	0.9790±0.0041	0.9836±0.0014	0.9789±0.0024	0.9802±0.0024	**0.9874±0.0015**	**0.9892±0.0013**
	FM	0.8538±0.0149	0.8559±0.0131	0.8459±0.0191	**0.8949±0.0088**	0.8427±0.0132	0.8277±0.0174	0.8914±0.0135	**0.9286±0.0070**
	GM	0.9453±0.0060	0.9439±0.0049	0.9394±0.0082	0.9569±0.0038	0.9391±0.0057	0.9331±0.0065	**0.9602±0.0051**	**0.9698±0.0030**
pima	AUC	0.8289±0.0020	0.8219±0.0014	0.8294±0.0019	**0.8313±0.0017**	**0.8305±0.0014**	0.8240±0.0017	0.8262±0.0012	0.8264±0.0014
	FM	0.6982±0.0027	0.6941±0.0032	0.6929±0.0030	0.6950±0.0034	**0.6986±0.0025**	0.6931±0.0020	0.6935±0.0025	**0.6993±0.0035**
	GM	0.7640±0.0025	0.7589±0.0038	0.7582±0.0037	0.7617±0.0034	**0.7645±0.0032**	0.7591±0.0025	0.7594±0.0027	**0.7642±0.0042**
segment	AUC	**0.9934±0.0003**	0.9898±0.0004	0.9882±0.0004	0.9924±0.0004	**0.9927±0.0004**	0.9909±0.0003	0.9899±0.0005	0.9906±0.0004
	FM	0.8174±0.0059	**0.8257±0.0085**	0.8205±0.0071	**0.8652±0.0071**	0.8187±0.0076	0.8208±0.0072	0.8077±0.0076	0.8212±0.0060
	GM	0.9575±0.0015	**0.9581±0.0024**	0.9554±0.0017	**0.9639±0.0016**	0.9576±0.0019	0.9578±0.0020	0.9540±0.0020	0.9562±0.0016
vehicle	AUC	0.9910±0.0005	**0.9932±0.0003**	0.9854±0.0006	0.9918±0.0006	**0.9919±0.0005**	0.9913±0.0003	0.9905±0.0006	0.9913±0.0005
	FM	0.8781±0.0075	**0.8852±0.0097**	0.8810±0.0038	0.8841±0.0069	0.8754±0.0077	0.8732±0.0051	0.8782±0.0077	**0.8932±0.0063**
	GM	0.9547±0.0034	**0.9575±0.0041**	0.9558±0.0017	0.9552±0.0028	0.9534±0.0034	0.9526±0.0023	0.9549±0.0034	**0.9609±0.0026**
Simulated1	AUC	0.9437±0.0014	0.9351±0.0022	**0.9461±0.0015**	0.9422±0.0013	0.9431±0.0012	0.9411±0.0015	0.9409±0.0024	**0.9476±0.0014**
	FM	0.7660±0.0027	0.7396±0.0041	**0.7736±0.0041**	0.7251±0.0074	0.7573±0.0057	0.7204±0.0092	**0.7709±0.0028**	0.7700±0.0027
	GM	**0.8966±0.0015**	0.8858±0.0019	0.8902±0.0022	0.8786±0.0033	0.8938±0.0024	0.8773±0.0041	**0.8957±0.0018**	0.8881±0.0017
Simulated2	AUC	0.9525±0.0007	0.9457±0.0009	0.9524±0.0009	**0.9535±0.0014**	0.9522±0.0010	0.9507±0.0009	0.9451±0.0013	**0.9527±0.0008**
	FM	0.8307±0.0029	0.7978±0.0062	**0.8342±0.0015**	0.8160±0.0055	0.8316±0.0021	0.8053±0.0088	0.8241±0.0029	**0.8359±0.0012**
	GM	0.8964±0.0019	0.8749±0.0042	**0.8971±0.0010**	0.8858±0.0034	0.8968±0.0015	0.8788±0.0061	0.8913±0.0020	**0.8978±0.0008**
Simulated3	AUC	0.9865±0.0004	0.9861±0.0006	0.9863±0.0004	**0.9868±0.0004**	0.9861±0.0004	0.9829±0.0006	**0.9882±0.0007**	0.9867±0.0006
	FM	0.9191±0.0045	0.9032±0.0055	0.9185±0.0038	**0.9372±0.0043**	0.9160±0.0061	0.8915±0.0056	0.9249±0.0035	**0.9272±0.0043**
	GM	0.9680±0.0013	0.9634±0.0019	0.9673±0.0012	**0.9733±0.0014**	0.9675±0.0016	0.9573±0.0020	**0.9698±0.0013**	0.9692±0.0011
Simulated4	AUC	0.9891±0.0003	0.9885±0.0003	0.9887±0.0003	**0.9902±0.0003**	0.9888±0.0003	0.9872±0.0004	**0.9893±0.0003**	0.9885±0.0003
	FM	0.9359±0.0031	**0.9419±0.0033**	0.9364±0.0039	**0.9460±0.0032**	0.9329±0.0033	0.9352±0.0044	0.9375±0.0034	0.9337±0.0040
	GM	0.9652±0.0013	**0.9682±0.0016**	0.9651±0.0018	**0.9701±0.0014**	0.9638±0.0015	0.9638±0.0021	0.9658±0.0015	0.9632±0.0019

of minority instance, τ_{min} as in (4) and τ_{maj} as in (5), respectively. The method compared the closeness factor between two classes in each minority instance.

$$\tau_{\text{min}}(i) = \sum_{j=1}^{K} \frac{\mathrm{C}\left(\boldsymbol{S}_{\text{min}}(i, j)\right)}{\mathrm{Dist}\left(\boldsymbol{X}_{\text{min}}(i), \boldsymbol{S}_{\text{min}}(i, j)\right)} \tag{4}$$

$$\tau_{\text{maj}}(i) = \sum_{j=1}^{K} \frac{\mathrm{C}\left(\boldsymbol{S}_{\text{maj}}(i.j)\right)}{\mathrm{Dist}\left(\boldsymbol{X}_{\text{min}}(i), \boldsymbol{S}_{\text{maj}}(i, j)\right)} \tag{5}$$

We defined the cutoff threshold CT as a threshold to select minority instances located in minority distribution to generate new instances. This step aimed to avoid the overlapping problem. We applied the minority and majority closeness factor, τ_{min} and τ_{maj}, to every minority instance to decide whether the method excluded the minority instance out of $\boldsymbol{X}_{\text{min}}$ or not. In each minority instance $\boldsymbol{X}_{\text{min}}(i)$, if $\tau_{\text{min}}(i)$ had a higher value than $\tau_{\text{maj}}(i)$ multiplied by CT, then we would keep this point, as shown in Fig. 1(d). This step focused on the minority instances located in the minority distribution and also avoided the majority distribution. We will discuss the appropriate cutoff threshold value in Section IV.

In equations (4) and (5), the method also applied the distance from the nearest neighbor to the referenced minority instance because the different neighbors of the referenced point may have the same closeness factor with different distances. Thus, we assigned a higher value to the nearer point than the further point.

