Loyalty-SMOTE: Data Synthesis Algorithm for Effective Imbalanced Data Classification

Abstract

Imbalanced datasets are always problematic in training machine learning models, such that classifiers often struggle to achieve satisfactory performance. Numerous approaches have been developed to tackle imbalanced data problems. Among them, some data-level methods perform linear interpolations between neighboring minority class samples to genrate new data points, while others focus on oversampling boundary samples which are specific to certain classes. However, many methods fail to consider scenarios involving noise susceptibility. In this paper, we propose a novel data-level method called the *Loyalty-SMOTE* algorithm. We introduce the concept of *Loyalty* to identify noise and boundaries within datasets. After identifying potential noisy datapoints, SMOTE (Synthetic Minority Oversampling Technique) algorithm is applied to oversample the minority class boundary data. Subsequently, a denoising process based on Loyality is conducted to obtain a balanced dataset. To extend our design, the concept of *Attraction* is introduced to generalize the denoising technique for multiclass problems. In our study, the SVM (Support Vector Machine) classifier is used as our base learner, extensive experiments are performed to evaluate and compare different algorithms. Our results demonstrate that Loyalty-SMOTE achieved superior performance across multiple metrics on both binary and multiclass UCI datasets. For 25 binary datasets, it achieved the highest scores in 21 datasets (84%) for F1-score, 24 datasets (96%) for AUROC, 21 datasets (84%) for recall, and 22 datasets (88%) for G-mean. For 5 multiclass datasets, our design achieved scores of 0.8317, 0.6153, 0.8537, and 0.6717, respectively.

Keywords: Classification, Imbalanced data, Loyalty-SMOTE, SMOTE

1. Introduction

Subsiding the adverse impact of imbalanced data to machine learning classification (Maheshwari, 2023) is always a critical research topic. Imbalanced datasets are prevalent across various areas, including medical disease detection (Li et al., 2022), user credit assessments (Abd El-Naby et al., 2023), financial risk prevention (Krasic & Celar, 2022), and cybersecurity (Talukder et al., 2024). These issues arise unavoidably due to inherent data collection biases (Mehrabi et al., 2021), the natural rarity of certain classes (Hasanin et al., 2020), and the variations among instances.

In practice, we often encounter numerous imbalanced datasets such that the sample size of one class significantly exceeds those of other classes. Unfortunately, traditional machine learning classifiers typically require a balanced number of samples for each class during model training processes (Shwartz-Ziv et al., 2023). Hence, when machine learning classifiers learn from imbalanced datasets, the classifications of minority classes often fail in performance. These classifiers are usually well trained with the larger amount of majority class data (Silva Filho et al., 2023), thus making it fail to recognize the minority classes, consequently lowering the overall accuracy and introducing systematic errors (Naim et al., 2022). In some specific scenarios, such errors can be intolerable. For instance, in medical disease detection, misdiagnosing a healthy person as a patient might be a marginally acceptable error, but misdiagnosing a patient as healthy could lead to catastrophic consequences. Therefore, researching reliable imbalanced data processing algorithms is crucial for improving the performance and generalization of machine learning classifiers.

As of today, methods for handling imbalanced data are commonly categorized into three main strategies: data-level, cost-sensitive, and ensemble learning. Furthermore, data-level strategies focus on processing the data through two sub-approaches: oversampling and undersampling. The objective of oversampling is to increase the number of samples in the minority class, while undersampling reduces the number of samples in the majority class in order to attain a balance in a dataset. A significant characteristic of data-level methods is that their designs are independent of the classifiers, making them a hot research topic. Various imbalanced data processing methods, such as SMOTE (Synthetic Minority Oversampling Technique) (Chawla et al., 2002), have been proposed in this context. SMOTE is the classical design that addresses the imbalanced data issue, and many SMOTE variants have been developed subsequently, for example, the Borderline-SMOTE (Sun et al., 2022).

As aforementioned, data-level methods are independent of the classifier designs. But their effectiveness can be constrained in certain situations; therefore, algorithms are developed to generate extra sample points to balance the classification performance. However, potential erratic sample points may be created, and these points may lead to inaccurate learning in classifiers (Wu et al., 2024). The generated points are usually based on the linear interpolation of two in-class samples basically. However, when linear interpolations are executed between normal samples and noise or outliners, the new samples may fail to accurately represent the minority class. As illustrated in Figure 1, the reasons for noise generation during oversampling in

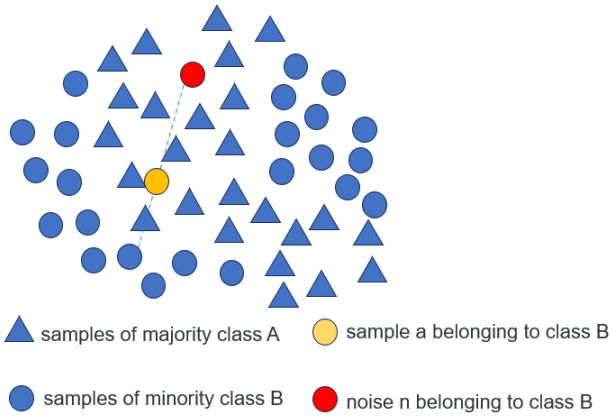


Figure 1: Oversampling of minority class in datasets.

imbalanced datasets, the circle symbols represent the minority class *A*, while the triangle symbols represent the majority class *B*. The samples from class *A* and noise *n* can be used for producing a new data point *a*. Unfortunately, it is evident that sample *a*, classified as minority class *A*, sits inside alone in the class *B* data points. This indicates that *a* is an incorrect sample. This phenomenon highlights a significant challenge in the application of data-level methods for processing imbalanced datasets. The generation of adversarial samples can negatively influence the performance of classifier models, necessitating further development and refinement of these techniques to mitigate their susceptibility to noise and inaccuracies.

