An Investigation of SMOTE based Methods for Imbalanced Datasets with Data Complexity Analysis

Nur Athirah Azhar, Muhammad Syafiq Mohd Pozi, Aniza Mohamed Din, Adam Jatowt

Abstract—Many binary class datasets in real-life applications are affected by class imbalance problem. Data complexities like noise examples, class overlap and small disjuncts problems are observed to play a key role in producing poor classification performance. These complexities tend to exist in tandem with class imbalance problem. Synthetic Minority Oversampling Technique (SMOTE) is a well-known method to re-balance the number of examples in imbalanced datasets. However, this technique cannot effectively tackle data complexities and it also has the capability of magnifying the degree of complexities. Also, the performance of the SMOTE is still not satisfactory. Therefore, various SMOTE variants have been proposed to overcome the downsides of SMOTE either by combining SMOTE with other algorithms or modifying the existing SMOTE algorithm. This paper aims to comparatively review the algorithms applied in SMOTE variants and investigate which data complexities are being addressed in what variants. Series of experiments are conducted on 24 binary class imbalanced datasets to observe the changes in the data complexity measures after SMOTE variants were applied in these datasets. The evaluation metrics like G-Mean and F1-Score are also analyzed to investigate the difference in classification performance between SMOTE variants.

Index Terms—SMOTE, imbalanced learning, data complexity, imbalanced data, oversampling, classification

1 Introduction

Nowadays, it is hard to see a commercialized computer-based application without some sort of machine learning implementation applied in it. Due to the advancement of computing infrastructure, the research paradigm in machine learning has significantly changed from almost pure theoretical model research [1], [2], [3] in the late 1990s, to rapid prototyping based research. Raw computing power and software increased and facilitated usage of those computing resources and made them accessible to many users. Consequently, problems related to handling a large amount of data, such as in computer vision [4], [5] and computational linguistics [6], [7], [8] could be dealt with in the recent years.

Regardless of the advancement in machine learning research, there are assumptions that need to be adhered (or at least used as guidelines) for machine learning methods to work effectively (which are built into many machine learning algorithms). Such assumptions are defined as follows:

Assumption 1. The objective is focused on maximizing the machine learning model accuracy.

Assumption 2. Future data has similar distribution characteristics as training data, from which the machine learning model is derived from [9].

- *Nur Athirah Azhar is with the Malaysian Investment Development Authority, Malaysia. E-mail:nurathirah@mida.gov.my*
- *Muhammad Syafiq Mohd Pozi is with the Institute for Advanced and Smart Digital Opportunities, School of Computing, Universiti Utara Malaysia and with the Institute of IR 4.0, Universiti Kebangsaan Malaysia. E-mail:syafiq.pozi@uum.edu.my*
- *Aniza Mohamed Din is with the Data Science Research Lab, School of Computing, Universiti Utara Malaysia. E-mail:anizamd@uum.edu.my*
- *Adam Jatowt is with the Department of Computer Science and Digital Science Center at the University of Innsbruck, Innsbruck, Tirol, Austria. E-mail:adam.jatowt@uibk.ac.at*

Machine learning relies on these assumptions to solve real-world problems. However, those assumptions are commonly violated due to various uncertainties in real-world cases during the learning stages. For example, due to the difficulty in obtaining real data describing the cancer cell [10], [11], the ability for a classifier to discriminate between cancerous and non-cancerous cells, in terms of classification accuracy can be superficial. The importance of the issue of class imbalance could be simply demonstrated by considering an extreme case when a naive machine learning model just predicts every cell as a non-cancerous cell, so it could easily accomplish the first assumption of maximizing the machine learning model accuracy. To illustrate, many classic machine learning classifiers are based on the minimization of training error, which is defined (loosely) by the following cost function J:

$$J = \frac{1}{2n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 \tag{1}$$

where $\hat{y}_i$ is the predicted value and y_i is the true value. Based on Equation 1, if the data consists of 10 positive samples and 90 negative samples, a naive classifier will just predict everything to be negative, so that it can achieve 90% classification accuracy, instead of predicting otherwise, which will only result in 10% accuracy. Thus, back to the cancer example, if we do not properly consider the true performance of the classifier, then, it will create more problems in nearest future. Some other example domains that suffer from imbalanced classification problems are fraud detection [12], network intrusion [13] and anomaly detection [14].

The definition of an imbalanced classification problem can be described whenever the number of representatives in one class is larger than in the other classes, thus the distribution of the dataset is skewed. Due to insufficient representation in minority classes, the underlying pattern of the minority class is then difficult to be captured and properly generalized. As a result, the classifier tends to exhibit bias towards the majority class that has prevalent examples [15], [16]. Therefore, one needs to verify whether the data at hand is suffering from imbalanced classification problem

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2022.3179381

or not. This can be checked either at the data level itself, that is, by measuring the class imbalance ratio such that, if the class imbalance ratio is very high, then it is very likely that the data is heavily in the imbalanced state. Another verification method is through the classification performance itself. Performance metrics such as F1-Score [17], [18] and G-Mean [19], [20] can be used to indicate how balanced the classification performance is for any newly derived classification model.

Once the degree of imbalanced classification problem is properly defined, then a mitigation plan can be drafted. If the definition of imbalanced classification problem is represented as a result of the classification performance itself (through F1-Score or G-Mean), then one can try to improve the classification performance through an exhaustive model selection task, so that the classification performance on minority classes is increased, without sacrificing the overall classification performance [21], [22], [23]. On the other hand, if the definition is represented as the imbalance ratio of the dataset, then one can try to solve the issue by acquiring more data on the less represented class. Since usually the data is difficult to be collected or is impossible to obtain, the class distribution can be balanced by removing redundant examples (undersampling) from the existing dataset or adding synthetic examples (oversampling) into it through various resampling techniques [24], [25], [26], [27].

Another option aside from data level approach for solving imbalanced datasets could be the algorithm level approach. While the former approach alters the dataset to rebalance the class distribution, the latter approach changes the classifiers that are directly involved in the learning of imbalance class distribution. Cost-sensitive learning is one example of the methods belonging to this approach. It amplifies the significant class with less examples by imposing a high cost if its examples are misclassified as other class [15].

Oversampling approaches are however easier to be applied and reason about as well as there are multiple variants of such techniques; hence, they have been commonly utilized. **S**ynthetic **M**inority **O**ver-sampling **TE**chnique (**SMOTE**) [24] is the most well-known oversampling technique proposed to even the data distribution between majority and minority classes. The technique has been successfully used to solve the imbalanced dataset problem in various researches and real-world domains [28], [29], [30]. Fig. 1 illustrates the citation trend for the original paper of SMOTE [24].

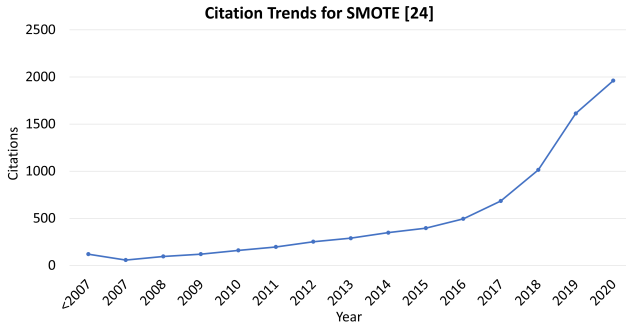


Fig. 1: Total number of citations for the original SMOTE algorithm [24] since before year 2007 to year 2020 (Source: SCOPUS[1]).

Based on Fig. 1, from before year 2007 until year 2020, we can see an increasing pattern of citation counts based on the original SMOTE paper [24]. In fact, despite the popularity of deep learning, SMOTE citations increase exponentially after year 2017. Nevertheless, this does not stop machine learning researchers to propose various new SMOTE variants in solving imbalanced dataset. Due to the fact that SMOTE is unable to handle very well the data complexities like noise examples, class overlap and small disjuncts [31], [32], [33], many SMOTE variants have been proposed to solve or overcome the SMOTE limitations.

This paper presents a review of various SMOTE variants that have been proposed in literature for solving imbalanced dataset. The main contributions of this survey are three-fold:

1) The evolution of SMOTE variants from the introduction of the SMOTE algorithm [24] until current year (2021) is thoroughly reviewed, and the variants are categorized into specific groups.
2) We quantitatively analyze the different variants in our experimentation framework.
3) A connection to describe a relation between the change of data complexity and the change of classification performance that are provided by SMOTE variants to machine learning classifier is established.

Other papers have reviewed and discussed the application of SMOTE variants on imbalanced datasets in terms of different aspects such as the performance metrics, time complexity, application of different operating principles and challenges that emerge [34], [35], [36], [37]. The reviews have been done either empirically or theoretically in order to depict the variability of performance since SMOTE variants are proposed with different problem domains and various combinations of original SMOTE with other methods. The author in [34] evaluates the performance of SMOTE variants on datasets that have variety of intrinsic characteristics in order to guide the researchers on which SMOTE variants should be applied for a certain type of a dataset. Apart from that, the discussion also incorporates the time complexity and operating principles of SMOTE variants since they might render useful understanding regarding the performance of the variants. A review of SMOTE variants in a conceptual manner which contrasts with the previous review paper has been proposed in [35]. The discussion was done on the development of SMOTE variants alongside with the issues related to the application of these variants. The author manages to deliver an extensive future perspectives that can be done for SMOTE improvements especially in big data issues.

Comparative analysis of SMOTE variants based on performance metrics presented in [36] only covers some of the earlier proposed variants. In contrast, we overview and analyze 5 more latest variants than this paper so that the readers could gain more information about how good these SMOTE variants work on imbalanced datasets.

Similar work has been done in [37] but the authors attempt to include the analysis of how the oversampling strategies adapted by some SMOTE variants affect the effectiveness of these variants' performances on various imbalanced datasets. We expand the analysis of the SMOTE variants by adding data complexity into the review instead of merely observing the evaluation metrics.

Nevertheless, the number of review papers related to SMOTE variants is not that extensive and many of them are quite old, as well as there is still a large area of aspects that can be observed and analyzed. Therefore, this paper attempts to investigate the relationship between data complexity and SMOTE variants with

1. https://www.scopus.com/home.uri

the objective to observe how the latter's performance will affect the changes in the former's measurement. No existing works have ever associated data complexity in imbalanced datasets with SMOTE variants. All of the prior papers focused on IR aspect which is typically deemed as the prime issue related to imbalanced classification problem. Comparison analysis based on IR is however rather simplistic because datasets in real-life tend to be complicated. For example, linearly separable imbalanced dataset could be tackled effortlessly by any SMOTE variant in lieu of the degree of IR. For this reason, the complexity measure is employed in our experiments to distinguish the performance of each variant.

The rest of this paper is organized as follows. Section 2 overviews different types of data complexities and complexity measurements for imbalanced datasets, and how they might affect the learning performance. Section 3 reviews how SMOTE variants solve common issues on imbalanced datasets, corresponding to the discussed data complexities in the first fold. This paper will review SMOTE variants that were published in the top-tier machine learning journal and conferences as well as ones considered to have important contribution in SMOTE improvements. Section 4 describes the conducted experimentation study for performance review to empirically illustrate the capability of SMOTE variants in improving classification task on imbalanced dataset. Section 5 discusses the significance of the obtained experimentation results. Finally, Section 6 presents the conclusion and future perspectives related to the main findings.

2 Data Complexities in Imbalanced Dataset

In general, the class imbalance problem can be observed in the form of causal effect, that is, when a particular class distribution is skewed in one direction, and the classifier fails to appropriately capture the underlying pattern hidden in the minority class, leading to poor classification performance on minority class' examples. While there are many reasons of why the classifier performs poorly on any given dataset, from the perspective of data, it is found that there are three other sources besides class imbalance problem that are usually accompanying imbalanced dataset, thus contributing to the poor classification performance. Those sources are noise examples, class overlap, and small disjuncts which are known as data complexities in imbalanced dataset. As stated in [38], analyzing data complexity is crucial in determining a suitable approach to solve the problem. Each of the data complexity type like noise examples, class overlap and small disjuncts will be discussed in this section.

2.1 Noise Examples

Noise examples have been a challenge to be addressed by researchers as they are common in real-word datasets. These kind of examples can poorly affect the classifier performance in terms of prediction accuracy rates [39]. The existence of an example with a different class label within another class region is what is considered as noise or label noise as presented in Fig. 2, which indicates that there could be various degree of noise in an imbalanced dataset. Label noise linked to anomalies, abnormalities and novelties is another type of noisy data apart from feature noise [31] (one associated with missing, incorrect and incomplete feature values). Data labelling which is done by humans who naturally may make mistakes could be one of the triggers that can induce label noise as humans tend to be biased and there might be element of subjectivity included during labelling. According to the study proposed in [40], malicious attack as in data poisoning [41] where attackers intentionally modify the class label could corrupt the data and consequently result in label noise.

Insufficient useful information is one of the factors that can lead to noisy environment [42] besides misconceptions during data labelling [40]. Noisy instances with minority class label located inside the majority class region should be given more attention than noise examples with majority class label within minority region. This is due to the fact that the minority class itself does not provide enough information and its examples are prone to be misclassified during classification as suggested by [43]. Aggravation of complexity of induced models might happen due to noisy environment where the classifier will face difficulties in distinguishing the class label of an example and in determining the complex decision boundary thus prolonging the training running time [44].

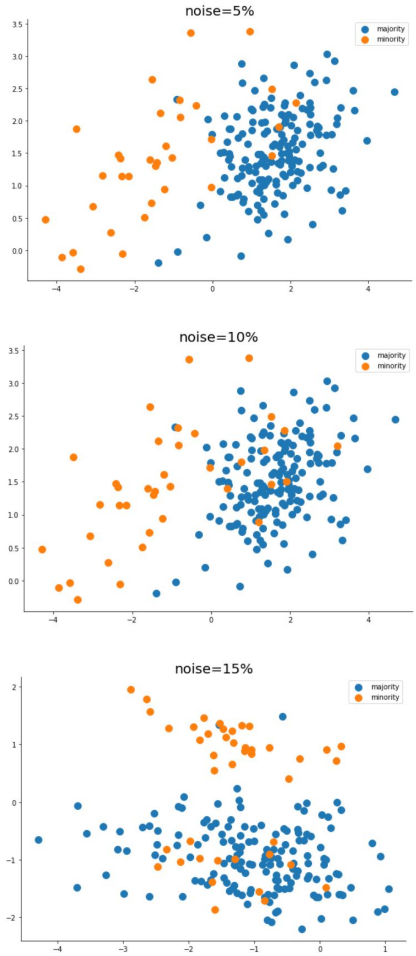


Fig. 2: Examples of imbalanced data affected by different levels of label noise; 0.05, 0.10, 0. 15. Such noisy environment hinders the predictive performance of a classifier due to insufficient information related to minority class which eventually will reduce the significance of this class while increasing the misclassification rate.

2.2 Class Overlap/Separability

Class overlap or class separability is one of the major problems responsible for degradation of classifier's performance. It refers to examples that belong to different classes and are found in an ambiguous region due to the closeness in their feature values [45], [46] which can be observed in Fig. 3. Studies conducted in [47], [48] suggest most standard classifiers manage to produce an

acceptable predictive accuracy rate despite the imbalance ratio of datasets. Therefore, it can be observed that imbalance ratio is not the only factor needed to be looked upon when addressing imbalanced classification problem, but also class overlap is, since it impacts the accuracy of classification [49], [50], [51].