After that, the method assigned a ratio of the closeness factor φ between minority and majority group in each minority instance $\boldsymbol{X}_{\text{min}}(i)$ as in (6) and converted into the probability distribution as in (7).

$$\varphi(i) = \frac{\tau_{\text{min}}(i) + 1}{\tau_{\text{maj}}(i) + 1} \tag{6}$$

In equation (6), we added one into minority and majority

TABLE 3. Performance evaluations of the oversampling methods using Logistic Regression algorithm

Data	Metric	SMOTE	BorSMOTE	KSMOTE	SLSMOTE	CluSMOTE	ADASYN	A-SUWO	SyMProD
balance	AUC	0.5516±0.0068	0.5529±0.0057	0.5159±0.0119	0.5084±0.0024	0.5546±0.0097	**0.5566±0.0056**	0.5342±0.0119	**0.5806±0.0065**
	FM	0.2844±0.0064	0.2808±0.0072	0.2847±0.0067	**0.3170±0.0038**	0.2979±0.0092	0.2821±0.0087	0.2760±0.0130	**0.3576±0.0059**
	GM	0.5057±0.0068	0.5021±0.0078	0.5059±0.0069	0.5176±0.0043	**0.5198±0.0096**	0.5030±0.0091	0.4940±0.0136	**0.5637±0.0064**
ecoli	AUC	0.9372±0.0015	0.9342±0.0019	**0.9375±0.0017**	0.9078±0.0042	0.9256±0.0035	0.9319±0.0024	0.9351±0.0018	**0.9377±0.0016**
	FM	0.7049±0.0102	0.6866±0.0113	**0.7394±0.0075**	0.6337±0.0123	0.6485±0.0145	0.6418±0.0146	0.7178±0.0083	**0.7475±0.0067**
	GM	0.9001±0.0043	0.8892±0.0054	**0.9096±0.0031**	0.8437±0.0070	0.8651±0.0088	0.8693±0.0080	0.8980±0.0040	**0.9048±0.0029**
glass	AUC	0.8249±0.0051	0.8217±0.0052	**0.8252±0.0049**	0.8176±0.0058	**0.8264±0.0045**	0.8243±0.0047	0.8233±0.0052	0.8150±0.0050
	FM	0.6683±0.0064	0.6563±0.0068	**0.6741±0.0072**	0.6399±0.0066	**0.6737±0.0060**	0.6649±0.0068	0.6580±0.0065	0.6666±0.0054
	GM	0.7224±0.0082	0.6989±0.0089	**0.7402±0.0085**	0.6690±0.0118	0.7320±0.0075	0.7126±0.0085	0.7063±0.0099	**0.7331±0.0053**
haberman	AUC	0.6766±0.0060	0.6677±0.0081	**0.6911±0.0052**	0.5681±0.0042	0.6834±0.0031	0.6761±0.0051	0.6805±0.0055	**0.6858±0.0070**
	FM	0.4851±0.0114	0.4718±0.0111	**0.5491±0.0077**	0.3309±0.0116	0.4829±0.0069	0.4912±0.0078	0.4781±0.0099	**0.5407±0.0085**
	GM	0.6291±0.0095	0.6217±0.0097	**0.6895±0.0059**	0.4969±0.0107	0.6237±0.0062	0.6405±0.0070	0.6064±0.0062	**0.6785±0.0086**
heart	AUC	**0.9020±0.0024**	0.9013±0.0029	0.9019±0.0024	0.8977±0.0025	0.9017±0.0027	0.9016±0.0026	0.8997±0.0029	**0.9026±0.0025**
	FM	0.8122±0.0053	0.8120±0.0076	0.8093±0.0061	0.8009±0.0083	0.8147±0.0052	**0.8154±0.0071**	0.8095±0.0056	**0.8160±0.0053**
	GM	0.8134±0.0061	0.8126±0.0086	0.8110±0.0074	0.7928±0.0132	0.8149±0.0059	**0.8169±0.0094**	0.8096±0.0077	**0.8185±0.0065**
ionosphere	AUC	0.8633±0.0055	0.8633±0.0050	**0.8639±0.0053**	**0.8676±0.0066**	0.8602±0.0060	0.8616±0.0059	0.8363±0.0058	0.8630±0.0056
	FM	0.7897±0.0062	0.7856±0.0071	**0.7945±0.0051**	0.7916±0.0080	0.7860±0.0086	0.7829±0.0074	0.7853±0.0069	**0.7963±0.0050**
	GM	0.8340±0.0050	0.8307±0.0059	**0.8364±0.0038**	0.8355±0.0062	0.8312±0.0067	0.8297±0.0061	0.8207±0.0053	**0.8375±0.0038**
libra	AUC	0.7341±0.0092	0.7333±0.0097	0.7341±0.0091	**0.7363±0.0110**	0.7342±0.0090	0.7344±0.0094	**0.7467±0.0106**	0.7342±0.0089
	FM	0.5628±0.0162	0.5629±0.0146	0.5623±0.0162	0.5567±0.0157	**0.5662±0.0136**	0.5620±0.0137	**0.5812±0.0139**	0.5609±0.0146
	GM	0.7095±0.0130	0.7084±0.0118	0.7080±0.0129	**0.7106±0.0132**	**0.7108±0.0115**	0.7086±0.0113	0.7086±0.0110	0.7053±0.0122
pima	AUC	0.8289±0.0013	0.8258±0.0014	0.8272±0.0016	0.8278±0.0011	0.8275±0.0010	0.8261±0.0013	**0.8292±0.0009**	**0.8290±0.0012**
	FM	0.6846±0.0027	0.6808±0.0034	0.6859±0.0029	**0.6864±0.0028**	0.6839±0.0031	0.6778±0.0026	0.6829±0.0019	**0.6872±0.0023**
	GM	0.7483±0.0035	0.7424±0.0044	0.7503±0.0036	**0.7523±0.0030**	0.7473±0.0039	0.7398±0.0050	0.7459±0.0023	**0.7514±0.0026**
segment	AUC	**0.9449±0.0005**	0.9349±0.0008	0.9430±0.0009	**0.9447±0.0007**	0.9435±0.0005	0.9384±0.0006	0.9321±0.0009	0.9437±0.0005
	FM	0.6288±0.0048	0.6222±0.0064	**0.6519±0.0045**	0.6392±0.0043	0.6312±0.0042	0.6265±0.0050	0.6221±0.0041	**0.6646±0.0049**
	GM	**0.8866±0.0016**	0.8837±0.0026	0.8710±0.0027	0.8853±0.0011	**0.8872±0.0013**	0.8865±0.0020	0.8864±0.0018	0.8719±0.0027
vehicle	AUC	**0.9937±0.0005**	0.9932±0.0007	**0.9938±0.0005**	0.9912±0.0007	0.9936±0.0007	0.9935±0.0005	0.9921±0.0010	0.9935±0.0007
	FM	0.9325±0.0030	0.9339±0.0026	**0.9342±0.0029**	0.8771±0.0054	0.9325±0.0032	0.9333±0.0032	0.9222±0.0023	**0.9360±0.0030**
	GM	0.9670±0.0017	0.9671±0.0018	**0.9680±0.0017**	0.9535±0.0023	0.9664±0.0019	**0.9672±0.0017**	0.9609±0.0017	0.9669±0.0019
Simulated1	AUC	0.9568±0.0004	**0.9569±0.0004**	0.9565±0.0005	0.9565±0.0005	0.9568±0.0004	0.9567±0.0004	0.9568±0.0003	**0.9569±0.0004**
	FM	0.7605±0.0059	0.7276±0.0068	**0.7867±0.0039**	0.7753±0.0079	0.7558±0.0066	0.7458±0.0077	0.7498±0.0057	**0.7891±0.0023**
	GM	0.8993±0.0019	0.8860±0.0035	**0.9038±0.0016**	0.9026±0.0026	0.8967±0.0028	0.8915±0.0031	0.8949±0.0028	**0.9060±0.0013**
Simulated2	AUC	0.9617±0.0003	0.9616±0.0003	**0.9617±0.0003**	0.9614±0.0003	0.9617±0.0003	0.9616±0.0003	0.9616±0.0003	**0.9618±0.0003**
	FM	0.8377±0.0040	0.8226±0.0053	**0.8488±0.0018**	0.8417±0.0036	0.8366±0.0037	0.8330±0.0049	0.8237±0.0057	**0.8497±0.0017**
	GM	0.9016±0.0023	0.8925±0.0034	**0.9073±0.0012**	0.9031±0.0021	0.9012±0.0023	0.8984±0.0029	0.8934±0.0034	**0.9081±0.0010**
Simulated3	AUC	0.9912±0.0003	**0.9919±0.0003**	0.9914±0.0003	0.9901±0.0004	0.9909±0.0003	0.9867±0.0007	**0.9922±0.0002**	0.9914±0.0003
	FM	0.9353±0.0040	0.9495±0.0020	0.9605±0.0018	0.9228±0.0069	0.9287±0.0034	0.8511±0.0064	**0.9791±0.0015**	**0.9619±0.0020**
	GM	0.9750±0.0012	0.9791±0.0006	0.9822±0.0005	0.9711±0.0022	0.9731±0.0010	0.9459±0.0025	**0.9862±0.0009**	**0.9826±0.0005**
Simulated4	AUC	0.9914±0.0002	0.9910±0.0002	**0.9914±0.0002**	0.9910±0.0002	0.9914±0.0002	0.9897±0.0003	0.9910±0.0005	**0.9915±0.0002**
	FM	0.9579±0.0025	0.9646±0.0018	0.9682±0.0017	0.9476±0.0032	0.9568±0.0020	0.9142±0.0048	**0.9754±0.0017**	**0.9686±0.0018**
	GM	0.9774±0.0012	0.9805±0.0008	0.9822±0.0008	0.9724±0.0015	0.9769±0.0009	0.9554±0.0026	**0.9830±0.0012**	**0.9824±0.0008**