To address the issue of imbalanced data processing algorithms which are sensitive to noise, extensive research has also been conducted. For example, Rekha et al. (2020) proposed a classification method that used an adaptive boosting (AdaBoost) algorithm for noise filtering during oversampling, which effectively reduced the negative impact of noise on the performance of oversampling techniques. And Vashisht & Rizvi (2023) introduced a Block Balancing Algorithm (BBA) technique to handle imbalanced data, which minimized the effect of noise on the accuracy of defect predictions in imbalanced datasets.

In addition, enhancing the effectiveness of generated samples is also an important research direction. Mainstream methods apply the oversampling process to the boundaries of denoised classes, specifically targeting minority class samples sitting on the class boundaries. One such well-known boundary oversampling method is the Borderline-SMOTE algorithm (Han et al., 2005). Then, further improvements on the Borderline-SMOTE algorithm was made by Boonchuay et al. (2017). The design is a hybrid oversampling technique that combines Borderline-SMOTE with GANs (Generative Adversarial Networks) to generate new synthetic data that follows a Gaussian distribution. Besides, Majzoub & Elgedawy (2020) introduced an improved Borderline-SMOTE algorithm called AB-SMOTE, which calculated class boundary affinity to achieve more accurate identification of sample boundaries. The aforementioned algorithms have made outstanding contributions to enhancing classifier performance. Unfortunately, most boundary-based oversampling methods primarily focus on feature relationships of the samples without considering sufficiently the relationships among samples.

In this paper, we propose a novel imbalanced data processing algorithm called *Loyalty-SMOTE*. The concept of *Loyalty* is introduced to measure the distance relationships between a given sample point and its neighboring data points. The Loyalty-SMOTE algorithm first identifies potential noise in the dataset by calculating the Loyalty of the samples, and then introduces an assimilation process for denoising. During the assimilation process, we fully consider the relationships between samples and their neighboring samples of different classes through the introduction of a concept called *Attraction*. By measuring the Attraction levels of different classes around a sample, the values assist in denoising the dataset and enable the extension of Loyalty-SMOTE to multiclass scenarios. Additionally, Loyalty is used to identify the data boundaries, where the SMOTE (Synthetic Minority Oversampling Technique) algorithm is applied specifically to these boundaries for data balancing. Finally, the potential noise we discovered is subjected to a secondary screening and noise removal by integrating Loyalty with LOF (Local Outlier Factor) in order to enhance the algorithm's stability and robustness. Our contributions are listed as follows:

- A new concept, Loyalty, is introduced to quantify the overall influence of other samples within the neighborhood on the data. Based on the Loyalty value, the relationship between low Loyalty samples and their associated classes is assessed, enabling the identification of noise in the dataset. Furthermore, Loyalty aids in determining the positions of class boundaries within the dataset.
- A new strategy is proposed to extend the algorithm to multiclass data by introducing Attraction. By calculating the Attraction values of all classes for a given sample, the tendency of the sample towards a particular class can be determined, effectively denoising multiclass data and identifying decision boundaries.
- A novel imbalanced data processing method, the Loyalty-SMOTE algorithm, is presented for handling imbalanced datasets. The Loyalty-SMOTE algorithm enhances the diversity and accuracy of synthesized minority class samples, reduces the probability of generating noisy samples, and ultimately improves the classification performance of classifiers.

The rest of the paper is organized as follows. Section 2 provides the related work that deals with similar problems. Section 3 proposed the Loyalty-SMOTE algorithm that provides the overall design structure of the proposed method. Section 4 provides the experimental comparative results of the proposed method. Finally, Sections 5 and 6 give the discussion and conclusion of the paper, respectively.

2. Related Work

There are different synthetic data and data augmentation techniques in dealing with imbalanced datasets. Further, im-

balanced data processing methods can be classified (Montesinos Lopez et al., 2022) into three categories: data-level, cost-sensitive, and ensemble learning methods. The classical design, SMOTE (Chawla et al., 2002), is a synthetic approach which belongs to a data-level design. It is widely popular due to its simplicity in design. It uses linear interpolations between in-class samples to generate synthetic sample points. Today, numerous improved designs are developed based on SMOTE. There are examples such as SMOTE-ENN (Husain et al., 2025), SMOTE-Tomek Links (Swana et al., 2022), Borderline-SMOTE (Sun et al., 2022), and Safe-Level-SMOTE (Dolo & Mnkandla, 2022).