Region around decision boundary usually contains examples from both minority and majority classes where the distribution of minority examples in this overlapping region is sparser and less dense compared to majority class. Hence, minority examples in this region could be easily mistaken as majority examples as they are outnumbered by majority examples. However, generating more minority examples within borderline region or overlapping region could tackle this class overlap problem and enhance the classification of minority class [52].

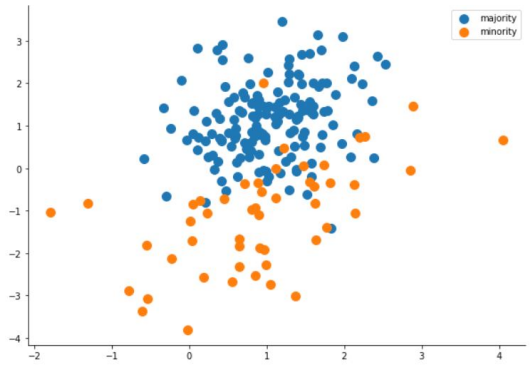


Fig. 3: The class overlap between minority and majority examples so that the decision boundary cannot be figured out clearly. This problem could lead to an increment in the classification complexity.

2.3 Small Disjuncts

The presence of small-sized clusters containing examples from one class and located in the region of another class due to underrepresented subconcepts in imbalanced problem is known as small disjuncts [53] (presented in Fig. 4). Within-class imbalance which corresponds to this small disjuncts problem is yet to be solved as most of the recent research works manage to alleviate only the between-class imbalance. [54], [55], [56] imply that small disjuncts problem is closely related to noise since an increase in the number of noise examples could induce more small disjuncts in the dataset. Most classifier models have difficulty in covering minority class that has been divided into several sub-concepts, leading to lower accuracy of identifying minority examples.

Clustering-based resampling method is one of the solutions proposed in solving within-class imbalance besides the detection of small disjuncts where the data is grouped, forming clusters, and then resampling is performed in the clusters either by generating useful examples or removing some useless examples [57], [58], [59], [60].

2.4 Complexity Measures in Imbalanced Dataset

The common and simple complexity measure available for imbalanced dataset is the ratio between sizes of minority class distribution and majority class distribution. It is commonly known as Imbalance Ratio (IR). Equation 2 gives the IR definition:

$$\text{IR} = \frac{\#neg}{\#pos} \tag{2}$$

where $\#neg$ is the total number of samples in majority class and $\#pos$ is the total count of samples in minority class.

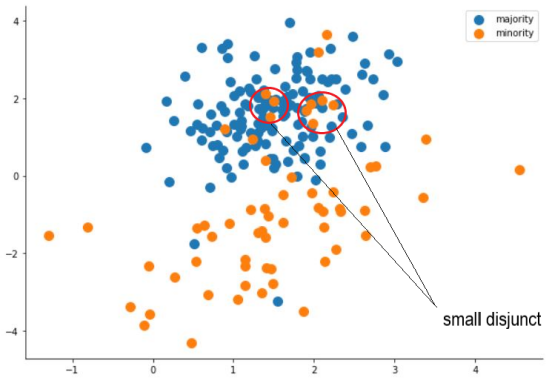


Fig. 4: Sub-clusters of minority class located inside majority class region which could give arise to small disjuncts problem.

Regardless, there are other complexity measures proposed to compliment IR values. In [61], a Fischer's discriminant ratio (Fischer's ratio) has been proposed to measure the means and variance of dataset classes. Equation 3 defines the Fischer's ratio.

$$\text{Fischer's ratio} = \frac{(\mu_{c_i} - \mu_{c_j})^2}{\sigma_{c_i}^2 + \sigma_{c_j}^2} \tag{3}$$

In Equation 3, μ_{c_i} and μ_{c_j} is the mean of feature values associated with class c_i and class c_j, respectively, while $\sigma_{c_i}^2$ and $\sigma_{c_j}^2$ is the variance of features values associated with class c_i and c_j, respectively. Fischer's ratio outputs the maximum f among all input features. The higher the Fischer's ratio value, the simpler is the classification problem concerning feature separability, and vice versa.

Barella et. al. [62] proposed to use Minimum Spanning Tree (MST) that connects all the examples from the dataset based on their distance, regardless of which class the example belongs to. The idea is to identify at least two examples from two different classes that are close to each other. It is assumed that those examples exist in the borderline of two classes, and their frequency relative to all examples in the dataset is defined as $N1$. Thus, the complexity measure $N1$ is defined such as follows:

$$N1 = \frac{1}{n}\sum_{i=1}^{n} I((\mathbf{x_i}, \mathbf{x_j}) \in \text{MST} \wedge y_i \neq y_j) \tag{4}$$

where $I(\mathbf{x_i}, \mathbf{x_j})$ represents a connection between instances $\mathbf{x_i}$ and $\mathbf{x_j}$, and MST represents the set of all connections in the tree. Higher N1 value indicates the need for more complex boundaries to separate the classes and/or that there is a large amount of overlapping between the classes.

Another unique approach is to consider the existence of small disjuncts in the imbalanced dataset in the form of Geometric Small Disjuncts Index (GSDI) [63]. Once the small disjuncts are identified in dataset, the GSDI will measure the complexity of those disjuncts based on formula defined in Equation 5.

$$GSDI = \left(\prod_{c=1}^{C} \frac{\sum_{i=1}^{\delta_c} \exp(-|\Delta_i^c|) \times \frac{m_c^{i'}}{m_c^i}}{\sum_{i=1}^{\delta_c} \exp(-|\Delta_i^c|)} \right)^{\frac{1}{C}} \tag{5}$$

In Equation 5, δ_c is the number of disjuncts within the c-th class. The Δ_i^c is the i-th disjunct within the c-th class containing m_c^i test points, where $m_c^{i'}$ are the points that are correctly classified. Higher GSDI indicates the identified small disjuncts in particular dataset will help in reducing the learning complexity, while lower GSDI gives arise to the learning complexity.

3 SMOTE: Synthetic Minority Oversampling Technique

There are several ways of handling imbalanced datasets, which can be categorized into two methods; data level and algorithm level. Algorithm level method enhances a classifier's accuracy on the minority class by tuning the existing classification algorithm. For example, to highlight any misclassification or correct classification in the positive (minority) class, the cost imposed on the false negatives can be set to be larger than the one for false positives. This paper focuses on the data level methods which use preprocessing steps in rebalancing the class distribution by either implementing oversampling or undersampling to decrease the imbalance ratio (IR) of data-sets.

Oversampling techniques are defined as replicating minority examples randomly to increase the minority class size or generating new examples synthetically from the interpolation of minority examples. In the undersampling technique, the process occurs in the majority class wherein the examples are eliminated blindly or by using a heuristic approach to balance the number of examples in both classes [64]. However, both oversampling and undersampling have their own drawbacks where replicating examples could lead to over-fitting as it does not give additional information about the minority class while examples reduction in majority class could cause the loss of potentially valuable information especially in the case of small-sized datasets [27], [60], [65].

In the case of large-sized imbalanced datasets, the undersampling method performance is quite effective and convenient because the information loss is insignificant due to downsizing the size of majority samples [66], [67]. Despite all that, it has been observed that there is lesser risk when applying the oversampling method to real-world datasets compared to undersampling [68], [69]. As a result, the oversampling method has gained more attention, resulting in many variants created to make sure that misclassification of minority class samples in imbalanced dataset problems can be reduced, thus enhancing the accuracy of classification [70], [71], [72], [73]. Fig. 5 illustrates the example resulting data after performing undersampling and oversampling process, over imbalanced dataset.

Synthetic Minority Oversampling Technique (SMOTE) was introduced by Chawla et. al. [24] to mitigate the overfitting problem occurring from random oversampling. [66], [74], [75] and [76] show that SMOTE has good robustness in noisy environment compared to random oversampling. SMOTE, as described in Algorithm 1, adds synthetic data using k-nearest neighbour algorithm by calculating the Euclidean distance and chooses the nearest neighbours for a given example from the neighbourhood provided as an input. The synthetic data is created by identifying the density of examples clustered in minority class so that SMOTE can find the actual examples which will give valuable information about the minority class labels and outliers that might influence the outcome incorrectly. The description of SMOTE can be concisely expressed in the following form [37]:

$$D_{new} = D_i + (D_l - D_i) \times w \tag{6}$$

Where D_{new} = synthetic data, D_i = examples from minority class, D_l = one of k-nearest neighbour from D_i, w = random number between 0 and 1.

SMOTE obviously could address class imbalance problem through generation of synthetic examples without involving removal of existing examples - the procedure which can cause information loss. Moreover, linear interpolation between selected minority examples may prevent redundant and replicated examples from being produced as in random oversampling algorithm [77]. Therefore, SMOTE successfully overcomes over-fitting problems and enhances the learning task of most classifiers.

Algorithm 1 SMOTE

Input:
$\mathbf{X}_+ \leftarrow [x_1, x_2, \ldots x_n], x \in \mathbb{R}^m, m \in \mathbb{Z}^+$ ▷ Minority class samples
$T \leftarrow n$ ▷ Number of minority class samples
$N \leftarrow$ *Amount of SMOTE*
$k \leftarrow$ *Number of nearest neighbors*
Output: *(N/100) * T synthetic minority class samples*

procedure SMOTE Procedure
  if $N < 100$ **then**
    $\mathbf{X}_+ \leftarrow Permutate(\mathbf{X}_+)$ ▷ Permutate the order of $\mathbf{X}_+$
    $T \leftarrow (N/100) \times T$
    $N \leftarrow 100$
  end if
  $N \leftarrow (int)(N/100)$ ▷ Assumed to be integral multiples of 100.
  $\mathbf{X}_+^{'} \leftarrow [x_1, x_2, \ldots x_T] \in \mathbf{X}_+$ ▷ T minority class samples
  $\mathbf{D} \leftarrow \emptyset$ ▷ Initialize empty set of distance matrix
  $\mathbf{X}^{''} \leftarrow \emptyset$ ▷ Initialize empty set of selected samples
  $\mathbf{S} \leftarrow \emptyset$ ▷ Initialize empty set of synthetic samples

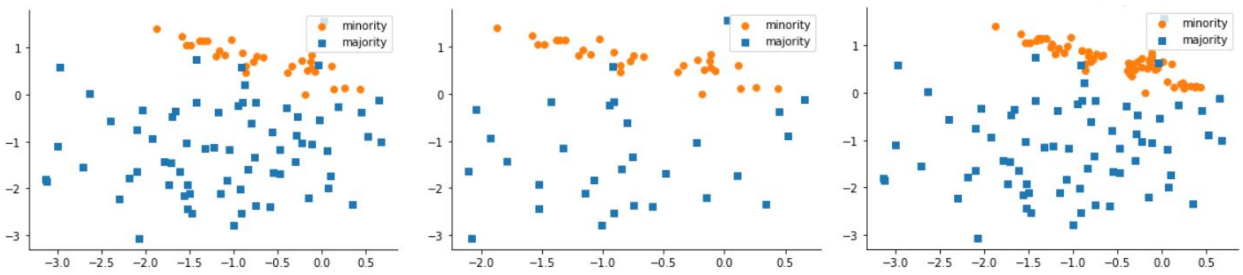
  ▷ Computing the nearest neighbour of each selected sample
  for $x_i \in \mathbf{X}_+^{'}$ **do**
    for $x_j \in \mathbf{X}_+^{'}$ and $i \neq j$ **do**

      ▷ Computing the distance between x_i and x_j, where $p = 2$
      $\mathbf{D}_{i,j} \leftarrow \left(\sum_{k=1}^{m} |x_i^k - x_j^k|^p\right)^{\frac{1}{p}}$
    end for
  end for
  for $x_i \in \mathbf{X}_+^{'}$ **do**
    for $g \leftarrow 1$ to k **do**
      $\mathbf{X}^{''} \leftarrow x_j, argmin_j(D_i)$ ▷ Select x_j with closest distance to x_i
      $\mathbf{D}_{i,j} \leftarrow \emptyset$ ▷ To prevent reselecting the same sample again
    end for
  end for

  ▷ Generating synthetic data
  for $x_i \in \mathbf{X}_+^{'}$ **do**
    $N^{'} \leftarrow N$
    $q \leftarrow Randomize(1, k)$ ▷ A random integer between 1 and k
    while $N^{'} \neq 0$ **do**
      $gap \leftarrow Randomize(0, 1)$ ▷ A vector of random number
      $\triangle x = x_i - \mathbf{X}^{''}[i \times q + q]$ ▷ Compute a difference vector
      $s \leftarrow x_i + gap \odot \triangle x$ ▷ A synthetic sample is generated
      $\mathbf{S}.append(s)$ ▷ Populating the set $\mathbf{S}$
      $N^{'}$--
    end while
  end for
end procedure

Regardless, there is still much to be improved when it comes to conventional SMOTE. For the start, one of its drawbacks is that SMOTE does not have a mechanism to control where the synthetic example should be placed. It is not uncommon that new synthetic examples appear at inappropriate positions in the hyperspace especially in the opposite region because the noisy information is being distributed [32] as shown in Fig. 6. Increasing noise examples will eventually lead to over-generalization problem. Apart from that, SMOTE often exacerbates class overlap problem by adding more overlapping examples as it cannot differentiate overlapping regions from so-called safe regions [68]. Next, there is danger of generating useless synthetic examples as well as the algorithm is only able to tackle between-class imbalance while setting aside the small disjuncts or within-class imbalance. Taking within-class imbalance into account can give quite an impact when trying to classify minority examples more accurately [60].

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2022.3179381



(a) Original data. (b) Undersampled data. (c) Oversampled data.

Fig. 5: Three figures show the difference between (a) original data, (b) undersampled data, (c) oversampled data, performed in order to solve imbalanced classification problem.

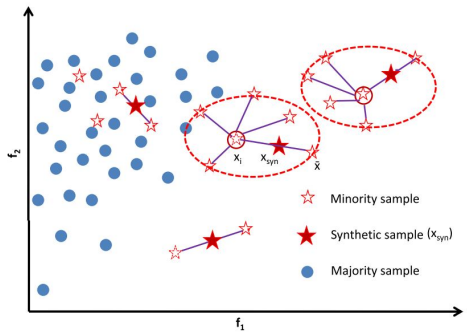


Fig. 6: SMOTE linearly interpolates a randomly chosen minority class example and one of its k=5 nearest neighbors. The presence of noise might cause minority examples generated in the majority class region [78].

3.1 Overview of SMOTE Variants Strategies

Before diving into SMOTE discussion, we first overview some common strategies used in order to improve the effectiveness of SMOTE. The excellent performance shown by SMOTE algorithm has recently encouraged the appearance of many variants either through modification of SMOTE or extension of SMOTE [79]. The former considers the characteristics of examples when changing the region of oversampling whereas the latter refers to the combination of SMOTE algorithm either with pre-processing or post-processing approaches while preserving the basic algorithm. The proposal of SMOTE variants can be better understood by looking into the provenance relation plot as shown in Fig. 9.

Modification of SMOTE is normally associated with change-direction algorithm where it attempts to redirect the synthetic examples into certain preferred areas so that it can control the distribution of these examples. Techniques proposed in [25], [33], [75], [80], [81] are of change-direction-based SMOTE variants which produce synthetic examples as close as possible to specific regions and most of these techniques prefer regions with high concentration of minority examples.