closeness factor to avoid the infinity value of ratio since the majority closeness factor can be almost zero if the majority nearest neighbors were far from the minority instance.

$$P(i) = \frac{\varphi(i)}{\sum_{j=1}^{n_{\min}} \varphi(j)} \tag{7}$$

The method assigned a probability distribution to each minority instance as in (7). The method did not consider minority data in the majority region based on the cutoff threshold CT to decrease the possibility of the overlapping problem. Therefore, $P(i)$ was the probability from which data point i^{th} of minority instances was selected to synthesize new points. The referenced data points were selected, and the number of generated samples was assigned in each point based on the probability distribution $P(i)$, as shown in Fig. 1(e).

In this step, we selected minority instances to create synthetic points, which determined the minority distribution. This procedure solved the overlapping and overgeneralization problem by selecting the minority instances based on the probability distribution instead of the same weight. In addition, the selected minority instances located in the minority distribution instead of majority distribution.

3) Instance Synthesis

In the last stage, the method generated synthetic instances from data points that we selected in the Minority Point Selection step. We modified the Sigma Nearest Oversampling based on Convex Combination (SNOCC) [15] algorithm that determined several nearest neighbors with the probability to create a new sample for determining the minority distribution instead of generating along the line between two points as SMOTE performed.

In each referenced minority instance, the method searched for M minority nearest neighbors where M was the number of neighbors to generate new samples. Then, the real value of the referenced minority instance and its neighbors were collected in a set $\boldsymbol{R}$, $\{R(1), R(2), \ldots, R(M+1)\}$. $\boldsymbol{R}$ was a set of real value to calculate new samples that the value of new samples was within this range. The probability distribution P of each instance in set $\boldsymbol{R}$ was collected in set $\boldsymbol{Pr}$. This step created the new sample $\boldsymbol{X}_{\text{new}}$ and added to the oversampled set $\boldsymbol{O}$ as follows:

$$\boldsymbol{X}_{\text{new}} = \sum_{j=1}^{M+1} \beta(j) Pr(j) R(j), \tag{8}$$

where $\beta(j)$ was a random of the positive value between 0 and 1, $Pr(j)$ was a probability distribution of sample j^{th} from referenced instance and its neighbors. Both $\beta(j)$ and $Pr(j)$ were normalized to one as the coefficient factor where $\beta(1)Pr(1)+\beta(2)Pr(2)+\ldots+\beta(M+1)Pr(M+1) = 1$, and $R(j)$ was a real value of sample j^{th}. We can claim that the value of new sample $\boldsymbol{X}_{\text{new}}$ will be created within the range of real value in set $\boldsymbol{R}$ because of the normalization step of the coefficient factors.

In this step, the method generated synthetic instances from multiple data points, which determined the minority class distribution. SyMProD also solved the overfitting problem that the method did not replicate the data as random oversampling performed. Fig. 1(f) shows the generated sample that the method created in a minority region.

B. DATASETS

To evaluate the oversampling technique, we selected ten imbalanced datasets from the UCI Machine Learning Repository [25] and four artificial datasets as benchmark datasets. Table 1 summarizes the characteristics of these datasets. We converted the multi-class dataset into binary class by labeling a particular class as a minority and merging the others as a majority class. Additionally, the four artificial datasets were a binary classification of 1000 samples generated by using the make_classification function from a scikit-learn library [26] with different imbalanced ratio. We labeled these artificial datasets as simulated datasets.

C. OVERSAMPLERS

Our oversampling technique, SyMProD, was evaluated by 14 datasets and compared with seven conventional oversampling techniques, i.e., SMOTE [16], Borderline-SMOTE (BorSMOTE) [17], Safe-Level SMOTE (SLSMOTE) [20], Cluster SMOTE (CluSMOTE) [21], k-means SMOTE (KSMOTE) [9], A-SUWO [18], and ADASYN [22]. The standard oversampling algorithms were implemented by imblearn [27] and smote-variant libraries [28] based on Python. We applied three classifiers, i.e., Random Forest, Logistic Regression, and Support Vector Machine to create prediction models using scikit-learn library [26] for the assessment of the proposed oversampling technique. The stratified five folds cross-validation was applied to randomly split data. The cross-validation avoids the undesirable shift by preserving the original distribution between classes in training and validation sets. Each cross-validation repeats twenty times and represents the average of the model performances. Moreover, the proposed oversampling technique only performed on the training set, whereas the validation set was used to evaluate the performance.

Algorithm 1: Synthetic Minority based on Probabilistic Distribution (SyMProD) Oversampling

```
Inputs : I : Original dataset,
    $NT$: Noise threshold to detect and remove noisy data,
    $CT$: Cutoff threshold to filter minority instances out of majority region,
    $K$: Number of nearest neighbors to compare the closeness factor between groups,
    $M$: Number of nearest neighbors to generate synthetic instances.
Output: O: Oversampled data after rebalancing.
Procedure:
1) Calculate the number of generated instances based on an imbalanced gap, $n_{\text{gen}}$, according to $n_{\text{gen}} = n_{\text{maj}} - n_{\text{min}}$, where $n_{\text{maj}}$ is the number of majority class instances and $n_{\text{min}}$ is the number of the minority class instances.
i) Noise Removal step: Remove outliers to reduce the possibility of noise generation.
    2) Apply the Z-score normalization to the minority and majority class of original dataset, $\boldsymbol{I}$.
    3) Remove instances if an absolute value of normalization is higher than $NT$ and collect the valid instances in the noise filtered dataset, $\boldsymbol{X}$.
ii) Minority Point Selection step: Select minority instances based on a probability distribution for the generating process.
    4) Calculate the Euclidean distance in each instance using (2).
    5) Determine the closeness factor and normalize using (3).
    6) For all minority instances, find $K$ minority nearest neighbors and $K$ majority nearest neighbors to define minority closeness factor $\tau_{\text{min}}$ and majority closeness factor $\tau_{\text{maj}}$ using (4) and (5), respectively.
    7) Exclude minority instances if the minority closeness factor is less than the majority closeness factor multiplies by $CT$.
    8) Define a ratio of closeness factor $\varphi$ using (6).
    9) Transform $\varphi$ into a probability distribution $P$ in each minority instance using (7).
    10) Select $n_{\text{gen}}$ minority instances based on $P$
iii) Instance Synthesis step : Generate the synthetic instances that cover the minority distribution.
- For each Minority Point Selection
    11) Find $M$ minority nearest neighbors
    12) Collect the real value of selected minority point and minority nearest neighbors in $\boldsymbol{R}$.
    13) Generate a random positive vector of $M+1$ dimensions where the value is between 0 and 1.
    14) Generate a synthetic instance from the selected point and nearest neighbors by set $\boldsymbol{R}$ and the normalization of the random positive vector and the probability distribution, as shown in (8).
    15) Add a generated instance into $\boldsymbol{O}$.
Return $\boldsymbol{O}$
```

