A SYNTHETIC MINORITY BASED ON PROBABILISTIC DISTRIBUTION (SYMPROD) OVERSAMPLING FOR IMBALANCED DATASETS

MR. INTOUCH KUNAKORNTUM

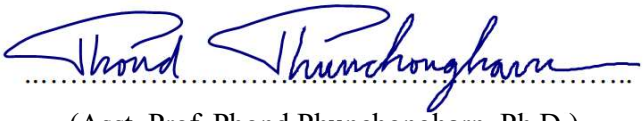
A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR
THE DEGREE OF MASTER OF ENGINEERING
(COMPUTER ENGINEERING)
FACULTY OF ENGINEERING
KING MONGKUT'S UNIVERSITY OF TECHNOLOGY THONBURI
2019

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

Mr. Intouch Kunakorntum B. Eng. (Computer Engineering)

A Thesis Submitted in Partial Fulfillment
of the Requirements for
the Degree of Master of Engineering (Computer Engineering)
Faculty of Engineering
King Mongkut's University of Technology Thonburi
2019

Thesis Committee



..... Thesis Advisor

(Asst. Prof. Phond Phunchongharn, Ph.D.)

..... Member

(Asst. Prof. Santitham Prom-on, Ph.D.)

..... Member

(Assoc. Prof. Peerapon Siripongwutikorn, Ph.D.)

..... Member

(Assoc. Prof. Anan Banharnsakun, Ph.D.)

..... Member

(Lect. Kejkaew Thanasuan, Ph.D.)

Thesis Title	A Synthetic Minority Based on Probabilistic Distribution (SyMProD) Oversampling for Imbalanced Datasets
Thesis Credits	12
Candidate	Mr. Intouch Kunakorntum
Thesis Advisor	Asst. Prof. Phond Phunchongharn, Ph.D.
Program	Master of Engineering
Field of Study	Computer Engineering
Department	Computer Engineering
Faculty	Engineering
Academic Year	2019

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 classes and fail to identify the minority classes 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 method 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 the 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 17 benchmark datasets and three classifiers. Moreover, this study compares the performance of the method with other eight conventional oversampling algorithms. The empirical results show that our method achieves better performance than other oversampling techniques.

Keywords: Classification/ Imbalanced Dataset/ Machine Learning/ Oversampling/ Probabilistic Distribution.

หัวข้อวิทยานิพนธ์	อัลกอริทึมการสร้างข้อมูลด้วยวิธีการแจกแจงค่าความน่าจะเป็นสำหรับข้อมูลที่ไม่สมดุล
หน่วยกิต	12
ผู้เขียน	นายอินทัช คุณากรธรรม
อาจารย์ที่ปรึกษา	ผศ.ดร.พร พันธุ์จงหาญ
หลักสูตร	วิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
ภาควิชา	วิศวกรรมคอมพิวเตอร์
คณะ	วิศวกรรมศาสตร์
ปีการศึกษา	2562

บทคัดย่อ

การจัดการข้อมูลที่ไม่สมดุลเป็นงานที่ท้าทายสำหรับระบบการวิเคราะห์ข้อมูลในปัจจุบัน ปัญหานี้จะส่งผลต่อโมเดล ทำให้ทำนายผลลัพธ์ได้เพียงแค่กลุ่มข้อมูลที่มีจำนวนมาก แต่ไม่สามารถพบกลุ่มข้อมูลที่มีจำนวนน้อยกว่าเนื่องจากชุดข้อมูลมีจำนวนที่แตกต่างกันมากระหว่างกลุ่ม สำหรับการแก้ไขปัญหาที่เกิดขึ้น การเพิ่มจำนวนของข้อมูลเป็นวิธีที่จัดการปัญหาข้อมูลที่ไม่สมดุลได้ อย่างไรก็ตามหลายเทคนิคในปัจจุบันไม่ได้นำการกระจายตัวของข้อมูลมาพิจารณา รวมถึงเกิดปัญหาการสร้างข้อมูลที่มีการเหลื่อมล้ำระหว่างกลุ่ม ดังนั้นงานวิทยานิพนธ์ฉบับนี้จึงได้คิดค้นวิธีการเพิ่มจำนวนของข้อมูลที่มีชื่อว่าการสร้างข้อมูลของกลุ่มที่น้อยกว่าด้วยวิธีการแจกแจงความน่าจะเป็น โดยมีทั้งหมด 3 ขั้นตอน ขั้นตอนแรกจะจัดข้อมูลให้อยู่ในช่วงเดียวกันและลบข้อมูลที่ผิดปกติโดยวิเคราะห์จากส่วนเบี่ยงเบนมาตรฐาน หลักจากนั้นจะเลือกกลุ่มข้อมูลที่มีจำนวนน้อยกว่าเพื่อเป็นจุดอ้างอิงในการสร้างข้อมูลใหม่ด้วยวิธีการกระจายตัวของความน่าจะเป็น รวมถึงวิเคราะห์การกระจายตัวทั้งกลุ่มที่มีจำนวนมากกว่าและน้อยกว่า สำหรับขั้นตอนการสร้างข้อมูลใหม่นั้น ตำแหน่งของข้อมูลใหม่จะอยู่ในข้อมูลที่เลือกมาและข้อมูลที่อยู่ใกล้เคียงหลายตัว เป้าหมายของวิธีที่นำเสนอนี้เพื่อให้ข้อมูลใหม่มีการครอบคลุมการกระจายตัวของข้อมูล, ลดโอกาสที่จะเกิดข้อมูลผิดปกติ, ลดปัญหาการเหลื่อมล้ำของกลุ่มและการให้ความสำคัญที่เท่ากันทุกจุด ผลลัพธ์จะถูกวัดผลและเปรียบเทียบกับวิธีการเพิ่มจำนวนข้อมูลอื่นจำนวน 8 วิธี ใช้ชุดข้อมูล 17 ชุดและ 3 โมเดลการทำนาย โดยผลลัพธ์แสดงให้เห็นว่าวิธีที่นำเสนอสามารถเพิ่มประสิทธิภาพของโมเดลทำนายได้มากกว่าเมื่อเทียบกับวิธีอื่น

คำสำคัญ: การกระจายตัวของความน่าจะเป็น/ การจำแนกประเภท/ การเรียนรู้ด้วยเครื่อง/ ข้อมูลที่ไม่สมดุล/ เพิ่มจำนวนข้อมูล

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to everyone who always supports and encourages me during the period of this research. Firstly, I would like to take this opportunity to acknowledge the guidance of my advisor, Asst. Prof. Phond Phunchongharn, who always provides helpful suggestions, feedback, and motivation. Several problems which I faced throughout the research have been solved by her advice. Moreover, I am extremely grateful for the opportunities that she provided to me. This research would not be finished without her assistance.

Moreover, this research collected data from Chulabhorn Hospital. I would like to thank Dr. Woranich Hinthong and Dr. Sumet Amornyingchareon for their support and guidance on the medical experience. The knowledge in the healthcare domain from their recommendation greatly improved the prediction model. Furthermore, I would like to thank all people who have been related through the process of data gathering.

Besides my thesis advisor, I would like to express sincere appreciation to all my lecturers; Prof. Tiranee Achalakul, Assoc. Prof. Peerapon Siripongwutikorn, Asst. Prof. Santitham Prom-on, Lect. Thanis Bunsom, and all members of the committee for the great advice.

Further, this research was supported by the Big Data Experience Center, Petchra Pra Jom Klao Scholarship Program for a Master's Degree, King Mongkut's University of Technology Thonburi (KMUTT), the Faculty of Medicine and Public Health HRH Princess Chulabhorn College of Medical Science, Chulabhorn Royal Academy, and Chulabhorn Hospital.

I would also like to thank all people from the Big Data Experience Center and Government Big Data Institute for the opportunities and the knowledge. I have learned lots of things from there and could not publish this research without this knowledge.

Lastly, I would like to thank my family and my friends, especially the ASquare team for their support and encouragement.

CONTENTS

PAGE

CONTENTS (Cont'd)

LIST OF TABLES

LIST OF FIGURES

FIGURE **PAGE**

LIST OF SYMBOLS

SYMBOL		UNIT
C	Closeness factor	
CT	Cutoff threshold	
I	Input dataset	
K	Number of nearest neighbors to compare the closeness factors between groups in the Minority Point Selection step	
M	Number of nearest neighbors to generate synthetic instances in the Instance Synthesis step	
NT	Noise threshold	
n_{maj}	Number of majority classes in the input dataset	
n_{min}	Number of minority classes in the input dataset	
n_{gen}	Number of generated instances	
O	Set of oversampled data	
P	Probability distribution	
R	Set of real values in the Instance Synthesis step	
X	Noise filtered dataset after applying the Noise Removal step	
X_{min}	Minority instances from noise filtered dataset	
X_{maj}	Majority instances from noise filtered dataset	
τ_{maj}	Majority closeness factor of minority instance	
τ_{min}	Minority closeness factor of minority instance	
φ	Ratio of closeness factor	
β	A random of the positive values between 0 and 1	

LIST OF TECHNICAL VOCABULARY AND ABBREVIATIONS

ADASYN	=	Adaptive Synthetic
ALT	=	Alanine Aminotransferase
ALP	=	Alkaline Phosphatase
AST	=	Aspartate Aminotransferase
A-SUWO	=	Adaptive Semi-unsupervised Weighted Oversampling
AUC	=	Area Under the Curve
cGAN	=	Conditional Generative Adversarial Networks
CPU	=	Central Processing Unit
ENN	=	Edited Nearest Neighbors
FN	=	False Negative
FP	=	False Positive
GAN	=	Generative Adversarial Networks
HEOM	=	Heterogeneous Euclidean-Overlap Metric
MCHC	=	Mean Corpuscular Hemoglobin Concentration
MCV	=	Mean Corpuscular Volume
MOPH	=	Ministry of Public Health
MWMOTE	=	Majority Weighted Minority Oversampling Technique
PCA	=	Principal Component Analysis
RF	=	Random Forest
ROC	=	Receiver Operating Characteristic Curve
SIMO	=	Synthetic Informative Minority Over-sampling
SMOTE	=	Synthetic Minority Over-sampling Technique
SNOCC	=	Sigma Nearest Oversampling Based on Convex Combination
SVM	=	Support Vector Machine
SyMProD	=	Synthetic Minority Based on Probabilistic Distribution
TN	=	True Negative
TP	=	True Positive
UCI	=	University of California, Irvine
WHO	=	World Health Organization

CHAPTER 1 INTRODUCTION

1.1 Problem Statement

Nowadays, several researchers and practitioners have implemented prediction models in their applications, which support decision making and solve the current problems in various businesses. In the real-world situation, some datasets contain a skew distribution and a class imbalance that cause an imbalanced dataset problem and decrease 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 class. A class having the most observation is called a majority class whereas the other is called a minority class.

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 handle the imbalanced class problem, which are: 1) algorithm level; 2) cost-sensitive; and 3) data level method [9]. Firstly, the algorithm level method creates new classifiers or modifies existing algorithms to handle this problem. This method also adjusts a decision threshold to solve the imbalanced class. However, the change of the decision threshold produces a trade-off between precision and recall metrics, which could not improve the Area Under the Curve (AUC) metric. Secondly, the cost-sensitive method considers the higher cost for the misclassification. This method requires the domain knowledge of a dataset, which is hard to estimate an appropriate value [9], [10]. Lastly, the data level method balances the number of samples between the majority and minority classes. Any algorithm can be trained with the data after balancing, which is more versatile than other methods [11]. The techniques in this method are undersampling and oversampling. The undersampling eliminates the majority class instances whereas the oversampling produces minority class samples. Both approaches intend to balance the classes of the dataset. This paper focuses on the data level method with an oversampling technique because the undersampling may drop insight data. Besides, if the size of the minority class is small, which is the main problem of the imbalanced class, the technique will remove most of the majority class instances that will affect the prediction performance.

Moreover, the medical prediction is one of the applications that suffers from the imbalanced class problem. In this study, the authors collaborate with Chulabhorn Hospital to research the liver cancer model. Over the past decade, liver cancer is one of the most incidences worldwide. The World Health Organization (WHO) announced that in 2018, the liver cancer incidence rate was about 841,080 cases, the sixth most frequently diagnosed cancer. Furthermore, liver cancer is the fourth cause of cancer mortality that estimated 781,631 deaths occurred [12]. In Thailand, the Ministry of Public Health (MOPH) [13] reported that cancer is the first leading cause of deadly diseases, and the

mortality rate increased every year from 2007 to 2014. For this problem, the authors received patients' data from the hospital to analyze the patterns of the data and create a liver cancer prediction model by using a machine learning algorithm.

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 the 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. The method aims to eliminate the overgeneralization problem by assigning the different probabilities to the data points and solve the overlapping class problem by considering the majority and minority class distribution. In the last step, SyMProD generates synthetic instances from the minority referenced data and multiple neighbors instead of creating them along a line between two points.

Our approach is evaluated using ten public UCI datasets, six scikit-learn simulated datasets, and a liver cancer dataset from Chulabhorn Hospital. The proposed method is compared with other eight oversampling techniques. This study employs three classifiers to train the prediction model and measures the Area Under the Curve (AUC), F-Measure, and G-Mean as performance metrics. Moreover, this research focuses on the binary classification problem.

The remainder of this paper is organized as follows: Section 2 summarizes related works of the existing data level methods to handle the imbalanced class problem. Section 3 explains the detail of our proposed algorithm, its characteristics, and the evaluation method. Section 4 shows analytical results, the performance of SyMProD, and discussion. Last, the authors conclude the algorithm in Section 5.

1.2 Objectives

1. To study the concept of oversampling techniques, machine learning algorithms, and the adaptation of the oversampling technique to improve the model performance.
2. To design and develop a new oversampling technique to resolve the imbalanced dataset problem.
3. To evaluate the performance of our implemented oversampling algorithm among the general oversampling techniques.
4. To design and develop a machine learning model to predict liver cancer with demographic and laboratory tests.

1.3 Scopes

1. The solution to solve the imbalanced dataset is the oversampling technique which is the data level method.
2. The oversampling technique and prediction model are created by Python programming and the open-source library.
3. The datasets are gathered from UCI Machine Learning Repository [14]. The simulated datasets are created by the scikit-learn library [15]. The patients' data and normal persons' data are received from the Chulabhorn Hospital, Thailand.

4. The dataset in our experimental design is a binary classification dataset.

1.4 Expected Benefit

1. The imbalanced problem is solved when using the machine learning model with our oversampling technique.
2. The oversampling algorithm may increase the performance of the prediction models which may be adapted to real-world datasets.

CHAPTER 2 LITERATURE REVIEW

In this chapter, the theoretical background and the definition will be discussed, which relate to oversampling techniques to handle the imbalanced dataset problem. The remainder of this topic is organized into four parts. Firstly, details of the various oversampling techniques are mentioned, including the advantages and limitations of each procedure. Secondly, the adaptations of the oversampling method to the medical application are summarized. Thirdly, the theories of the classification models are described. Finally, the conclusion of our literature survey is presented.

2.1 Oversampling Technique

Several researchers have reported that the imbalanced class dataset which consists of an unequal distribution of classes can decrease the prediction model performance [16] [17]. There were several techniques in the data level method that handled the distribution of data before performing the prediction model [18]. 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 the majority class was equal to the minority class. However, the data for training the classifier might not be enough because the removal of the samples might eliminate valuable insight [11]. Random oversampling method duplicated minority class instances until the artificial data reached the desired size. Nevertheless, this technique had the main drawback of the overfitting problem [19]. The classifier was only learned in the specific decision region of the replicated data [20] [21].

To overcome this problem, Chawla et al. [22] proposed the oversampling technique called Synthetic Minority Oversampling Techniques (SMOTE).

2.1.1 SMOTE

SMOTE was a type of oversampling technique that generated synthetic instances to balance the minority and majority datasets. The k nearest neighbors were searched based on the feature space similarities and one neighbor was randomly selected (x') in each minority data point (x_i). SMOTE created the synthetic instances along the line between the point and its neighbor, where the position was randomly chosen between zero and one as equation (2.1).

$$x_{new} = x_i + random(0,1) * (x' - x_i) \quad (2.1)$$

The SMOTE created synthetic instances between two points, as shown in Figure 2.1. Compared to the random oversampling algorithm, the SMOTE resolved the overfitting problem and expanded the classifiers' decision region because this technique did not apply the replication method [23] [24]. However, the SMOTE assigned equal weight to every instance that caused overgeneralization [7] [25].

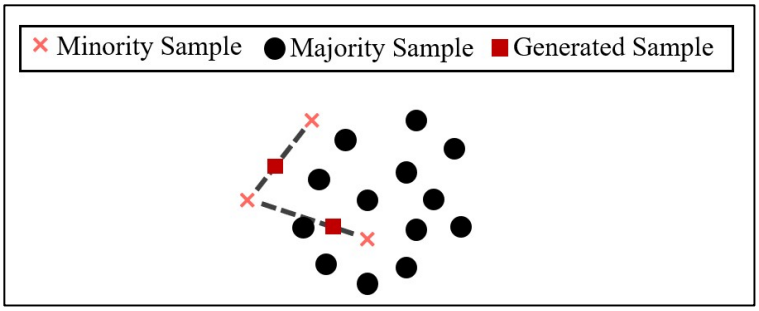


Figure 2.1 Samples were generated through oversampling with the SMOTE algorithm

Moreover, noise data could be generated that decreased the prediction model performance and caused an overlapping problem [26]. The new instances also did not provide the distribution of the data because it was created in the line between data points [20]. Several modification methods have been adapted from the SMOTE and proposed to handle the imbalanced problem, e.g., Borderline SMOTE [23], Safe-level SMOTE [27], and ADASYN [18].