Filtering, clustering and weighting are the most common techniques integrated with SMOTE algorithm in extension of SMOTE. Different types of filtering techniques are implemented in [68], [82], [83], [84] for data cleaning purpose by detecting and removing either noise examples or overlap examples. Some SMOTE extensions [45], [60], [85], [86] suggested that clustering and weighting algorithm are best to be applied together while other techniques [83] only combined SMOTE with a clustering technique. Clustering algorithm could guarantee that generation of synthetic examples occurs within the minority class region and weighting algorithm is applied to determine which minority examples are appropriate and suitable for oversampling to produce useful information. Both algorithms are frequently found in SMOTE extensions that try to handle data with class overlap or small disjuncts problems.

Various algorithms have been intentionally incorporated into the original SMOTE algorithm to mitigate the aforementioned weaknesses as well as to enhance the effectiveness of generation of synthetic examples. Table 2 shows SMOTE variants categorized into groups based on the algorithm implemented in the main method. It can also be stated that these variants also are capable of tackling one or more previously mentioned data complexities due to the fact that SMOTE variants are often developed with multi-purpose objectives such as alleviating the data complexities that originally co-exist in the data or those that have been intensified by limitations of SMOTE.

The next three subsections will be discussing SMOTE variants in a chronological order starting from the oldest techniques to the most recent ones. The explanation is done in such a way that it should provide useful insights regarding the evolution of SMOTE variants. Besides we believe that it should be intuitive for the reader to follow the timeline of SMOTE variants.

3.2 Mitigating Noise Examples Problem

In the application of SMOTE algorithm, the presence of noise examples in imbalanced datasets can aggravate the noisy environment and degrade the performance of classifiers. Instead of generating synthetic examples in the minority region, the examples might be introduced in the majority region and this phenomenon is called over-generalization. Thus, it is important to remove the noise examples prior to oversampling or to filter them posterior to oversampling.

SMOTE-Edited Nearest Neighbour Rule (**SMOTE-ENN**) is a hybrid sampling technique which combines SMOTE and data cleaning [82]. The motivation behind this technique is to discard noise examples both from minority and majority classes using ENN after application of SMOTE on the training set [87]. ENN detects noise examples by randomly choosing any examples and considers their three nearest neighbours [88]. Therefore, the noise examples that are generated during SMOTE can be deleted by applying ENN afterwards in order to reduce noisy environment that will confuse the classifier. Even though SMOTE-ENN is able to remove noise examples using data cleaning approach, a problem might happen when the minority class has sparse distribution and small in size since the hard-to-learn minority examples overlapping with majority examples tend to be removed by this technique. This phenomena can easily downgrade the generalization of underpresented minority class due to compression of class space [32].

Modification of SMOTE like Borderline-SMOTE (**BD-SMOTE**) [80] does not employ noise removal, instead it does ignore noise examples and does not include them in the oversampling process. This technique takes into account the majority examples distribution and makes assumption that examples existing close to borderline have more contribution to classification success [89]. As minority examples are grouped into three categories which are safe, borderline and noise based on the number of minority and majority examples in their neighbourhood prior to oversampling, any noise examples that exist in minority class are filtered effectively [90]. Then, only examples in the borderline region are used for oversampling as they are easily misclassified during classification. It can be said that noise filtering also occurs in majority class as noise minority examples are being excluded. There are two versions of BD-SMOTE, such as **B-SMOTE1** and **B-SMOTE2** [80], in which the synthetic examples are created between the borderline examples from minority class and their nearest neighbours from the same class in the former, while the latter generates synthetic examples by adding majority class examples in the interpolation of a minority example. The downside of this technique is that an example can only be identified as noise if all of its nearest neighbours are from majority class. If there are two minority examples placed closely to each other within the majority class region, they are not considered as noise since one of the nearest neighbours is of the same class [91].

While BD-SMOTE manages to filter noise examples by choosing only borderline examples for oversampling, Safe-Level-SMOTE **(SL-SMOTE)** [25] does noise filtering based on the safe level value of minority examples. This technique generates synthetic examples in the safe region which contains minority examples with higher safe level value [92]. Firstly, the safe value for every minority examples is measured based on the number of minority examples in its nearest neighbours. A ratio of safe level between a minority example and a nearest neighbour is then calculated to determine the position of synthetic examples as well as to identify noise examples that need to be filtered. More synthetic examples are located around examples with highest safe level value which means safe region has higher density of synthetic examples and noisy environment is not further aggravated. However, oversampling near the safe region might introduce repeated and non-valuable synthetic examples [29] thus leading to overfitting [93].

SMOTE-RSB* introduced by [83] is another extensions of SMOTE that also performs data cleaning after SMOTE application besides SMOTE-ENN. This technique removes synthetic examples by using Rough Set Theory (RST) and the lower approximation (B_*) of a subset. Data cleaning could be effective to enhance classification since SMOTE can generate low-quality synthetic examples from noise examples. The approximation of similarity relations between synthetic examples and the minority examples by using RST allows the synthetic examples that do not belong to the minority class to be excluded from the final result [94], [95]. After applying SMOTE for minority class in the training dataset, this technique creates a set of final output to include all the original dataset. Next, in order to detect which synthetic examples belong to lower approximation of minority class, the similarity value for every synthetic examples is measured and compared with a given threshold. The examples are then included in the set of algorithm's final output as they do not have similarities in majority class which means noise examples are removed effectively. Similarity value parameter in this technique can be considered as the degree of similarity between synthetic examples and majority examples.

ENN filtering algorithm might be suffering from failure of detecting all noise examples that are identified based on the distance of nearest neighbours. To improve the solution, authors in [84] proposed **SMOTE-IPF**, a combination of SMOTE with another noise filter known as Iterative-Partitioning Filter (IPF). This technique depends on classifiers for identification of noise examples, instead of using distance metric just like SMOTE-ENN. Noise examples introduced in the data as well as the existing ones from both majority and minority classes can be removed easily. In addition, this technique is effective in addressing the problems caused by noise and borderline examples [96], [97]. The algorithm starts with division of the oversampled training data into n equal-sized subsets where union of any $n-1$ subsets will be used independently for training multiple n classifiers. Noise identification is made with two voting schemes called consensus and majority. An example is considered as noisy if all classifiers misclassify it in the consensus scheme whilst in majority scheme, if more than half of the classifiers do misclassification of the example. Then, the noise examples are added into a new set, S. For a number of consecutive iterations t, if the size of noise examples detected in each of the iterations lower than a $p\%$ of the original training dataset size, then the iterations will stop. Lastly, all the noise examples in set S are deleted from the training dataset before classification is further performed. Even though IPF relies on certain classifiers, it can face similar risk as ENN in handling small-sample imbalanced dataset [32] such as removing minority examples from the overlapping region where minority class has lower local distribution compared to majority class.

Most of the variants of SMOTE that have been proposed are able to enhance the performance of essentially any classifiers except for **SMOTE-PSO** [98]. This technique is devoted on mitigating the problem of Support Vector Machine (SVM) in imbalanced datasets since SVM performance is seriously affected while being able to remove noise examples as Particle Swarm Optimization (PSO) algorithm is employed to evolve the synthetic examples [33]. Support vectors (SVs) are identified first from the dataset to be used for generating minority examples in the region between the decision boundary and a SV called margin where the density of minority class is getting lower. The generated synthetic examples are evolved by using PSO algorithm to discard noise examples therefore moving the separating hyperplanes towards the majority class. However, SMOTE-PSO is only effective on small datasets since it is computationally expensive.

SMOTE variant such as Noise Reduction A Priori Synthetic Over-Sampling method (**NRAS**) [99] focuses on eliminating noise examples before oversampling with goal to improve the prediction of minority class examples. In imbalanced datasets, classification is more biased towards majority class but this technique shifts the bias towards the minority class to achieve the intended goal with the risk of misclassification of majority class examples [100]. Rather than using typical distance metric such as Euclidean distance to identify noise examples like BD-SMOTE and SL-SMOTE [101], the measure for distance metric in NRAS is the difference of conditional probability of belonging to the minority class which has never been applied in any other SMOTE variants. Despite that, NRAS also determines a noise example by doing comparison between the number of nearest neighbours from majority examples and a threshold value, which is done in BD-SMOTE algorithm.

A grouped SMOTE algorithm with noise filtering mechanism

(**GSMOTE-NFM**) [68] implements a different way of identifying and filtering noise examples by comparing the probability density of the same example in two different classes. Unlike SMOTE-ENN and SMOTE-IPF which use distance between nearest neighbours and relies on the voting schemes of classifiers respectively. The prior distributions of both majority and minority classes are learned respectively beforehand through construction of Gaussian-Mixture Model (GMM). After filtering noise examples, GMMs are adopted on the remainder data and the probability distribution information is used to split the minority class into three different groups; safe, borderline and outlier. Lastly, before SMOTE implementation, examples of each group is given an individual parameter k which depends on their characteristics of distributions. The demerits of GSMOTE-NFM are the construction of GMM is time-consuming and if the data has high imbalance ratio, the extremely sparse minority examples will jeopardize the accuracy of GMM model [33].

SMOTE algorithm of limiting the radius (**LR-SMOTE**) [75] modifies the original SMOTE algorithm just like BD-SMOTE does but their difference can be observed from the objectives perspective. While BD-SMOTE aims to change the region of oversampling, LR-SMOTE goal is to set a limit range to the oversampling space since noise generation can happen due to boundless space for synthetic examples generation using SMOTE. Existing noise examples can worsen the noise generation as these examples uncontrollably invade the opposite class region [102]. Therefore, LR-SMOTE performs denoising step by using SVM algorithm for classifying the training data to find misclassified minority examples. These examples are then placed into one sample set and their k number of minority nearest neighbours is observed for detecting and deleting noise examples. Then, K-means clustering algorithm [103] is adopted to find the centre point of minority class for calculating the Euclidean distance, d between this point and every non-noise minority examples as well as the average of all distances, d_{mean}. Synthetic examples generation formula is described such in Equation 7:

$$D_{new} = D_i + (D_c - D_i) \times rand(0, M) \tag{7}$$

where D_{new} = synthetic data, D_i = examples from minority, D_c = minority class centre point, $rand(0,M)$ = random number between 0 and 1. For this technique, the range for generation of synthetic examples is changed into 0 and M where M is the ratio between d_{mean} and d thus synthetic examples are produced in the region closer to this point. The final step includes filtering the excessive synthetic examples to balance the size of dataset. Experimentation results have shown that the effectiveness LR-SMOTE algorithm is proportional to the imbalance ratio of imbalanced dataset.

SMOTE-ENN, SMOTE-RSB*, SMOTE-IPF, SMOTE-PSO and GSMOTE-NFM are SMOTE variants that implement particular noise filter and introduce additional parameters in the algorithm. On the other hand, self-adaptive robust SMOTE (**RSMOTE**) [33] is able to boost the performance of classifiers without applying these two steps. Noise examples in minority class region are identified and filtered out heuristically based on the number of its k nearest neighbours [80] so that these examples will not be operated in the next step. Next, a new dataset is created to gather all the noise-free remaining minority examples, resulting in only safe and borderline minority examples are included in this set. In order to differentiate between these two type of minority examples, the ratio of the total distance of every minority examples with their nearest neighbours from the same class to the total distance of this example from the nearest opposite class examples, also known as relative density is calculated. 2-means clustering is deployed to divide these examples inside the set into two clusters, one with higher relative density of examples are residing inside it while the other has smaller relative density of examples. For each minority examples in both clusters, the number of majority examples in its nearest neighbours helps to define chaotic level which is used to reweigh the amount of synthetic examples needs to be generated. Safe examples are given higher weights to be oversampled so that fewer synthetic examples are placed near the borderline region. In this way, decision boundary will be more obvious besides the overlapping can be avoided effectively.

Authors in [104] proposed **SMOTE-LOF** technique that integrates SMOTE with an unsupervised anomaly detection algorithm known as local outlier factor algorithm (LOF) [105]. Even though LOF was initially applied to distinguish outliers in a dataset, this algorithm is employed in this study for noise detection since most outliers can be considered as noise. SMOTE-LOF aims to detect noise from synthetic minority examples, not from the original dataset which is similar to SMOTE-RSB* algorithm. In other words, noise detection and deletion takes place after implementation of SMOTE just like other SMOTE variants such as SMOTE-ENN, SMOTE-RSB* and SMOTE-IPF except for NRAS which removes noise examples before oversampling process. LOF algorithm is only performed on minority class where the distance between every minority examples are calculated using Euclidean distance and reachability distance calculations so that the LOF value of a minority example can be measured. Minority example that is considered as outlier based on the LOF value is removed from the datasets except for original minority example despite it is identified as one. SMOTE-LOF manages to avoid repetition of noise deletion which happens in some SMOTE variants [84], [106] by using LOF algorithm to identify outlier (noise).

In summary, there are two SMOTE variant approaches when considering mitigating the noise examples' problem. Noise examples can be filtered either prior to SMOTE application (pre-processing) or post SMOTE application (post-processing). Table 1 tabulates the SMOTE variants that fall into these two approaches.

TABLE 1: Two SMOTE variants' approaches in solving noise examples problem.

Approach	SMOTE Variants
Pre-processing	BD-SMOTE [80], SL-SMOTE [25], NRAS [99], GSMOTE-NFM [68], LR-SMOTE [75], R-SMOTE [33]
Post-processing	SMOTE-ENN [82], SMOTE-RSB* [83], SMOTE-IPF [84], SMOTE-PSO [98], SMOTE-LOF [104]

3.3 Mitigating Class Overlap/Separability Problem

The existing class overlap problem might be aggravated after SMOTE is applied to address the imbalance problem since it can introduce more examples in the overlapping region and make decision boundary becomes unclear. The importance of borderline examples that are located in the overlapping region is an issue that need to be given more attention because they can either be noise examples or valuable examples that contain information [35]. Thus, most of SMOTE variants which trying to overcome class overlap focus on the hard-to-learn examples located around class boundaries.

SMOTE-Tomek Links (**SMOTE-TL**) [82] alleviates the overlap class problem by removing the pairs of examples that belong to different classes. Tomek links can be defined as pair of minimally Euclidean distanced neighbours of different classes and it can either be used to do under-sampling task or data cleaning task [107]. In SMOTE-TL, synthetic examples are initially generated before Tomek Links are identified and later the distance between a synthetic examples and both examples in the pair are measured. The distance must be smaller than the distance of the example pair for elimination of the pair that is considered as overlapped, thus any examples that cause overlapping are removed effectively. Tomek links that are going to be eliminated mostly found in the borderline region due to overlapping tends to take place here. Nevertheless, noise generation cannot be successfully avoided as SMOTE-TL depends on the adjacencies between borderline examples [47]. Fig. 7 presents how this method operates.

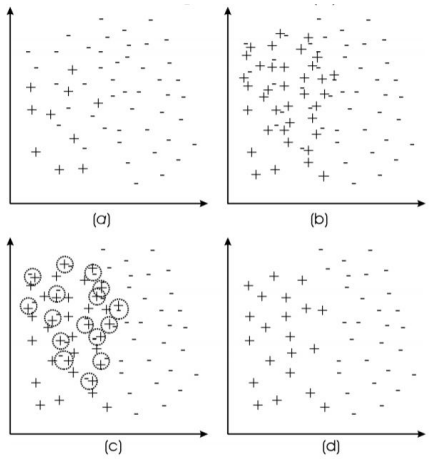


Fig. 7: (a) presents an imbalanced dataset with little information regarding minority class. In (b) and (c), SMOTE-TL is implemented to generate more synthetic minority examples and to identify pairs of overlapping examples from both minority and majority classes are detected respectively. Lastly, (d) shows the case when the pairs have been deleted from the dataset according to (c), thus removing the overlapping region that could increase the complexity of decision boundary [82].