TABLE 4. Performance evaluations of the oversampling methods using Support Vector Machine algorithm

Data	Metric	SMOTE	BorSMOTE	KSMOTE	SLSMOTE	CluSMOTE	ADASYN	A-SUWO	SyMProD
balance	AUC	0.5317±0.0079	0.5301±0.0082	0.5058±0.0136	0.5056±0.0028	0.5283±0.0129	**0.5387±0.0086**	0.5050±0.0141	**0.5924±0.0049**
	FM	0.2600±0.0082	0.2731±0.0084	0.2767±0.0071	0.2665±0.0029	**0.2957±0.0090**	0.2548±0.0085	0.2530±0.0155	**0.3683±0.0046**
	GM	0.4748±0.0088	0.4896±0.0092	0.4950±0.0078	0.4793±0.0071	**0.5107±0.0092**	0.4687±0.0094	0.4538±0.0185	**0.5742±0.0052**
ecoli	AUC	**0.9365±0.0015**	0.9328±0.0019	0.9345±0.0016	0.9308±0.0032	0.9284±0.0028	0.9348±0.0022	0.9328±0.0013	**0.9353±0.0016**
	FM	0.7075±0.0104	0.6782±0.0114	**0.7356±0.0079**	0.7022±0.0093	0.6828±0.0121	0.6584±0.0134	0.7156±0.0095	**0.7364±0.0076**
	GM	0.9009±0.0039	0.8856±0.0049	**0.9097±0.0037**	0.8937±0.0045	0.8870±0.0066	0.8784±0.0067	0.9036±0.0035	**0.9077±0.0038**
glass	AUC	**0.8172±0.0053**	0.8117±0.0065	0.8126±0.0071	0.8073±0.0071	0.8136±0.0048	**0.8136±0.0055**	0.8119±0.0063	0.8133±0.0055
	FM	**0.6502±0.0088**	0.6358±0.0070	0.6495±0.0091	0.6376±0.0042	0.6460±0.0080	0.6434±0.0069	0.6377±0.0063	**0.6624±0.0075**
	GM	0.6846±0.0136	0.6587±0.0097	**0.6998±0.0125**	0.6546±0.0084	0.6853±0.0107	0.6685±0.0120	0.6588±0.0122	**0.7109±0.0113**
haberman	AUC	0.6863±0.0055	0.6670±0.0054	**0.6971±0.0044**	0.5670±0.0074	0.6807±0.0058	0.6730±0.0037	0.6854±0.0058	**0.6938±0.0039**
	FM	0.4338±0.0066	0.4423±0.0110	**0.5494±0.0055**	0.3382±0.0098	0.4349±0.0068	0.4327±0.0087	0.4403±0.0063	**0.5230±0.0128**
	GM	0.5722±0.0056	0.5838±0.0098	**0.6853±0.0143**	0.4986±0.0097	0.5718±0.0060	0.5761±0.0079	0.5786±0.0054	**0.5986±0.0433**
heart	AUC	0.9001±0.0022	0.8985±0.0030	**0.9008±0.0026**	0.8942±0.0029	0.9001±0.0030	0.8993±0.0029	0.8975±0.0028	**0.9034±0.0023**
	FM	0.8140±0.0068	0.8159±0.0071	**0.8179±0.0069**	0.8036±0.0071	0.8137±0.0079	0.8135±0.0070	0.8067±0.0051	**0.8224±0.0078**
	GM	0.8139±0.0091	0.8168±0.0108	**0.8208±0.0093**	0.7974±0.0121	0.8154±0.0115	0.8161±0.0103	0.8056±0.0075	**0.8269±0.0101**
ionosphere	AUC	0.8655±0.0071	0.8586±0.0055	0.8721±0.0073	**0.8740±0.0065**	0.8597±0.0062	0.8580±0.0057	0.8586±0.0069	**0.8739±0.0066**
	FM	0.7861±0.0098	0.7788±0.0089	**0.7902±0.0098**	**0.8084±0.0074**	0.7814±0.0082	0.7713±0.0064	0.7865±0.0077	0.7849±0.0113
	GM	0.8330±0.0076	0.8273±0.0071	**0.8363±0.0080**	**0.8484±0.0061**	0.8285±0.0064	0.8215±0.0050	0.8302±0.0066	0.8317±0.0092
libra	AUC	**0.8375±0.0064**	0.8350±0.0070	0.7981±0.0104	0.8280±0.0071	**0.8353±0.0065**	0.8337±0.0072	0.8167±0.0063	0.8114±0.0072
	FM	0.6831±0.0128	0.6647±0.0096	0.6603±0.0142	**0.7023±0.0126**	0.6915±0.0117	0.6464±0.0121	0.6779±0.0117	**0.6922±0.0115**
	GM	**0.8314±0.0078**	0.8208±0.0064	0.8069±0.0083	0.8206±0.0079	**0.8339±0.0062**	0.8126±0.0066	0.8198±0.0067	0.8253±0.0064
pima	AUC	0.8283±0.0013	0.8243±0.0016	0.8255±0.0017	**0.8300±0.0011**	**0.8287±0.0012**	0.8258±0.0013	0.8271±0.0011	0.8275±0.0014
	FM	0.6856±0.0028	0.6801±0.0025	0.6817±0.0027	**0.6895±0.0031**	0.6852±0.0024	0.6814±0.0024	0.6833±0.0027	**0.6859±0.0028**
	GM	0.7497±0.0038	0.7444±0.0034	0.7471±0.0032	**0.7572±0.0029**	0.7503±0.0034	0.7472±0.0026	0.7472±0.0033	**0.7506±0.0039**
segment	AUC	0.9221±0.0010	0.9026±0.0015	0.9382±0.0010	**0.9436±0.0007**	0.9113±0.0009	0.9041±0.0005	0.8914±0.0018	**0.9409±0.0007**
	FM	0.6427±0.0027	0.6414±0.0016	0.6334±0.0047	**0.6638±0.0066**	0.6417±0.0020	0.6415±0.0018	0.6409±0.0019	**0.6473±0.0039**
	GM	0.8979±0.0012	0.8987±0.0007	0.8624±0.0029	0.8776±0.0014	0.8977±0.0010	**0.8987±0.0008**	**0.8992±0.0009**	0.8768±0.0021
vehicle	AUC	**0.9943±0.0005**	0.9940±0.0005	0.9931±0.0005	0.9908±0.0005	**0.9944±0.0005**	0.9940±0.0006	0.9912±0.0006	0.9942±0.0004
	FM	0.9190±0.0025	0.9204±0.0019	0.9177±0.0022	0.9033±0.0059	**0.9197±0.0024**	0.9150±0.0026	0.9091±0.0024	**0.9222±0.0025**
	GM	**0.9687±0.0016**	**0.9690±0.0012**	0.9672±0.0013	0.9645±0.0025	0.9684±0.0015	0.9681±0.0013	0.9649±0.0016	0.9669±0.0012
Simulated1	AUC	0.9566±0.0004	**0.9568±0.0004**	0.9562±0.0006	0.9563±0.0005	0.9566±0.0005	0.9567±0.0005	0.9567±0.0004	**0.9570±0.0003**
	FM	0.7661±0.0063	0.7318±0.0059	**0.7837±0.0032**	0.7752±0.0068	0.7599±0.0049	0.7464±0.0095	0.7556±0.0079	**0.7895±0.0033**
	GM	0.9008±0.0023	0.8878±0.0025	**0.9035±0.0017**	0.9023±0.0025	0.8988±0.0022	0.8915±0.0038	0.8971±0.0034	**0.9068±0.0010**
Simulated2	AUC	0.9616±0.0003	0.9617±0.0003	0.9616±0.0004	0.9613±0.0003	0.9616±0.0003	**0.9618±0.0003**	0.9615±0.0003	**0.9617±0.0003**
	FM	0.8365±0.0044	0.8194±0.0066	**0.8478±0.0028**	0.8417±0.0030	0.8369±0.0030	0.8343±0.0056	0.8192±0.0044	**0.8495±0.0023**
	GM	0.9008±0.0026	0.8900±0.0042	**0.9069±0.0015**	0.9031±0.0018	0.9011±0.0017	0.8989±0.0033	0.8909±0.0029	**0.9079±0.0011**
Simulated3	AUC	0.9912±0.0003	0.9913±0.0003	**0.9918±0.0002**	0.9898±0.0003	0.9910±0.0003	0.9879±0.0007	0.9912±0.0003	**0.9920±0.0002**
	FM	0.9407±0.0042	0.9461±0.0019	0.9582±0.0030	0.9425±0.0036	0.9328±0.0044	0.8692±0.0083	**0.9768±0.0011**	**0.9604±0.0028**
	GM	0.9766±0.0012	0.9783±0.0006	0.9816±0.0008	0.9772±0.0010	0.9744±0.0013	0.9534±0.0029	**0.9867±0.0003**	**0.9822±0.0008**
Simulated4	AUC	0.9915±0.0002	0.9909±0.0002	**0.9917±0.0002**	0.9912±0.0002	0.9915±0.0002	0.9906±0.0003	0.9916±0.0003	**0.9920±0.0002**
	FM	0.9593±0.0017	0.9608±0.0013	0.9676±0.0022	0.9519±0.0025	0.9559±0.0026	0.9275±0.0034	**0.9769±0.0013**	**0.9684±0.0019**
	GM	0.9780±0.0008	0.9788±0.0006	0.9819±0.0010	0.9745±0.0012	0.9764±0.0012	0.9623±0.0018	**0.9861±0.0006**	**0.9822±0.0009**