2.1.2 Borderline SMOTE and Safe-Level SMOTE

Han et al. [23] 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 categorized the data points into three groups which were the safe, danger, and noise groups. For every minority class instance, the k nearest neighbors were searched and stored as Set of nearest neighbors (S_{NN}) where k was a user-defined value. Then, if all instances of S_{NN} were majority classes, the method classified the instances as the noise group. If the majority classes occurred more than half of S_{NN}, then the method classified them as the danger group. The last group, safe group, was assigned when the appearance of the minority class instances was more than the majority class instances in S_{NN}. The definition of the Borderline SMOTE is summarized in Table 2.1.

Table 2.1 The definition of the Borderline SMOTE

Category	Definition
Safe	$0 \leq m' < k/2$
Danger	$k/2 \leq m' < k$
Noise	$m' = k$

The parameter m' was defined as the amount of the majority nearest neighbors from the instance P_i. This method only focused on and oversampled the danger group to generate new instances in the decision boundary. Equation (2.2) presents the set of the danger group which is focused in the method.

$$DANGER = \{ P'_1, P'_2, \ldots, P'_n \}, 0 < n < |S_{min}| \tag{2.2}$$

P' represented the minority instances which were categorized into the danger group, n was the number of the danger group instances, and $|S_{min}|$ was the number of the minority class instances. Figure 2.2 shows the samples generated by this technique.

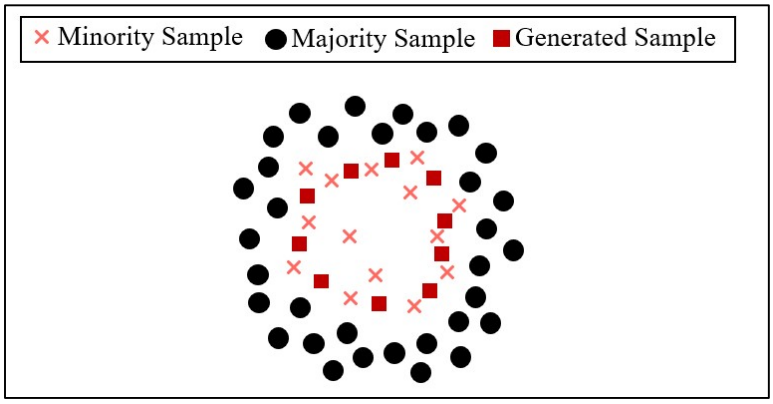


Figure 2.2 Samples were generated through oversampling with the Borderline SMOTE algorithm

This method could remove the noise genaration because it did not select the minority points in case all nearest neighbors were the majority class instances. However, the Borderline SMOTE technique had a limitation on the border finding process that the decision boundary was hard to find. Furthermore, there were some cases that the Borderline SMOTE incorrectly detected the decision boundary [28].

On the contrary, Bunkhumpornpat et al. [27] proposed the Safe-Level SMOTE to generate new points that were farther away from the borderline based on a safe level value and safe level ratio. The safe level value was the number of the minority nearest neighbors in the k nearest neighbors whereas the safe level ratio was calculated by the safe level value of the minority instance divided by the safe level value of the nearest neighbor.

The Safe-Level approach categorized the characteristics of the dataset into five cases to decide whether to generate new instances or not. Firstly, the safe level was zero, and the safe level ratio was infinity. This case was categorized as noise because both points were surrounded by the majority classes. Secondly, the safe level was not zero, and the safe level ratio was infinity. The method duplicated the minority instances because the nearest neighbor was a noise. Thirdly, the safe level ratio was one, which indicated both points were located in the minority region. The method generated the synthetic instance in the line between two points. Fourthly, the safe level ratio was higher than one. The position of the new sample which was generated by the method was closer to the selected minority point than the neighbor point. In the last case, the safe level ratio was lower than one. The synthetic instance was generated closer to the nearest neighbor. Nonetheless, Nekooeimehr and Lai-Yuen [26] reported that the method could cause an overfitting problem. Figure 2.3 illustrates the samples generated by the Safe-Level algorithm.

2.1.3 ADASYN

He et al. [18] developed an Adaptive Synthesis (ADASYN) Sampling Approach to resolve the overgeneralization of the SMOTE problem. The method applied the concept of weighting the minority class instance based on the ratio of the majority neighbors. Then, the new samples were employed by the SMOTE algorithm. The algorithm could decide how many samples should be produced for the minority class and focus on hard to learn data. Firstly, the number of synthetic instances, N_{syn}, was calculated by equation (2.3).

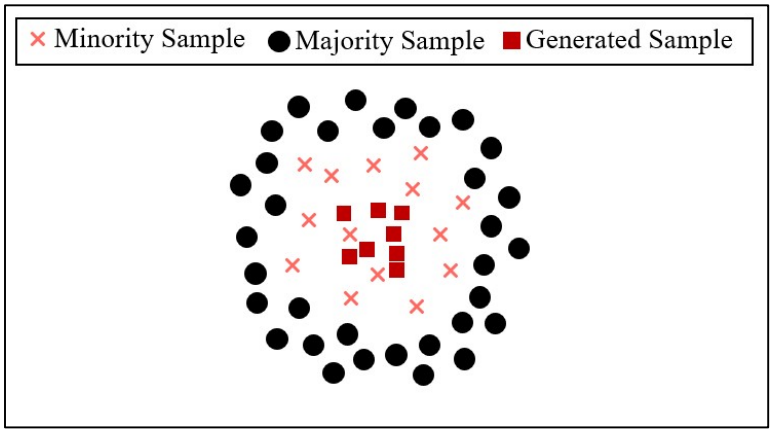


Figure 2.3 Samples were generated through oversampling with the Safe-Level SMOTE algorithm

$$N_{syn} = \left(\left| S_{maj} \right| - |S_{min}|\right) * \beta \tag{2.3}$$

The variables S_{maj} and S_{min} were the sets of majority classes and minority classes, respectively, β was a parameter to specify the balance between the majority and minority classes, which $\beta \in [0,1]$. If β was set to 1, then the number of the minority classes will be equal to the number of the majority classes after oversampling the data. The method searched the k nearest neighbor in every minority class instance to calculate the ratio, r_i, of the majority neighbor over the nearest neighbor as equation (2.4) and normalized $\hat{r_i}$ as equation (2.5).

$$r_i = \frac{\Delta_i}{K}, i = 1, \dots, |S_{min}| \tag{2.4}$$

$$\hat{r_i} = \frac{r_i}{\sum_{i=1}^{|S_{min}|} r_i}, i = 1, \dots, |S_{min}| \tag{2.5}$$

Δ_i was the number of majority nearest neighbors in each minority class instance. k was the number of neighbors. If the ratio of the majority neighbor was high, then the weight of the minority dataset would be high. After that, the number of the generated data of each minority class, g, depended on the weight that was defined as equation (2.6).

$$g_i = \hat{r_i} * N_{syn}, i = 1, \dots, |S_{min}| \tag{2.6}$$

Finally, the ADASYN applied the concept of the SMOTE in equation (2.1) to generate synthetic data, as shown in Figure 2.4. The minority instances with a higher number of the majority nearest neighbors were a higher possibility to be selected for the instance synthesis process.

The main differences between the ADASYN and the SMOTE algorithms were the weight of each point. The SMOTE assigned equal weight to every data point that caused the

overgeneralization whereas the ADASYN applied the density distribution to create a different number of new samples in each minority class. However, the ADASYN algorithm also generated noisy instances in the same way as the SMOTE technique [29]. The weighting by using the density caused the problem on the minority class instance located on the decision boundary. This point was the high possibility of the zero weight if the neighbors of this point were minority datasets. Then, new points were not generated by this data point, even if this point was on the boundary decision [26] [28].

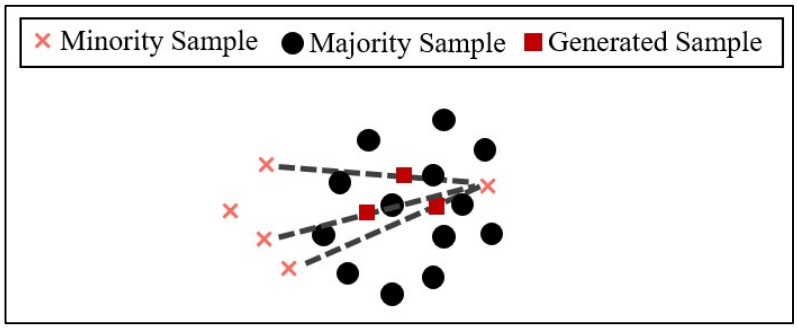


Figure 2.4 Samples were generated through oversampling with the Adaptive Synthesis algorithm

2.1.4 K-means SMOTE

In terms of the clustering techniques, several authors have applied the clustering algorithm to solve the imbalanced class problem. Cluster-SMOTE [30] utilized the clustering to group the minority classes 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 to improve the performance of the oversampling technique.

Firstly, the method clustered the data using Euclidean distance and chose the cluster to generate new samples based on the class ratio in each group. If the number of majority instances in the cluster was higher than the number of minority instances, then the cluster was not selected to produce samples. In each selected cluster, the method computed a sampling weight to define the number of the generated samples. The sparse group was likely to create new samples more than the dense based on the sampling weight. The density factor and sparsity factor are described in equation (2.7) and (2.8), respectively.

$$Density\ factor\ (f) = \frac{minority\ count\ (f)}{average\ minority\ distance^{m}} \tag{2.7}$$

$$Sparsity\ factor\ (f) = \frac{1}{Density\ factor(f)} \tag{2.8}$$

The variable f was the filtered cluster from the previous step, and m was the number of features. If the cluster was dense, then the average distance would be low, which increased the value of the density factor. The method converted the density factor to the sparsity factor and calculated the sampling weight as in equation (2.9).

$$Sampling\ weights\ (f) = \frac{Sparsity\ factor(f)}{\sum_{i}^{f \in filtered\ clustered} sparsity\ factor(i)} \tag{2.9}$$

Lastly, synthetic instances were generated by the SMOTE that the nearest neighbor had to be located in the same cluster, as shown in equation (2.1).

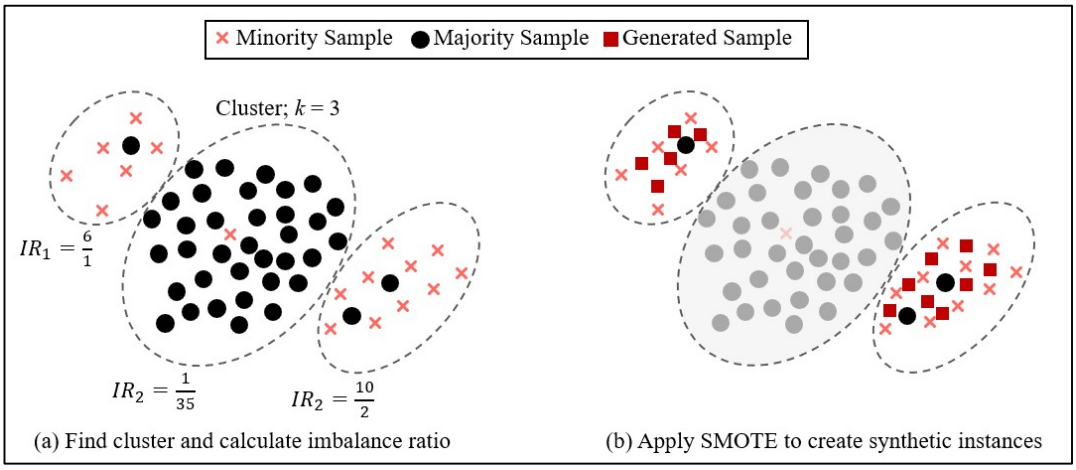


Figure 2.5 Samples were generated through oversampling with the k-means SMOTE algorithm

Santos et al. [4] proposed the clustering technique to cluster data and balance the class in each group. After that, they sampled the data in each group and merged them to create the prediction model. However, this technique might not solve the problem because the data used for training the model might not include the original data which were the practical insight, and the sampled data might contain an imbalanced dataset.

2.1.5 Adaptive semi-unsupervised weighted oversampling (A-SUWO)

Nekooeimehr and Lai-Yuen [26] presented adaptive semi-unsupervised weighted oversampling (A-SUWO) to solve the binary imbalanced dataset problem. This technique was another approach in the clustering algorithm. The method consisted of three main steps. Firstly, Semi-unsupervised clustering, the method applied Agglomerative Complete-Linkage Hierarchical Clustering to group the minority classes and avoid the majority class instances located between minority points. The main goal was to reduce the overlapping problems between both classes. Moreover, the process removed the noisy cases based on the nearest neighbors. In each minority point, if all k-nearest neighbors were the majority classes, the method categorized that point as noise and removed it from the dataset.

Secondly, Adaptive sub-cluster sizing, the method determined the number of synthetic samples based on the complexity of classification. The cluster with a higher misclassification rate received more attention and created more synthetic instances to that cluster to provide more information. The misclassification rate was calculated using the classifier and cross-validation.

Lastly, Synthetic instance generation, the authors assigned a weight in each instance to generate points, which depended on the distance to the nearest neighbors of the majority classes. The SMOTE was used to generate new samples after selection based on these weights.

This oversampling technique could handle the overlapping problem because the synthetic instances were created inside the cluster. However, the execution time in this technique might be higher than other methods because this method trained the prediction model to calculate the misclassification error whereas other conventional methods did not execute any classifiers.

2.1.6 Majority Weighted Minority Oversampling Technique (MWMOTE)

Barua et al. [28] developed the Majority Weighted Minority Oversampling Technique (MWMOTE) to improve the decision boundary detection to solve the imbalanced class problem. This technique adapted the clustering and SMOTE algorithms as synthetic data were produced inside their cluster.

MWMOTE consisted of three important parts. Firstly, it identified which instances of the minority class were hard to learn and tried to find the decision boundary based on the nearest neighbors of the majority and minority instances. Figure 2.6 presents the process of minority selection and decision boundary detection. The method found the borderline majority from the minority instance and combined all points to create the set S_{bmaj}, as shown in Figure 2.6(b). In each majority point in S_{bmaj}, it found the minority nearest neighbors and determined them as the informative minority points, S_{imin}, as shown in Figure 2.6(c). This step aimed to improve the detection of the decision boundary. Moreover, the method removed the minority instances if all nearest neighbors were majority instances.

Secondly, the method indicated the weight assigned to the minority class instances based on the closeness of the decision boundary and density factor. A higher weight implied the point was more important than the others, and the method focused on this point for the generation process.

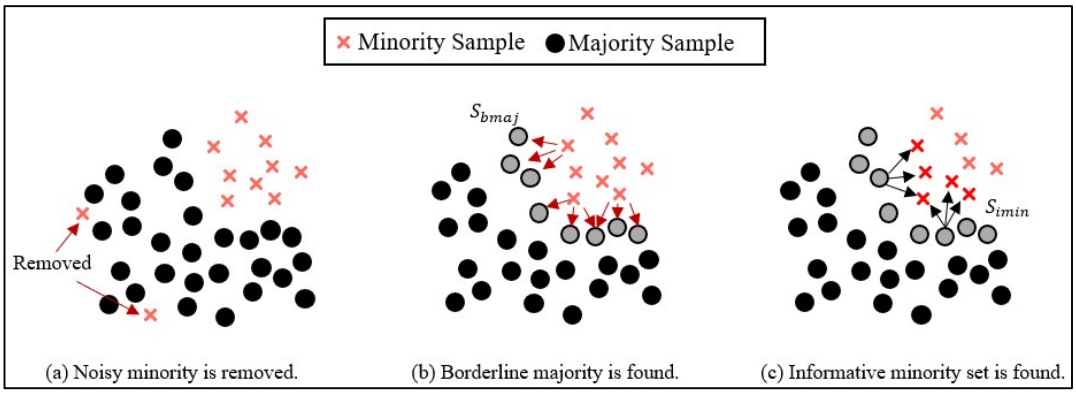


Figure 2.6 The MWMOTE algorithm detected the decision boundary

Thirdly, the synthesized process adapted the clustering technique to group the minority class instances before generating new samples to avoid the overlapping and noisy data

generation problems. However, the authors reported that their technique was worse than the SMOTE and the ADASYN in case of a recall metric because the traditional methods created the erroneous data points, which produced the wider minority class region. Moreover, the MWMOTE did not support the nominal variable [28]. Chennai also reported that the MWMOTE was sensitive to noise and outlier [31]. Nekooeimehr and Lai-Yuen [26] reported that the MWMOTE did not detect the minority instances that were far from the boundary region even if they consisted of valuable information for the classifier.

2.1.7 Other oversampling techniques

Several researchers proposed various kinds of oversampling techniques as solutions to the imbalanced problem such as SIMO [11] which applied the Support Vector Machine to detect valuable points, cGAN [32] which used deep learning based, RACOG [10] which utilized the joint distribution and Gibbs sampling, and SMOTEBoost [33] which applied the boosting algorithm.

2.1.7.1 A synthetic informative minority over-sampling (SIMO)

Piri et al. [11] proposed an oversampling technique using the Support Vector Machine (SVM) algorithm to define a decision boundary between two classes. Firstly, SIMO applied the initial SVM model to the original imbalanced dataset to indicate the important points. Then, the minority points located close to the decision boundary were oversampled. The method worked iteratively, which created a new SVM model and tried the various number of generated samples. In each iteration, the G-mean metric was collected to find the highest value as the final oversampled dataset for the training procedure in the final model. Moreover, the authors also developed the W-SIMO which considered the incorrectly classified points as weight. The method aimed to generate new samples on the points that had a high misclassification rate. Then, the SVM was developed iteratively and found the highest G-Mean.

The main difference between this algorithm and other techniques was the number of the minority generations. This technique tried to generate the least amount of minority points, which decreased the transformation of the data distribution.