BD-SMOTE tackles the class overlap problem by highlighting the importance of minority examples in the borderline region and these examples are chosen to be oversampled using linear interpolation which can increase the density of minority class in this region and subsequently might enhance the visibility of class border. Where as SL-SMOTE focuses on the region with high concentration of minority examples for generating synthetic examples which could prevent adding more examples in the existing overlapping region like SMOTE does. Apart from that, the synthetic examples in SL-SMOTE are well separated from the majority class compared to BD-SMOTE because they are found far from the borderline region [81]. In contrast to SMOTE-TL, overlapping examples from the training data still can be found since SL-SMOTE does not dispose them, instead it just avoids the overlapping problem from being exacerbated.

By using the large margin principle, Margin-guided Synthetic Over-sampling (**MSYN**) [108] technique chooses synthetic examples with large margin on both minority and majority classes which eventually could make the separability better [109]. SMOTE does not consider the number of synthetic examples that need to be generated whereas MSYN determines this amount both for minority class itself and for each minority examples before oversampling [79]. Then, the algorithm performs oversampling using SMOTE on every minority examples and selects the top best synthetic examples to be minority synthetic examples based on minimizing the margin loss for the major class and maximizing margin gains for the minority class while the rest are eliminated.

Clustering is one of the algorithms used to deal imbalanced datasets with class overlap, therefore Density-Based Synthetic Minority Over-sampling Technique (**DBSMOTE**) [110] is proposed where the algorithm consists integration of Density-Based Spatial Clustering of Applications with Noise (DBSCAN) and SMOTE. Motivated by BD-SMOTE, this technique carefully oversamples minority examples in the borderline region to maintain the accuracy of majority class that declines in BD-SMOTE. DBSCAN algorithm [111] is applied because of its ability to find cluster with an arbitrary shape by using the minimum density level estimation where the minimum number of neighbours, $minPts$ and radius eps are the parameters.

Firstly, the algorithm constructs a new data structure called directly density-reachable graph [112] which transformed from the DBSCAN algorithm and works as an underlying weighted directed graph. This graph contains core examples that have examples more than $minPts$ within eps distance and border examples which lie in their neighbourhoods. We can say the latter are directly density-reachable to the former. The border example can be formed if at least one of a pair examples is directly density-reachable to others, that is how an arbitrarily shaped cluster can be discovered. Then, generation of synthetic example occurs along the shortest path between a pseudo-centroid of a minority class cluster and each example in that class. Subsequently, overlapping can be avoided as the synthetic example is located on the shortest path that is formed inside the minority cluster. However, over-fitting still could not be solved due the density of synthetic examples are higher close to the cluster centroid [113].

Majority Weighted Minority Oversampling Technique (**MWMOTE**) [85] also employs clustering algorithm to make sure that all synthetic examples lie inside the minority class regions, such that it prevents them from overlapping with majority class regions [114], [115]. This technique aims to improve the initial selection process besides producing informative and necessary synthetic examples from hard-to-learn minority examples as shown in Fig. 8. When compared to MWMOTE, other SMOTE variants like SMOTE, BD-SMOTE and SL-SMOTE tend to synthesize duplicated and meaningless examples. Essentially, MWMOTE has three steps involved. Initially,it identifies the most difficult to learn minority class examples from the original set of minority class which can be found near to the borderline of both classes. Instead of applying the well-known k-NN algorithm, nearest neighbours are observed by using the euclidean distance. k-NN algorithm cannot find the hard-to-learn minority examples appropriately even though they are located within borderline region if these examples k-nearest neighbours are of the same class. Next, three factors are taken into consideration before assigning selection weight to every example in the set of identified examples: 1) the closeness of minority examples to majority class region, 2) the sparsity of minority class cluster and 3) the sparsity of majority class cluster. Lastly, this technique constructs M sub-clusters from S_{min} using modified hierarchical clustering algorithm prior to oversampling. This can prevent the synthetic examples from overlapping with the opposite class examples such that they all lie in the minority class cluster.

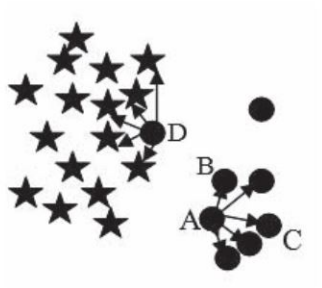


Fig. 8: *A* and *B* which are placed near the borderline of majority class cannot be identified as hard-to-learn examples by using *k*-nearest neighbour (*k*-NN) since the nearest neighbours have the same class label. MWMOTE algorithm is able to detect these examples as hard-to-learn by implementing a different approach nearest neighbours. The identification of these examples is crucial in MWMOTE as it focuses on generating more synthetic examples nearby example that probably has more helpful information compared to the others [85].

Even though MWMOTE can identify borderline examples better than BD-SMOTE, small-sized sub-clusters containing significant information that located far away from the majority class can be easily undetected [86]. Therefore, Adaptive semi-unsupervised weighted oversampling (**A-SUWO**) [45] is introduced to overcome the above problem by implementing a semi-unsupervised hierarchical clustering approach that enables detection of small clusters. A-SUWO technique can be considered as an extension of MWMOTE as this technique also focuses on finding the hard-to-learn minority examples and performs oversampling based on the weights assigned to each minority examples in the sub-clusters while applying different clustering algorithm that adaptively determines the size of synthetic examples in every sub-cluster.

Before clustering the minority examples, noise examples are identified and eliminated using BD-SMOTE method such that they will not be included in the next process. While MWMOTE only performs hierarchical clustering on minority class, this technique performs the clustering on both minority and majority classes so that the number of synthetic minority examples overlapping with majority examples can be minimized . However, the hierarchical clustering is modified for application on minority class in order to observe the existence of majority sub-clusters. Considering two minority sub-clusters are going to be combined, if the Euclidean distance of a majority cluster to both of them less than the a distance threshold T, there will be no combination happening, preventing majority examples from being included in the cluster and that is how semi-unsupervised hierarchical clustering works in this technique. On the assumption that more synthetic examples needed in sub-clusters with higher misclassification error rate, cross validation is used to calculate the misclassification error of each minority sub-cluster or in other words, the classification complexity. Sub-clusters that have higher misclassification error will be given a bigger size compared to those with lower error rates since they need more information in the cluster that will increase the classification accuracy [86].

After that, the probability distribution of every minority examples in each sub-clusters needed to be measured based on the weights before oversampling can be done. Instead of assigning weights on minority examples in each sub-cluster based on the distance to all candidate borderline majority examples as in MWMOTE, A-SUWO finds the average Euclidean distance to the *k*-nearest neighbours from majority class. This new approach gives more chances for minority examples that nearer to majority examples and have high misclassification rate to be oversampled. Apart from that, it performs weighting algorithm on all sub-clusters separately which result in small-sized sub-clusters are oversampled and not overlooked, contrary to MWMOTE that tends to ignore them. Then, generation of synthetic examples occurs only between minority examples from the same cluster where the initial selection of example is made by sampling from the pre-determined probability distribution, thus overlapping could be avoided as all synthetic examples fall within the cluster. Despite being able to alleviate over-generalization due to weights assignment on hard-to-learn minority examples and also overlapping problem, this algorithm has a higher complexity which might lead to poor sampling [115]. In addition, over-fitting might even happen when synthetic examples are concentrated near the borderline regions, encouraging duplication of examples [86].

Most of the listed SMOTE extension such as BD-SMOTE, SL-SMOTE, DBSMOTE, MWMOTE, and A-SUWO address the class overlap problem through initial selection either aiming to enhance the separability of examples from different class or to increase the density of synthetic examples near the decision boundary. Contrary to those methods, SMOTE-IPF [84] applies iterative filtering approach to handle the aforementioned problem by not filtering out the sparse minority examples in the overlapping region that usually assumed to be noise, instead eliminating more majority examples. This technique prone to remove majority examples from overlapping region in some type of datasets which causes the density of minority examples becomes higher. Because of that, visibility of minority class becomes clear and the decision boundary is more regular [52] since class overlap is removed.

A-SUWO is an effective technique in solving the imbalanced datasets problem so the algorithm is further improved to synthesize more useful and expanded borderline minority examples which introduces a new technique called an improving adaptive semi-unsupervised weighted oversampling (**IA-SUWO**) [86]. This technique follows directly the first two steps of A-SUWO algorithm. Firstly, the examples from both minority and majority classes are clustered using semi-unsupervised hierarchical clustering. Then, the size of synthetic examples in every minority sub-clusters are decided according to classification complexity and cross validation. The improvements made by IA-SUWO can be found in the last step, the generation of synthetic examples.

IA-SUWO proposes a new weighting algorithm by combining an improving majority weighted minority oversampling (IMWMO) method with α-method. The former measures the average Euclidean distance which has been implemented in A-SUWO while the latter uses least squares supported number spectrum values as the distance metric. The combination of these two method forms a new weight assignment method that could make a good decision boundary. The ratio of influence between these two can be calibrated so that mutual correction can be attained. Instead of implementing *k*-NN algorithm to generate synthetic examples like A-SUWO does, *k** information nearest neighbors (*k**INN) approach is used to avoid the density of these examples becomes too high near the decision boundary. *k**INN indicates the example with the smallest sample information distance (SID) from a candidate example in the same cluster for this technique. Overall, the improvements introduced in IA-SUWO makes this technique better at handling imbalanced datasets with higher complexity compared to A-SUWO due to the ability of generates more valuable and scattered examples.

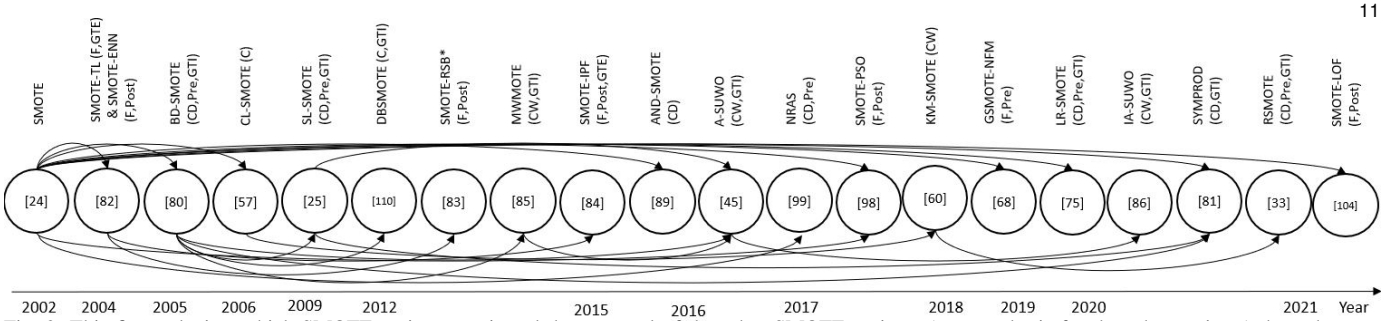


Fig. 9: This figure depicts which SMOTE variants motivated the proposal of the other SMOTE variants (or were basis for the other variants), how they are related to each other and their algorithm and approaches. The following are the abbreviations used for describing different variants: Filtering-based - F, change-direction-based - CD, clustering-based - C, clustering+weighting-based - CW, noise pre-processing - Pre, noise post-processing - Post, "generate-to-eliminate" examples in the class border region - GTE, "generate-to-increase" examples in the class border region - GTI.

Taking into account the distribution of classes in imbalanced datasets when oversampling could alleviate the class overlap problem which has been done in a new technique called a synthetic minority based on probabilistic distribution (**SyMProD**) oversampling [81] apart from MWMOTE, A-SUWO and IA-SUWO. The algorithm starts with implementation of Z-score normalization on both majority and minority classes and the Z-score value is used for comparison with a noise threshold value (NT), to identify any noise examples exist in the original data which are then not included in the training data. Next, rather than calculating the probability distribution according to weight [45], [86], this technique converts the ratio of closeness factor derived from both classes. Minority examples that will be selected for oversampling based on the probability distribution possibly can reduce the overlapping problem. This is due to class closeness factor is compared to the cutoff threshold (CT), which helps to only choose minority examples located in the minority class region, ignoring the examples found in the opposite region. The number of selected minority examples follows the difference between majority and minority examples in the original dataset. Lastly, the generation of synthetic examples step involves searching for M minority nearest neighbours of the referenced minority example. The real value of synthetic example will be generated within the range of real value of the nearest neighbours, ensuring the example is not duplicated and over-generalization could be hindered. Apart from that, there will be no overlapped synthetic examples generated since they are found in the minority distribution.

Even though LR-SMOTE [75] and RSMOTE [33] are direction-based algorithm, they still manage to handle class overlap problem in some way without identifying the hard-to-learn examples. generating more synthetic examples near the centre point of minority class or around safe region reduce addition of overlapping examples thus avoiding the problem from being exacerbated.

In conclusion, there are two main approaches in dealing with class overlap and class separability problem. The first approach is based on a "generate-to-eliminate" examples in the class border region. SMOTE-TL and SMOTE-IPF fall into this approach. The second approach is based on a "generate-to-increase" examples in the class border region so the visibility of the class border is increasingly clear to learning algorithm. The SMOTE variants that fall into this approach are BD-SMOTE, SL-SMOTE, MSYN, DBSMOTE, MWMOTE, A-SUWO, IA-SUWO, SyMProD, LR-SMOTE, and RSMOTE.

3.4 Mitigating Small Disjuncts Problem

Existence of small-sized clusters or sub-concepts in imbalanced datasets is easily overlooked by SMOTE therefore this technique fail to address within-class imbalance while being able to solve between-class imbalance. To make it worse, small disjuncts encourage the degradation of classifiers since they are easily overlooked due to their sparse distribution and usually being mistaken as noise. Apart from that, small disjuncts might increase noisy environment if they are considered for oversampling process.

Cluster-SMOTE (**CL-SMOTE**) [57] integrates k-means clustering with SMOTE where it applies SMOTE oversampling in every k clusters of minority class after dividing the class using k-means clustering. CL-SMOTE manages to handle within-class imbalance only to some extent [86] since the number of synthetic examples in each cluster is not considered besides it has difficulty in determining the optimal number of clusters. Apart from that, [60] claims that this technique fails at completely removing small disjuncts problem.

Small disjuncts might induce noise examples when example from one small disjunct is identified as nearest neighbour of seed example from other disjuncts during oversampling due to inappropriate k value of nearest neighbours. Hence, **AND-SMOTE** [89] instinctively determines the optimal k-values for every minority examples as different examples required different number of nearest neighbours used in SMOTE depending on the characteristics of data. Minority example and its nearest neighbours are contained in a rectangular region and then this region is expanded and accumulated towards the next nearest neighbour. The change of precision values based on the size of nearest neighbour in every iteration helps to identify the optimal k-value by choosing the drop point precision. Instead of using random number in the range $[0, 1]$ to generate examples on the lines between two points, this technique uses a vector of random number in the same range. The difference between CL-SMOTE and AND-SMOTE is obvious where the former has the ability to lift within-class imbalance but cannot eliminate small disjuncts where as the latter avoids noisy environment caused by small disjuncts but does not attempt to solve within-class imbalance.