In this section, we provide a brief review of SMOTE-based data-level processing methods. The SMOTE algorithm, introduced by Chawla et al. (2002), is the classical approach in handling imbalanced datasets. It starts by randomly selecting a sample x_i from the minority class, and finds its k nearest neighbors, S_k. Then for each neighbors, $x_{nn} \in S_k$, it generates

$$x_j = x_i + \lambda \cdot (x_{nn} - x_i) \tag{1}$$

where $\lambda \in [0, 1]$ is a random number, and x_j is a new sample. The SMOTE algorithm effectively augments the minority class and improves classifier performance. One drawback is that it does not consider the data sample distributions. To address the issue, the Borderline-SMOTE (Han et al., 2005) addresses this issue through identifying class boundaries, and focuses on oversampling the minority class samples setting at the boundaries, thereby enhancing the ability of the classifier to recognize minority class samples.

Another approach is to increase the inter-class distance or to identify these class boundaries clearly. The SMOTE-Tomek Links algorithm (Swana et al., 2022) follows this design concept to handle imbalanced datasets. It applies the SMOTE algorithm to oversample minority class samples, and generates synthetic samples to increase the number of minority class instances. Next, it identifies the Tomek Links, which are defined as pairs of samples that are nearest neighbors and belong to different classes in the dataset. That is, one sample is from the minority class and the other is from the majority class. Finally, the algorithm removes the Tomek Links to obtain the final balanced dataset.

Some other designs focus on addressing the noise sensitivity issue in SMOTE. One example is the SMOTE-ENN algorithm (Husain et al., 2025). The ENN (Ensemble Neural Network) (Chen et al., 2019) is a cleaning technique for identifying samples that are misclassified. Specifically, for each sample, the algorithm calculates the class of its nearest neighbors and decides whether to retain the sample based on the voting results of its nearest neighbors. If the class of the sample is different from the majority class of its nearest neighbors, the sample is removed. The SMOTE-ENN algorithm combines SMOTE and ENN by oversampling minority class samples using the SMOTE algorithm, followed by applying ENN to clean the noise, ultimately resulting in a denoised dataset.

The Safe-Level-SMOTE algorithm is also a classic improvement of SMOTE algorithm. It generates synthetic samples by considering the *safe level* of samples, thereby enhancing the quality and diversity of minority class samples. The fundamental idea of the algorithm is to select appropriate samples for synthesis based on the density of the proximity relationships. That is, it avoids the generation of noisy samples. The proximity relationships is defined using the k-nearest neighbors. The proximity relationships of each sample consists of the sample itself and the k nearest samples. The definition of the *safe level* is

$$SL(x_i) = \frac{N_{minority}(x_i)}{N_{total}(x_i)} \tag{2}$$

where $SL(x_i)$ represents the safe level of the sample x_i, $N_{minority}(x_i)$ denotes the number of minority class samples within the neighborhood of sample x_i, $N_{total}(x_i)$ is the total number of samples in the neighborhood of x_i. Typically, Safe-Level-SMOTE selects with a safe level above a certain threshold for synthesis, ensuring that the generated synthetic samples have higher quality.

Building on the aforementioned works, we propose a new oversampling method known as Loyalty-SMOTE algorithm. Our design performs denoising by considering the relationships between samples and their neighboring samples prior to synthesizing new samples. During the synthesis of new samples, it determines the positions of the samples by taking into account their relationship with the class boundaries. After synthesizing the new samples, it reanalyzes the relationship between the samples and their neighboring samples to perform a second round of denoising. We shall discuss the designs of Loyalty-SMOTE thoroughly in the next section.

3. Loyalty-SMOTE

In real life, there are phenomena where certain entities are influenced by their surrounding environment. For instance, the beliefs and ideas of a person can be shaped by the people they interact with daily. This idea can be extended such that sample points located close to each other should likely belong to the same class. In machine learning, samples in general tend to be similar to their surrounding samples. Even if their labels are different, they often exhibit similar in their feature attributes.

With these considerations, we propose an improved SMOTE method called Loyalty-SMOTE to mitigate the negative impact of imbalanced data on machine learning classifiers. There are four main steps in our Loyalty-SMOTE design:

1. The first part of the calculation is to find the Loyalty of each sample based on its definition.
2. The second part involves the assimilation process, where the isolated points and noise in each class are identified according to their Loyalty values. The Attraction is defined with an equation for determining the class of each sample. Those isolated points and noise are cleaned by assigning them to the class with the highest Attraction in achieving a denoising effect.
3. The third part is sample generation. Since the expansion of boundary samples between classes has a beneficial effect on classifiers (Boonchuay et al., 2017), the Loyalty-SMOTE algorithm determines the boundary samples of