2.1.7.2 Conditional generative adversarial networks (cGAN)

Nowadays, Deep Learning has a high impact on the prediction model classification and can increase the performance of the prediction model in various domain knowledge, especially image processing. Goodfellow et al. [34] introduced a generative adversarial network (GAN) to create new data, e.g., images which were applied in neural networks and game theory. In terms of oversampling data, Douza and Bacao [32] modified the GAN technique by conditioning procedures on the external information. There were two models in GAN, which were generator (G) to generate new samples and discriminator (D) to detect the source of the dataset. Both networks tried to beat each other in order to improve the overall performance. cGAN modified the value function from the traditional GAN by including the additional data. The functions of the generator and discriminator models are shown in equation (2.10) and (2.11), respectively.

$$G: Z \times Y \rightarrow X \tag{2.10}$$

$$D: X \times Y \rightarrow [0,1] \tag{2.11}$$

$$min_G max_D V(D,G) = E_D + E_G \tag{2.12}$$

where

$$E_D = E_{x,\ y\sim p_{data}(x,y)} [\log D\ (x,y)] \tag{2.13}$$

$$E_G = E_{z\sim p_z(z),\ y\sim p(y)} [\log(1 - D(G(z,y),y))] \tag{2.14}$$

Z was the noise space, Y was the additional information from the training data, and X was the data space. The generator tried to capture the data distribution whereas the discriminator estimated the probability of the data source that was brought from the real data instead of the generator function. The cost functions of cGAN for updating the weight of generator, J_D, and discriminator, J_G, were proposed as follows:

$$J_D = -\frac{1}{2m}\left(\sum_{i=1}^{m} \log D(x_i,y_i) + \sum_{i=1}^{m} \log\left(1 - D(G(z_i,y_i),y_i)\right)\right) \tag{2.15}$$

$$J_G = -\frac{1}{m}\sum_{i=1}^{m} \log D(G(z_i,y_i),y_i) \tag{2.16}$$

The variable m was the number of the training samples. The authors claimed that this technique improved the model performance. However, the neural networks needed enough training sets to train the model. If the minority class was rare, then the model might not be effective. Moreover, the training time was longer than other oversampling techniques because of the complexity of the networks. Nevertheless, if the data were enough and the model was trained nicely, then the performance of the neural networks would be higher than the conventional techniques. It also can work with the noise data.

2.1.7.3 Sampling with data preprocessing technique

Several pieces of research applied the preprocessing technique to handle the imbalanced dataset. Tomek Link [6] removed noises in the data by selecting the nearest pair of the data. If the class of the pair was different, then it could be borderline, or one of the two instances was noise. The majority class instance or both could be removed to clean up the data. Figure 2.7 represents the dataset after being removed by the Tomek Link technique. Besides, Edited Nearest Neighbors (ENN) eliminated the data using the K-nearest neighbor. If the class neighbor was different from the data point at least two of three neighbors, then the data point would be removed.

Furthermore, in the real-world application, the problem of the missing value affected the prediction model performance. There were several solutions to solve this problem, for example, removing rows and inserting value procedure. The removing rows method caused the loss of insight problems, especially the real-world dataset that the user might not fill in every data. Thus, filling artificial values into the null record was appropriate to

resolve the missing value problem. The Heterogeneous Euclidean-Overlap Metric (HEOM) technique [21] handled this problem by averaging the value of neighbor data to fill in the null value. This technique worked with continuous and discrete variables.

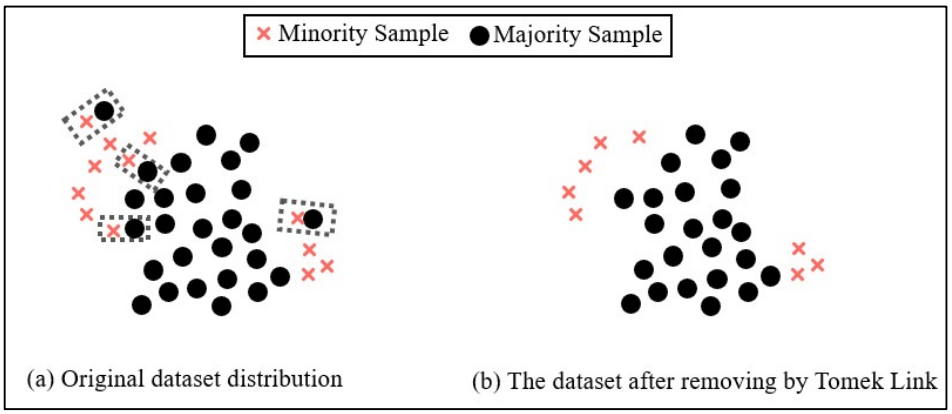


Figure 2.7 Data preprocessing using Tomek Link method

2.2 The Adaptation of Oversampling Technique to Medical Application

The oversampling technique can be adapted to work with several applications to balance the distribution and create a better prediction model, e.g., medical dataset. Fallahi and Jafari proposed the breast cancer detection system using a relief algorithm to find the related features. The method removed missing values, applied the SMOTE to balance the data, and utilized the Bayesian network to create the model. The authors presented that the SMOTE could improve the accuracy of the model [35]. Wang and Adrian proposed the SMOTE technique to balance the data with the Artificial Immune Recognition System to predict breast cancer. They claimed that the combination of the oversampling methods and classifiers increased the performance of prediction because the data were balanced [17]. Naseriparsa and Kashani developed the system to classify lung cancer. The method employed the SMOTE and the principal component analysis (PCA) to determine the related factors, reduce the dimensions of data without much loss of information, and balance the distribution. The performance of the model before using the oversampling was 80% accuracy whereas it reached 97% accuracy after the SMOTE was applied. The method also raised the precision and recall metrics [16]. Sug proposed the liver disease prediction using the oversampling technique. The accuracy of the model was 0.4% better than the model that did not perform the oversampling based on seven datasets and decision tree algorithm [36]. Furthermore, Lokanayaki and Malathi analyzed the balanced dataset in liver disease with several oversampling techniques. The SMOTE provided a better result than the Random Oversampling and Undersampling techniques [37]. Santos et al. developed the liver cancer prediction by using new cluster-based oversampling with the imputation method. This technique had a better result in accuracy and area under the curve, which showed the oversampling technique improved the prediction model [4].

2.3 Theory of Classification Model

This section presents a survey of the classification model which was used for the assessment of oversampling performance. In each experimental design, the method applied the same classifiers which were random forest, support vector machine, and logistic regression, with the same parameters. A description and theory of each classifier are presented below.

2.3.1 Random Forest Classification

Random Forest is an algorithm that applies multiple decision trees to vote for the final result, which is called the bootstrap aggregation technique. In each decision tree, the algorithm creates a decision rule to classify the new inputs. There are multiple criteria to calculate the decision rule and construct the decision tree, e.g., entropy and information gain.

Entropy is a measure of the variable homogeneity of the dataset. Equation (2.17) presents the calculation of entropy. If the dataset consists of one class which is the purity dataset, then the entropy of this dataset is zero. On the contrary, the dataset of a mixed class has high entropy.

$$E(S) = \sum_{i=1}^{c}(-p_i log_2 p_i) \tag{2.17}$$

E is the entropy of set S, c is the number of classes, and p_i is the probability of randomly selecting an element of class i.

The objective of the decision rule is to separate the classes of the dataset by finding the best features to split the data. Information Gain (IG) is a metric to find the difference in the entropy of each branch from the original entropy that is calculated using equation (2.18).

$$Information\ Gain(T, X) = Entropy(T) - Entropy(T, X) \tag{2.18}$$

T is the dataset before being split by the feature, and X is the feature to consider. If the feature can completely distinguish the target variable, then the information gain is one, and the entropy of the data after being split in each group is zero. The method will select this feature as a root node. On the contrary, if the feature does not separate the target group, then the information gain is low. Moreover, the method calculates the leaf node recursively and creates the rule until the value reaches the parameter setting, e.g., the number of tree depth.

In terms of the random forest algorithm, it produces multiple decision trees and selects results based on the highest voting. In each tree of the random forest, the algorithm samples the record and column. This situation causes a higher variation among the trees, which may improve the performance because the method tries to create the uncorrelation of trees. The random forest performs the following steps:

1. Randomly select features and records from the total dataset.
2. Calculate the nodes among the selected features by using the splitting criteria, e.g., information gain.
3. Repeat steps 1. and 2. until the number of depth trees reaches the desired size which is set as a parameter of the algorithm. Moreover, the algorithm may have other parameters, e.g., min impurity decrease, max features, etc. as in the sklearn library.

After the random forest model is created, new data can be classified by the following steps:

1. Choose features of the new data, apply them to the rule of each decision tree, and collect the result of each tree.
2. Consider the highest frequency results from all decision trees as the output result of the random forest algorithm.

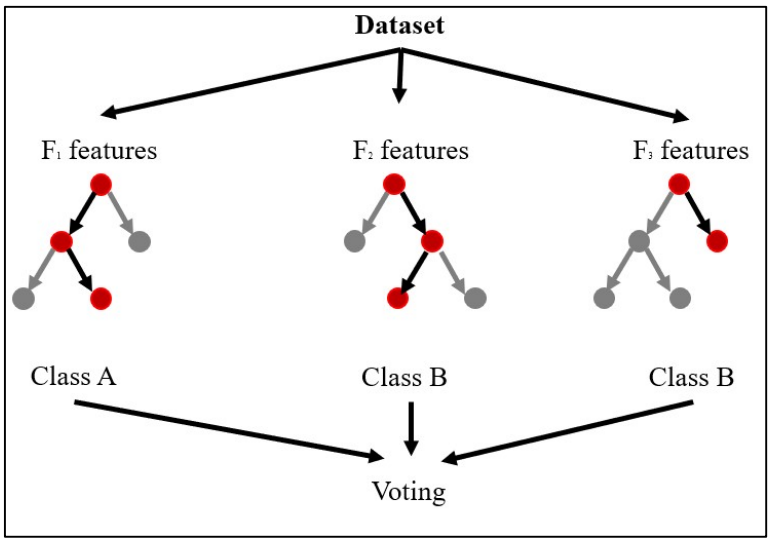


Figure 2.8 The simplified random forest algorithm

2.3.2 Support Vector Machine Classification

Support Vector Machine is a supervised machine learning that applies a hyperplane to separate the target variable. The algorithm calculates a hyperplane that maximizes a margin between two classes. The hyperplane is called the decision boundary which can be a linear plane or a radial basis function based on the kernel. The appropriate kernel depends on the characteristics of the dataset. If the target variable of the dataset can be separated by a single line, then the linear kernel can be applied to classify the data. However, some datasets cannot be solved by a linear kernel, for example, a non-linear dataset, a polynomial kernel, can be employed to fit the dataset.

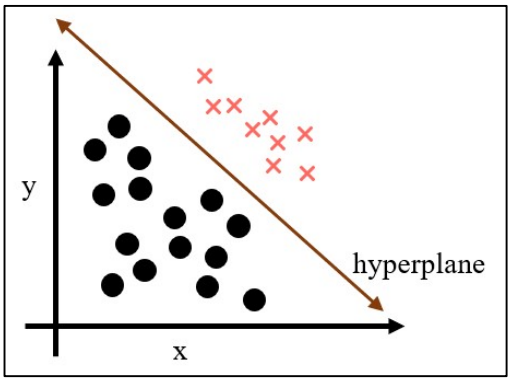


Figure 2.9 The simplified support vector machine algorithm

Moreover, the regularization is applied to optimize the bias-variance tradeoff to increase the performance of the classification model. In terms of the sklearn library, a large value of C which is the regularization parameter can decrease a bias. The hyperplane tries to classify all points correctly that can cause an overfitting problem. On the contrary, a small value of C can solve the overfitting problem, but the performance of the model will be low, which is a high bias. The default value of the regularization of the sklearn library is 1.0. Figure 2.9 presents a simple linear plane on the SVM algorithm.

2.3.3 Logistic Regression Classification

Logistic Regression is a predictive model which provides a probability of the target variable. This method employs linear regression before applying the sigmoid function to classify the results. The linear regression intends to predict the numerical variable whereas the logistic regression predicts the nominal class, e.g., binary classification. Equations (2.19) and (2.20) present the workflow of the logistic regression algorithm.

$$Z = WX + B \tag{2.19}$$

$$y = \frac{1}{1 + e^{-z}} \tag{2.20}$$

W is a weight in each feature input considered as a coefficient, y is an output, and B is a constant. Z is considered a linear function which takes the input to find the best parameters of the model based on cost-function. Furthermore, the method needs a sigmoid function to convert the result of the linear function into the range of zero to one as output for the target variable.

2.4 Literature Review Conclusion

In this section, the techniques that relate to our study are summarized for more understanding from the main topics. Nowadays, the imbalanced class is a problem that affects a prediction model, e.g., fraud detection, medical prediction, and oil spill detection. The model of the imbalanced class may only predict the majority class and cannot adapt to the real-world applications although it has high accuracy. Our work tries to solve this problem with the oversampling technique.

In terms of the oversampling technique, several pieces of research handled the imbalanced dataset by producing new instances to improve the classification performance with several methods. 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 consider the distribution of data [3] [20].

To overcome these problems, the authors have proposed 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 aims to create minority instances that cover the minority distribution. The data from our method are generated based on the probability distribution. Moreover, our proposed algorithm expands the boundaries of the minority class, avoids the overlapping region, removes the overfitting and overgeneralization problems, and may increase the prediction model performance.

To assess the performance of our algorithm, the proposed method is evaluated by ten UCI imbalanced datasets, six simulated imbalanced datasets, and a liver cancer dataset from Chulabhorn Hospital. Our oversampling algorithm is compared to other eight conventional oversampling techniques, i.e., SMOTE, ADASYN, k-means SMOTE, MWMOTE, Safe-Level SMOTE, Cluster SMOTE, Borderline SMOTE, and A-SUWO techniques, which are the data level methods to resolve the imbalanced dataset. Three metrics are used with the stratified five-fold cross-validation.

CHAPTER 3 DESIGN AND METHODOLOGY

This chapter describes the main workflow of the prediction model, the algorithm of our oversampling technique which is SyMProD, the evaluation metric, and the experimental design. Firstly, the overall framework of the prediction model and the detail of each procedure are presented. Secondly, the design of our oversampling technique is proposed to increase the prediction model performance. Thirdly, the evaluations of the experiments are explained to show the meaning and the formula in terms of algorithm and model performance. Lastly, the authors present the experimental design and statistical method in this research.

3.1 Overall Framework

Figure 3.1 shows the diagram of the prediction model workflow. Each task in the sequence intends to increase the performance that is the standard method in the data science process. However, imbalanced data is the problem that needs to be solved. In our method, the oversampling technique was applied and integrated before training the prediction model.

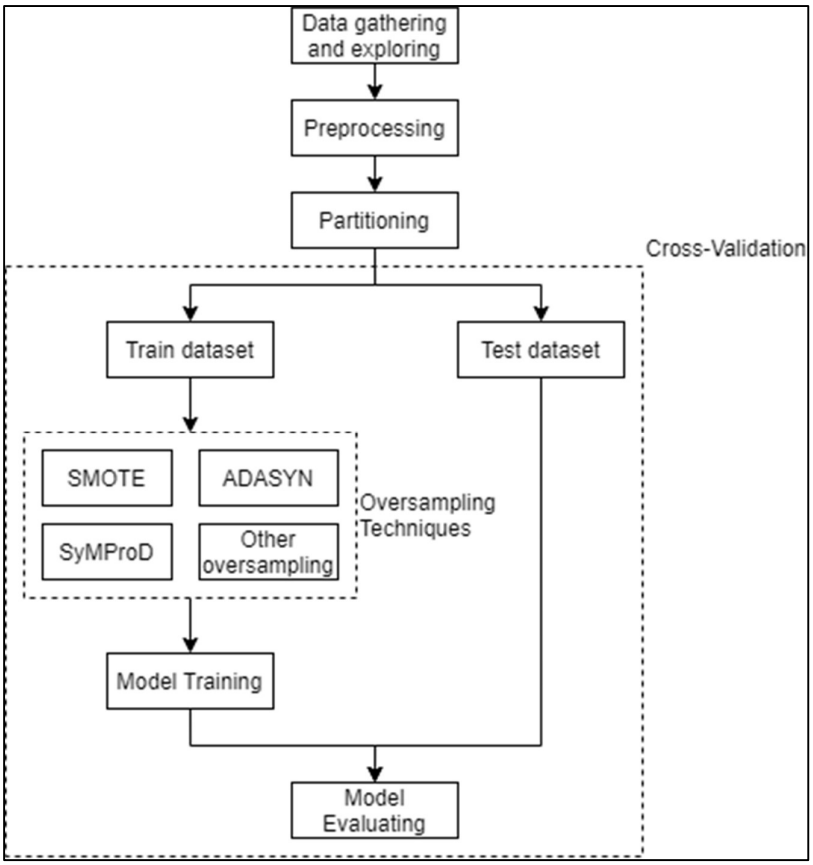


Figure 3.1 The diagram of prediction model workflow

3.1.1 Data Gathering and Exploring

This research selected 17 imbalanced datasets, i.e., ten datasets from UCI Machine Learning Repository [14], six datasets from the scikit-learn library [15], and one dataset from Chulabhorn Hospital. Table 3.1 summarizes the characteristics of these datasets. The imbalanced datasets from UCI consisted of several imbalanced ratios and multiple topics, e.g., weather dataset and healthcare dataset. Additionally, the six artificial datasets were a binary class dataset of 1000 samples generated by using the make_classification function from a scikit-learn library with different imbalanced ratios, from two to eight imbalanced ratios. This study labeled these artificial datasets as simulated datasets. The last dataset, liver cancer dataset, was gathered from Chulabhorn Hospital. The authors collaborated with the hospital and received the 589 patient and 3,709 normal people data records to generate the prediction model and analysis. The main features in the experiment of the liver cancer prediction were demographics and laboratory tests, which were Basophil, Eosinophil, Hematocrit, MCHC, MCV, Monocyte, Neutrophil, Platelet, ALT, AST, Albumin, ALP, and Total Bilirubin.