D. METRICS

There are several measurement metrics to assess the performance of the oversampling technique and prediction model. However, some metrics cannot cope with the imbalanced dataset; therefore, the result might be unreliable, e.g., accuracy [29], [30]. The accuracy is the ratio between the number of correctly predicted and total observations. If the dataset is significantly imbalanced, the accuracy metric can reach 99 percent without the prediction of the positive class. Our study applied three classification metrics: 1) Area Under the Curve (AUC); 2) G-Mean (GM); and 3) F-Measure (FM). We considered the minority class as a positive class and the majority class as a negative class. Furthermore, the confusion matrix in Table 5 was required to calculate the evaluation.

The confusion matrix classifies the result into four groups according to actual and predicted classes. True Positive (TP) and True Negative (TN) are the numbers of the correctly predicted positive and negative values, respectively. On the other hand, False Positive (FP) and False Negative (FN) are the numbers of the incorrectly predicted because the actual classes are the negative and positive class, respectively.

TABLE 5. Confusion Matrix

		Predicted Class	
		Positive	Negative
Actual Class	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

Sensitivity is the proportion of positive cases that the model identifies correctly. If the dataset is imbalanced and no positive class is predicted, then the model will perform low Sensitivity. Specificity is the proportion of negative cases that the model identifies correctly. Precision explains the correctly positive prediction among the entire positive

TABLE 6. Results for mean ranking of oversampling methods over 3 classifiers and 13 datasets

Metric	SMOTE	BorSMOTE	KSMOTE	SLSMOTE	CluSMOTE	ADASYN	A-SUWO	SyMProD
Algorithm : Random Forest								
AUC	3.954	5.282	4.214	4.179	4.139	5.632	5.236	**3.336**
FM	4.200	4.929	4.150	4.918	4.550	5.514	4.368	**3.371**
GM	4.136	5.071	4.204	4.732	4.382	5.643	4.329	**3.504**
Algorithm : Logistic Regression								
AUC	3.764	4.825	3.886	5.779	4.207	5.254	4.768	**3.421**
FM	4.689	5.246	3.111	5.300	4.707	5.636	4.679	**2.582**
GM	4.371	5.082	3.354	5.425	4.511	5.300	4.821	**3.086**
Algorithm : Support Vector Machine								
AUC	3.793	4.989	4.225	5.325	4.375	4.964	5.382	**2.904**
FM	4.614	5.293	3.714	4.525	4.700	5.957	4.668	**2.514**
GM	4.379	5.061	3.832	5.039	4.486	5.564	4.464	**3.161**

predicted. G-Mean aggregates Sensitivity and Specificity to detect the overfitting of negative class with underfitting of the positive class. In an imbalanced dataset, a classifier may only predict the majority class with high accuracy, but low G-Mean metric. Hence, G-Mean indicates the performance of both positive and negative classes. The equation is given as follows:

$$\text{G-Mean} = \sqrt{\frac{\text{TP}}{\text{TP}+\text{FN}} \cdot \frac{\text{TN}}{\text{TN}+\text{FP}}} \tag{9}$$

$$= \sqrt{\text{Sensitivity} \cdot \text{Specificity}}. \tag{10}$$

F-Measure is the harmonic mean of Sensitivity and Precision that measures the performance of positive predicted. The γ adjusts the importance of Sensitivity and Precision. The equation of F-Measure is given as follows:

$$\text{F-Measure} = \frac{\left(1+\gamma^2\right) \cdot \text{Sensitivity} \cdot \text{Precision}}{\left(\gamma^2 \cdot \text{Precision}\right) + \text{Sensitivity}} \tag{11}$$

$$= \frac{\left(1+\gamma^2\right) \cdot \text{TP}}{\left(1+\gamma^2\right) \cdot \text{TP} + \left(\gamma^2 \cdot \text{FN}\right) + \text{FP}}. \tag{12}$$

Area Under the Curve (AUC) is the metric that measures from the Receiver Operating Characteristic curve (ROC), which displays the true positive rate (sensitivity) and the false positive rate (1-specificity) with many thresholds. The higher value of AUC means the better performance of the prediction model.

IV. RESULT AND DISCUSSION

A. EXPERIMENTAL RESULT

This section presents results that experimentally evaluate from seven oversampling techniques, three classifiers, 14 datasets, and three evaluation metrics. The evaluation method applied stratified five folds cross-validation to assess the generalization of prediction models with repeated twenty times and tuned the classification threshold. The results were averaged, and 95% confidence interval was calculated to represent the model performance. Tables 2, 3, and 4 illustrate the metric results of Random Forest, Logistic Regression, and Support Vector Machine, respectively. The top two measurements are represented in bold from each dataset and classifier.

To compare the results among the oversampling techniques, we applied the Friedman test and Holm's test as a post-hoc analysis. Other studies in oversampling techniques topic also used these statistical tests [4], [9], [18], [31]. We applied a ranking score to assess the performance of the oversampling technique in every combination. The best performing technique received a ranking one, while the worst received ranking eight.