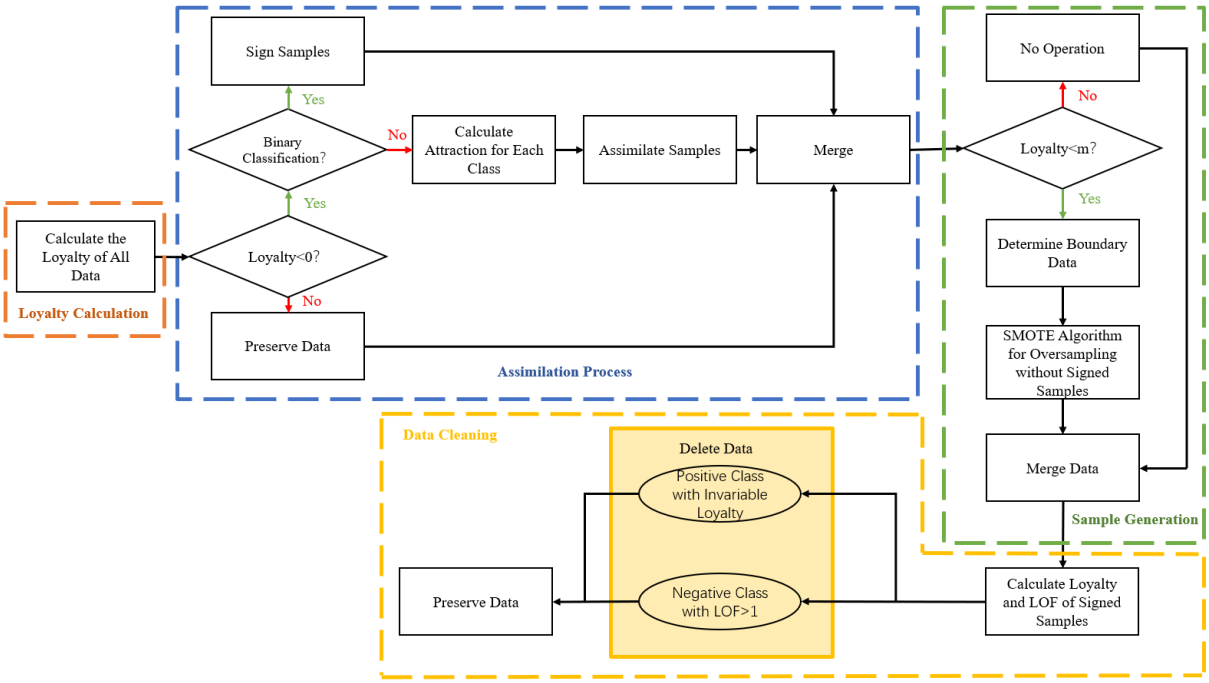


Figure 2: Flow diagram of the Loyalty-SMOTE.

the classes based on their Loyalty values. It then uses the SMOTE algorithm to expand the boundary samples of the minority class, thereby achieving data balance.

4. Finally, the last part is data re-cleaning. Considering that the data processed by SMOTE may introduce new noise (Cheng et al., 2019), there is a need to clean the newly introduced noise in the dataset. Loyalty-SMOTE will recalculate the Loyalty of each signed sample in the new dataset and calculate their LOF to make a second judgment on potential noise in the dataset, and then remove the noise from it.

The flowchart of the Loyalty-SMOTE algorithm is shown in Figure 2, and the pseudocode for Loyalty-SMOTE is presented in Figure 3.

Figure 2 illustrates the four stages of the Loyalty-SMOTE algorithm: Loyalty calculation (orange box), assimilation process (blue area), sample generation (green box), and data re-cleaning (yellow box). The first stage is to compute the Loyalties of all samples in the dataset. The next stage is the assimilation process, where it identifies samples with $Loyalty < 0$ and applies different processing methods depending on the classification problem. For binary classification problems, it will sign samples with $Loyalty < 0$. These samples will be judged as potential noise and not be involved in the subsequent sample generation. For multi-class problems, it calculates the Attraction among all classes surrounding the sample. By assigning the sample to the class with the highest Attraction, it effectively assimilates the sample. The assimilated samples are combined with the unassimilated samples to become a new dataset. The algorithm then proceeds to the sample generation stage, where it identifies the boundary samples of each class based on the criterion of $Loyalty < m$, where m is a predefined constant, and performs SMOTE oversampling on these boundary samples. The Loyalty values of all samples are then updated. Finally, during the data cleaning stage, the Loyalty-SMOTE algorithm adopts distinct secondary noise judgment and cleansing strategies for signed samples from majority and minority classes. For signed minority-class samples, the algorithm recalculates their Loyalty and removes those still with $Loyalty < 0$. For labeled majority-class samples, it calculates the LOF values and deletes samples with $LOF < 0$.

3.1. Loyalty Calculations

Considering that whether a sample is an isolated point or noise is closely related to its surrounding data, the influence of a single neighboring sample on the sample is limited (Najman & Zielinski, 2021). Therefore, it is necessary to assess the distribution of all neighboring samples within the neighborhood to determine the degree of influence on the sample. To address this issue, this paper proposes a new isolated point identification method that uses Loyalty to determine whether a sample is an isolated point or noise. By calculating the Loyalty for each sample, we can assess whether the sample is an isolated point or noise based on the magnitude of Loyalty. The definition of Loyalty is presented in (3).

$$L(x_i) = \begin{cases} \sum_{x_j \in N_k(x_i)} \left(\frac{1}{d(x_i, x_j)} \right) & \text{for } y_i = y_j \\ \sum_{x_j \in N_k(x_i)} \left(\frac{-1}{d(x_i, x_j)} \right) & \text{for } y_i \neq y_j \end{cases}, \tag{3}$$

where y_i and y_j represent the class labels of samples x_i and x_j, respectively. The $L(x_i)$ denotes the Loyalty of x_i, and $N_k(x_i)$ represents the set of samples in the k-nearest neighbors of the sample x_i. The term $d(x_i, x_j)$ denotes the squared Euclidean