Table 3.1 The characteristics of datasets

Dataset	**Minority Class**	**Majority Class**	**# of vari-ables**	**# of rows**	**# of minor-ity**	**# of major-ity**	**Imba-lanced Ratio**
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
Libras	Class "1", "2", and "3"	All other	90	360	72	288	4.00
Segment	Class "WINDOW"	All other	19	2310	330	1980	6.00
Vehicle	Class "van"	All other	18	846	199	647	3.25
Pima	Class "1"	Class "0"	8	768	268	500	1.87
Liver cancer	Class "Patient"	Class "Normal"	14	4298	589	3709	6.30
Simulated1	Class "1"	Class "0"	2	1000	105	895	8.52
Simulated2	Class "1"	Class "0"	2	1000	205	795	3.87
Simulated3	Class "1"	Class "0"	2	1000	304	696	2.28
Simulated4	Class "1"	Class "0"	10	1000	104	896	8.61
Simulated5	Class "1"	Class "0"	10	1000	203	797	3.92
Simulated6	Class "1"	Class "0"	10	1000	300	700	2.33

The description of ten datasets from the UCI repository are described below:

1. Balanced dataset: The goal is to classify the balance scales of the psychological experiments, which consist of the left, right, and balanced scales.

2. Ecoli dataset: The localization site of the Ecoli is the result of the model, that contains different sites. The minority class is the Ecoli that is located in the periplasm whereas the other locations are cytoplasm, inner membrane, and outer membrane.
3. Glass dataset: The type of float glass is determined in this dataset by considering several features, e.g., Magnesium, Aluminum.
4. Haberman dataset: The goal is to classify the survival of patients after undergoing surgery for breast cancer into two fields which are survival after five years or death within five years.
5. Heart dataset: The goal is to analyze the presence of heart disease in patients from the demographics and laboratory tests.
6. Ionosphere dataset: The target variable is the type of ionosphere which is good or bad based on the signal from the radar in a different situation.
7. Libras dataset: The goal is to classify the sign languages from several classes based on the hand movement.
8. Segment dataset: The purpose is to distinguish the image from seven outdoor types. The minority class is the window, whereas the others are the brick face, sky, foliage, cement, path, and grass.
9. Vehicle dataset: The purpose is to classify the types of vehicles from the features extracted from the silhouette.
10. Pima dataset: The goal is to diagnose a liver patient or normal person from the blood test.

3.1.2 Preprocessing

In terms of the UCI datasets, this study converted the multi-class into binary classification by labeling a particular class as a minority and merging the others as a majority class. Table 3.1 presents the labels of the minority and majority classes. In terms of the Chulabhorn Hospital's dataset, this study received the laboratory test results from the hospital that were collected in the report format which was the unreliable form to be analyzed by the machine learning model. Consequently, the authors transformed the report format into the table format and performed the integration of multiple tables into one table for the analysis. Moreover, this study performed feature selection and applied the HEOM method for imputation to decrease the distortion of data. The average value of the neighbors replaced the missing value in each record.

3.1.3 Partitioning

In the general machine learning process, it splits data into training and testing datasets to create the prediction model and evaluate it. However, this technique causes the bias because the training set has only a specific group. Then, the overfitting problem will occur. To handle this problem and generalize the model prediction, this study applied stratified five-fold cross-validation by randomly splitting the data. Each fold had approximately an equal number of the sample data. Then, this study trained the model five times so that each iteration had different training and testing datasets. The training dataset was selected from four out of five folds, and another fold was a testing set. Moreover, the oversampling technique only performed on the training dataset, whereas the testing dataset was used to evaluate the performance.

Lastly, the performance of each iteration will be averaged and represent the model production. The cross-validation avoids the undesirable shift by preserving the original distribution between the classes in the training and testing sets. However, this process has

a drawback which is time-consuming. The model has to train five times more than the train/test split method, but it is useful and worth to adapt this technique to assess the overfitting problem.

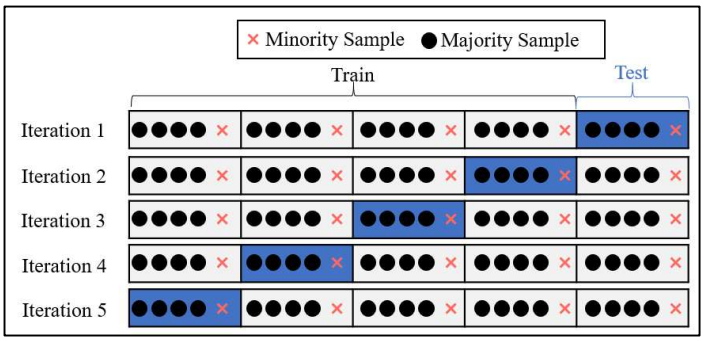


Figure 3.2 Stratified five-fold cross-validation procedure

3.1.4 Oversampling

To handle the imbalanced dataset problem, this study performed the oversampling technique to increase minority class samples before training the classifier. This research implemented an oversampling technique called Synthetic Minority Based on Probabilistic Distribution (SyMProD) to balance skewed data. Moreover, our proposed method was compared with eight conventional oversampling techniques, i.e., SMOTE [22], Borderline SMOTE [23], Safe-Level SMOTE (SLSMOTE) [27], Cluster SMOTE [30], MWMOTE [28], k-means SMOTE [9], A-SUWO [26], and ADASYN [18]. The standard oversampling algorithms were implemented by imblearn [38] and smote-variant libraries [39] based on Python. This section describes the details of our oversampling technique.

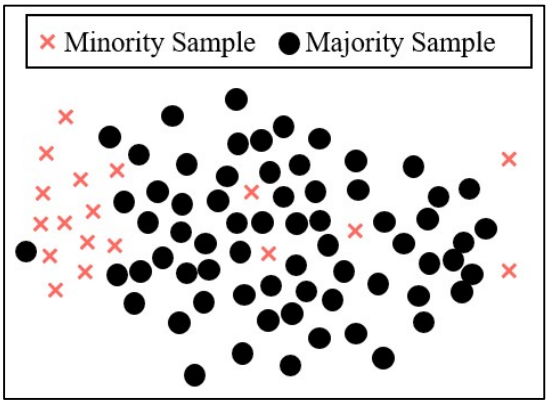


Figure 3.3 The sample dataset for visualizing the framework of SyMProD

3.1.4.1 Synthetic Minority Based on Probabilistic Distribution (SyMProD) Oversampling Framework

Our proposed method applied a concept of the probability distribution to assign a probability to each minority instance. The main purpose was to balance the skewed data. SyMProD consisted of three main steps: i) Noise Removal; ii) Minority Point Selection; and iii) Instance Synthesis. Algorithm 3.1 shows the process of the oversampling framework. Firstly, the method calculated the number of the generated instances, n_{gen}, by

determining the difference between the majority and minority classes before employing the oversampling technique. The n_{gen} minority instances were created to balance the class.

Algorithm 3.1 : Synthetic Minority Based on Probabilistic Distribution (SyMProD) Oversampling

---

Inputs : **I** : Original dataset as input
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 factors between groups
M : Number of nearest neighbors to generate synthetic instances
Output : **O** : Oversampled data after rebalancing
Procedure:

1) Calculate the number of the generated instances based on an imbalanced gap, n_{gen}, according to $n_{gen} = n_{maj} - n_{min}$, where n_{maj} is the number of the majority class instances and n_{min} is the number of the minority class instances.

i) Noise Removal Step: Remove the outliers to reduce the possibility of noise generation.

2) Apply the Z-score normalization to the minority and majority classes of the original dataset, **I**.
3) Remove the instances if an absolute value of normalization is higher than *NT* and collect the valid instances in the noise filtered dataset, **X**.

ii) Minority Point Selection Step: Select the minority instances based on the probability distribution for the generating process.

4) Calculate the Euclidean distance in each instance using equation (3.2).
5) Determine the closeness factor and normalize it using equation (3.3).
6) For all minority instances, find *K* minority nearest neighbors and *K* majority nearest neighbors to define the minority closeness factor, τ_{min}, and the majority closeness factor, τ_{maj}, using equation (3.4) and equation (3.5), respectively.
7) Exclude the minority instances if the minority closeness factor is less than the majority closeness factor multiplied by *CT*.
8) Define a ratio of the closeness factor, φ, using equation (3.6).
9) Transform φ into the probability distribution, *P*, in each minority instance using equation (3.7).
10) Select n_{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 the selected minority point and minority nearest neighbors in **R**.
13) Generate a random positive vector of *M+1* dimension, where the value is between 0 and 1.
13) Generate a synthetic instance from the selected instance and nearest neighbors by set **R** and the normalization of the random positive vector and the probability distribution, as shown in equation (3.8).
14) Add a generated instance into the oversampled dataset, **O**.

Return **O**

3.1.4.1.1 Noise Removal

The first step of our approach was to remove the outliers from the data. The synthetic instance could decrease the model performance if it was generated from the noisy data. This step tried to reduce the chance of noise generation. Our method applied the Z-score normalization to both classes from the original dataset, **I**, and indicated a noise by comparing it with a noise threshold, *NT*. The method selected a point if the absolute value of the normalization was lower than *NT,* as shown in Figure 3.4. The non-outlier data were added into a set of the noise filtered dataset, **X**.

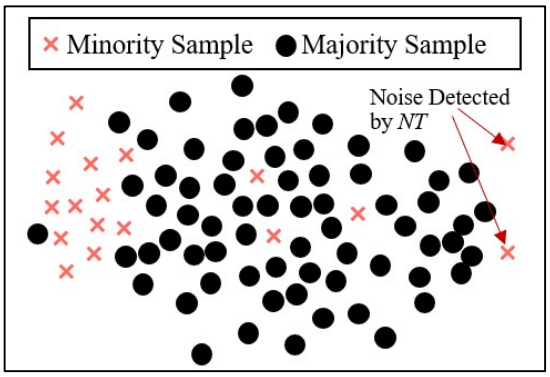


Figure 3.4 The noise removal step of SyMProD algorithm

3.1.4.1.2 Minority Point Selection

This step selected the referenced minority instances to generate new samples in the next step based on the probability distribution. The number of the selected instances was the difference between the number of the majority and minority classes from the original dataset, n_{gen}. The method considered the regions 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.

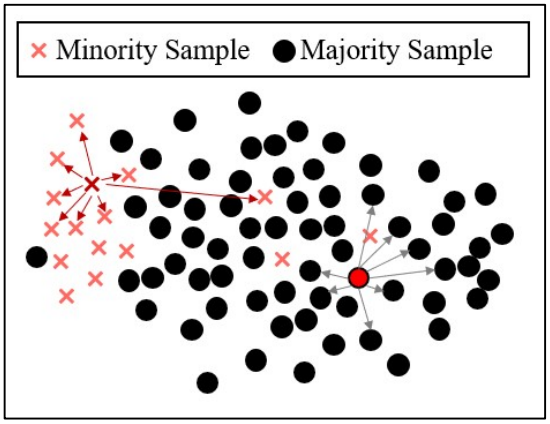


Figure 3.5 The SyMProD calculates closeness factor of each instance

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

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

$$d(p,q) = \sqrt{\sum_{i=1}^{N}(p(i) - q(i))^2} \tag{3.1}$$

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

$$DIST(X_{min}(i)) = \sum_{j=1}^{n_{min}} d(X_{min}(i), X_{min}(j)) \tag{3.2}$$

The method defined the closeness factor, C, in each minority instance using equation (3.3). Our proposed algorithm assigned a higher value of the closeness factor to the point that was close to other points in the same class. Then, the method applied the min-max normalization to scale the data. For the majority group, the approach also assigned the distance and closeness factors among the majority classes to determine the majority distribution. Figure 3.5 illustrates the sample connection among the points. In summary, the closeness factor was calculated from an instance to all points in the same class.

$$C(X_{min}(i)) = \frac{1}{Dist(X_{min}(i))} \tag{3.3}$$

The method selected the minority instances based on the minority class distribution. Moreover, the proposed step avoided the overlapping problem by deciding whether to choose the minority instances based on the minority and majority class distribution. For each minority instance, $\mathbf{X_{min}(i)}$, the method found the *K* nearest neighbors of both classes. The *K* majority and minority nearest neighbors of each minority instance, $\mathbf{X_{min}(i)}$, were collected in a set of $\mathbf{S_{maj}(i)}$ and $\mathbf{S_{min}(i)}$, respectively.

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

To reduce the possibility of the overlapping problem, the method defined the minority and majority closeness factors of the minority instance, τ_{min} as in equation (3.4) and τ_{maj}

as in equation (3.5), respectively. The main purpose of this step was to compare the values between both classes. In equations (3.4) and (3.5), the proposed algorithm also applied the distance from the nearest neighbor to the referenced minority instance because the different neighbors of the referenced point might have the same closeness factor with different distances. Thus, the method assigned a higher value to the nearer point than the further point, as shown in Figure 3.6.

$$\tau_{min}(i) = \sum_{j=1}^{K} \frac{C(S_{min}(i,j))}{DIST\left(X_{min}(i), S_{min}(i,j)\right)} \tag{3.4}$$

$$\tau_{maj}(i) = \sum_{j=1}^{K} \frac{C(S_{maj}(i,j))}{DIST\left(X_{min}(i), S_{maj}(i,j)\right)} \tag{3.5}$$

This step defined the cutoff threshold, *CT*, as a threshold to select the minority instances located in the minority distribution. The method applied the minority and majority closeness factors, τ_{min} and τ_{maj}, to decide whether the method excluded the minority instances out of $\mathbf{X_{min}}$ or not. In each minority instance, $\mathbf{X_{min}(i)}$, if τ_{min}(i) had a higher value than τ_{maj}(i) multiplied by *CT*, then the method would keep this point, as shown in Figure 3.6. This step focused on the minority instances located in the minority distribution and avoided the majority distribution. The appropriate cutoff threshold value will be discussed in Section 4.

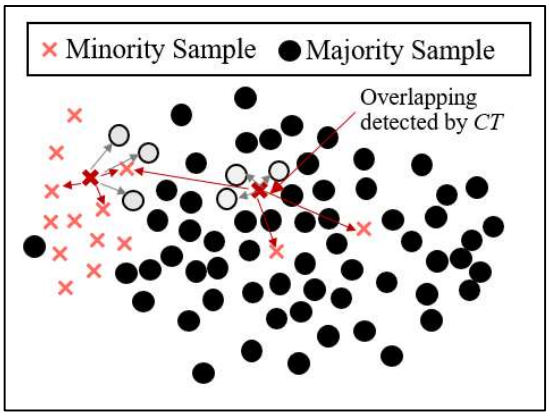


Figure 3.6 The overlapping instances are excluded by the algorithm

After that, the method assigned a ratio of the closeness factor, φ, between the minority and majority groups to each minority instance, as in equation (3.6). Then, the ratio of the closeness factor was converted into the probability distribution, as in equation (3.7).

$$\varphi(i) = \frac{\tau_{min}(i) + 1}{\tau_{maj}(i) + 1} \tag{3.6}$$

In equation (3.6), the method added one into the minority and majority closeness factors to avoid the infinity value of the ratio because the majority closeness factor could be almost zero if the majority nearest neighbors were far from the minority instance.

$$P(i) = \frac{\varphi(i)}{\sum_{j=1}^{n_{min}} \varphi(j)} \tag{3.7}$$

The method assigned a probability distribution to each minority instance, as in equation (3.7). Our approach did not consider the 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 the data point i^{th} of the minority instances was selected to synthesize the new points. The referenced data points were selected, and the number of the generated samples was assigned in each point based on the probability distribution *P(i),* as shown in Figure 3.7.

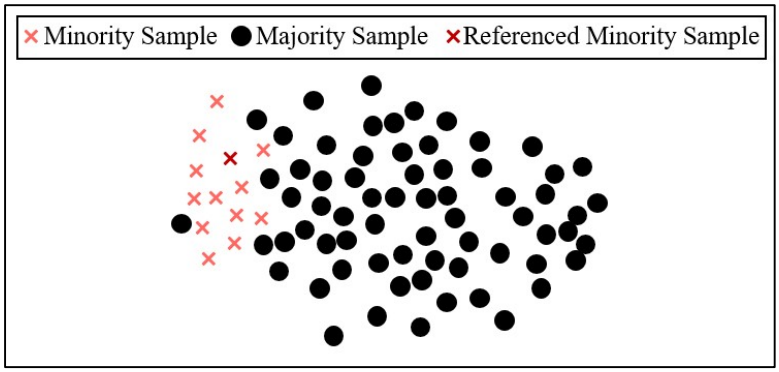


Figure 3.7 The SyMProD calculates probability distribution and selects referenced minority instance

In this step, the method selected the minority instances to create synthetic points which determined the minority distribution. The method solved the overlapping and overgeneralization problems by selecting the minority instances based on the probability distribution instead of the same weight. Moreover, the selected minority instances were located in the minority distribution instead of the majority distribution.

3.1.4.1.3 Instance Synthesis

In the last stage, the method generated synthetic instances from the data points that were selected in the minority point selection step. This step modified the Sigma Nearest Oversampling Based on Convex Combination (SNOCC) [20] algorithm that determined several nearest neighbors with the probability. Our method created new samples by this technique instead of generating them along the line between two points as the 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 **R**, **{R(1)**, **R(2)**,…, **R(M+1)}**. **R** was a set of real values to calculate new samples, and the

value of a new sample was within this range. The probability distribution, P, of each instance in set **R** was collected in set **Pr.** This step created the synthetic instance, X_{new}, and added it to the oversampled set, **O**, as follows:

$$X_{new} = \beta(1)Pr(1)R(1) + \beta(2)Pr(2)R(2) + \cdots + \beta(M+1)Pr(M+1)R(M+1) \quad (3.8)$$
$$= \sum_{j=1}^{M+1} \beta(j)Pr(j)R(j)$$

β(j) was a random number of the positive values between 0 and 1, *Pr(j)* was a probability distribution of the sample j^{th} from the referenced instance and its neighbors. *β(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 the sample j^{th} in set **R**. The authors can claim that the value of a new sample, X_{new}, will be created within the range of the real value in set **R** because of the normalization step of the coefficient factor.