KMeans-SMOTE (**KM-SMOTE**) [60] applies k-means clustering together with SMOTE to tackle class imbalance problem as well as noisy environment issue. In [57], clustering algorithm is implemented only on minority class where as in KM-SMOTE, clusters are formed for both minority and majority classes. In this cluster, the number of minority and majority examples will be used to calculate the imbalance ratio of the cluster which then determines either this cluster is suitable for oversampling or not. Then, KM-SMOTE observes the distribution of minority examples in the filtered cluster by measuring its density and sparsity so that it can assign appropriate number of synthetic examples to be generated for each clusters before implementing oversampling [30]. This is how within-class imbalance is tackled

as this technique does not blindly generate the same amount of synthetic examples in every cluster. Apart from that, KM-SMOTE properly prevents introducing more noise examples and overfitting problem by generating informative examples inside the minority class cluster [116]. Nonetheless, this technique still faces the same limitations as CL-SMOTE where the optimal number of clusters yet to be looked upon and clustering algorithm used does not applicable when the clusters have irregular shape.

Despite the fact that A-SUWO [45] and IA-SUWO [86] alleviate within-class imbalance due to their high capability of detecting minority class sub-concepts located far from decision boundary, it is still not clear whether small disjuncts problem are being taken care of.

In summary, cluster-based oversampling techniques are widely practiced by researchers when the data contains small disjuncts problem that degrades the classification performance gravely. Observing the density of a cluster as well as the sparsity of examples are very crucial in determining the amount of synthetic examples needed by the cluster. Because of that, the synthetic examples are distributed appropriately between the clusters and eventually solves within-class imbalance.

Lastly, we show the overview picture of how SMOTE variants relate to each other in terms of their provenance. Fig. 9 shows the provenance relation observed from SMOTE and its variants. For example, we can see that the original SMOTE [24] gives rise to many other variants such as BD-SMOTE [80] which further is basis for SL-SMOTE [25], A-SUWO [45], MWMOTE [85], KM-SMOTE [60] and many more variants. SMOTE-TL [82] and SMOTE-ENN [82] which are ones of the earliest SMOTE variants also do motivate the proposal of cleaning-based variants like SMOTE-RSB* [83] and SMOTE-IPF [84].

TABLE 2: Classification of SMOTE variants (chronologically ordered), with algorithm and data complexity categorization. The following are the abbreviations used for describing different variants: Filtering-based - F, change-direction-based - CD, clustering-based - C, clustering+weighting-based - CW, noise pre-processing - Pre, noise post-processing - Post, "generate-to-eliminate" examples in the class border region - GTE, "generate-to-increase" examples in the class border region - GTI.

SMOTE Variant	Algorithm	Approach	Data Complexities		
			Noise	Class Overlap	Small Disjuncts
SMOTE-TL [82]	F	GTE		√	
SMOTE-ENN [82]	F	Post	√		
BD-SMOTE [80]	CD	Pre,GTI	√	√	
CL-SMOTE [57]	C	-		√	
SL-SMOTE [25]	CD	Pre,GTI	√	√	
DBSMOTE [110]	C	GTI		√	
SMOTE-RSB* [83]	F	Post	√		
MWMOTE [85]	CW	GTI		√	
SMOTE-IPF [84]	F	Post,GTE	√	√	
AND-SMOTE [89]	CD	-			√
A-SUWO [45]	CW	GTI		√	
NRAS [99]	CD	Pre	√		
SMOTE-PSO [98]	F	Post	√		
KM-SMOTE [60]	CW	-		√	√
GSMOTE-NFM [68]	F	Pre	√		
LR-SMOTE [75]	CD	Pre,GTI	√	√	
IA-SUWO [86]	CW	GTI		√	
SyMProD [81]	CD	GTI	√	√	
RSMOTE [33]	CD	Pre,GTI	√	√	
SMOTE-LOF [104]	F	Post	√		

4 Experimental Study

4.1 Experimentation Design

For the experimental review, based on selected 8 SMOTE-variants and 24 binary imbalanced datasets, we evaluated each SMOTE variant performance through testing 5 commonly used machine learning classifiers, such as Gaussian Naive Bayes (**GNB**) [117], *k*-Nearest Neighbour (*k*-**NN**) [118], Decision Tree (**DT**) [119], and Random Forest (**RF**) [120]. Those 8 selected SMOTE-variants are the ones that have relatively high citations, with publicly available source code and ones that cover different period of time from 2002 to 2021. Note that, most of the datasets are originally multi-class imbalanced datasets obtained from the UCI Machine Learning Repository but they have been reorganized as binary imbalanced datasets to fit the purpose of our paper. Other researchers [34], [50], [77], [81], [98] commonly use these datasets as the benchmark datasets when it comes to imbalanced classification problem. This is because of the variety of IR values, sizes of datasets, and even the amounts of minority examples in the datasets that allow portraying the general performance of SMOTE variants on datasets with diverse properties. The datasets are tabulated in Table 3, and are sorted based on IR value in ascending order.

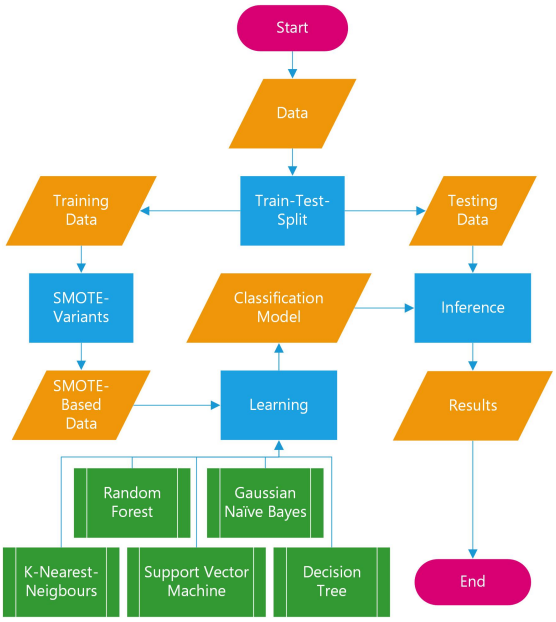


Fig. 10: Flowchart describing the experimentation design.

The experimentation parameters are summarized in Table 4, which consists of the name of each SMOTE variant, the name of each classifier and evaluation method. The parameters for SMOTE variants and classifiers essentially employ default values except for SVM in which the parameter *C* is set to a range of values so that we can determine the best performance metric result provided by this classifier.

Fig. 10 describes the flowchart of the experiments conducted for this paper. Five-fold cross validation is performed in every experiment by using stratified shuffle split to make sure the ratios of classes are approximately preserved and every generated split is not similar with others to an extent. 70% of the dataset is constructed for training set whereas the remaining 30% is assigned as the test set. Then SMOTE variants (as tabulated in Table 4) are applied on the training set in order to balance the imbalance ratio in that training set. Next, a classification model is fit on the transformed training set so that the learning algorithm (classifier) can discover the relationship between the input values and the class labels (output). The learning process is significant in order to proceed with prediction task. This classification model is then used on test set to predict the class labels where the predicted class

TABLE 3: Datasets used in the experiment, sorted by IR value in ascending order.

Dataset	#Instances	#Features	#Positive	#Negative	IR
ecoli	336	7	35	301	8.60
optical_digits	5620	64	554	5066	9.14
satimage	6435	36	626	5809	9.28
pen_digits	10992	16	1055	9937	9.42
abalone	4177	10	391	3786	9.68
sick_euthyroid	3163	42	293	2870	9.80
car_eval_34	1728	21	134	1594	11.90
isolet	7797	617	600	7197	12.00
us_crime	1994	100	150	1844	12.29
yeast_ml8	2417	103	178	2239	12.58
scene	2407	294	177	2230	12.60
libras_move	360	90	24	336	14.00
thyroid_sick	3772	52	231	3541	15.33
coil_2000	9822	85	586	9236	15.76
arrythmia	452	278	25	427	17.08
solar_flare_m0	1389	32	68	1321	19.43
oil	937	49	41	896	21.85
car_eval_4	1728	21	65	1663	25.58
wine_quality	4898	300	183	4715	25.77
letter_img	20000	16	734	19266	26.25
yeast_me2	1484	8	51	1433	28.10
ozone_level	2536	72	73	2463	33.74
mammography	11183	6	260	10923	42.01
abalone_19	4177	10	32	4145	129.53

TABLE 4: Parameters used in the experimentation design.

Parameter Type	Name	Parameters
SMOTE Variants	SMOTE-TL [82] SMOTE-ENN [82] BD-SMOTE [80] SL-SMOTE [25]	$p = 1.0,\ k = 5$
	DBSMOTE [110]	$p = 1.0,\ \varepsilon = 0.8,$ $min\_samples = 3$
	SMOTE-RSB* [83]	$p = 2.0,\ k = 5$
	MWMOTE [85]	$p = 1.0,\ k_1 = 5,$ $k_2 = 5,\ k_3 = 5$
	KM-SMOTE [60]	$p = 1.0,\ k = 5,$ $n\_clusters = 10,\ irt = 2.0$
Classifiers	GNB [117]	$var\_smoothing = 10^{-9}$
	k-NN [118]	$k = \{1,3,5\}$
	DT [119] RF [120]	$criterion = gini,$ $min\_samples\_split = 2$
	SVM [2]	$C = \{10^{-3}, 10^{-2}, \ldots, 10^{3}, 10^{4}\},$ $\gamma = \frac{1}{\#\text{Features} \times Var(X)}$
Evaluation	Stratified Random Permutations Cross-Validation	5 Splits

labels are compared to the expected class labels.

4.2 Evaluation Metrics

As our discussion revolves around imbalanced dataset, hence, it is highly appropriate to use a performance measure that can emphasize the importance of positive class during experimentation stage. Hence, we used G-Mean value and F1-Score as our performance measures. G-Mean is a metric that measures the balance between classification performance on both the majority and minority classes. G-Mean can be considered as replacement metric for accuracy in the context of imbalanced dataset. F1-Score however is the harmonic mean of other measures called precision and recall. These performance measures are calculated using the numbers of correctly classified positive or negative class examples as well as the number of negative class examples misclassified as positive and vice versa.

4.3 Complexity Measure

In this experiment, two complexity measures are selected, which are maximum Fischer's discriminant ratio (Fischer's ratio) and the fraction of points on the class boundary (N1) which both are calculated before and after implementation of SMOTE variants. The complexity measures for both Fischer's ratio and N1 are defined in Equation 3 and Equation 4 respectively. This is done to observe to what extent SMOTE variants could affect the complexity of data besides understanding the performance of these variants. The Geometric Small Disjuncts Index (GSDI) is excluded from this experiment, as it has extensive model selection requirements for performing small disjuncts' detection task.

4.4 Results and Analysis

In this section, the analysis of obtained results is divided into two parts. The first part is for performance analysis based on SMOTE and classifiers. In this part, G-Mean and F1-Score are used to evaluate the capability of each classifier. Next, in the second part, the complexity analyses of each SMOTE variant are carried based on Fischer's ratio complexity and N1 complexity. For this part, due to the space limitation, each data complexity metric will be evaluated based on F1-Score only.

4.4.1 SMOTE-Classifier Performance Analysis

This subsection will describe the performance of five classifiers tested over eight popular SMOTE variants, as elaborated in Table 4. The performance is analyzed based on G-Mean and F1-Score taken from each experiment.

To show how balance the classification performance is, for each classifier, the difference between the G-Mean of each SMOTE variant for each dataset with the G-Mean without SMOTE (baseline) is calculated, and then divided by the G-Mean of the baseline, as in Equation 8.

$$\Delta\text{G-Mean} = \frac{\text{G-Mean in SMOTE Variant} - \text{G-Mean in Baseline}}{\text{G-Mean in Baseline}} \tag{8}$$

The difference in G-Mean gained by a SMOTE variant for each dataset was then averaged over all the 24 datasets.

TABLE 5: Percentage of average difference in G-Mean between SMOTE variants and without SMOTE (baseline).

SMOTE Variant	GNB	DT	SVM	1-NN	3-NN	5-NN	RF
SMOTE-TL [82]	2.00	2.44	17.11	8.92	14.75	**19.34**	8.61
SMOTE-ENN [82]	**2.96**	**5.32**	**17.34**	**10.66**	**15.29**	19.03	**11.87**
BD-SMOTE [80]	1.36	1.29	16.25	5.92	13.23	17.74	7.18
SL-SMOTE [25]	-0.97	1.53	3.97	1.59	4.26	8.19	3.18
DBSMOTE [110]	2.69	0.44	7.25	0.52	2.13	4.61	0.61
SMOTE-RSB* [83]	-0.72	0.80	6.90	3.64	5.30	8.19	4.46
MWMOTE [85]	1.10	2.49	15.48	7.09	13.06	17.08	7.41
KM-SMOTE [60]	-0.75	-0.03	0.63	0.33	0.72	0.68	0.03

The highest value for each classifier was highlighted in bold, as shown in Table 5. The following observations can be concluded from the table:

1) SMOTE-ENN recorded the highest average difference in G-Mean for Gaussian Naïve Bayes (2.96), Decision Tree (5.32), Support Vector Machine (17.34), k-Nearest Neighbour (1-NN (10.66), 3-NN (15.29)) and Random Forest (11.87). SMOTE-TL produced the highest average difference in G-Mean for k-Nearest Neighbour (5-NN (19.34)).

2) Among the SMOTE variants, SMOTE-ENN performed better than the other variants, as it produced the best average difference in G-Mean for five classifiers.
3) *k*-Nearest Neighbour (5-NN) is the best classifier for all SMOTE variants, as it recorded the highest percentage of average difference in G-Mean for seven SMOTE variants (out of eight SMOTE variants experimented) compared to other classifiers. Gaussian Naïve Bayes performed the worst as it recorded the lowest difference for most variants.

TABLE 6: Number of datasets achieving best performance based on G-Mean (out of 24 datasets).

SMOTE Variant	GNB	DT	SVM	1-NN	3-NN	5-NN	RF
SMOTE-TL [82]	16	17	22	24	23	23	22
SMOTE-ENN [82]	17	17	22	23	23	23	23
BD-SMOTE [80]	15	13	20	21	21	23	21
SL-SMOTE [25]	8	17	16	14	15	22	17
DBSMOTE [110]	14	13	15	9	13	13	13
SMOTE-RSB* [83]	8	13	16	18	19	17	19
MWMOTE [85]	14	16	21	20	22	23	22
KM-SMOTE [60]	12	13	14	12	13	11	17

Table 6 shows the number of datasets, out of 24 datasets used in the experiments that achieved the best performance based on G-Mean for each SMOTE variant and classifier. The observations from the table can be concluded as follows:

1) Out of all eight SMOTE variants, SMOTE-TL, SMOTE-ENN, BD-SMOTE, DBSMOTE and MWMOTE were able to have more than half of the total datasets achieving the best performance for all classifiers.
2) For classifiers, Decision Tree, Support Vector Machine, *k*-Nearest Neighbour (3-NN) and Random Forest were able to produce best performance for more than half of the total datasets.

Next, we compute F1-Score as a metric in evaluating the classifier performance in term of minority class performance rate (t_{pos}) relative to the f_{pos} and f_{neg}.

TABLE 7: Percentage of average difference in F1-Score between SMOTE variants and without SMOTE (baseline).