Algorithm 1: Loyalty-SMOTE

input : K-nearest neighbors parameter: k,
Assimilation threshold for loyalty: m
DataSet: N

output: Balanced Data

```
begin
    Step 1: Calculate Loyalty
    For each sample x_i in dataset N: begin
        Loyalty_value[x_i] ← calculate_loyalty(x_i)
    end
    Step 2: Assimilation Process
    For each class in dataset: begin
        Identify isolated points and noise based on Loyalty values
        For each sample x_i with Loyalty ¡ 0: begin
            Sign sample x_i as potential noise
            Use Attraction equation to determine class
            Assign x_i to class with highest Attraction
            Clean the dataset by removing noise
        end
    end
    Step 3: Sample Generation
    For each class in dataset: begin
        Identify boundary samples based on Loyalty values
        If class is minority: begin
            Generate new samples using SMOTE(boundary_samples)
        end
    end
    Step 4: Data Cleaning
    For each sample in new dataset: begin
        Recalculate Loyalty for each sample
        If Loyalty ¡ 0: begin
            Remove sample from dataset
        end
        For signed majority-class samples: begin
            Calculate LOF for each sample
            If LOF ¡ 0: begin
                Remove sample from dataset
            end
        end
    end
    Return Balanced Data;
end
```

Figure 3: Loyalty-SMOTE pseudocode.

distance between samples x_i and x_j, as shown in (2). That is

$$d(x_i, x_j) = \sum_{m=1}^{n} \left(x_{im} - x_{jm}\right)^2 \tag{4}$$

where x_{im} and x_{jm} represent the m-th attribute values of the samples x_i and x_j, respectively. If there are a samples of the same class within the k-nearest neighbors of x_i, the total contribution of the Loyalty for all samples of the same class, denoted as $L_{\text{sameclass}}(x_i)$, is given by:

$$L_{\text{sameclass}}(x_i) = \sum_{j=1}^{a} \frac{1}{d(x_i, x_j)}. \tag{5}$$

According to the definition of loyalty, it is evident that $L_{\text{sameclass}}() > 0$. If there are b samples of a different class within the k-nearest neighbors of x_i, the total contribution of the Loyalty for all samples of the different class, denoted as $L_{\text{differentclass}}(x_i)$, is given by: total contribution of the Loyalty for all samples of the same class, denoted as $L_{\text{sameclass}}(x_i)$, is given by:

$$L_{\text{differentclass}}(x_i) = -\sum_{j=1}^{a} \frac{1}{d(x_i, x_j)}. \tag{6}$$

According to the definition of loyalty, it is clear that $L_{\text{differentclass}}() < 0$ holds true. Therefore, we can provide the second definitional form of $L(x_i)$ as follows:

$$L(x_i) = L_{\text{sameclass}}(x_i) + L_{\text{differentclass}}(x_i). \tag{7}$$

The purpose of introducing Loyalty is to measure the degree of isolation of a sample and determine its class belongingness. The neighboring sample points of each sample may inevitably influence the sampe itself, with the strongest influence from the closest neighbor, and the least impact from the farthest one. This influence is reflected in the class similarity between samples. The principle of Loyalty-SMOTE is visualized in Figure 4:

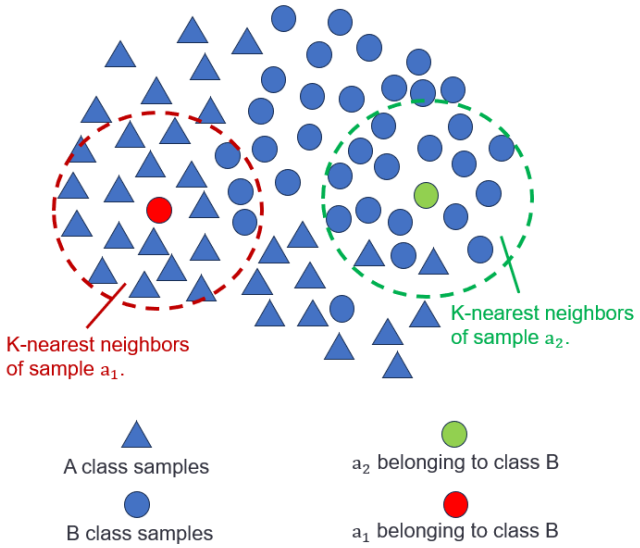


Figure 4: Principle of the Loyalty-SMOTE algorithm.

As shown in Figure 4, the triangles represent class A objects, and the circles represent class B objects. There are a large number of class A objects in the neighborhood of the red circle sample, and these samples are closer to the red circle sample a_1. Therefore, the class A samples in the neighborhood have a significant influence on a_1. Although there are some class B samples in the neighborhood, they are too few in number and farther away from a_1, resulting in a smaller influence of class B samples on a_1. Using (5), the $L_{\text{sameclass}}(a_1)$ is calculated, and the $L_{\text{differentclass}}(a_1)$ is obtained through (6). It is obvious that the class B samples in the neighborhood have a significant influence on the red circle sample, while the class A samples have weaker influence on it; that is,

$$L(a_1) = L_{\text{sameclass}}(a_1) + L_{\text{differentclass}}(a_1) < 0. \tag{8}$$

Since $L(a_1)$, the sample a_1 is more inclined to be similar to class A samples, indicating that a_1 is "unloyal" to its originally assigned class B, and thus, it is considered a "unloyal point." In contrast, the a_2 is more greatly influenced by class B samples in the neighborhood than by class A samples, making it more similar to class B samples. Therefore, the a_2 is "loyal" to its originally assigned class B and is classified as a "loyal point." Ir also illustrates that the unloyal points are relatively isolated

compared to other samples in the same class, while loyal points exhibit the opposite behavior.