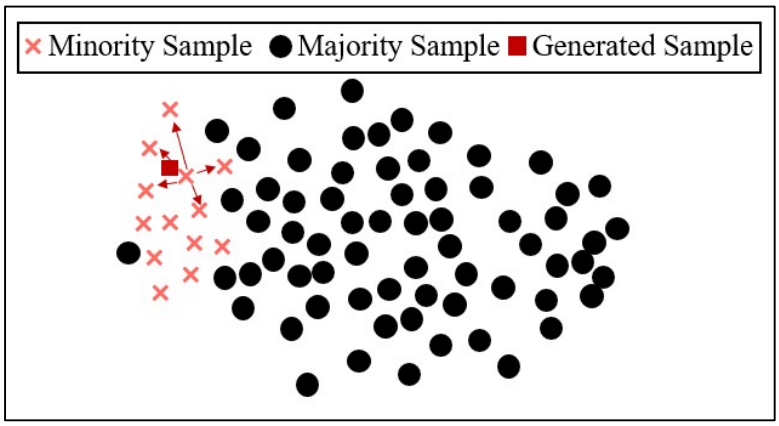


Figure 3.8 The SyMProD generates new instance from multiple nearest neighbors

In this step, the method generated synthetic instances from the multiple data points, which covered the minority class distribution. SyMProD also solved the overfitting problem since our approach did not replicate the data as the random oversampling performed. Figure 3.8 shows the generated sample that was created in the minority region instead of being produced in the line between two points.

3.1.5 Model Training

The dataset after oversampling was trained by the classification models which were supervised learning. This study employed three classifiers, i.e., Random Forest, Logistic Regression, and Support Vector Machine, to create prediction models using scikit-learn library [15] and assess the oversampling technique. This section summarized the theory of each classifier from Section 2.

Random Forest (RF) is an ensemble learning method that combines several decision tree algorithms to resolve the overfitting problem. In each decision tree, it has different records

and features to generalize the training model. After that, the voting method is applied to select the most result from each tree.

Logistic Regression (LR) is a classification technique that applies the sigmoid function of linear combination with given variables to classify the labels. This algorithm measures the correlation between independent variables and the categorical labels. The result is the probability of each class, which is between zero and one.

Support Vector Machine (SVM) is a supervised learning method that applies the hyper-plane to differentiate the class of the dataset. The algorithm tries to find the best location of the plane based on the distance between the plane and the nearest point of each class. There are several plane types, kernel functions, in this algorithm, e.g., linear, polynomial, and radial basis functions.

3.1.6 Model Evaluating

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 [40] [41]. The accuracy is the ratio between the number of the correctly predicted observations and the total observations. When 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). This research considered the minority class as a positive class and the majority class as a negative class. Furthermore, the confusion matrix in Table 3.2 was required to calculate the evaluation.

Table 3.2 Confusion Matrix

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

The confusion matrix classifies the results into four groups according to the actual and predicted classes.

True Positive (TP) is the number of the correctly predicted positive values. For example, the actual condition of the patient is liver cancer, and the model also predicts the same class.

True Negative (TN) is the number of the correctly predicted negative values. For example, the actual condition of the patient is non-cancerous, and the model predicts the same class.

False Positive (FP) is the number of the incorrectly predicted results that the actual condition of the person is the negative class, whereas the model predicts the positive class.

For example, the actual condition of the patient is non-cancerous, but the model predicts that the class of this patient is liver cancer.

False Negative (FN) is the number of the incorrectly predicted results that the actual condition of the data is positive whereas the model predicts the negative class. For example, the actual condition of the patient is liver cancer, but the model predicts that the class of this patient is non-cancerous.

The sensitivity, also referred to recall or true positive rate, is the proportion of the positive cases that the model identifies correctly. If the dataset is imbalanced and the positive class is not predicted, then the model will perform low sensitivity. It is determined using this equation.

$$Sensitivity = \frac{TP}{TP + FN} \tag{3.9}$$

Specificity is the proportion of the negative cases that the model identifies correctly, as shown in equation (3.10). Precision explains the correctly positive prediction among the entire positive predictions.

$$Specificity = \frac{TN}{TN + FP} \tag{3.10}$$

G-Mean aggregates Sensitivity and Specificity to detect the overfitting of the negative class with the 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:

$$G - Mean = \sqrt{Sensitivity * Specificity} = \sqrt{\frac{TP}{TP + FN} * \frac{TN}{TN + FP}} \tag{3.11}$$

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

$$F - Measure = \frac{(1 + \gamma^2) * Sensitivity * Precision}{(\gamma^2 * Precision) + Sensitivity} \tag{3.12}$$

$$F - Measure = \frac{(1 + \gamma^2) * TP}{(1 + \gamma^2) * TP + (\gamma^2 * FN) + FP} \tag{3.13}$$

Moreover, sensitivity and specificity metrics in the prediction model are necessary for the medical sector. The best case is that the sensitivity and specificity are 100 percent, and the prediction model can classify both classes flawlessly. However, the chance of 100

percent of the sensitivity and specificity in the real-world problems is impossible, especially in the medical datasets. The doctors and scientists have to maximize the performance of the prediction model and select the best threshold for classification. This value is the trade-off between these two metrics. If users choose a high threshold, it decreases the performance of the sensitivity while increasing the specificity. In the same way, if users select a lower threshold, it increases the performance of the sensitivity while decreasing the specificity. This trade-off can be represented by the Receiver Operating Characteristic curve (ROC) which displays the true positive rate (sensitivity) and false positive rate (1-specificity) with many thresholds in the graph, and calculates the Area Under the Curve (AUC). This metric compares the performance among the prediction models. The higher value of AUC presents a better performance of the prediction model.

3.2 Experimental Design

In this section, the authors present the experimental setup, design of experiment, and the hardware and software specifications. Firstly, the authors conclude the overview of the general oversampling techniques compared to our proposed method. Secondly, the design of experiment is introduced to evaluate the performance among algorithms. Finally, the machine environment and software libraries are provided.

3.2.1 Experimental Setup

Our oversampling technique was evaluated with eight popular oversampling techniques which were SMOTE, Borderline SMOTE, Safe-Level SMOTE, Cluster SMOTE, MWMOTE, k-means SMOTE, A-SUWO, and ADASYN. The summary of each method is provided below. The details of each technique were introduced in Section 2. The characteristics of 17 datasets to compare the results were shown in Table 3.1. Moreover, this study applied three algorithms, three evaluation metrics, and the stratified five-fold cross-validation with repeated twenty times in the experiment.

1. Synthetic Minority Oversampling Techniques (SMOTE): It applies the k-nearest neighbor method to search based on the similarity of feature and randomly select one neighbor. It generates the synthetic instances along the line between data points.
2. Borderline-SMOTE: The method divides the minority instances into safe, borderline, and danger regions based on the ratio of the majority class. The Borderline-SMOTE adapts the SMOTE technique by creating the new synthetic instance on the border of the decision region.
3. Safe-Level SMOTE: Unlike the Borderline-SMOTE, the Safe-Level SMOTE only focuses the minority instances in the safe region. The safe level depends on the minority instances in the k nearest neighbors. Then, the method applies the SMOTE to generate new samples in a safe area.
4. Cluster SMOTE: The method applies clustering to partition the dataset and enable the SMOTE to generate new samples in the cluster.
5. MWMOTE: The method improves the performance of decision boundary detection. The MWMOTE identifies which instances of the minority class were hard to learn, indicates the weight to minority class instances, adapts the clustering technique to group the minority class instances before generating new samples, and employs the SMOTE to produce new samples.
6. K-means SMOTE: The method uses the k-means clustering to group datasets and then classifies which clusters have to be oversampled based on the ratios of both classes and the number of synthetic instances in each cluster; whereas the Cluster-SMOTE

does not define the number of the generated instances in each cluster and the optimal number of clusters. After that, the SMOTE is applied to generate new samples in the cluster.

7. Adaptive semi-unsupervised weighted oversampling (A-SUWO): The method employs the hierarchical clustering and classification model to identify the number of new samples in each sub-cluster. The method determines the probability distribution based on the distance and creates new samples on the borderline between two classes.
8. Adaptive Synthetic (ADASYN): It applies the concept of weighting the minority class instances to generate data. The algorithm can decide how many samples should be produced and focus on hard to learn data based on the number of the majority nearest neighbors.

3.2.2 Design of Experiment

There are two main experiments which evaluate our proposed algorithm. This study assessed the performance of our oversampling technique and compared it to the well-known methods with the standard dataset. In the experiments, this research applied the Friedman test and Holm's test as a post-hoc analysis. Other studies in the oversampling technique topic also applied these statistical tests [4] [9] [26] [42] [43]. Firstly, the method applied a ranking score to assess the performance of the oversampling technique in every combination. The best performing technique received the rank one whereas the worst received the rank nine. To compare the results statistically, this study employed the Friedman test and Holm-Bonferroni method that took the SyMProD as a control method to verify the statistical significance.

The Friedman test is a non-parametric test that works the same as the repeated measures of ANOVA. This statistical test compares 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 is 0.05.

After that, if the null hypothesis is rejected, then a post-hoc test is applied. This study used the Holm-Bonferroni test as the post-hoc test. This test is the non-parametric test and counteracts multiple testing using the step-down technique with an adjusted p-value. According to the Holm-Bonferroni procedure, this research set the SyMProD as a control method to compare our proposed technique with other methods. The null hypothesis in each pairwise is to check whether the SyMProD does not perform better than the others. The authors conclude the statistical test below.

Objective (1)

To detect the differences in mean rankings among oversampling techniques in each metric and classifier, i.e., AUC, F-Measure, and G-Mean.

Input

The inputs of the experimental design were the rankings of the oversampling methods in each prediction model and evaluation metric. In each experiment, the method applied the same random state in the stratified cross-validation and parameters. The 17 datasets were included in this experiment, which are provided in Table 3.1.

Output
The research applied the Friedman test to compare the performance. The null hypothesis test is whether there is a similar performance among all classifiers in terms of mean ranking. After that, if the null hypothesis of the Friedman test is rejected, then a post-hoc is applied.

Objective (2)
To compare the performance of our proposed technique with other conventional algorithms.

Input
The method only applied the Holm-Bonferroni method if the Friedman test was rejected. The inputs were the ranking of our proposed method and the ranking of the conventional technique which was needed to compare the metrics.

Output
The null hypothesis test is whether our oversampling algorithm does not perform better than another algorithm. The output of this test verifies the performance statistically by comparing the rankings while our proposed method is the control method.

3.2.3 Hardware and Software Specifications

Our experiments were performed on Google Cloud's Linux instances. Table 3.3 presents the detail of the specifications.

Table 3.3 Specifications of hardware and software

Operating System	Linux Debian 9.5 64 bits
CPU	Intel(R) Xeon(R) CPU @ 2.20GHz
Memory (RAM)	16 GB
Programming Language	Python 3.7.3
Machine learning open source	Pandas version 0.25.0 Numpy version 1.16.4 Scikit learn version 0.21.2 Scipy version 1.2.1 Imblearn version 0.5.0 Smote-Variant version 0.3.3

The prediction model and our proposed oversampling technique were developed based on Python and open-source libraries as the information below:

1. Pandas is the importing and analyzing python package. It provides the manipulation tools to read the data from several formats, preprocess, and transform them to the appropriate format.
2. Numpy is the package to handle the multidimensional dataset with a fast and powerful function.
3. Scikit-learn is the machine learning python package. It contains several algorithms to create a machine learning model from the data, e.g., Random Forest, Logistic Regression, and Support Vector Machine.

4. Scipy is the mathematics and statistics python package. It provides technical computing, especially the stats algorithms to analyze the data.
5. Imblearn is a python package to handle the imbalanced dataset. It contains general methods, e.g., SMOTE and ADASYN.
6. Smote-Variant is another package to deal with the imbalanced dataset. It contains several methods that the Imblearn package does not maintain, e.g., MWMOTE and A-SUWO.

CHAPTER 4 RESULTS AND DISCUSSION

This chapter presents results that were experimentally evaluated and discussion of the algorithm. There are two sections which are experimental results and parameter sensitivity analysis of our algorithm. The first section shows the results from the proposed oversampling technique and other conventional methods over three classifiers and 17 datasets in terms of Area Under the Curve, F-Measure, and G-mean metrics. Moreover, this study compares our algorithm among eight conventional oversampling techniques in order to present the improvement of the performance. The statistical results, e.g., Friedman test and Holm-Bonferroni, are presented in this section. The second section demonstrates the parameters of our algorithm and the results of the sensitivity analysis.

4.1 Experimental Results

This section presents the results that were assessed with eight conventional oversampling techniques, three classifiers, 17 datasets, and three evaluation metrics. The method applied the stratified five-fold cross-validation to evaluate the generalization of the prediction models with repeated twenty times and tuned the threshold. The results were averaged, and 95% confidence interval was calculated to represent the performance.

According to the experimental design in Section 3, this study measured the performance metrics of our oversampling algorithm compared with other oversampling methods. The main goal was to improve the AUC metric of the classifiers after being oversampled by our proposed technique. In each classifier, the ranking of the oversampling techniques was calculated to determine the mean ranking, and presented in Table 4.1 as the metric assessment summary. For the detailed results, Tables 4.2, 4.3, and 4.4 illustrate the metric results of Random Forest, Logistic Regression, and Support Vector Machine, respectively. These tables show the results in each classifier, dataset, oversampling technique, and metric. The top two measurements are represented in bold.

Table 4.1 Results for mean ranking of oversampling methods over 3 classifiers and 17 datasets

	SMOTE	Borderline SMOTE	K-means SMOTE	MW MOTE	Safe Level SMOTE	Cluster SMOTE	ADASYN	A-SUWO	SyMProD
Algorithm: Random Forest									
AUC	4.176 ± 0.224	5.832 ± 0.261	4.756 ± 0.288	5.218 ± 0.253	4.850 ± 0.297	4.553 ± 0.250	6.332 ± 0.227	5.638 ± 0.297	**3.615 ± 0.241**
FM	4.771 ± 0.233	5.632 ± 0.260	4.215 ± 0.256	5.556 ± 0.241	5.482 ± 0.307	5.006 ± 0.232	6.465 ± 0.271	4.421 ± 0.258	**3.453 ± 0.269**
GM	4.326 ± 0.238	5.591 ± 0.267	5.003 ± 0.277	5.215 ± 0.264	5.350 ± 0.302	4.603 ± 0.253	6.288 ± 0.253	4.679 ± 0.266	**3.944 ± 0.267**
Algorithm: Logistic Regression									
AUC	4.188 ± 0.206	5.359 ± 0.265	4.409 ± 0.272	4.553 ± 0.251	6.700 ± 0.290	4.671 ± 0.241	6.068 ± 0.245	5.126 ± 0.319	**3.744 ± 0.219**
FM	5.247 ± 0.209	5.853 ± 0.249	3.124 ± 0.225	5.388 ± 0.219	5.759 ± 0.289	5.376 ± 0.214	6.612 ± 0.252	4.994 ± 0.300	**2.588 ± 0.202**
GM	4.729 ± 0.218	5.532 ± 0.237	4.053 ± 0.290	4.815 ± 0.239	6.041 ± 0.296	5.006 ± 0.236	6.235 ± 0.265	5.068 ± 0.294	**3.462 ± 0.255**
Algorithm: Support Vector Machine									
AUC	4.268 ± 0.197	5.568 ± 0.246	4.685 ± 0.297	4.856 ± 0.238	6.159 ± 0.304	4.712 ± 0.247	5.844 ± 0.262	5.782 ± 0.277	**3.044 ± 0.227**
FM	5.250 ± 0.215	6.000 ± 0.238	3.709 ± 0.258	5.318 ± 0.232	4.956 ± 0.279	5.332 ± 0.226	6.909 ± 0.242	4.950 ± 0.295	**2.553 ± 0.199**
GM	4.844 ± 0.223	5.621 ± 0.240	4.541 ± 0.302	4.906 ± 0.239	5.515 ± 0.269	4.918 ± 0.253	6.462 ± 0.266	4.653 ± 0.301	**3.518 ± 0.258**

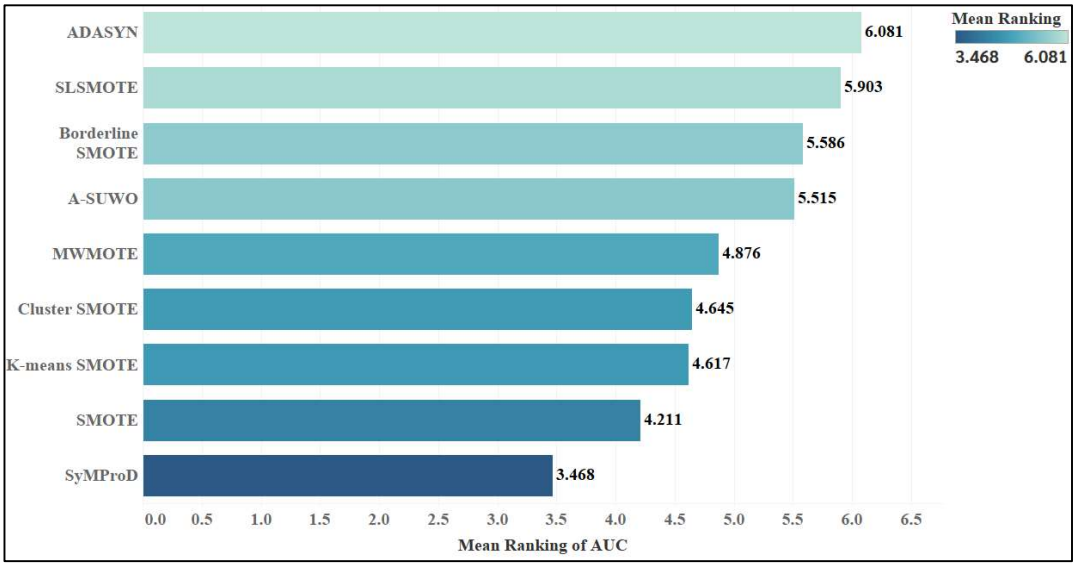


Figure 4.1 Mean ranking of AUC in each oversampling technique over all classifiers

To fairly compare the oversampling techniques, this study applied the same number of features, the parameter in classifiers, and the random state of each cross-validation. Thus, there was the same training dataset before oversampling the data in each algorithm. In Table 4.1, the SyMProD performs better than other oversampling techniques in every classifier and metric from the benchmark datasets. Aside from our approach, the k-means SMOTE and SMOTE achieved the best result. Moreover, the method calculated the mean ranking of the AUC across classifiers and presented the results in Figure 4.1. The tables below present the detailed results in each classifier.