SMOTE Variant	GNB	DT	SVM	1-NN	3-NN	5-NN	RF
SMOTE-TL [82]	-1.90	0.04	11.54	1.23	4.56	6.01	6.51
SMOTE-ENN [82]	-1.93	**1.61**	10.26	1.00	3.90	5.45	**8.22**
BD-SMOTE [80]	-3.34	0.09	**11.96**	**2.21**	**5.71**	**7.15**	5.98
SL-SMOTE [25]	-2.62	-3.12	8.77	-0.10	2.58	3.73	1.17
DBSMOTE [110]	-1.32	-0.31	9.09	-0.23	2.19	2.15	-1.17
SMOTE-RSB* [83]	-0.58	0.75	8.71	1.06	3.18	4.49	2.63
MWMOTE [85]	-1.53	0.88	10.70	0.34	4.64	6.02	6.03
KM-SMOTE [60]	**-0.29**	-1.75	4.22	0.61	1.80	1.14	-0.21

For each classifier, Table 7 shows the percentage of average difference in F1-Score between SMOTE variants and without SMOTE. The values were calculated by getting the difference between the F1-Score of each SMOTE variant for each dataset with the F1-Score of without SMOTE (baseline), divided by the baseline, as shown in Equation 9.

$$\Delta\text{F1-Score} = \frac{\text{F1-Score in SMOTE Variant} - \text{F1-Score in Baseline}}{\text{F1-Score in Baseline}} \quad (9)$$

The total of the differences in F1-Score for each SMOTE variant was then divided by the number of (24) datasets to get the average. The highest value for each classifier was highlighted in bold. Based on Table 7, the observation concluded were as follows:

1) KM-SMOTE recorded the highest average difference in F1-Score for Gaussian Naïve Bayes (-0.29), while SMOTE-ENN is the best for Decision Tree (1.61) and Random Forest (8.22). Borderline SMOTE recorded the best for Support Vector Machine (11.96) and *k*-Nearest Neighbour (1-NN (2.21), 3-NN (5.71) and 5-NN (7.15)).
2) Among the SMOTE variants, BD-SMOTE and SMOTE-ENN show the best performance compared to the other variants. Both variants produced the highest average difference in F1-Score for two classifiers.
3) Support Vector Machine is found to be the best classifier for all SMOTE variants, recording the highest average difference in F1-Score for all eight SMOTE variants compared to other classifiers, while Gaussian Naïve Bayes recorded the lowest difference for most variants.

TABLE 8: Number of datasets achieving best performance based on F1-Score (out of 24 datasets).

SMOTE Variant	GNB	DT	SVM	1-NN	3-NN	5-NN	RF
SMOTE-TL [82]	11	12	20	16	15	14	21
SMOTE-ENN [82]	10	11	15	14	14	13	18
BD-SMOTE [80]	10	13	17	16	16	14	19
SL-SMOTE [25]	6	5	13	9	14	15	16
DBSMOTE [110]	13	14	18	15	20	20	9
SMOTE-RSB* [83]	10	15	20	20	20	18	15
MWMOTE [85]	11	15	17	13	15	14	20
KM-SMOTE [60]	14	8	17	21	22	21	13

The number of datasets, out of 24 datasets used in the experiments, that achieved the best performance based on F1-Score for each SMOTE variant and classifier are shown in Table 8. The following observations can be made:

1) None of the SMOTE variants were able to have more than half of the total datasets achieving the best performance for all classifiers.
2) For classifiers, only Support Vector Machine and *k*-Nearest Neighbour (3-NN, 5-NN) were able to produce best performance for more than half of the total datasets.

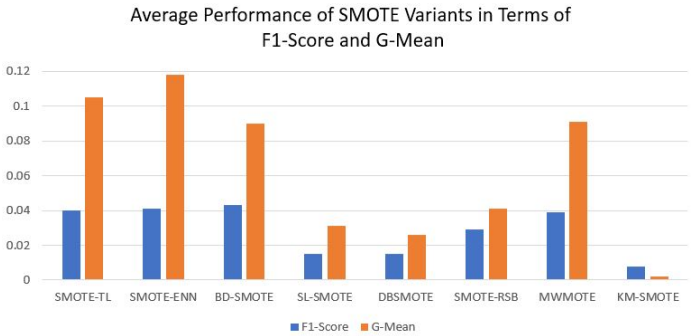


Fig. 11: Average performance of SMOTE variants F1-Score and G-Mean when combined with different base classifiers

Fig. 11 illustrates the average performance of SMOTE variants in terms of F1-score and G-Mean when combined with different base classifiers. According to the figure, SMOTE-ENN (0.118) recorded the best average performance based on G-Mean, while BD-SMOTE (0.043) is the best SMOTE variant as it produced the best average ranking based on F1-Score.

4.4.2 SMOTE Variants Complexity Analysis

In brief, a high value of Fischer's ratio indicates lower complexity in terms of feature overlapping among different classes, while low value of Fischer's ratio indicates there is at least one feature whose values show little overlap among different classes. In terms of N1 complexity, due to N1 being very sensitive to noise examples, higher values of N1 indicate high degree of noise examples existing in the dataset, while low values of N1 indicate fewer noise examples in the dataset.

Hence, this subsection will discuss the performance of SMOTE variants based on the complexity measures, Fischer's ratio and N1 values as depicted in Fig. 12 and Fig. 13. The box plot graphs present the differences of complexity measures after SMOTE variants are applied on all 24 datasets. The differences of Fischer's ratio and N1 values are calculated using Equation 10.

$$\Delta\text{Complexity} = \frac{\text{Complexity After SMOTE Variant} - \text{Complexity in Baseline}}{\text{Complexity in Baseline}} \quad (10)$$

The line in the middle of the box plot acts as the median value and the symbol '$\times$' marks the mean value. Outlier dataset is shown as a dot, '$\cdot$', which lies outside the range of the box plot. The following observations have been concluded based on these two figures:

1) SL-SMOTE has the narrowest range ($-0.30745 \leq \Delta$Fischer's ratio ≤ 0.07629) besides having a higher median (ΔFischer's ratio $= -0.05255$) and the highest mean (ΔFischer's ratio $= -0.07333$) of differences in Fischer's ratio values compared to other SMOTE variants which mean that there are no significant changes in the complexity related to the feature separability. Some of the datasets even have higher Fischer's ratio values after synthetic examples are generated using SL-SMOTE, thus minimizing the degree of overlapping between classes in the features.
2) The lowest median (ΔFischer's ratio $= -0.27275$) and mean (ΔFischer's ratio $= -0.29246$) values can be observed in SMOTE-ENN box plot where this technique encourages overlapping examples to be generated in the dataset. There is a wide range of difference between after and before this technique is implemented ($-0.60002 \leq \Delta$Fischer's ratio ≤ -0.08357) since the maximum and the third quartile values also shown as the lowest compared to other SMOTE variants. Therefore, SMOTE-ENN has the ability to worsen the complexity and reduce data separability.
3) For N1 complexity, the performance of SL-SMOTE and SMOTE-ENN are swapped where the latter reduces the complexity of decision boundary where as the former makes the decision boundary becoming more complex. There is a large decrease in N1 values for SMOTE-ENN as the median values is the lowest (ΔN1 $= -0.59394$) with lower mean value (ΔN1 $= -0.16953$). This proves that the technique eliminates overlapping as much as possible in the class boundary.
4) Despite having the best performance in avoiding a substantial decrease in Fischer's ratio value, SL-SMOTE tends to increase the complexity of boundaries. From the box plot graph, it can be identified that the median (ΔN1 $= 0.61685$) and mean (ΔN1 $= 0.57931$) values for this technique are the highest as well as more than 90% of the differences in N1 values are located above zero.

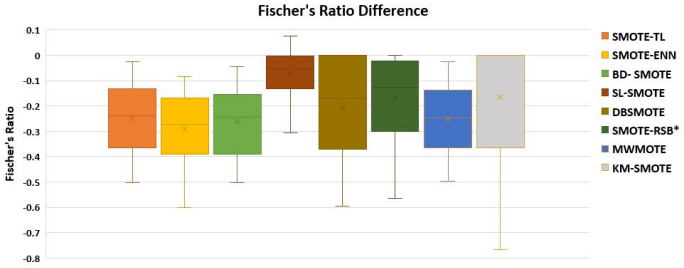


Fig. 12: The difference of Fischer's ratio values between before and after SMOTE variants were implemented from all datasets.

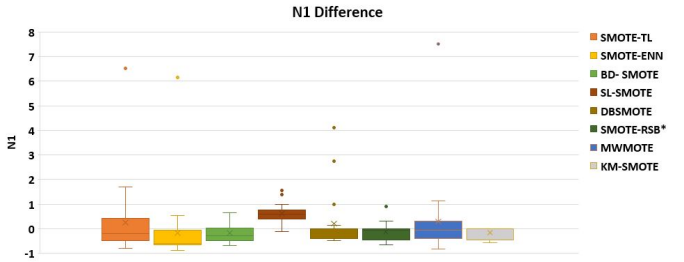


Fig. 13: The difference of N1 values between before and after SMOTE variants were implemented from all datasets.

4.4.3 Fischer's ratio Complexity

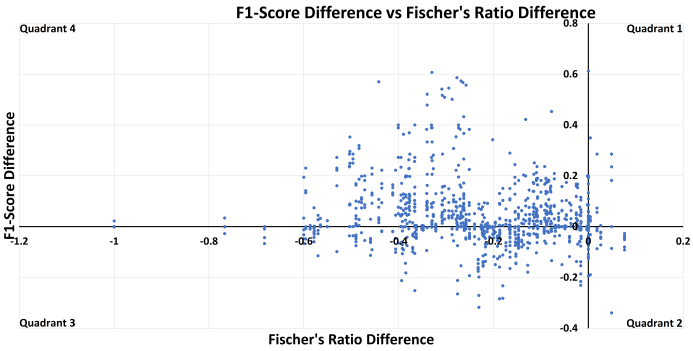


Fig. 14: The changes in Fischer's ratio and F1-Score value from classification task without SMOTE processing (baseline) to classification task with SMOTE processing, respectively.

Fig. 14 describes the distribution of changes in F1-Score over the change in Fischer's ratio, from without SMOTE processing to with SMOTE processing. It can be observed that the increasing complexity in Fischer's ratio is represented by how far the result of Fischer's ratio difference is moving to the left. Thus, several key observations can be extracted from Fig. 14, such as follows:

1) For most experiments, SMOTE increased the feature complexity as most of the experimentation results are located at the left axis of Fig. 14.
2) Hence, in the first 2 quadrants (clockwise order), only few experimentation results illustrated a reduced Fischer's ratio compared to third quadrant and fourth quadrant.
3) The third quadrant represents experimentation results that suffered from worse F1-Score difference due to the increasing complexity Fischer's ratio, while the fourth quadrant represent experimentation results that have better F1-Score value compared to the baseline, despite of having high complexity Fischer's ratio.

This article has been accepted for publication in IEEE Transactions on Knowledge and Data Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2022.3179381

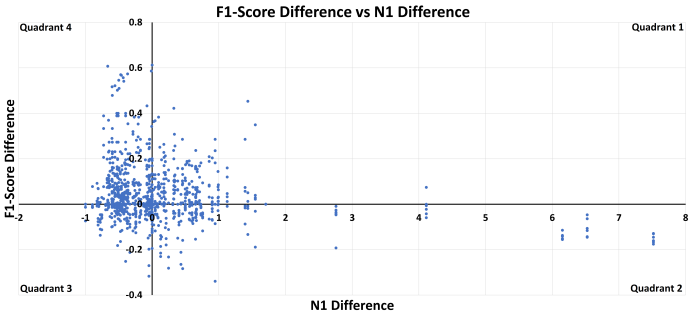


Fig. 15: The changes in N1 and F1-Score value from classification task without SMOTE processing (baseline) to classification task with SMOTE processing, respectively.

4.4.4 N1 Complexity

Fig. 15 describes the distribution of changes in F1-Score over the change in N1 complexity, from without SMOTE processing (baseline) to with SMOTE processing. Several observations can be extracted from Fig. 15, such as follows:

1) The distribution are scattered in 4 different quadrants (clockwise order). On the first quadrant, it is observed that most of the distribution are scattered near the origin. This illustrates that, there is a F1-Score improvement, as long as the increase in N1 complexity difference is not significant. Only one experimentation result with the highest N1 complexity difference ($\Delta N1 = 4.11$), has shown a slight improvement on F1-Score metric.
2) The distribution scattered in the second quadrant represent experimentation results that are decreasing in F1-Score performance due to the increase in N1 complexity. It is clear that as N1 complexity is getting higher, the F1-Score performance is degraded (especially within $6 < \Delta N1 < 8$). Regardless, some experimentation results that have shown significant degradation in F1-Score performance does not have obvious (increasing) N1 complexity difference.
3) The third quadrant represent experimentation results that have performance reduction in F1-Score metric, while having reduction in N1 complexity value. Most of the distribution are scattered near the origin. Only few experimentation results are plotted at ΔF1-Score < -0.2, illustrating that despite of the reduction in N1 complexity value, F1-Score does not show any improvement, rather, major degradation over baseline performance. However, those few experimentation results are exceptional, as it can be observed later in the fourth quadrant, where many of the experimentation results are showing major improvement in F1-Score, when degradation in N1 complexity values are observed.
4) Then, the fourth quadrant represents experimentation results that have performance increment in F1-Score metric, while also having reduction in N1 complexity value. Again, much of the distribution is scattered near the origin. However, unlike in the third quadrant, experimentation results that undergo increment in F1-Score performance are many compared to the experiments that suffered from F1-Score reduction, such as illustrated in the third quadrant. For example, 18 experimentation results with $\Delta N1 < 0$ have F1-Score performance difference within $0.4 \leq \Delta$F1-Score < 0.8.

TABLE 9: Correlation analysis for changes in Fischer's Ratio and N1 complexity metrics towards changes in F1-Score.

Complexity Metric	Correlation to F1-Score
Fischer's Ratio	-0.15529
N1	-0.20685

Table 9 concludes that N1 complexity metric and F1-Score have stronger negative correlation compared to correlation between Fischer's ratio and F1-Score. Again, to reiterate, as N1 complexity is decreased, the performance of F1-Score will increase.

5 Discussion & Limitations

5.1 Discussion

In the recent years, problems induced from imbalanced datasets have motivated the proposal of SMOTE variants, each with different aims and approaches. However, none of the existing review papers attempts to discover the effects of these SMOTE variants on the data complexity associated with imbalanced datasets as well as on the evaluation metrics, such as G-Mean and F1-Score. This study investigates if there is any relation between data complexity and the changes of classification performance in imbalanced dataset after implementation of SMOTE variants. Eventually, we can identify which variants might have good performance in regards to decreasing complexity besides improving evaluation metrics and vice versa.

First, in Section 4.4.1, we have found that there are two SMOTE variants which perform well in two different metrics: SMOTE-ENN for G-Mean metric (Table 5) and BD-SMOTE for F1-Score metric (Table 7). Referring back to Table 2, SMOTE-ENN is designed to solve noise issue, while BD-SMOTE is designed to solve both noise and class overlap issues. Hence, it can be concluded that most of our datasets that we experimented on in this paper suffer from noises and class overlap issues.

Next, further experiments conducted in this paper show in Section 4.4.2 that there is a notable relation between N1 complexity and F1-score which can be proved by the results in Table 7 and Fig. 13. Any SMOTE variants that are able to decrease the decision boundary complexity (N1) have higher probability of giving better F1-score and vice versa. On the other hand, the relation between Fischer's ratio and F1-score is the opposite of the previous relation as observed in Table 7 and Fig. 12. Despite most of the SMOTE variants being prone to worsen the features overlapping complexity (Fischer's ratio) in the datasets, either insignificant increase or noticeable decrease in the degree of complexity does not guarantee a good classification performance.

BD-SMOTE and SMOTE-ENN which generally have negative N1 average difference for most of the datasets (lower complexity) are found to be among SMOTE variants with the highest F1-score values. These variants generate synthetic examples that are able to reduce the decision boundary complexity in a dataset thus suggesting the class overlap problem could be successfully elevated. Learning and classification task are affected in a way that the classifiers' performances are excellently improved.