The introduction of Loyalty quantifies the influence of neighboring samples on a sample's belonging class, which helps determine whether the sample is an isolated point or noise. It also allows for the assessment of whether the sample is located at a class boundary. It lays the groundwork for subsequent steps in the algorithm. Typically, samples for which $L(x_i) < 0$ are considered non-loyal points. This is because if $L(x_i) < 0$ holds true, then $|L_{\text{sameclass}}(x_1)| < |L_{\text{differentclass}}(x_1)|$ also holds true, indicating that the sample is more influenced by neighboring samples from other classes than by those from its own class.

3.2. Assimilation Process

After calculating the Loyalty of each sample using (1), we can determine whether a sample is an potential isolated point or noise based on its Loyalty value. Therefore, the next step is to process these identified outliers, specifically the samples where $Loyalty < 0$. Current traditional outlier handling methods primarily focus on directly deleting the detected outliers, such as the Interquartile Range (IQR) (Vinutha et al., 2018) method and isolation forests (Hariri et al., 2021). However, directly removing outliers can alter the distribution of the samples, leading to newly generated samples during the subsequent oversampling that may not accurately represent the minority class, thereby reducing the classifier's effectiveness.

To achieve above objectives and overcoming the issues mentioned above, this paper introduces the concept of the assimilation process, which retains those unloyal samples identified as outliers. The assimilation process is divided into binary classification assimilation and multiclass assimilation (Nugroho et al., 2021):

Binary Classification Assimilation For samples with $Loyalty < 0$, the Loyalty-SMOTE algorithm identifies them as potential noise samples and signs them. The signed samples will not participate in the subsequent sample generation stage. This operation prevents potential noise from adversely affecting the subsequent oversampling process, as illustrated in Figure 1. The workflow then proceeds to the next step.

Multiclass Assimilation In the case of multiclass problems, to quantify a sample's belongingness to a certain class, we introduce the concept of Attraction with definition:

$$Att_A(x_i) = \sum_{x_j \in N_{k_A}(x_i)} \frac{1}{d(x_i, x_j)} \tag{9}$$

where $Att_A(x_i)$ represents the Attraction of x_i to class A, N denotes a set, and $N_{k_A}(x_i)$ represents the set of samples of class A within the k-nearest neighbors of the sample x_i. y_i and y_j are the class labels of samples x_i and x_j, respectively. The $d(x_i, x_j)$ is the distance between samples x_i and x_j, which also uses the squared Euclidean distance.

If there are n classes in the k-nearest neighbors of the sample x_i, then n values of $Att_A(x_i)$ can be calculated. If class A has the highest Attraction to x_i, it is considered that class A has the maximum influence on x_i, indicating that x_i belongs to class A to the highest degree.

The design principle of Attraction is similar to that of Loyalty, as it also posits that closer neighboring samples have a greater impact on the sample, and this influence is reflected in the similarity of classes. The key difference is that Attraction represents the degree of influence from another class on the sample, while Loyalty represents the sample's belongingness to its own class. The conceptual framework for Attraction is illustrated in Figure 5.

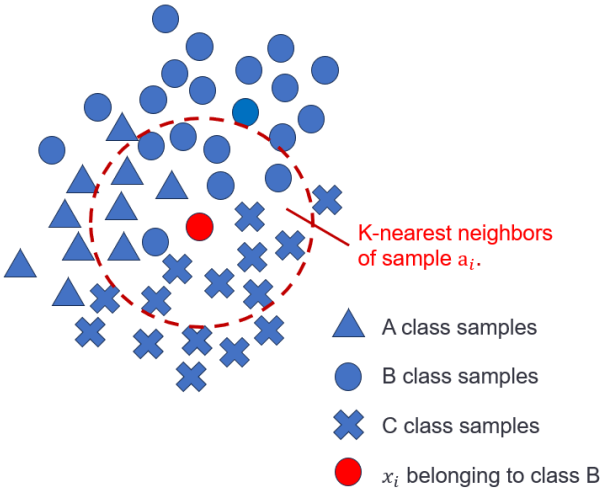


Figure 5: Principle diagram of Loyalty-SMOTE for multiclass processing.

In the above figure, the red circle sample x_i has three types of samples in its neighborhood: triangles represent class A, circles represent class B, and crosses represent class C. It can be observed that class C has more samples that are closer to x_i and in greater quantity, followed by class B and then class A. Therefore, for sample x_i, we have $Attraction_C > Attraction_B > Attraction_A$. This indicates that class C has the greatest influence on x_i, meaning that x_i belongs to class C to the highest degree.

Based on the concept of *Attraction*, we first need to calculate the *Attraction* of all unloyal samples, and then assign these samples to the class with the highest *Attraction*.