Table 4.2 Performance evaluations of the oversampling methods using Random Forest algorithm

Data	Metric	SMOTE	Borderline SMOTE	K-means SMOTE	MW MOTE	Safe Level SMOTE	Cluster SMOTE	ADASYN	A-SUWO	SyM ProD
Balance	AUC	0.3754 ±0.0060	0.3741 ±0.0049	0.3959 ±0.0097	0.3818 ±0.0075	**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	0.2648 ±0.0085	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**	0.4156 ±0.0054	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**	0.9629 ±0.0033	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**	0.8085 ±0.0117	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	0.9283 ±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**	0.9221 ±0.0033	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**	0.7459 ±0.0086	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**	0.8119 ±0.0081	0.7462 ±0.0150	0.8393 ±0.0077	0.8055 ±0.0114	0.8125 ±0.0103	**0.8664 ±0.0086**
Haber man	AUC	0.6869 ±0.0076	0.6825 ±0.0075	0.6973 ±0.0069	0.6932 ±0.0064	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	0.5185 ±0.0075	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	0.6470 ±0.0090	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**	0.9085 ±0.0027	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	**0.8206 ±0.0047**	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	0.8206 ±0.0076	**0.8210 ±0.0050**	0.8151 ±0.0075	0.8146 ±0.0080	0.8121 ±0.0080	0.8118 ±0.0115

Table 4.2 Performance evaluations of the oversampling methods using Random Forest algorithm (Cont'd)

Data	Metric	SMOTE	Borderline SMOTE	K-means SMOTE	MW MOTE	Safe Level SMOTE	Cluster SMOTE	ADASYN	A-SUWO	SyM ProD
Ionos phere	AUC	0.9763 ±0.0010	0.9766 ±0.0015	0.9757 ±0.0012	0.9758 ±0.0011	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**	0.8933 ±0.0082	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**	0.9227 ±0.0062	0.9228 ±0.0041	0.9254 ±0.0033	0.9245 ±0.0060	0.9254 ±0.0033	**0.9299 ±0.0035**
Libras	AUC	0.9846 ±0.0021	0.9801 ±0.0027	0.9790 ±0.0041	0.9715 ±0.0023	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	0.8225 ±0.0115	**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	0.9331 ±0.0047	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	0.8245 ±0.0016	**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	0.6966 ±0.0036	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	0.7600 ±0.0046	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	0.9903 ±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	0.8048 ±0.0115	**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	0.9527 ±0.0025	**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	0.9915 ±0.0005	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	**0.8930 ±0.0072**	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	**0.9606 ±0.0029**	0.9552 ±0.0028	0.9534 ±0.0034	0.9526 ±0.0023	0.9549 ±0.0034	**0.9609 ±0.0026**
Cancer	AUC	0.9470 ±0.0005	0.9440 ±0.0006	0.9386 ±0.0007	0.9379 ±0.0005	0.9382 ±0.0008	0.9487 ±0.0005	0.9438 ±0.0006	**0.9478 ±0.0006**	**0.9471 ±0.0006**
	FM	0.6709 ±0.0099	0.6582 ±0.0062	**0.6953 ±0.0047**	0.6480 ±0.0076	0.6358 ±0.0083	0.6801 ±0.0075	0.6507 ±0.0077	**0.6977 ±0.0061**	0.6847 ±0.0056
	GM	0.8841 ±0.0017	0.8786 ±0.0016	0.8598 ±0.0018	0.8698 ±0.0014	0.8656 ±0.0021	**0.8856 ±0.0014**	0.8775 ±0.0013	**0.8886 ±0.0011**	0.8854 ±0.0017
Simu lated1	AUC	**0.9310 ±0.0023**	0.9258 ±0.0023	0.9282 ±0.0030	**0.9343 ±0.0024**	0.9255 ±0.0014	0.9247 ±0.0027	0.9258 ±0.0023	0.9127 ±0.0033	0.9284 ±0.0026
	FM	0.5959 ±0.0080	0.6030 ±0.0053	**0.6849 ±0.0072**	0.6107 ±0.0066	0.6280 ±0.0082	0.5972 ±0.0073	0.5701 ±0.0077	0.6409 ±0.0053	**0.6840 ±0.0054**
	GM	**0.8830 ±0.0043**	0.8826 ±0.0041	0.8602 ±0.0057	**0.8888 ±0.0029**	0.8822 ±0.0039	0.8820 ±0.0041	0.8792 ±0.0033	0.8722 ±0.0040	0.8690 ±0.0042
Simu lated2	AUC	0.9437 ±0.0014	0.9351 ±0.0022	0.9461 ±0.0015	**0.9464 ±0.0010**	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**	0.7523 ±0.0063	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	0.8917 ±0.0026	0.8786 ±0.0033	0.8938 ±0.0024	0.8773 ±0.0041	**0.8957 ±0.0018**	0.8881 ±0.0017
Simu lated3	AUC	0.9525 ±0.0007	0.9457 ±0.0009	0.9524 ±0.0009	0.9519 ±0.0008	**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**	0.8263 ±0.0027	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**	0.8937 ±0.0018	0.8858 ±0.0034	0.8968 ±0.0015	0.8788 ±0.0061	0.8913 ±0.0020	**0.8978 ±0.0008**
Simu lated4	AUC	0.9751 ±0.0009	0.9726 ±0.0012	**0.9757 ±0.0012**	0.9750 ±0.0007	0.9748 ±0.0010	0.9731 ±0.0012	0.9685 ±0.0010	**0.9762 ±0.0011**	0.9752 ±0.0012
	FM	0.8503 ±0.0073	0.8049 ±0.0114	0.8617 ±0.0104	0.8275 ±0.0089	0.8547 ±0.0095	0.8448 ±0.0113	0.7800 ±0.0133	**0.8743 ±0.0072**	**0.8844 ±0.0077**
	GM	**0.9617 ±0.0015**	0.9506 ±0.0030	**0.9617 ±0.0024**	0.9583 ±0.0017	0.9571 ±0.0021	0.9605 ±0.0023	0.9467 ±0.0024	0.9589 ±0.0023	0.9614 ±0.0017
Simu lated5	AUC	0.9865 ±0.0004	0.9861 ±0.0006	0.9863 ±0.0004	0.9859 ±0.0003	**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	0.9095 ±0.0056	**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	0.9652 ±0.0016	**0.9733 ±0.0014**	0.9675 ±0.0016	0.9573 ±0.0020	**0.9698 ±0.0013**	0.9692 ±0.0011
Simu lated6	AUC	0.9891 ±0.0003	0.9885 ±0.0003	0.9887 ±0.0003	0.9887 ±0.0002	**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	0.9363 ±0.0044	**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	0.9651 ±0.0021	**0.9701 ±0.0014**	0.9638 ±0.0015	0.9638 ±0.0021	0.9658 ±0.0015	0.9632 ±0.0019

Table 4.3 Performance evaluations of the oversampling methods using Logistic Regression algorithm

Data	Metric	SMOTE	Borderline SMOTE	K-means SMOTE	MW MOTE	Safe Level SMOTE	Cluster SMOTE	ADASYN	A-SUWO	SyM ProD
Balance	AUC	0.5516 ±0.0068	0.5529 ±0.0057	0.5159 ±0.0119	**0.5650 ±0.0099**	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	0.3088 ±0.0090	**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	**0.5302 ±0.0094**	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**	0.9375 ±0.0014	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**	0.6991 ±0.0081	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**	0.8977 ±0.0032	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**	0.8233 ±0.0046	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**	0.6714 ±0.0066	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**	0.7249 ±0.0084	0.6690 ±0.0118	0.7320 ±0.0075	0.7126 ±0.0085	0.7063 ±0.0099	**0.7331 ±0.0053**
Haber man	AUC	0.6766 ±0.0060	0.6677 ±0.0081	**0.6911 ±0.0052**	0.6728 ±0.0073	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**	0.4774 ±0.0090	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**	0.6249 ±0.0075	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	**0.9022 ±0.0023**	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	**0.8165 ±0.0056**	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	**0.8187 ±0.0069**	0.7928 ±0.0132	0.8149 ±0.0059	0.8169 ±0.0094	0.8096 ±0.0077	**0.8185 ±0.0065**
Ionos phere	AUC	0.8633 ±0.0055	0.8633 ±0.0050	**0.8639 ±0.0053**	0.8632 ±0.0047	**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**	0.7927 ±0.0054	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**	0.8360 ±0.0041	0.8355 ±0.0062	0.8312 ±0.0067	0.8297 ±0.0061	0.8207 ±0.0053	**0.8375 ±0.0038**
Libras	AUC	0.7341 ±0.0092	0.7333 ±0.0097	0.7341 ±0.0091	0.7331 ±0.0095	**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	0.5605 ±0.0165	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	0.7074 ±0.0133	**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	0.8279 ±0.0010	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	0.6811 ±0.0020	**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	0.7454 ±0.0023	**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	**0.9455 ±0.0004**	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**	0.6381 ±0.0043	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	**0.8903 ±0.0017**	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**	0.9933 ±0.0007	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**	0.9333 ±0.0031	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**	0.9668 ±0.0018	0.9535 ±0.0023	0.9664 ±0.0019	**0.9672 ±0.0017**	0.9609 ±0.0017	0.9669 ±0.0019
Cancer	AUC	**0.9170 ±0.0005**	0.9124 ±0.0006	0.9118 ±0.0020	0.9165 ±0.0007	0.9120 ±0.0009	0.9169 ±0.0005	0.9130 ±0.0007	0.9142 ±0.0009	**0.9169 ±0.0005**
	FM	0.6105 ±0.0083	0.5854 ±0.0082	**0.6587 ±0.0043**	0.5995 ±0.0055	0.6101 ±0.0067	0.6096 ±0.0084	0.5831 ±0.0084	0.6009 ±0.0072	**0.6593 ±0.0036**
	GM	**0.8468 ±0.0015**	0.8399 ±0.0021	0.8346 ±0.0016	0.8447 ±0.0015	0.8386 ±0.0013	**0.8471 ±0.0017**	0.8388 ±0.0018	0.8448 ±0.0016	0.8454 ±0.0016
Simu lated1	AUC	0.9478 ±0.0005	0.9477 ±0.0005	0.9478 ±0.0005	**0.9479 ±0.0004**	0.9458 ±0.0005	0.9478 ±0.0005	0.9478 ±0.0005	0.9476 ±0.0004	**0.9480 ±0.0004**
	FM	0.6117 ±0.0116	0.6025 ±0.0099	**0.6962 ±0.0070**	0.6116 ±0.0105	0.6376 ±0.0089	0.6005 ±0.0139	0.5927 ±0.0132	0.6257 ±0.0056	**0.6894 ±0.0066**
	GM	0.9017 ±0.0028	0.8997 ±0.0027	0.8944 ±0.0037	**0.9023 ±0.0024**	**0.9027 ±0.0023**	0.8982 ±0.0037	0.8954 ±0.0040	0.9013 ±0.0024	0.8983 ±0.0024
Simu lated2	AUC	0.9568 ±0.0004	0.9569 ±0.0004	0.9565 ±0.0005	**0.9570 ±0.0003**	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**	0.7641 ±0.0071	0.7753 ±0.0079	0.7558 ±0.0066	0.7458 ±0.0077	0.7498 ±0.0057	**0.7891 ±0.0023**

Table 4.3 Performance evaluations of the oversampling methods using Logistic Regression algorithm (Cont'd)

Data	Metric	SMOTE	Borderline SMOTE	K-means SMOTE	MW MOTE	Safe Level SMOTE	Cluster SMOTE	ADASYN	A-SUWO	SyM ProD
Simulated2	GM	0.8993 ±0.0019	0.8860 ±0.0035	**0.9038 ±0.0016**	0.9009 ±0.0028	0.9026 ±0.0026	0.8967 ±0.0028	0.8915 ±0.0031	0.8949 ±0.0028	**0.9060 ±0.0013**
Simulated3	AUC	0.9617 ±0.0003	0.9616 ±0.0003	0.9617 ±0.0003	0.9616 ±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**	0.8292 ±0.0047	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**	0.8965 ±0.0028	0.9031 ±0.0021	0.9012 ±0.0023	0.8984 ±0.0029	0.8934 ±0.0034	**0.9081 ±0.0010**
Simulated4	AUC	0.9813 ±0.0005	0.9825 ±0.0004	**0.9827 ±0.0003**	0.9799 ±0.0005	0.9796 ±0.0011	0.9811 ±0.0004	0.9721 ±0.0008	**0.9850 ±0.0003**	0.9821 ±0.0004
	FM	0.8449 ±0.0071	0.8987 ±0.0058	0.9148 ±0.0040	0.8250 ±0.0101	0.8475 ±0.0100	0.8416 ±0.0070	0.7234 ±0.0092	**0.9640 ±0.0033**	**0.9182 ±0.0046**
	GM	0.9617 ±0.0012	0.9699 ±0.0009	0.9721 ±0.0006	0.9579 ±0.0020	0.9575 ±0.0025	0.9610 ±0.0012	0.9319 ±0.0023	**0.9750 ±0.0016**	**0.9726 ±0.0007**
Simulated5	AUC	0.9912 ±0.0003	**0.9919 ±0.0003**	0.9914 ±0.0003	0.9905 ±0.0004	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	0.9279 ±0.0036	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	0.9728 ±0.0010	0.9711 ±0.0022	0.9731 ±0.0010	0.9459 ±0.0025	**0.9862 ±0.0009**	**0.9826 ±0.0005**
Simulated6	AUC	0.9914 ±0.0002	0.9910 ±0.0002	0.9914 ±0.0002	0.9912 ±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	0.9530 ±0.0020	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	0.9751 ±0.0010	0.9724 ±0.0015	0.9769 ±0.0009	0.9554 ±0.0026	**0.9830 ±0.0012**	**0.9824 ±0.0008**

Table 4.4 Performance evaluations of the oversampling methods using Support Vector Machine algorithm

Data	Metric	SMOTE	Borderline SMOTE	K-means SMOTE	MW MOTE	Safe Level SMOTE	Cluster SMOTE	ADASYN	A-SUWO	SyM ProD
Balance	AUC	0.5317 ±0.0079	0.5301 ±0.0082	0.5058 ±0.0136	0.5284 ±0.0117	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	**0.3226 ±0.0083**	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	**0.5324 ±0.0088**	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	**0.9369 ±0.0015**	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**	0.7021 ±0.0110	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**	0.8988 ±0.0043	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	0.8150 ±0.0052	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	**0.6514 ±0.0076**	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**	0.6848 ±0.0120	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**	0.6802 ±0.0058	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**	0.4401 ±0.0064	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**	0.5807 ±0.0056	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**	0.8995 ±0.0024	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**	0.8132 ±0.0083	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**	0.8127 ±0.0128	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	0.8676 ±0.0048	**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**	0.7865 ±0.0088	**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**	0.8324 ±0.0068	**0.8484 ±0.0061**	0.8285 ±0.0064	0.8215 ±0.0050	0.8302 ±0.0066	0.8317 ±0.0092
Libras	AUC	**0.8375 ±0.0064**	0.8350 ±0.0070	0.7981 ±0.0104	0.8251 ±0.0077	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	0.6636 ±0.0148	**0.7023 ±0.0126**	0.6915 ±0.0117	0.6464 ±0.0121	0.6779 ±0.0117	**0.6922 ±0.0115**

Table 4.4 Performance evaluations of the oversampling methods using Support Vector Machine algorithm (Cont'd)