The performance of SL-SMOTE, DBSMOTE and KM-SMOTE are not comparable to those BD-SMOTE and SMOTE-ENN. Moreover, these techniques have negative percentage of average difference in F1-score values more often compared to others. Fact is that these three techniques either rarely decrease or usually maintain the degree of complexity.

The relation is further observed in Fig. 15 such that reduction in complexity is most likely to raise the classification performance. This graph shows that several SMOTE variants might have the same F1-score difference value with various N1 difference values. According to Occam's Razor Principle [121], when two models have the same error or performance, model with lesser complexity should be preferred. This principle is normally utilized in statistical model selection. In this situation, any SMOTE variants that are able to lower the complexity of decision boundary should be chosen if they have the same classification performance which is F1-Score.

To conclude our experiments, we observed the runtime (in seconds) for each SMOTE variant on dataset *wine_quality*, as shown in Fig. 16 where BD-SMOTE has the shortest runtime (0.0098 s) while SMOTE_RSB* has the longest runtime (3.1125 s), 316 times longer than BD-SMOTE.

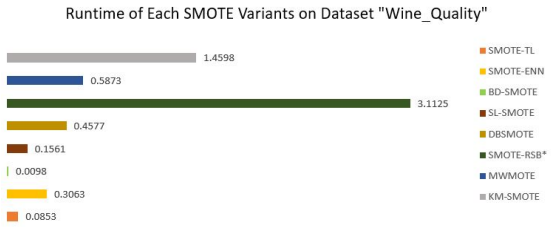


Fig. 16: Runtime (in seconds) of each SMOTE variant operated on dataset *wine_quality*.

5.2 Limitations

Due to insufficient computing resources, the exploration of changes in data complexities induced by SMOTE variants is limited and obstructed. More information could have been gathered if more data complexity measures had been calculated, which would elaborate explicitly the relation between data complexity and classification performance. Apart from that, this paper does not take into consideration the decision on the best parameters for every SMOTE variant as well as the classifier (instead it uses suggested default implementation values), since it is beyond the scope of this study. Most of the time, we have been using default parameters provided by the Python library[2] in the experiments. Apparently, this shortcoming can be observed in the performance of KM-SMOTE in which this technique does not yield good results as it is supposed to. The same issue goes to SL-SMOTE in real-world applications. Despite of performing well in Fischer's ratio analysis, this technique does not show significant improvement during evaluation using performance measures. We believe that more precise classification performances and complexity measures values can be expected from SMOTE variants application if this issue is being reviewed.

6 Conclusion and Future Perspectives

In the real world, classification task is not a straightforward task. Especially, when dealing with imbalanced dataset, one needs to ensure that performance metric is representing the classification performance of each targeted class. Various strategies have been proposed to solve this issue, in which SMOTE is one of them.

In this paper, we give a taxonomy of SMOTE variants to best highlight the similarities and differences among the SMOTE variants proposed in the literature. The dimensions we compare are based on SMOTE algorithm and which data complexity the SMOTE variant is solving.

In addition, we have also conducted 40 experiments on 24 benchmark binary datasets to explain the behaviour of 8 SMOTE variants during the classification task. We found that SMOTE variants indeed give a performance gain ranging up to approximately 12% on classification performance, compared to non-SMOTE implementations. We also noted that SMOTE variants and Support Vector Machines are the best SMOTE-classifier combination. In contrast, we found that Gaussian Naive Bayes does not complement well with any SMOTE variant.

We also discovered that SMOTE variants influence the data complexity of imbalanced dataset, in relation to the classification performance. We discussed these findings and noted that if SMOTE variants increase the Fischer's ratio and decrease the N1, the probability of having good classification model is increasing.

We conclude that SMOTE variants are useful in practice and that there is ample evidence that better SMOTE based algorithms can be devised for improved classification performance on imbalanced dataset, and decreased data complexity. For further study, we would like to investigate other data complexity measures that can be used to benchmark and correlate the SMOTE behaviour in relation to classification performance. In addition to that, we also would like to identify the proper complexity measure for each type of problem that is discussed in this paper, which is the noise example problem, the class overlap/separability problem and the small disjuncts problem. Future SMOTE variant experimentation should attempt on decreasing the complexity of a dataset due to higher probability of getting good classification performance. To do that, we suggest three guidelines:

- Generation of synthetic examples in the region with high concentration of minority examples should be avoided since it does not guarantee that the complexity could be reduced. Thus, the actual performance of such a variant might not be any better.
- The oversampling strategy should incorporate the examples located near the overlapping region as well as working on the the separability between different classes. Not only the required decision boundary would be then less complex but also the synthetic examples generated could provide effective information that might enhance the classification task.
- Filtering-based algorithm should have been widely implemented with SMOTE since SMOTE variants that use this type of combination could positively affect the classification performance. This is due to the degree of complexity in a dataset that would subside.

Acknowledgement

This research is funded by the Ministry of Higher Education Malaysia (MOHE) under the Research Excellence Consortium Grant Scheme (KKP): JPT(BPKI)1000/016/018/25(54).

References

[1] Y. Freund and R. E. Schapire, "Large margin classification using the perceptron algorithm," *Machine learning*, vol. 37, no. 3, pp. 277–296, 1999.

[2] J. A. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural processing letters*, vol. 9, no. 3, pp. 293–300, 1999.

[3] W. S. Noble, "What is a support vector machine?" *Nature biotechnology*, vol. 24, no. 12, pp. 1565–1567, 2006.

2. https://smote-variants.readthedocs.io/en/latest/oversamplers.html

[4] K. Jiang, Z. Wang, P. Yi, G. Wang, T. Lu, and J. Jiang, "Edge-enhanced gan for remote sensing image superresolution," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 8, pp. 5799–5812, 2019.

[5] X. Xuan, B. Peng, W. Wang, and J. Dong, "On the generalization of gan image forensics," in *Chinese Conference on Biometric Recognition*. Springer, 2019, pp. 134–141.

[6] Y. Goldberg and O. Levy, "word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method," *arXiv preprint arXiv:1402.3722*, 2014.

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[8] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz *et al.*, "Huggingface's transformers: State-of-the-art natural language processing," *ArXiv*, pp. arXiv–1910, 2019.

[9] J. Quiñonero-Candela, M. Sugiyama, N. D. Lawrence, and A. Schwaighofer, *Dataset shift in machine learning*. Mit Press, 2009.

[10] B. Krawczyk, M. Galar, Ł. Jeleń, and F. Herrera, "Evolutionary undersampling boosting for imbalanced classification of breast cancer malignancy," *Applied Soft Computing*, vol. 38, pp. 714–726, 2016.

[11] M. Saini and S. Susan, "Deep transfer with minority data augmentation for imbalanced breast cancer dataset," *Applied Soft Computing*, p. 106759, 2020.

[12] J. O. Awoyemi, A. O. Adetunmbi, and S. A. Oluwadare, "Credit card fraud detection using machine learning techniques: A comparative analysis," in *2017 International Conference on Computing Networking and Informatics*, 2017, pp. 1–9.

[13] K. A. Taher, B. Mohammed Yasin Jisan, and M. M. Rahman, "Network intrusion detection using supervised machine learning technique with feature selection," in *2019 International Conference on Robotics,Electrical and Signal Processing Techniques*, 2019, pp. 643–646.

[14] M. Luo, K. Wang, Z. Cai, A. Liu, Y. Li, and C. F. Cheang, "Using imbalanced triangle synthetic data for machine learning anomaly detection," *Computers, Materials and Continua*, vol. 58, no. 1, pp. 15–26, 2019.

[15] B. Krawczyk, "Learning from imbalanced data: open challenges and future directions," *Progress in Artificial Intelligence*, vol. 5, no. 4, pp. 221–232, 2016.

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

[17] S. P. Parambath, N. Usunier, and Y. Grandvalet, "Optimizing f-measures by cost-sensitive classification," in *Advances in Neural Information Processing Systems*, 2014, pp. 2123–2131.

[18] K. Dembczynski, A. Jachnik, W. Kotlowski, W. Waegeman, and E. Hüllermeier, "Optimizing the f-measure in multi-label classification: Plug-in rule approach versus structured loss minimization," in *International Conference on Machine Learning*, 2013, pp. 1130–1138.

[19] Y. Tang, Y.-Q. Zhang, N. V. Chawla, and S. Krasser, "Svms modeling for highly imbalanced classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 1, pp. 281–288, 2008.

[20] T. Imam, K. M. Ting, and J. Kamruzzaman, "z-svm: An svm for improved classification of imbalanced data," in *Australasian Joint Conference on Artificial Intelligence*. Springer, 2006, pp. 264–273.

[21] M. S. M. Pozi, N. A. Azhar, A. R. A. Raziff, and L. H. Ajrina, "Svgpm: evolving svm decision function by using genetic programming to solve imbalanced classification problem," *Progress in Artificial Intelligence*, pp. 1–13, 2021.

[22] L. Gonzalez-Abril, H. Nuñez, C. Angulo, and F. Velasco, "Gsvm: An svm for handling imbalanced accuracy between classes inbi-classification problems," *Applied Soft Computing*, vol. 17, pp. 23–31, 2014.

[23] B. Richhariya and M. Tanveer, "A reduced universum twin support vector machine for class imbalance learning," *Pattern Recognition*, vol. 102, p. 107150, 2020.

[24] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.

[25] C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap, "Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2009, pp. 475–482.

[26] S. Wang, L. L. Minku, and X. Yao, "Resampling-based ensemble methods for online class imbalance learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 5, pp. 1356–1368, 2014.

[27] R. A. Sowah, M. A. Agebure, G. A. Mills, K. M. Koumadi, and S. Y. Fiawoo, "New cluster undersampling technique for class imbalance learning," *International Journal of Machine Learning and Computing*, vol. 6, no. 3, p. 205, 2016.

[28] A. J. Mohammed, M. M. Hassan, and D. H. Kadir, "Improving classification performance for a novel imbalanced medical dataset using smote method," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 9, no. 3, 2020.

[29] Y. Ge, D. Yue, and L. Chen, "Prediction of wind turbine blades icing based on mbk-smote and random forest in imbalanced data set," in *2017 IEEE Conference on Energy Internet and Energy System Integration*. IEEE, 2017, pp. 1–6.

[30] X. Zheng, "Smote variants for imbalanced binary classification: Heart disease prediction," Ph.D. dissertation, UCLA, 2020.

[31] M. Koziarski, B. Krawczyk, and M. Woźniak, "Radial-based oversampling for noisy imbalanced data classification," *Neurocomputing*, vol. 343, pp. 19–33, 2019.

[32] H. Guan, Y. Zhang, M. Xian, H. Cheng, and X. Tang, "Smote-wenn: Solving class imbalance and small sample problems by oversampling and distance scaling," *Applied Intelligence*, pp. 1–16, 2020.

[33] B. Chen, S. Xia, Z. Chen, B. Wang, and G. Wang, "Rsmote: A self-adaptive robust smote for imbalanced problems with label noise," *Information Sciences*, vol. 553, pp. 397–428, 2021.

[34] G. Kovács, "An empirical comparison and evaluation of minority oversampling techniques on a large number of imbalanced datasets," *Applied Soft Computing*, vol. 83, p. 105662, 2019.

[35] A. Fernández, S. Garcia, F. Herrera, and N. V. Chawla, "Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary," *Journal of Artificial Intelligence Research*, vol. 61, pp. 863–905, 2018.

[36] A. Gosain and S. Sardana, "Handling class imbalance problem using oversampling techniques: A review," in *2017 International Conference on Advances in Computing, Communications and Informatics*. IEEE, 2017, pp. 79–85.

[37] B. Santoso, H. Wijayanto, K. Notodiputro, and B. Sartono, "Synthetic over sampling methods for handling class imbalanced problems: a review," in *IOP Conference Series: Earth and Environmental Science*, vol. 58, no. 1, 2017, p. 012031.

[38] J. Kong, W. Kowalczyk, D. A. Nguyen, T. Bäck, and S. Menzel, "Hyperparameter optimisation for improving classification under class imbalance," in *2019 IEEE Symposium Series on Computational Intelligence*. IEEE, 2019, pp. 3072–3078.

[39] L. P. Garcia, A. C. de Carvalho, and A. C. Lorena, "Effect of label noise in the complexity of classification problems," *Neurocomputing*, vol. 160, pp. 108–119, 2015.

[40] M. Koziarski, M. Woźniak, and B. Krawczyk, "Combined cleaning and resampling algorithm for multi-class imbalanced data with label noise," *Knowledge-Based Systems*, vol. 204, p. 106223, 2020.

[41] B. Li, Y. Wang, A. Singh, and Y. Vorobeychik, "Data poisoning attacks on factorization-based collaborative filtering," *arXiv preprint arXiv:1608.08182*, 2016.

[42] N. Nigam, T. Dutta, and H. P. Gupta, "Impact of noisy labels in learning techniques: a survey," in *Advances in Data and Information Sciences*. Springer, 2020, pp. 403–411.

[43] L. Ju, X. Wang, L. Wang, D. Mahapatra, X. Zhao, M. Harandi, T. Drummond, T. Liu, and Z. Ge, "Improving medical image classification with label noise using dual-uncertainty estimation," *arXiv preprint arXiv:2103.00528*, 2021.

[44] C. Pelletier, S. Valero, J. Inglada, N. Champion, C. Marais Sicre, and G. Dedieu, "Effect of training class label noise on classification performances for land cover mapping with satellite image time series," *Remote Sensing*, vol. 9, no. 2, p. 173, 2017.

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

[46] P. Vuttipittayamongkol, E. Elyan, and A. Petrovski, "On the class overlap problem in imbalanced data classification," *Knowledge-Based Systems*, p. 106631, 2020.

[47] T. Sasada, Z. Liu, T. Baba, K. Hatano, and Y. Kimura, "A resampling method for imbalanced datasets considering noise and overlap," *Procedia Computer Science*, vol. 176, pp. 420–429, 2020.

[48] E. R. Fernandes and A. C. de Carvalho, "Evolutionary inversion of class distribution in overlapping areas for multi-class imbalanced learning," *Information Sciences*, vol. 494, pp. 141–154, 2019.

[49] K. Napierala and J. Stefanowski, "Types of minority class examples and their influence on learning classifiers from imbalanced data," *Journal of Intelligent Information Systems*, vol. 46, no. 3, pp. 563–597, 2016.

[50] B.-W. Yuan, X.-G. Luo, Z.-L. Zhang, Y. Yu, H.-W. Huo, T. Johannes, and X.-D. Zou, "A novel density-based adaptive k nearest neighbor method for dealing with overlapping problem in imbalanced datasets," *Neural Computing and Applications*, pp. 1–25, 2020.

[51] P. Vorraboot, S. Rasmequan, K. Chinnasarn, and C. Lursinsap, "Improving classification rate constrained to imbalanced data between overlapped and non-overlapped regions by hybrid algorithms," *Neurocomputing*, vol. 152, pp. 429–443, 2015.

[52] P. Vuttipittayamongkol and E. Elyan, "Improved overlap-based undersampling for imbalanced dataset classification with application to epilepsy and parkinson's disease," *International Journal of Neural Systems*, vol. 30, no. 08, p. 2050043, 2020.

[53] W. Almutairi and R. Janicki, "On relationships between imbalance and overlapping of datasets." in *Proceedings of 35th International Conference on Computers and Their Applications*. EasyChair, 2020, pp. 141–150.