3.3. Sample Generation

One key distinction of Loyalty-SMOTE from traditional SMOTE is that Loyalty-SMOTE performs oversampling specifically at the boundaries of samples rather than across the entire dataset. Therefore, accurately identifying the boundaries is crucial. To facilitate the identification of class boundary samples, this paper proposes a method that utilizes the Loyalty of samples to recognize these boundaries.

Considering that Loyalty represents the degree to which a sample belongs to its class, a higher Loyalty indicates a stronger belonging to the class, while a lower Loyalty indicates the opposite. This study finds that samples with the strongest belonging to their class are more likely to be distributed in the central

region of the class. As Loyalty decreases, the distribution of samples gradually tends to shift toward the edges. This phenomenon occurs because there are fewer samples from other classes in the central region and more samples from the same class, resulting in a greater influence of homologous samples on the sample. As a sample approaches the boundary, its influence from same-class samples diminishes.

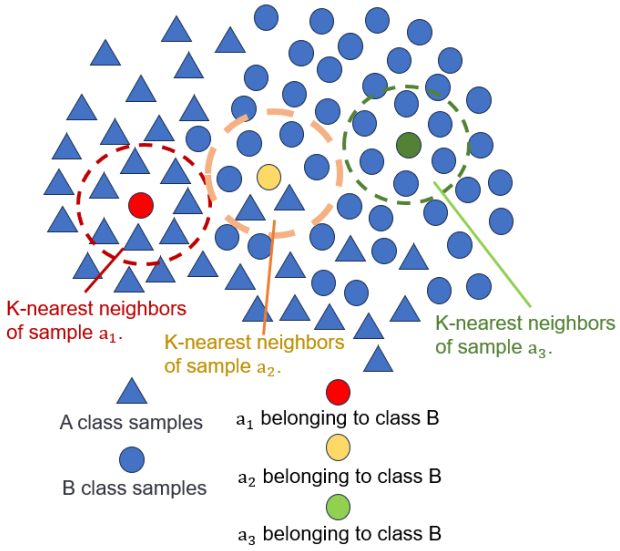


Figure 6: Relationship between Loyalty and sample distributions.

Figure 6 illustrates the relationship between Loyalty and sample distribution, it can be observed that when a sample is located at the edge, there are a few neighboring samples of the same class. This results in a small Loyalty value. Conversely, when a sample is positioned in the middle with numerous neighboring samples of the same class, it leads to a larger Loyalty value. When Loyalty falls below a certain threshold, such as 0, the sample is classified as an isolated point.

Based on the relationship between Loyalty and sample distribution, the Loyalty-SMOTE algorithm considers samples with $Loyalty \in [0, m]$ as boundary samples, where m is the boundary threshold that determines the level of Loyalty below which samples are regarded as boundary samples – typically set to 0.5. $Loyalty > 0$ ensures that the samples are not outliers, while $Loyalty < m$ ensures that the samples are distributed as far from the center as possible.

By identifying the set of boundary samples X_{border} through Loyalty, the SMOTE algorithm is applied to the X_{border} to augment the minority class within X_{border} (where the minority class refers to the minority class of the original dataset), thereby achieving data balance. Additionally, the generated samples are focused on the class boundaries, which is beneficial for the classifier (Sun et al., 2022).

3.4. Data Cleaning

Considering that the introduction of new noise in the data processed by the SMOTE algorithm is inevitable (Ou, 2024), it is necessary to perform an additional denoising step after applying the SMOTE algorithm to enhance the performance of the classifier. However, changes in the dataset will alter the neighborhoods of some samples, so during the data re-cleaning process, it is essential to first recalculate the Loyalty of all samples and then remove the isolated points where $Loyalty < 0$.

To improve classifier performance, cleaning noise in datasets is crucial (Ou, 2024). In sensitive fields like medical testing, improper data cleaning can lead to severe consequences. Loyalty-SMOTE employs a dual-validation strategy for noise detection. During the initial assimilation phase, samples with $Loyalty < 0$ are signed as potential noise. In the subsequent data cleaning phase, a secondary verification is applied:

(1) For signed minority-class samples, Loyalty-SMOTE recalculates their Loyalty values. Samples retaining $Loyalty < 0$ are confirmed as noise and removed.

(2) For signed majority-class samples, the $LOF < 0$ is calculated, and those with $LOF < 0$ are deleted. The LOF can be obtained by using Eqn. (10).

$$\mathrm{LOF}(p) = \frac{\sum_{o \in N_k(p)} \frac{\mathrm{LRD}(o)}{\mathrm{LRD}(p)}}{|N_k(p)|} \tag{10}$$

where p represents a sample point. $N_k(p)$ denotes the k-nearest neighbor set of point p. LRD stands for Local Reachability Density, and its definition is provided in Eqn. (11).

$$\mathrm{LRD}(p) = \frac{|N_k(p)|}{\sum_{o \in N_k(p)} \text{reach-dist}(p, o)}. \tag{11}$$

where reach-dist() denotes the reachability distance, which is defined in Eqn. (12).

$$\text{reach-dist}(p, o) = \max\{k\text{-dist}(o), \text{dist}(p, o)\} \tag{12}$$

where o represents another sample point.