Data	Metric	SMOTE	Borderline SMOTE	K-means SMOTE	MW MOTE	Safe Level SMOTE	Cluster SMOTE	ADASYN	A-SUWO	SyM ProD
Libras	GM	**0.8314 ±0.0078**	0.8208 ±0.0064	0.8069 ±0.0083	0.8165 ±0.0082	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	0.8265 ±0.0012	**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	0.6852 ±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	0.7496 ±0.0027	**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	0.9227 ±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	0.6404 ±0.0022	**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	0.8956 ±0.0012	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	**0.9944 ±0.0004**	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	0.9200 ±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	0.9687 ±0.0013	0.9645 ±0.0025	0.9684 ±0.0015	0.9681 ±0.0013	0.9649 ±0.0016	0.9669 ±0.0012
Cancer	AUC	0.9159 ±0.0005	0.9114 ±0.0005	0.9083 ±0.0025	0.9154 ±0.0007	0.9121 ±0.0007	**0.9170 ±0.0005**	0.9114 ±0.0007	0.9134 ±0.0008	**0.9164 ±0.0006**
	FM	0.6095 ±0.0070	0.5846 ±0.0078	**0.6533 ±0.0045**	0.6026 ±0.0066	0.6103 ±0.0045	0.6213 ±0.0063	0.5866 ±0.0067	0.6033 ±0.0063	**0.6567 ±0.0035**
	GM	**0.8457 ±0.0013**	0.8394 ±0.0019	0.8286 ±0.0018	0.8455 ±0.0016	0.8394 ±0.0013	**0.8492 ±0.0011**	0.8381 ±0.0011	0.8445 ±0.0017	0.8452 ±0.0017
Simulated1	AUC	0.9477 ±0.0005	0.9476 ±0.0005	0.9476 ±0.0005	0.9475 ±0.0004	0.9466 ±0.0005	0.9476 ±0.0004	**0.9477 ±0.0005**	0.9475 ±0.0005	**0.9481 ±0.0005**
	FM	0.6104 ±0.0125	0.6002 ±0.0092	**0.6928 ±0.0059**	0.6118 ±0.0103	0.6325 ±0.0125	0.5990 ±0.0148	0.5982 ±0.0140	0.6278 ±0.0093	**0.6823 ±0.0056**
	GM	0.9012 ±0.0030	0.8986 ±0.0026	0.8959 ±0.0030	**0.9024 ±0.0024**	0.9017 ±0.0025	0.8971 ±0.0041	0.8964 ±0.0039	**0.9025 ±0.0023**	0.9006 ±0.0020
Simulated2	AUC	0.9566 ±0.0004	**0.9568 ±0.0004**	0.9562 ±0.0006	0.9568 ±0.0003	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**	0.7633 ±0.0058	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**	0.9011 ±0.0021	0.9023 ±0.0025	0.8988 ±0.0022	0.8915 ±0.0038	0.8971 ±0.0034	**0.9068 ±0.0010**
Simulated3	AUC	0.9616 ±0.0003	0.9617 ±0.0003	0.9616 ±0.0004	0.9616 ±0.0003	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**	0.8322 ±0.0046	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**	0.8985 ±0.0028	0.9031 ±0.0018	0.9011 ±0.0017	0.8989 ±0.0033	0.8909 ±0.0029	**0.9079 ±0.0011**
Simulated4	AUC	0.9806 ±0.0006	0.9821 ±0.0005	**0.9838 ±0.0004**	0.9794 ±0.0006	0.9797 ±0.0006	0.9812 ±0.0005	0.9747 ±0.0011	0.9833 ±0.0006	**0.9842 ±0.0004**
	FM	0.8468 ±0.0078	0.8971 ±0.0045	**0.9241 ±0.0059**	0.8421 ±0.0093	0.8977 ±0.0077	0.8587 ±0.0074	0.7463 ±0.0135	**0.9492 ±0.0031**	0.9227 ±0.0057
	GM	0.9622 ±0.0013	0.9696 ±0.0007	**0.9733 ±0.0008**	0.9612 ±0.0016	0.9697 ±0.0011	0.9639 ±0.0011	0.9389 ±0.0030	**0.9766 ±0.0004**	0.9732 ±0.0008
Simulated5	AUC	0.9912 ±0.0003	0.9913 ±0.0003	**0.9918 ±0.0002**	0.9904 ±0.0003	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	0.9341 ±0.0033	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	0.9748 ±0.0010	0.9772 ±0.0010	0.9744 ±0.0013	0.9534 ±0.0029	**0.9867 ±0.0003**	**0.9822 ±0.0008**
Simulated6	AUC	0.9915 ±0.0002	0.9909 ±0.0002	**0.9917 ±0.0002**	0.9911 ±0.0003	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	0.9543 ±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	0.9757 ±0.0010	0.9745 ±0.0012	0.9764 ±0.0012	0.9623 ±0.0018	**0.9861 ±0.0006**	**0.9822 ±0.0009**

According to Table 4.2, Table 4.3, and Table 4.4, our method is not the best performing algorithm in all datasets. The performance of the classifier also depends on the distribution of the data and the complexity of the dataset. Therefore, this study aimed to solve the drawbacks of the previous oversampling algorithms, generalize our method to handle various types of imbalanced ratios, and cope with multiple datasets. Based on three evaluation metrics, this study mainly focused on the AUC because the G-Mean and F-Measure could change if the threshold was adjusted whereas the AUC did not change.

Compared with the SMOTE, our results indicated that the SyMProD could increase the AUC metric in 37 out of 51 cases over 17 datasets in three classifiers. The biggest gain of the AUC was 0.06, 11.42% increment, which improved the balance's dataset when the classifier was Support Vector Machine. Figures 4.2 and 4.3 present the relative change in the AUC of our proposed algorithm compared with other oversampling techniques in each classifier.

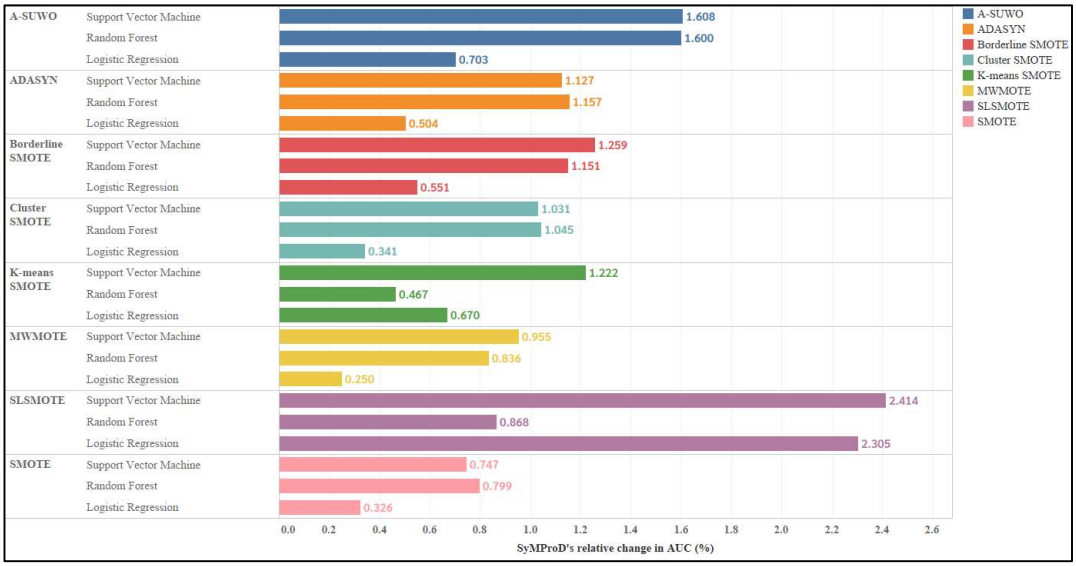


Figure 4.2 SyMProD's relative change in AUC with other oversampling techniques

Figure 4.2 shows the relative change in the AUC metric of the SyMProD when compared with other oversampling techniques. This study calculated the relative change in each dataset and determined the average of the difference to evaluate the generalization of the method. The results showed that our proposed method performed better than other oversampling algorithms. In terms of the AUC, the highest relative change among the oversampling techniques was 2.41% when compared with the Safe Level SMOTE method in the Support Vector Machine algorithm. This situation implied that our proposed method could improve the model performance. The main reason that caused the difference in the AUC was the function in each method. With the different operations, the oversampled data were generated in a different location, which affected the classifier performance.

Figure 4.3 shows the relative change of the SyMProD in each classifier. The median of increment was 0.527%, 0.957%, and 1.175% for Logistic Regression, Random Forest, and Support Vector Machine, respectively. The results showed that our proposed method could work with any classifiers. In this experiment, the authors applied the same parameters of classifiers and conventional oversampling techniques.

This study employed the Friedman test to detect the differences in the mean rankings to compare the results statistically. If the previous test was rejected, then the Holm-Bonferroni method would perform with the SyMProD as the control method to verify the statistical significance. Both statistical analyses determined the significance level of 0.05.

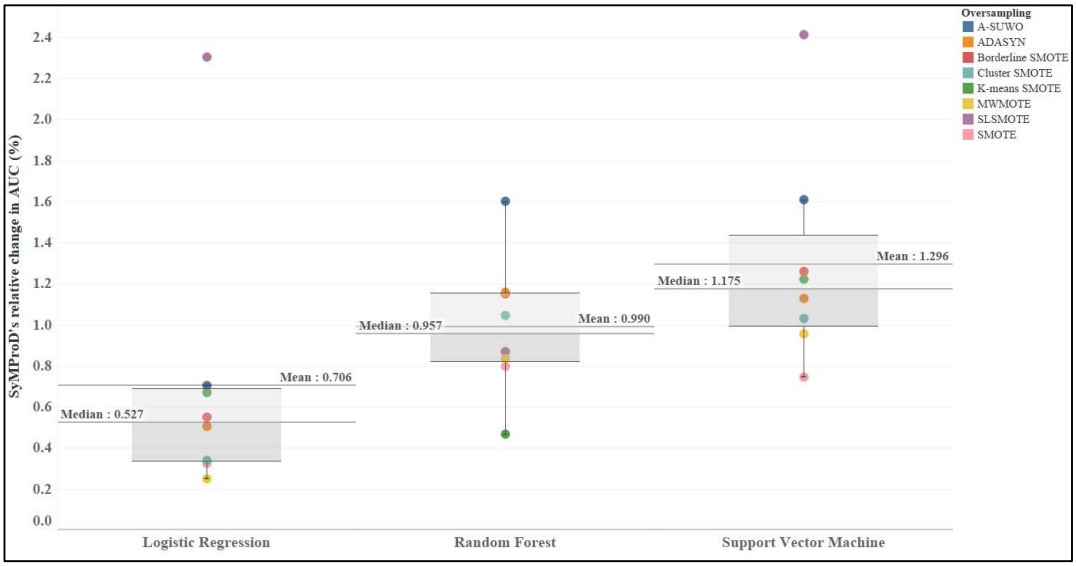


Figure 4.3 SyMProD's relative change in AUC among classifiers

This paper applied the Friedman test to compare the performance of the classifiers with the oversampling technique for each evaluation metric. Table 4.5 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 the mean rankings of the AUC, F-Measure, and G-Mean metrics. Therefore, a post-hoc test was applied.

Table 4.5 Result of Friedman test

Metric	p-value
Algorithm: Random Forest	
AUC	**4.26E-03**
F-Measure	**7.90E-04**
G-Mean	**1.26E-02**
Algorithm: Logistic Regression	
AUC	**2.34E-05**
F-Measure	**2.18E-07**
G-Mean	**3.25E-03**
Algorithm: Support Vector Machine	
AUC	**8.82E-06**
F-Measure	**3.94E-08**
G-Mean	**3.38E-03**

According to the Holm-Bonferroni procedure, this study applied the SyMProD as the control method to evaluate the performance of the oversampling technique and compared it to other methods. Table 4.6 shows the results of the Holm-Bonferroni and an adjusted p-value. The adjusted p-value was compared with a significant level to indicate the hypothesis testing. In terms of the AUC metric, our proposed method defeated other oversampling techniques 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 the SMOTE. In terms of the F-Measure metric, the SyMProD outperformed other techniques in every classifier. In terms of the G-Mean metric, the SyMProD was significantly better than other methods when the authors applied the Support Vector Machine and Logistic Regression. When the Random Forest algorithm was trained as a prediction model, our method outperformed other oversampling techniques except the SMOTE technique.

Table 4.6 Results of Holm-Bonferroni test with SyMProD as the control method

	AUC		**F-Measure**		**G-Mean**	
	Method	p-value	**Method**	p-value	**Method**	p-value
Algorithm: Random Forest						
1	ADASYN	**4.04e-44**	ADASYN	**2.36E-54**	ADASYN	**4.39E-32**
2	Borderline SMOTE	**9.26e-30**	Borderline SMOTE	**4.04E-29**	Borderline SMOTE	**4.25E-16**
3	A-SUWO	**6.55e-25**	MWMOTE	**3.56E-27**	SLSMOTE	**8.57E-12**
4	MWMOTE	**8.24e-16**	SLSMOTE	**2.23E-25**	MWMOTE	**1.13E-09**
5	SLSMOTE	**1.62e-09**	Cluster SMOTE	**4.81E-15**	K-means SMOTE	**9.20E-07**
6	K-means SMOTE	**3.75e-08**	SMOTE	**6.12E-11**	A-SUWO	**2.41E-03**
7	Cluster SMOTE	**1.20e-05**	A-SUWO	**4.83E-06**	Cluster SMOTE	**8.88E-03**
8	SMOTE	**2.89e-02**	K-means SMOTE	**6.34E-04**	SMOTE	4.61E-01
Algorithm: Logistic Regression						
1	SLSMOTE	**1.49E-52**	ADASYN	**8.80E-107**	ADASYN	**4.40E-46**
2	ADASYN	**3.30E-33**	Borderline SMOTE	**2.28E-72**	SLSMOTE	**4.54E-40**
3	Borderline SMOTE	**2.38E-16**	SLSMOTE	**1.78E-68**	Borderline SMOTE	**3.49E-26**
4	A-SUWO	**4.91E-12**	MWMOTE	**4.25E-54**	A-SUWO	**6.51E-16**
5	Cluster SMOTE	**1.27E-05**	Cluster SMOTE	**1.14E-53**	Cluster SMOTE	**9.16E-15**
6	MWMOTE	**2.56E-04**	SMOTE	**4.76E-49**	MWMOTE	**1.95E-11**
7	K-means SMOTE	**4.66E-03**	A-SUWO	**1.96E-40**	SMOTE	**4.27E-10**
8	SMOTE	1.29E-01	K-means SMOTE	**2.81E-02**	K-means SMOTE	**1.84E-02**
Algorithm: Support Vector Machine						
1	SLSMOTE	**5.63E-59**	ADASYN	**2.21E-121**	ADASYN	**5.70E-51**
2	ADASYN	**3.63E-48**	Borderline SMOTE	**1.04E-78**	Borderline SMOTE	**1.37E-26**
3	A-SUWO	**3.89E-46**	Cluster SMOTE	**2.42E-52**	SLSMOTE	**4.83E-24**
4	Borderline SMOTE	**1.92E-39**	MWMOTE	**7.89E-52**	Cluster SMOTE	**5.07E-12**
5	MWMOTE	**9.55E-21**	SMOTE	**1.94E-49**	MWMOTE	**7.71E-12**
6	Cluster SMOTE	**1.20E-17**	SLSMOTE	**1.33E-39**	SMOTE	**7.70E-11**
7	K-means SMOTE	**4.22E-17**	A-SUWO	**1.99E-39**	A-SUWO	**5.73E-08**
8	SMOTE	**1.02E-09**	K-means SMOTE	**1.11E-09**	K-means SMOTE	**1.59E-06**

In terms of the imbalanced ratio, this study determined the correlation between the ranking result of our oversampling technique and the imbalanced ratio of datasets in each classifier. The correlation values of our method were -0.24, -0.14, and -0.28 for the Logistic Regression, Random Forest, and Support Vector Machine classifier, respectively. If the absolute value of the correlation value was less than 0.3, then our method had a low correlation with the imbalanced ratio of datasets [44]. This condition indicated that our method could work with several imbalanced ratios because the correlation between the ranking results and imbalanced ratios was low. In terms of datasets, the SyMProD intended to remove the drawbacks of other oversampling

techniques. The method could handle the dataset when a few minority instances were located in the majority region, which might cause the overlapping problem in another oversampling technique. Moreover, our technique could work with multiple imbalanced ratios. However, if the dataset contained several features and one of its features was an outlier value, then our method would classify that point as an outlier. The method excluded the outlier point for oversampling, which might remove an insight, although the rest of the features were not outliers. In terms of the execution time, our oversampling technique used 1.27 seconds to create 1000 synthetic instances, which were higher than the SMOTE which used 0.02 seconds because of the operation in the algorithm.

In terms of the limitation of our proposed technique, the method only worked with binary classification. The datasets in the experimental designs were preprocessed to the binary classification if an original dataset was a multi-class problem. In the noise removal process, the Z-score normalization was applied for the univariate variable which could not cope with all possibilities of noisy instances, especially multivariate variable analysis. The future work could analyze the noisy instances by using the Mahalanobis distance to identify multivariate outliers. Besides, in the process of calculating the closeness factor, our proposed technique had a limitation on the multimodal dataset because the method determined all points to calculate the closeness factor. However, there was a possible solution that might solve this limitation, which was to apply an Agglomerative hierarchical clustering to group the data and process the entire step in each cluster. Moreover, most of the datasets from the UCI repository and the scikit learn library contained both unimodal and multimodal features, e.g., bimodal. However, if the method could handle the multimodal feature, the performance of oversampling would be increased.

In terms of the liver cancer dataset, the best results of our proposed oversampling technique were performed with the Random Forest classifier. The results were 0.9461, 87.53, and 89.63 for the AUC, Sensitivity, and Specificity metrics. There were several pieces of research from Chulabhorn Hospital that related to this topic. P.Sungkasubun et al. [45] published the research on detecting cholangiocarcinoma, which was a type of liver cancer, by using laboratory tests. Their results were 68.75 and 70.91 for the Sensitivity and Specificity metrics. For the benchmark evaluation, the results from our model could improve the overall performance of the liver cancer prediction model. Furthermore, the top five important features were Neutrophil, Eosinophil, ALP, Total Bilirubin, and Basophil laboratory test. This study also tried various sets of the input features and found that the prediction model provided the best performance with the features of demographic, hematology, and clinical laboratory. Although the other experiments of the liver cancer prediction contained more variables, e.g., Carbohydrate antigen 19-9 (CA199), the model performance was worse than the first set of the features because of the limited dataset. The method removed the record if it contained a missing value. This situation claimed that the limited dataset could decrease the performance of the prediction model. However, the authors expect that the performance of the prediction model will be increased if the authors gather more information on Immunology laboratory data.