According to Tables 2, 3, and 4, our method is not the best performing algorithm in all datasets. The performance of the classifier also depends on the distribution of data and the complexity of the dataset. So, we aimed to generalize our oversampling technique to handle various types of imbalance ratio and cope with multiple datasets. Table 6 summarizes the mean ranking of each classifier and oversampling technique. SyMProD performs better than other oversampling technique in every classifier and metric from benchmark datasets. Aside from our approach, k-means SMOTE and SMOTE achieve the best result. To compare the result statistically, we employed the Friedman test and Holm-Bonferroni method that SyMProD was the control method to verify the statistical significance.

Friedman test is a non-parametric test and is used to compare the performance of classifiers with the oversampling technique for each evaluation metric. The null hypothesis is whether there is a similar performance among all classifiers in terms of mean ranking. In this paper, the significance level was 0.05. Table 7 shows the results of the Friedman test. The null hypothesis was rejected in all oversampling techniques and classifiers, i.e., the classifiers did not perform similarly in mean rankings of AUC, F-Measure, and G-Mean metric.

TABLE 7. Results of Friedman Test

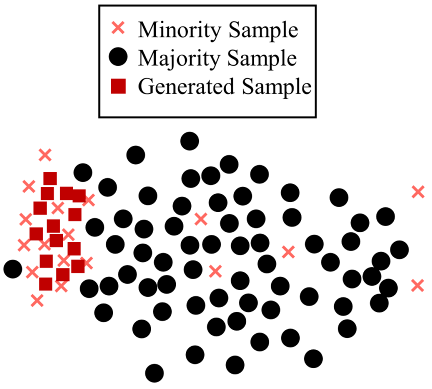
Metric	p-value
Algorithm : RF	
AUC	**1.78E-02**
FM	**2.70E-02**
GM	**2.63E-02**
Algorithm : LR	
AUC	**1.00E-03**
FM	**2.82E-05**
GM	**3.75E-03**
Algorithm : SVM	
AUC	**2.48E-04**
FM	**9.86E-06**
GM	**1.05E-02**

TABLE 8. Results of Holm's Test with SyMProD as the control method

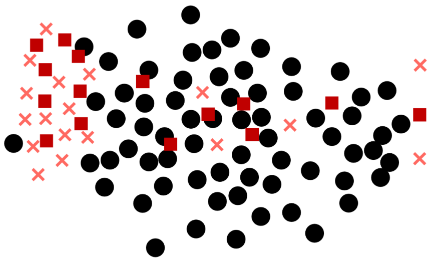
	AUC		FM		GM	
	Method	p-value	Method	p-value	Method	p-value
	Algorithm : Random Forest					
1	ADASYN	**2.52E-33**	ADASYN	**4.46E-28**	ADASYN	**5.06E-28**
2	BorSMOTE	**3.08E-24**	BorSMOTE	**4.14E-15**	BorSMOTE	**2.47E-15**
3	A-SUWO	**3.77E-23**	SLSMOTE	**6.35E-15**	SLSMOTE	**1.40E-09**
4	KSMOTE	**2.35E-05**	CluSMOTE	**8.44E-09**	CluSMOTE	**5.34E-05**
5	SLSMOTE	**5.57E-05**	A-SUWO	**2.36E-06**	A-SUWO	**1.84E-04**
6	CluSMOTE	**1.37E-04**	SMOTE	**1.91E-04**	KSMOTE	**2.81E-03**
7	SMOTE	**7.75E-03**	KSMOTE	**5.97E-04**	SMOTE	**9.62E-03**
	Algorithm : Logistic Regression					
1	SLSMOTE	**5.87E-35**	ADASYN	**5.20E-63**	SLSMOTE	**3.78E-35**
2	ADASYN	**1.26E-21**	SLSMOTE	**1.14E-50**	ADASYN	**1.18E-31**
3	BorSMOTE	**6.55E-13**	BorSMOTE	**8.19E-49**	BorSMOTE	**5.97E-26**
4	A-SUWO	**6.70E-12**	CluSMOTE	**8.15E-32**	A-SUWO	**7.96E-20**
5	CluSMOTE	**2.27E-04**	CluSMOTE	**2.52E-31**	CluSMOTE	**1.39E-13**
6	KSMOTE	8.06E-02	A-SUWO	**5.44E-31**	SMOTE	**3.82E-11**
7	SMOTE	2.42E-01	KSMOTE	**1.66E-02**	KSMOTE	6.97E-01
	Algorithm : Support Vector Machine					
1	A-SUWO	**3.96E-39**	ADASYN	**3.44E-77**	ADASYN	**5.75E-36**
2	SLSMOTE	**2.06E-37**	BorSMOTE	**8.60E-52**	BorSMOTE	**6.98E-23**
3	BorSMOTE	**3.87E-28**	CluSMOTE	**7.66E-33**	SLSMOTE	**2.14E-22**
4	ADASYN	**1.62E-27**	A-SUWO	**6.25E-32**	CluSMOTE	**2.27E-11**
5	CluSMOTE	**2.21E-14**	SMOTE	**1.87E-30**	A-SUWO	**5.04E-11**
6	KSMOTE	**1.13E-11**	SLSMOTE	**5.03E-28**	SMOTE	**1.08E-09**
7	SMOTE	**1.50E-05**	KSMOTE	**1.96E-10**	KSMOTE	**4.20E-03**

Therefore, a post-hoc test was applied.

According to the Holm-Bonferroni procedure, we set SyMProD as a control method to evaluate the performance of the oversampling technique and compare it to other methods. The procedure counteracts multiple testing using the step-down technique. The null hypothesis in each pairwise is to check whether the SyMProD does not perform better than the others. Table 8 shows the results of the Holm-Bonferroni and an adjusted p-value. Accordingly, we can compare the adjusted p-value with a significant level to indicate the hypothesis testing. For the AUC metric, our proposed method defeated the other oversampling technique when the classifiers were Random Forest and Support Vector Machine. However, our approach failed to reject the null hypothesis in Logistic Regression when compared to SMOTE and k-means SMOTE. For the F-Measure metric, SyMProD



(a) Synthetic instances were generated using SyMProD.



(b) Synthetic instances were generated using SMOTE.

FIGURE 2. Distribution of synthetic instances using SMOTE and SyMProD.

outperformed other techniques in every classifier. For the G-Mean metric, SyMProD was significantly better than the other methods when we applied the Support Vector Machine and Random Forest. When Logistic Regression was used, we outperformed the other oversampling technique except for the k-means SMOTE.

Based on three evaluation metrics, we mainly focus on AUC because G-Mean and F-Measure can be changed if the threshold is adjusted, whereas AUC does not change. Compared with SMOTE, the conventional oversampling technique, our result indicated that SyMProD could increase the AUC metric in 30 out of 42 cases over 14 datasets in three classifiers. The biggest gain of AUC was 0.06, 11.42% increment, that we could improve the balance's dataset when the classifier was Support Vector Machine. In terms of the generalization of our proposed technique, we evaluated the measurement metrics with several imbalanced datasets and multiple domain knowledge with cross-validation. The mean ranking of 14 datasets was applied to assess the performance. The results from our method were not the best in every dataset, but the outcomes showed that our process achieved the lowest mean ranking across datasets in every classifier. For the execution time, our oversampling technique used 1.27 seconds to create 1000 synthetic instances, which were higher than SMOTE because of the operation in the algorithm. For the dataset, SyMProD aimed to remove the drawbacks of other oversampling techniques. We can handle