[54] V. García, J. S. Sánchez, H. O. Domínguez, and L. Cleofas-Sánchez, "Dissimilarity-based learning from imbalanced data with small disjuncts and noise," in *Iberian Conference on Pattern Recognition and Image Analysis*. Springer, 2015, pp. 370–378.

[55] J. L. Leevy, T. M. Khoshgoftaar, R. A. Bauder, and N. Seliya, "A survey on addressing high-class imbalance in big data," *Journal of Big Data*, vol. 5, no. 1, p. 42, 2018.

[56] D. Chen, X.-J. Wang, and B. Wang, "A dynamic decision-making method based on ensemble methods for complex unbalanced data," in *Web Information Systems Engineering 2019*, R. Cheng, N. Mamoulis, Y. Sun, and X. Huang, Eds. Springer International Publishing, 2019, pp. 359–372.

[57] D. Cieslak, N. Chawla, and A. Striegel, "Combating imbalance in network intrusion datasets," in *2006 IEEE International Conference on Granular Computing*, 2006, pp. 732–737.

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

[59] W.-C. Lin, C.-F. Tsai, Y.-H. Hu, and J.-S. Jhang, "Clustering-based undersampling in class-imbalanced data," *Information Sciences*, vol. 409, pp. 17–26, 2017.

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

[61] T. K. Ho and M. Basu, "Complexity measures of supervised classification problems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 3, pp. 289–300, 2002.

[62] V. H. Barella, L. P. Garcia, M. P. de Souto, A. C. Lorena, and A. de Carvalho, "Data complexity measures for imbalanced classification tasks," in *2018 International Joint Conference on Neural Networks*. IEEE, 2018, pp. 1–8.

[63] S. Datta, S. Nag, S. S. Mullick, and S. Das, "Diversifying support vector machines for boosting using kernel perturbation: Applications to class imbalance and small disjuncts," *arXiv preprint arXiv:1712.08493*, 2017.

[64] V. S. Spelmen and R. Porkodi, "A review on handling imbalanced data," in *2018 International Conference on Current Trends towards Converging Technologies*. IEEE, 2018, pp. 1–11.

[65] T. Le, M. T. Vo, B. Vo, M. Y. Lee, and S. W. Baik, "A hybrid approach using oversampling technique and cost-sensitive learning for bankruptcy prediction," *Complexity*, vol. 2019, 2019.

[66] S. Maldonado, J. López, and C. Vairetti, "An alternative smote oversampling strategy for high-dimensional datasets," *Applied Soft Computing*, vol. 76, pp. 380–389, 2019.

[67] A. Guzmán-Ponce, R. M. Valdovinos, J. S. Sánchez, and J. R. Marcial-Romero, "A new under-sampling method to face class overlap and imbalance," *Applied Sciences*, vol. 10, no. 15, p. 5164, 2020.

[68] K. Cheng, C. Zhang, H. Yu, X. Yang, H. Zou, and S. Gao, "Grouped smote with noise filtering mechanism for classifying imbalanced data," *IEEE Access*, vol. 7, pp. 170 668–170 681, 2019.

[69] G. Douzas and F. Bacao, "Self-organizing map oversampling (somo) for imbalanced data set learning," *Expert Systems with Applications*, vol. 82, pp. 40–52, 2017.

[70] S. S. Mullick, S. Datta, and S. Das, "Generative adversarial minority oversampling," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 1695–1704.

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

[72] K. E. Bennin, J. Keung, P. Phannachitta, A. Monden, and S. Mensah, "Mahakil: Diversity based oversampling approach to alleviate the class imbalance issue in software defect prediction," *IEEE Transactions on Software Engineering*, vol. 44, no. 6, pp. 534–550, 2017.

[73] X. Wang, B. Yu, A. Ma, C. Chen, B. Liu, and Q. Ma, "Protein–protein interaction sites prediction by ensemble random forests with synthetic minority oversampling technique," *Bioinformatics*, vol. 35, no. 14, pp. 2395–2402, 2019.

[74] G. Kovács, "Smote-variants: A python implementation of 85 minority oversampling techniques," *Neurocomputing*, vol. 366, pp. 352–354, 2019.

[75] X. Liang, A. Jiang, T. Li, Y. Xue, and G. Wang, "Lr-smote–an improved unbalanced data set oversampling based on k-means and svm," *Knowledge-Based Systems*, p. 105845, 2020.

[76] P. Kaur and A. Gosain, "Comparing the behavior of oversampling and undersampling approach of class imbalance learning by combining class imbalance problem with noise," in *ICT Based Innovations*. Springer, 2018, pp. 23–30.

[77] N. Verbiest, "Fuzzy rough and evolutionary approaches to instance selection," Ph.D. dissertation, Ghent University, 2014.

[78] B. S. Raghuwanshi and S. Shukla, "Smote based class-specific extreme learning machine for imbalanced learning," *Knowledge-Based Systems*, vol. 187, p. 104814, 2020.

[79] P. Soltanzadeh and M. Hashemzadeh, "Rcsmote: Range-controlled synthetic minority over-sampling technique for handling the class imbalance problem," *Information Sciences*, vol. 542, pp. 92–111, 2021.

[80] H. Han, W.-Y. Wang, and B.-H. Mao, "Borderline-smote: a new over-sampling method in imbalanced data sets learning," in *International Conference on Intelligent Computing*. Springer, 2005, pp. 878–887.

[81] I. Kunakorntum, W. Hinthong, and P. Phunchongharn, "A synthetic minority based on probabilistic distribution (symprod) oversampling for imbalanced datasets," *IEEE Access*, vol. 8, pp. 114 692–114 704, 2020.

[82] G. E. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *ACM SIGKDD Explorations Newsletter*, vol. 6, no. 1, pp. 20–29, 2004.

[83] E. Ramentol, Y. Caballero, R. Bello, and F. Herrera, "Smote-rs b*: a hybrid preprocessing approach based on oversampling and undersampling for high imbalanced data-sets using smote and rough sets theory," *Knowledge and information systems*, vol. 33, no. 2, pp. 245–265, 2012.

[84] J. A. Sáez, J. Luengo, J. Stefanowski, and F. Herrera, "Smote–ipf: Addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering," *Information Sciences*, vol. 291, pp. 184–203, 2015.

[85] S. Barua, M. M. Islam, X. Yao, and K. Murase, "Mwmote–majority weighted minority oversampling technique for imbalanced data set learning," *IEEE Transactions on knowledge and data engineering*, vol. 26, no. 2, pp. 405–425, 2012.

[86] J. Wei, H. Huang, L. Yao, Y. Hu, Q. Fan, and D. Huang, "Ia-suwo: An improving adaptive semi-unsupervised weighted oversampling for imbalanced classification problems," *Knowledge-Based Systems*, vol. 203, p. 106116, 2020.

[87] Y. Yan, R. Liu, Z. Ding, X. Du, J. Chen, and Y. Zhang, "A parameter-free cleaning method for smote in imbalanced classification," *IEEE Access*, vol. 7, pp. 23 537–23 548, 2019.

[88] G. Douzas and F. Bacao, "Geometric smote: Effective oversampling for imbalanced learning through a geometric extension of smote," *arXiv preprint arXiv:1709.07377*, 2017.

[89] J. Yun, J. Ha, and J.-S. Lee, "Automatic determination of neighborhood size in smote," in *Proceedings of the 10th International Conference on Ubiquitous Information Management and Communication*, 2016, pp. 1–8.

[90] H. Al Majzoub, I. Elgedawy, Ö. Akaydın, and M. K. Ulukök, "Hcab-smote: A hybrid clustered affinitive borderline smote approach for imbalanced data binary classification," *Arabian Journal for Science and Engineering*, pp. 1–18, 2020.

[91] S. Piri, D. Delen, and T. Liu, "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, 2018.

[92] A. Gosain and S. Sardana, "Farthest smote: a modified smote approach," in *Computational Intelligence in Data Mining*. Springer, 2019, pp. 309–320.

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

[94] S. Susan and A. Kumar, "Ssomaj-smote-ssomin: Three-step intelligent pruning of majority and minority samples for learning from imbalanced datasets," *Applied Soft Computing*, vol. 78, pp. 141–149, 2019.

[95] K. Borowska and J. Stepaniuk, "Imbalanced data classification: A novel re-sampling approach combining versatile improved smote and rough sets," in *IFIP International Conference on Computer Information Systems and Industrial Management*. Springer, 2016, pp. 31–42.

[96] X. Chen, Q. Kang, M. Zhou, and Z. Wei, "A novel under-sampling algorithm based on iterative-partitioning filters for imbalanced classification," in *2016 IEEE International Conference on Automation Science and Engineering*. IEEE, 2016, pp. 490–494.

[97] K. Agrawal, Y. Baweja, D. Dwivedi, R. Saha, P. Prasad, S. Agrawal, S. Kapoor, P. Chaturvedi, N. Mali, V. U. Kala *et al.*, "A comparison of class imbalance techniques for real-world landslide predictions," in *2017 International Conference on Machine Learning and Data Science*. IEEE, 2017, pp. 1–8.

[98] J. Cervantes, F. Garcia-Lamont, L. Rodriguez, A. López, J. R. Castilla, and A. Trueba, "Pso-based method for svm classification on skewed data sets," *Neurocomputing*, vol. 228, pp. 187–197, 2017.

[99] W. A. Rivera, "Noise reduction a priori synthetic over-sampling for class imbalanced data sets," *Information Sciences*, vol. 408, pp. 146–161, 2017.

[100] D. Chen, X.-J. Wang, and B. Wang, "A dynamic decision-making method based on ensemble methods for complex unbalanced data," in *International Conference on Web Information Systems Engineering*. Springer, 2020, pp. 359–372.

[101] J. Zhang, T. Wang, W. W. Ng, W. Pedrycz, S. Zhang, and C. D. Nugent, "Minority oversampling using sensitivity," in *2020 International Joint Conference on Neural Networks*. IEEE, 2020, pp. 1–7.

[102] C. Liu, S. Jin, D. Wang, Z. Luo, J. Yu, B. Zhou, and C. Yang, "Constrained oversampling: An oversampling approach to reduce noise generation in imbalanced datasets with class overlapping," *IEEE Access*, 2020.

[103] G. Zhang, C. Zhang, and H. Zhang, "Improved k-means algorithm based on density canopy," *Knowledge-based systems*, vol. 145, pp. 289–297, 2018.

[104] N. U. Maulidevi, K. Surendro *et al.*, "Smote-lof for noise identification in imbalanced data classification," *Journal of King Saud University-Computer and Information Sciences*, 2021.

[105] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: identifying density-based local outliers," in *Proceedings of the 2000 ACM SIGMOD International Conference on Management of data*, 2000, pp. 93–104.

[106] T. Maciejewski and J. Stefanowski, "Local neighbourhood extension of smote for mining imbalanced data," in *2011 IEEE Symposium on Computational Intelligence and Data Mining*. IEEE, 2011, pp. 104–111.

[107] Y. Sanguanmak and A. Hanskunatai, "Dbsm: The combination of dbscan and smote for imbalanced data classification," in *2016 13th International Joint Conference on Computer Science and Software Engineering*. IEEE, 2016, pp. 1–5.

[108] X. Fan, K. Tang, and T. Weise, "Margin-based over-sampling method for learning from imbalanced datasets," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2011, pp. 309–320.

[109] K. Puntumapon, T. Rakthamamon, and K. Waiyamai, "Cluster-based minority over-sampling for imbalanced datasets," *IEICE TRANSACTIONS on Information and Systems*, vol. 99, no. 12, pp. 3101–3109, 2016.

[110] C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap, "Dbsmote: density-based synthetic minority over-sampling technique," *Applied Intelligence*, vol. 36, no. 3, pp. 664–684, 2012.

[111] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, and X. Xu, "Dbscan revisited, revisited: why and how you should (still) use dbscan," *ACM Transactions on Database Systems*, vol. 42, no. 3, pp. 1–21, 2017.

[112] C. Bunkhumpornpat and K. Sinapiromsaran, "Dbmute: density-based majority under-sampling technique," *Knowledge and Information Systems*, vol. 50, no. 3, pp. 827–850, 2017.

[113] S. C. Wong, A. Gatt, V. Stamatescu, and M. D. McDonnell, "Understanding data augmentation for classification: When to warp?" in *2016 International Conference on Digital Image Computing: Techniques and Applications*, 2016, pp. 1–6.

[114] J. Wei, H. Huang, L. Yao, Y. Hu, Q. Fan, and D. Huang, "New imbalanced fault diagnosis framework based on cluster-mwmote and mfo-optimized ls-svm using limited and complex bearing data," *Engineering Applications of Artificial Intelligence*, vol. 96, p. 103966, 2020.

[115] W. Jianan, H. Haisong, Y. Liguo, H. Yao, F. Qingsong, and H. Dong, "Ni-mwmote: An improving noise-immunity majority weighted minority oversampling technique for imbalanced classification problems," *Expert Systems with Applications*, vol. 158, p. 113504, 2020.

[116] Y. Gao, Y.-F. Li, Y. Lin, C. Aggarwal, and L. Khan, "Setconv: A new approach for learning from imbalanced data," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, 2020, pp. 1284–1294.

[117] T. M. Mitchell, *Machine Learning*. New York: McGraw-Hill, 1997.

[118] J. M. Keller, M. R. Gray, and J. A. Givens, "A fuzzy k-nearest neighbor algorithm," *IEEE transactions on systems, man, and cybernetics*, no. 4, pp. 580–585, 1985.

[119] W.-Y. Loh, "Classification and regression trees," *Wiley interdisciplinary reviews: data mining and knowledge discovery*, vol. 1, no. 1, pp. 14–23, 2011.

[120] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[121] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth, "Occam's razor," *Information processing letters*, vol. 24, no. 6, pp. 377–380, 1987.

NUR ATHIRAH AZHAR received the Bachelor of Science in Applied Mathematics from University of Malaya, Malaysia in 2020. She is now currently pursuing a master's degree in School of Computing, Universiti Utara Malaysia. Her research interests include machine learning and imbalanced data.

MUHAMMAD SYAFIQ MOHD POZI is an academician at School of Computing, Universiti Utara Malaysia. He received his Bachelor of Computer Science with Honours from Infrastructure University of Kuala Lumpur, Malaysia in 2012 and Doctor of Philosophy in Intelligent Computing from Universiti Putra Malaysia in 2016. He has been appointed as a postdoctoral researcher in Kyoto Sangyo University, Japan and at Universiti Tenaga Nasional, Malaysia. His main research interest is machine learning.

ANIZA MOHAMED DIN is an academician and researcher at School of Computing, Universiti Utara Malaysia. She received a Bachelor in Information Technology (with Honours) degree from Universiti Utara Malaysia in 1998. She then received her Master of Science in Computer Science from Universiti Sains Malaysia in 1999. Her research interests include resource allocation problem, operational research, optimization, metaheuristics, genetic algorithm, artificial immune systems and uncertainties.

ADAM JATOWT is a Professor at the Computer Science department of the University of Innsbruck. He is also affiliated with the Digital Science Center at the University of Innsbruck. He received his Ph.D. from the University of Tokyo, Japan in 2005 and has worked as an Assistant and Associate Professor at Kyoto University. His research interests include broad topics in natural language processing, information retrieval, digital humanities and digital libraries. Adam has published over 150 research papers in international conferences and journals including AAAI, IJCAI, WSDM, SIGIR, ACL, TKDE, TOIS, KDD, WWW, and CIKM. He is on the editorial board of IP&M, JASIST, IJDL, JIIS and IEEE JSC journals.