The result indicated that our oversampling technique could improve the prediction model. The SyMProD had several advantages compared to other data level methods. Firstly, our approach applied the oversampling technique to generate new points rather than the undersampling, which might depreciate the valuable insight. Secondly, the noisy data

were removed in the first step, which decreased the possibility of noise creation. Thirdly, our approach applied the probability distribution to create synthetic instances to cover minority regions by considering several minority points in the synthesis process. This procedure also eliminated the overfitting problem that the random oversampling faced. Further, the method avoided producing new points in the majority region to reduce the overlapping problem by excluding the minority samples that were located in the majority distribution with the parameter *CT*. The other techniques, e.g., the ADASYN and borderline SMOTE, focused on creating a sample in the hard-to-learn region, but they changed the distribution of the data massively. Fourthly, the SyMProD assigned the probability to each minority instance to eliminate the overgeneralization problem which the SMOTE faced. Lastly, any conventional classifiers can be trained with the oversampled data from our method, which is suitable for practitioners.

4.2 Parameter Sensitivity Analysis

The SyMProD requires 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.

Firstly, our method utilized the Noise Threshold, *NT*, to remove noise instances. An appropriate range of *NT* was between three to five. A lower value of *NT* eliminated valid points and decreased the classifier performance. On the other hand, a higher *NT* could increase the possibility of noise generation because the method did not remove the noise instances. Secondly, the Cutoff Threshold, *CT*, parameter performed with the K nearest neighbors, *K*, in the minority point selection step. These parameters reduced the possibility of the overlapping class problem by considering the closeness factor between the two groups. In each minority instance, the *K* majority and minority nearest neighbors were searched and used to calculate τ_{maj} and τ_{min}, respectively. The proposed method excluded the minority instances if τ_{maj} multiplied by the *CT* was more than τ_{min}. A reasonable value for the *CT* was between 0.8 and 1.2, and the *K* was five.

According to Table 4.7, when the *CT* was set to 0, the result was worse than a higher *CT* because the method did not consider the overlapping problem. However, a higher parameter of the *CT* could remove the insight data because it eliminated the minority instances.

Lastly, the number of the minority nearest neighbors, *M*, was determined to generate new instances from multiple points to cover the minority distribution. A default parameter value was five. If the *M* was defined as one, the synthetic instance generation process would perform the same as the SMOTE oversampling technique. The generated instances might not cover the minority distribution because the method produced new points in the line between two points.

In summary, there were four parameters in our proposed algorithm, i.e., Noise Threshold *NT*, Cutoff Threshold *CT*, *K* nearest neighbors in the minority point selection step, and *M* nearest neighbors in the instance synthesis step. The default parameter of the *NT* was five, which removed the noise data. The *CT* value was between 0.8 to 1.2 and the *K* was five for handling the overlapping problem. The default value of the *M* was five to generate points from several data.

Table 4.7 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

CHAPTER 5 CONCLUSION

The trend of machine learning algorithms is increasing every year, and several real-world industries apply this technique to achieve insight value from data. However, multiple datasets contain a problem that the number of instances in each class is different significantly and the classifier predicts only the majority class. The practitioners cannot employ this model in the production stage because it cannot distinguish any minority instance which is an important class. This problem is called the imbalanced class dataset. Moreover, this problem affects several applications, e.g., churn prediction and fraud detection. Therefore, the procedure to handle the problem becomes an important task in general.

Through the literature review, several methods solve the imbalanced class problem and improve the power of the prediction model. This research mainly focuses on the data level with an oversampling technique that generates minority class instances until the number of both classes is equal. This method persists the insight data of the majority class whereas the undersampling procedure removes the majority class, which may eliminate the data pattern. In the oversampling approach, SMOTE is one of the popular methods that is a baseline for other algorithms. Multiple techniques modify the SMOTE to enhance their oversampling algorithm and model performance.

However, the current algorithms have drawbacks on the generating data process. Firstly, the method may create noisy data if the referenced point or its nearest neighbor is noisy data. Secondly, some procedures assign the same weight to all instances, which cause an overgeneralization problem. Thirdly, the new points from the oversampling technique are located in the overlapping region, which may decrease the performance of the model. Lastly, some methods do not focus on the data distribution that may change the distribution of variables massively.

To handle these problems, this research has proposed the oversampling technique, namely Synthetic Minority Based on Probabilistic Distribution (SyMProD), to balance classes in the dataset. There are three main steps in our method. Firstly, the noise removal step applies the Z-score normalization and compares it to the noise threshold to eliminate the noise before taking forward to the next process. Secondly, the minority point selection step calculates and assigns the probability to the minority instances based on the probability distribution, which also avoids the overlapping region. Then, this step selects minority instances as referenced points to produce new points. Lastly, the instance synthesis step creates synthetic instances using the referenced points from the previous step and its nearest neighbors. This step generates points from multiple minority instances to cover the distribution of the minority class instead of creating them in the line between two points.

This study has assessed the performance of the proposed method using 17 imbalanced datasets, i.e., ten datasets from UCI imbalanced dataset repository, six simulated datasets, and the liver cancer dataset from Chulabhorn Hospital. The method has applied the stratified five-fold cross-validation with three classifiers which are random forest, logistic regression, and support vector machine classifiers. The evaluation metrics in our assessment are AUC, F-Measure, and G-Mean metric. Furthermore, our proposed method

was compared with eight conventional oversampling techniques, i.e., SMOTE, Borderline SMOTE, Safe-Level SMOTE, Cluster SMOTE, MWMOTE, k-means SMOTE, A-SUWO, and ADASYN. The results have shown that our proposed technique can enhance the performance of the prediction model. Moreover, the SyMProD is statistically better than other methods based on the evaluation metrics in most cases. In terms of the liver cancer dataset, this study has improved the prediction model compared to the previous research. The top five important features are Neutrophil, Eosinophil, ALP, Total Bilirubin, and Basophil laboratory test.

In summary, this research has proposed a new oversampling algorithm called Synthetic Minority Based on Probabilistic Distribution (SyMProD) oversampling to balance the imbalanced dataset. Our method has eliminated the noisy data and assigned the probability to each minority instance. These steps have avoided the noisy generation, overgeneralization, and overlapping class problems. The proposed method has generated samples that covered the minority distribution by considering the referenced minority points and multiple nearest neighbors. This study has assessed three evaluation metrics, i.e., AUC, G-Mean, and F-Measure, with other eight oversampling algorithms and three classifiers, i.e., Random Forest, Support Vector Machine, and Logistic Regression. The results show that the classification performance of the data from our proposed method is higher than the other conventional oversampling techniques in most datasets.

REFERENCES

1. Zhang, J., Bloedorn, E., Rosen, L. and Venese, D., 2004, "Learning Rules from Highly Unbalanced Data Sets", **Fourth IEEE International Conference on Data Mining (ICDM'04)**, 1-4 November 2004, United Kingdom.

2. Phua, C., Alahakoon, D. and Lee, V., 2004, "Minority Report in Fraud Detection: Classification of Skewed Data", **ACM SIGKDD Explorations Newsletter**, Vol. 6, pp. 50-59.

3. Amin, A., Anwar, S., Adnan, A., Nawaz, M., Howard, N., Qadir, J., Hawalah, A. and Hussain, A., 2016, "Comparing Oversampling Techniques to Handle the Class Imbalance Problem: A Customer Churn Prediction Case Study", **IEEE Access**, Vol. 4, pp. 7940-7957.

4. Santos, M.S., Abreu, P.H. , García-Laencina P. J., Simão A. and Carvalho A., 2015, "A New Cluster-Based Oversampling Method for Improving Survival Prediction of Hepatocellular Carcinoma Patients", **Journal of Biomedical Informatics**, Vol. 58, pp. 49-59.

5. Yu, H., Ni, J., Dan, Y. and Xu, S., 2012, "Mining And Integrating Reliable Decision Rules for Imbalanced Cancer Gene Expression Data Sets", **Tsinghua Science and Technology**, Vol. 17, pp. 666-673.

6. Elhassan, T. and Aljurf, M., 2016, "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.

7. He, H. and Garcia, E.A., 2009, "Learning from Imbalanced Data", **IEEE Transactions on Knowledge and Data Engineering**, Vol. 21, pp. 1263-1284.

8. Haixiang, G., Yijing, L., Shang, J., Mingyun, G., Yuanyue, H. and Bing, G., 2017, "Learning from Class-Imbalanced Data: Review of Methods and Applications", **Expert Systems with Applications**, Vol. 73, pp. 220-239.

9. Douzas, G., Bacao, F. and Last, F., 2018, "Improving Imbalanced Learning through a Heuristic Oversampling Method Based on K-Means and SMOTE", **Information Sciences**, Vol. 465, pp. 1-20.

10. Das, B., Krishnan, N.C. and Cook, D.J., 2014,"RACOG and wRACOG: Two Probabilistic Oversampling Techniques", **IEEE Transactions on Knowledge and Data Engineering**, Vol. 27, pp. 222-234.

11. Piri, S., Delen, D. and Liu, T., 2018, "A Synthetic Informative Minority Over-Sampling (SIMO) Algorithm Leveraging Support Vector Machine to Enhance Learning from Imbalanced Datasets", **Decision Support Systems**, Vol. 106, pp. 15-29.

12. International Agency for Research on Cancer, 2018, **Liver Cancer Fact Sheet** [Online], Available: http://gco.iarc.fr/today/data/factsheets/cancers/11-Liver-fact-sheet.pdf. [2020, April 13].

13. National Statistical Office of Thailand, 2014, **Number of Deaths by Leading Cause of Death and Sex, Whole Kingdom: 2007-2014** [Online]. Available: http://service.nso.go.th/nso/web/statseries/statseries09.html. [2019, June 6].

14. Dua, D. and Graff, C., 2019, **UCI Machine Learning Repository** [Online], Available: http://archive.ics.uci.edu/ml. [2020, January 10].

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

16. Naseriparsa, M. and Kashani, M.M.R., 2013, "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.

17. Wang, K.J. and Adrian, A.M., 2013, "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, pp. 408-412.

18. He, H., Bai, Y., Garcia, E.A. and Li, S., 2008, "ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning", **International Joint Conference on Neural Networks (IJCNN 2008)**, 1-8 June 2008, Hong Kong, pp. 1322-1328

19. Weiss, G.M., McCarthy, K. and Zabar, B., 2007, "Cost-Sensitive Learning vs. Sampling: Which is Best for Handling Unbalanced Classes with Unequal Error Costs?", **DMIN**, Vol. 7, pp. 35-41.

20. Zheng, Z., Cai, Y. and Li, Y., 2016, "Oversampling Method for Imbalanced Classification", **Computing and Informatics**, Vol. 34, pp. 1017-1037.

21. Batista, G., Prati, R.C. and Monard, M.C., 2004, "A Study of the Behavior of Several Methods for Balancing Machine Learning Training Data", **ACM SIGKDD Explorations Newsletter**, Vol. 6, pp. 20-29.

22. Chawla, N.V., Bowyer, K.W., Hall, L.O. and Kegelmeyer, W.P., 2002,"SMOTE: Synthetic Minority Over-Sampling Technique", **Journal of Artificial Intelligence Research**, Vol. 16, pp. 321-357.

23. Han, H., Wang, W.Y. and Mao, B.H., 2005, "Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning", **International Conference on Intelligent Computing**, 23-26 August 2005, Hefei, China, pp. 878-887.

24. Liu, A., Ghosh, J. and Martin, C.E., 2007, "Generative Oversampling for Mining Imbalanced Datasets", **DMIN**, pp. 66-72.

25. Gao, M., Hong, X., Chen, S., Harris, C.J. and Khalaf, E., 2014, "PDFOS: PDF Estimation Based Over-Sampling for Imbalanced Two-Class Problems", **Neurocomputing**, Vol. 138, pp. 248-259.

26. Nekooeimehr, I. and Lai-Yuen, S.K., 2016, "Adaptive Semi-Unsupervised Weighted Oversampling (A-SUWO) for Imbalanced Datasets", **Expert Systems with Applications**, Vol. 46, pp. 405-416.

27. Bunkhumpornpat, C., Sinapiromsaran, K. and Lursinsap, C., 2009, "Safe-level-SMOTE: Safe-level Synthetic Minority Over-Sampling Technique for Handling the Class Imbalanced Problem", **Pacific-Asia Conference on Knowledge Discovery and Data Mining**, 27-30 April 2009, Bangkok, Thailand, pp. 475-182.

28. Barua, S., Islam, M.M., Yao, X. and Murase, K., 2012, "MWMOTE-Majority Weighted Minority Oversampling Technique for Imbalanced Data Set Learning", **IEEE Transactions on Knowledge and Data Engineering**, Vol. 26, pp. 405-425.

29. Dattagupta, S.J., 2017, **A Performance Comparison of Oversampling Methods for Data Generation in Imbalanced Learning Tasks**, Master's Degree Dissertation, Department of Statistics and Information Management, NOVA Information Management School, pp. 1-28.

30. Cieslak, D.A., Chawla, N.V. and Striegel, A., 2006, "Combating Imbalance in Network Intrusion Datasets", **GrC**, pp. 732-737.

31. Madan, S. and Dana, K.J., 2016, "Modified Balanced Iterative Reducing and Clustering Using Hierarchies (M-BIRCH) for Visual Clustering", **Pattern Analysis and Applications**, Vol. 19, No. 4, pp. 1023-1040.

32. Douzas, G. and Bacao, F., 2018, "Effective Data Generation for Imbalanced Learning Using Conditional Generative Adversarial Networks", **Expert Systems with Applications***,* Vol. 91, pp. 464-471.

33. Chawla, N.V., Lazarevic, A., Hall, L.O. and Bowyer, K.W., 2003, "SMOTEBoost: Improving Prediction of the Minority Class in Boosting", **European Conference on Principles of Data Mining and Knowledge Discovery**, 22-26 September 2003, Cavtat-Dubrovnik, Croatia, pp. 107-119.

34. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y., 2014, "Generative Adversarial Nets", **Advances in Neural Information Processing Systems,** pp. 2672-2680.

35. Fallahi, A. and Jafari, S., 2011, "An Expert System for Detection of Breast Cancer Using Data Preprocessing and Bayesian Network", **International Journal of Advanced Science and Technology**, Vol. 34, pp. 65-70.

36. Sug, H., 2012, "Improving the Prediction Accuracy of Liver Disorder Disease with Oversampling" **Applied Mathematics in Electrical and Computer Engineering**, pp. 331-335.

37. Lokanayaki, K. and Malathi, A., 2013, "Data Preprocessing for Liver Dataset Using SMOTE", **International Journal of Advanced Research in Computer Science and Software Engineering**, Vol. 3, No. 11.

38. Lemaître, G. , Nogueira, F. and Aridas, C.K., 2017, "Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning," **Journal of Machine Learning Research***,* Vol. 18, pp. 1-5.

39. Kovács G., 2019, "Smote-Variants: a Python Implementation of 85 Minority Oversampling Techniques", **Neurocomputing**, Vol. 366, pp. 352-354.

40. Suh, Y., Yu J., Mo, J., Song, L. and Kim, C., 2017, "A Comparison of Oversampling Methods on Imbalanced Topic Classification of Korean News Articles", **Journal of Cognitive Science**, Vol. 18, pp. 391-437.

41. Zhang, H. and Wang, Z., 2011, "A Normal Distribution-based Over-sampling Approach to Imbalanced Data Classification", **International Conference on Advanced Data Mining and Applications**, 17-19 December 2011, Beijing, China.

42. Cieslak, D.A., Hoens, T.R., Chawla, N.V. and Kegelmeyer, W.P., 2012, "Hellinger Distance Decision Trees are Robust and Skew-Insensitive", **Data Mining and Knowledge Discovery**, Vol. 24, pp. 136-158.

43. Galar, M., Fernández, A., Barrenechea, E., Bustince, H. and Herrera, F., 2016, "Ordering-based Pruning for Improving the Performance of Ensembles of Classifiers in the Framework of Imbalanced Datasets", **Information Sciences**, Vol. 354, pp. 178-196.

44. Laerd Statistics, 2018, **Pearson Product-Moment Correlation** [Online], Available: https://statistics.laerd.com/statistical-guides/pearson-correlation-coefficient-statistical-guide.php [2020, March 3].

45. Sungkasubun, P., Siripongsakun, S., Akkarachinorate, K., Vidhyarkorn, S., Worakitsitisatorn, A., Sricharunrat, T., Singharuksa, S., Chanwat, R., Bunchaliew, C., Charoenphattharaphesat, S. and Molek, R. , 2016, "Ultrasound Screening for Cholangiocarcinoma Could Detect Premalignant Lesions and Early-Stage Diseases with Survival Benefits: A Population-based Prospective Study of 4,225 Subjects in an Endemic Area", **BMC Cancer**, No. 1, pp. 346-353.

CURRICULUM VITAE

NAME	Mr. Intouch Kunakorntum
Date of Birth	29 August 1995
EDUCATIONAL RECORD	
HIGH SCHOOL	High School Graduation Assumption College Thonburi School, 2012
BACHELOR'S DEGREE	Bachelor of Engineering (Computer Engineering) King Mongkut's University of Technology Thonburi, 2016
MASTER'S DEGREE	Master of Engineering (Computer Engineering) King Mongkut's University of Technology Thonburi, 2019
SCHOLARSHIP/ RESEARCH GRANT	Research Grant for Graduate Student Petchra Prajomklao Scholarship, 2017
PUBLICATION	1) Ongsuk, S., Komolvatin, S., Kunakorntum, I., Phunchongharn, P., Amonyingcharoen, S. and Hinthong, W., 2018, "An Adaptive Cancer Prognosis Framework for Cholangiocarcinoma Based on Machine Learning Techniques", **1st IEEE International Conference on Knowledge Innovation and Invention (ICKII)**, 23-27 July 2018, Jeju, South Korea, pp. 82-85. 2) Kunakorntum, I., Hinthong, W., Amonyingchareon, S. and Phunchongharn, P., 2019, "Liver Cancer Prediction Using Synthetic Minority Based on Probabilistic Distribution (SyMProD) Oversampling Technique", **IEEE 10th International Conference on Awareness Science and Technology (iCAST)**, 23-25 October 2019, Morioka, Japan, pp. 1-6.