TABLE 9. Sensitivity analysis on SyMProD using Support Vector Machine

Data	AUC value for different NT		AUC value for different CT		AUC value for different K		AUC value for different M	
	NT	value	CT	value	K	value	M	value
ecoli	2	0.9344±0.0017	0	0.9339±0.0015	1	0.9346±0.0015	1	0.9361±0.0014
	3	0.9346±0.0015	0.6	0.9340±0.0015	3	0.9344±0.0017	**3**	**0.9362±0.0016**
	4	0.9339±0.0015	0.8	0.9341±0.0016	**5**	**0.9353±0.0016**	5	0.9353±0.0016
	5	**0.9353±0.0016**	**1**	**0.9353±0.0016**	7	0.9351±0.0014	7	0.9346±0.0017
	6	**0.9353±0.0016**	1.2	0.9346±0.0019	9	0.9344±0.0016	9	0.9341±0.0018
	7	**0.9353±0.0016**	1.4	0.9338±0.0018	11	0.9351±0.0013	11	0.9348±0.0016
heart	2	0.9033±0.0022	0	0.9015±0.0026	1	0.9025±0.0022	1	0.9029±0.0023
	3	0.9027±0.0024	0.6	0.9020±0.0022	3	0.9030±0.0024	3	0.9030±0.0022
	4	0.9031±0.0025	0.8	0.9031±0.0022	**5**	**0.9034±0.0023**	**5**	**0.9034±0.0023**
	5	0.9034±0.0023	1	0.9034±0.0023	7	0.9031±0.0025	7	0.9033±0.0022
	6	**0.9036±0.0022**	**1.2**	**0.9036±0.0022**	9	0.9034±0.0023	9	0.9032±0.0023
	7	0.9035±0.0021	1.4	0.9033±0.0023	11	0.9024±0.0022	11	0.9030±0.0022
ionosphere	**2**	**0.8741±0.0067**	0	0.8672±0.0060	1	0.8707±0.0070	1	0.8707±0.0063
	3	0.8734±0.0070	0.6	0.8688±0.0060	3	0.8728±0.0068	3	0.8724±0.0066
	4	0.8738±0.0066	0.8	0.8724±0.0070	5	0.8739±0.0066	**5**	**0.8739±0.0066**
	5	0.8739±0.0066	**1**	**0.8739±0.0066**	7	0.8736±0.0064	7	0.8721±0.0073
	6	0.8739±0.0066	1.2	0.8738±0.0070	9	0.8739±0.0071	9	0.8730±0.0068
	7	0.8739±0.0066	1.4	0.8733±0.0069	**11**	**0.8742±0.0067**	11	0.8725±0.0066
pima	2	0.8246±0.0013	0	0.8275±0.0015	1	0.8273±0.0015	1	0.8274±0.0013
	3	0.8267±0.0015	0.6	0.8275±0.0014	**3**	**0.8278±0.0014**	3	0.8274±0.0015
	4	0.8268±0.0014	**0.8**	**0.8277±0.0016**	5	0.8275±0.0014	**5**	**0.8275±0.0014**
	5	**0.8275±0.0014**	1	0.8275±0.0014	7	0.8275±0.0013	7	0.8273±0.0014
	6	0.8273±0.0015	1.2	0.8272±0.0014	9	0.8269±0.0015	9	0.8268±0.0012
	7	0.8273±0.0016	1.4	0.8266±0.0010	11	0.8273±0.0012	11	0.8267±0.0013
segment	2	0.9253±0.0007	0	0.9366±0.0007	1	0.9411±0.0009	**1**	**0.9428±0.0006**
	3	**0.9422±0.0007**	0.6	0.9366±0.0006	3	0.9410±0.0007	3	0.9415±0.0006
	4	0.9402±0.0008	0.8	0.9392±0.0006	5	0.9409±0.0007	5	0.9409±0.0007
	5	0.9409±0.0007	1	0.9409±0.0007	7	0.9408±0.0007	7	0.9408±0.0007
	6	0.9410±0.0007	**1.2**	**0.9423±0.0007**	**9**	**0.9412±0.0007**	9	0.9407±0.0007
	7	0.9405±0.0005	1.4	0.9414±0.0007	11	0.9410±0.0008	11	0.9407±0.0007

the dataset when a few minority instances are located in the majority region, which may cause the overlapping problem in another oversampling technique. Moreover, our technique can work with multiple imbalance ratios. However, if the dataset contains several features and one of its features is an outlier value, our method will classify that point as an outlier and exclude it for oversampling, which can remove an insight if the rest of features are not outliers.

The result indicates that our oversampling technique can improve the prediction model. SyMProD has several advantages when compared with other data level methods. Firstly, our approach applies the oversampling technique to create new points rather than the undersampling, which may depreciate the valuable insight. Secondly, the noisy data are removed in the first stage to reduce the possibility of noise creation. Thirdly, our approach applies the probability distribution to create synthetic instances that cover minority regions by considering several minority points in the synthesis process. This procedure also eliminates the overfitting problem, which random oversampling faces. In addition, we avoid producing new points in the majority region to reduce the overlapping problem by excluding minority samples that locate in majority distribution with parameter CT. Moreover, our approach tries to maintain the original distribution of the dataset. The other techniques, e.g., ADASYN and borderline-SMOTE, focus on creating a sample on the hard to learn region, but it changes the distribution of the data massively. Fourthly, SyMProD assigns the probability to each minority instance to eliminate the overgeneralization problem as SMOTE faces. Lastly, any conventional classifiers can be trained with the oversampled data from our method, which is suitable for practitioners.

B. PARAMETER SENSITIVITY ANALYSIS

SyMProD acquires four parameters to oversample the dataset, i.e., Noise Threshold (NT), Cutoff Threshold (CT), K nearest neighbors (K) in the Minority Point Selection step, and M minority nearest neighbors (M) in the Instance Synthesis step. We assessed the performance in the different parameter values by performing the sensitivity analysis and presented results in Table 9. Firstly, our method utilizes the Noise Threshold NT to remove noise instances. An appropriate range of NT is between three to five since a lower value of NT eliminates valid points and decreases the classifier performance. On the other hand, a higher NT can increase the possibility of noise generation because the method does not remove the noise instances.

The Cutoff Threshold CT performs with K nearest neighbors K in the Minority Point Selection step. These parameters reduce the possibility of the overlapping class problem by considering the closeness factor between the two groups. In each minority instance, K majority and minority nearest neighbors are searched and used to calculate τ_{maj} and τ_{min}, respectively. We exclude the minority instances if τ_{maj} multiplied by CT is more than τ_{min}. A reasonable value for CT is between 0.8 and 1.2, and K is five. As shown in Table 9, when the CT is set to 0, the result is worse than a higher CT because the method does not consider the overlapping problem. However, a higher value of CT can remove the insight data because the method excludes the minority class instances.

Lastly, the number of minority nearest neighbors M is determined to generate new instances from multiple points and cover the minority distribution. A default parameter value is five. It means a new sample is created from the referenced point and five nearest neighbors of the referenced point. If M is defined as one, the Instance Synthesis process will perform as same as SMOTE oversampling technique. Besides, the generated instances may not determine the minority distribution because the method produces new points in the line between two points [15], as shown in Fig. 2.

V. CONCLUSION

In this paper, we have proposed a new oversampling algorithm called Synthetic Minority based on Probabilistic Distribution (SyMProD) oversampling to balance the imbalanced dataset. Our method has assigned a probability to each minority instance to generate synthetic instances based on probability distribution. SyMProD has avoided the noisy generation, overgeneralization, and overlapping class problem. The objective of the method is to generate samples that determine the minority distribution by considering referenced minority points and multiple neighbors. We have assessed the evaluation metrics, i.e., AUC, G-Mean, and F-Measure, with seven other oversampling algorithms and three classifiers, i.e., Random Forest, Support Vector Machine, and Logistic Regression. The results show that the data oversampled by our method had higher classification performance than other conventional oversampling techniques in most datasets.

REFERENCES

[1] J. Zhang, E. Bloedorn, L. Rosen, and D. Venese, “Learning rules from highly unbalanced data sets,” in *Fourth IEEE International Conference on Data Mining*, Brighton, UK, 2004, pp. 571–574.

[2] C. Phua, D. Alahakoon, and V. Lee, “Minority report in fraud detection: classification of skewed data,” *ACM SIGKDD*, vol. 6, no. 1, pp. 50–59, 2004.

[3] A. Amin, S. Anwar, A. Adnan, M. Nawaz, N. Howard, J. Qadir, A. Hawalah, and A. Hussain, “Comparing oversampling techniques to handle the class imbalance problem: A customer churn prediction case study,” *IEEE Access*, vol. 4, pp. 7940–7957, 2016.

[4] M. S. Santos, P. H. Abreu, P. J. García-Laencina, A. Simão, and A. Carvalho, “A new cluster-based oversampling method for improving survival prediction of hepatocellular carcinoma patients,” *Journal of biomedical informatics*, vol. 58, pp. 49–59, 2015.

[5] H. Yu, J. Ni, Y. Dan, and S. Xu, “Mining and integrating reliable decision rules for imbalanced cancer gene expression data sets,” *Tsinghua Science and technology*, vol. 17, no. 6, pp. 666–673, 2012.

[6] T. Elhassan and M. Aljurf, “Classification of imbalance data using tomek link (t-link) combined with random under-sampling (rus) as a data reduction method.” *Journal of Informatics and Data Mining*, vol. 1, pp. 1–12, 2016.

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

[8] G. Haixiang, L. Yijing, J. Shang, G. Mingyun, H. Yuanyue, and G. Bing, “Learning from class-imbalanced data: Review of methods and applications,” *Expert Systems with Applications*, vol. 73, pp. 220–239, 2017.

[9] G. Douzas, F. Bacao, and F. Last, “Improving imbalanced learning through a heuristic oversampling method based on k-means and smote,” *Information Sciences*, vol. 465, pp. 1–20, 2018.

[10] B. Das, N. C. Krishnan, and D. J. Cook, “Racog and wracog: Two probabilistic oversampling techniques,” *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 1, pp. 222–234, 2014.

[11] S. Piri, D. Delen, and T. Liu, “A synthetic informative minority oversampling (simo) algorithm leveraging support vector machine to enhance learning from imbalanced datasets,” *Decision Support Systems*, vol. 106, pp. 15–29, 2018.

[12] M. Naseriparsa and M. M. R. Kashani, “Combination of pca with smote resampling to boost the prediction rate in lung cancer dataset,” *International Journal of Computer Applications*, vol. 77, no. 3, pp. 33–38, 2013.

[13] K. J. Wang and A. M. Adrian, “Breast cancer classification using hybrid synthetic minority over-sampling technique and artificial immune recognition system algorithm,” *International Journal of Computer Science and Electronics Engineering (IJCSEE)*, vol. 1, no. 3, pp. 408–412, 2013.

[14] G. M. Weiss, K. McCarthy, and B. Zabar, “Cost-sensitive learning vs. sampling: Which is best for handling unbalanced classes with unequal error costs?” *DMIN*, vol. 7, no. 24, pp. 35–41, 2007.

[15] Z. Zheng, Y. Cai, and Y. Li, “Oversampling method for imbalanced classification,” *Computing and Informatics*, vol. 34, no. 5, pp. 1017–1037, 2016.

[16] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: synthetic minority over-sampling technique,” *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[17] H. Han, W. Y. Wang, and B. H. Mao, “Borderline-smote: a new over-sampling method in imbalanced data sets learning,” in *International conference on intelligent computing*, Hefei, China, 2005, pp. 878–887.

[18] I. Nekooeimehr and S. K. Lai-Yuen, “Adaptive semi-unsupervised weighted oversampling (a-suwo) for imbalanced datasets,” *Expert Systems with Applications*, vol. 46, pp. 405–416, 2016.

[19] S. Barua, M. M. Islam, X. Yao, and K. Murase, “Mwmote–majority weighted minority oversampling technique for imbalanced data set learning,” *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 2, pp. 405–425, 2012.

[20] C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap, “Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem,” in *Pacific-Asia conference on knowledge discovery and data mining*, Bangkok, Thailand, 2009, pp. 475–482.

[21] D. A. Cieslak, N. V. Chawla, and A. Striegel, “Combating imbalance in network intrusion datasets.” in *GrC*, 2006, pp. 732–737.

[22] H. He, Y. Bai, E. A. Garcia, and S. Li, “Adasyn: Adaptive synthetic sampling approach for imbalanced learning,” in *IEEE International Joint Conference on Neural Networks*, Hong Kong, 2008, pp. 1322–1328.

[23] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, “Smoteboost: Improving prediction of the minority class in boosting,” in *European conference on principles of data mining and knowledge discovery*, Cavtat-Dubrovnik, Croatia, 2003, pp. 107–119.

[24] L. Ma and S. Fan, “Cure-smote algorithm and hybrid algorithm for feature selection and parameter optimization based on random forests,” *BMC bioinformatics*, vol. 18, no. 1, pp. 1–18, 2017.

[25] D. Dua and C. Graff, “UCI machine learning repository,” 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[26] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[27] G. Lemaître, F. Nogueira, and C. K. Aridas, “Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning,” *Journal of Machine Learning Research*, vol. 18, no. 17, pp. 1–5, 2017. [Online]. Available: http://jmlr.org/papers/v18/16-365

[28] G. Kovács, “smote-variants: a python implementation of 85 minority oversampling techniques,” *Neurocomputing*, vol. 366, pp. 352–354, 2019.
[29] Y. Suh, J. Yu, J. Mo, L. Song, and C. Kim, “A comparison of oversampling methods on imbalanced topic classification of korean news articles,” *Journal of Cognitive Science*, vol. 18, no. 4, pp. 391–437, 2017.
[30] H. Zhang and Z. Wang, “A normal distribution-based over-sampling approach to imbalanced data classification,” in *International Conference on Advanced Data Mining and Applications*, Beijing, China, 2011, pp. 83–96.
[31] G. Douzas and F. Bacao, “Effective data generation for imbalanced learning using conditional generative adversarial networks,” *Expert Systems with applications*, vol. 91, pp. 464–471, 2018.

INTOUCH KUNAKORNTUM received a B.Eng. in Computer Engineering from King Mongkut’s University of Technology Thonburi, Bangkok, Thailand, in 2017. He is currently a postgraduate student at the Department of Computer Engineering, King Mongkut’s University of Technology Thonburi, Bangkok. His research interests are machine learning and oversampling technique.

WORANICH HINTHONG received a D.V.M. from the Faculty of Veterinary Medicine from Kasetsart University, Bangkok, Thailand, in 2008, and a Ph.D. degree in tropical medicine from Mahidol University, Nakhon Pathom, Thailand, in 2015. She is currently a lecturer at HRH Princess Chulabhorn College of Medical Science, Bangkok, Thailand. Her research interests include antibiotic resistance in bacteria, metagenomics, and transcriptomic.

PHOND PHUNCHONGHARN received a B.Eng. and an M.Eng. in Computer Engineering from King Mongkut’s University of Technology Thonburi, Bangkok, Thailand, in 2005 and 2007, respectively, and a Ph.D. in Electrical and Computer Engineering from the University of Manitoba, Canada, in 2013. She is currently an assistant professor at the Department of Computer Engineering and the director of Big Data Experience Center (BX), King Mongkut’s University of Technology Thonburi. Her research interests include data analytics, machine learning, cognitive radio networks, resource allocation, and optimization.