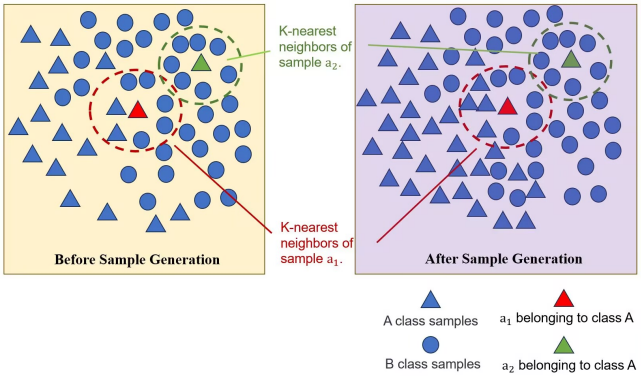


Figure 7: Dataset changes caused by oversampling.

Figure 7 illustrates dataset changes before and after the Sample Generation phase. Post-generation, the expanded negative class leads to a decline in Loyalty for signed unloyal samples and positive samples near negative-class boundaries, as their neighborhood density increases. Conversely, in the positive class, signed unloyal samples and negative samples near positive-class boundaries exhibit rising Loyalty due to reduced isolation. This dynamic implies that signed positive-class samples with negligible Loyalty changes post-generation are likely true anomalies.

In Figure 7, A is the negative dataset, and B is the positive dataset. a_2, being far from A, keeps its loyalty unchanged after Sample Generation due to the stable local data distribution. Similarly, a_1, though marked, is closer to A, so its loyalty should increase more.

However, this logic fails for signed negative-class samples, as their Loyalty inherently fluctuates due to dataset expansion. To address this, Loyalty-SMOTE integrates the LOF, an unsupervised noise detection method. LOF leverages a key property: the positive class distribution remains unchanged post-generation, enabling reliable noise detection even with increased negative samples. So LOF identifies outliers in the majority class without being misled by density shifts.

After completing secondary validation, Loyalty-SMOTE removes confirmed noise, ensuring a cleansed dataset robust for downstream tasks. This dual-mechanism balances precision and adaptability, particularly critical in high-stakes domains.

Based on the proposed algorithm steps, combined with the pseudocode shown in Figure 3, the time complexity of the Loyalty-SMOTE algorithm can be derived. First, for the Loyalty calculation, for each sample x_i, it is necessary to traverse the k-nearest neighbors, $N_k(x_i)$. Assuming the total number of samples is N and the total number of minority class samples is n, the time complexity for calculating Loyalty is $O(N \cdot k)$. In the assimilation process for multiclass problems, the Attraction for each sample is calculated. For each sample x_i, it is required to traverse its k neighboring samples $N_k(x_i)$. Assuming the number of classes is c, the time complexity for the assimilation process is $O(N \cdot k \cdot c)$. In the assimilation process for binary classification, let $n_{unloyal}$ represent the number of samples where $Loyalty < 0$. The time complexity for inverting these samples is $O(n_{unloyal})$. On this basis, to identify the boundary samples and perform the SMOTE algorithm oversampling on these samples with their k-nearest neighbors, the time complexity for sample generation is $O(n \cdot k + n \cdot m)$. Finally, the time complexity for data re-cleaning is $O(n_{unloyal})$.

Combining the above steps, the time complexity of the Loyalty-SMOTE algorithm for binary classification is

$$O(N \cdot k + n \cdot k + n \cdot m).$$

For multiclass classification, the time complexity is

$$O(N \cdot k \cdot c + n \cdot k + n \cdot m).$$

Since the c is a constant, the time complexity of the Loyalty-SMOTE algorithm is

$$O(N \cdot k + n \cdot k + n \cdot m), \tag{13}$$

for both binary and multiclass classification.

4. Experiments

4.1. Datasets

To evaluate the performance of the Loyalty-SMOTE algorithm, this paper uses four groups of commonly used imbalanced datasets. The sources and details of these datasets are shown in Table 1.

Table 1: Sources and information of datasets.

Data Group	Source	Description
Group 1	UCI	25 real-world imbalanced datasets
Group 2	Kaggle	5 popular imbalanced datasets
Group 3	Kaggle	7 large-scale datasets and 2 small-scale datasets
Group 4	UCI	5 multiclass datasets
Group 5	Kaggle	5 high-dimensional datasets and 5 large-scale datasets.

Table 2: Description of Group 1 datasets (IR = imbalanced Rate).

Dataset	Instances	Attributes	IR
adult	32562	14	3.1528
ai4i2020	10000	5	8.9701
banknote	1373	5	1.2508
drug	1886	13	3.3759
AIDS_5000	5000	23	1.3494
Maternal	1015	7	1.4938
national	2279	9	5.2438
occupancy	2666	7	1.7428
online	39645	38	6.6387
Shoppers	12331	11	5.4628
Academic	4425	37	2.0825
AIDS_15000	15000	23	2.2396
german	1000	21	2.3333
statlog	1000	21	2.3333
Shuttle	14500	10	16.7914
Fault	1942	28	11.2911
ICR	634	33	6.2511
Bankruptcy	6820	94	31.0000
breast	699	10	1.9004
wdbc	570	31	1.6761
wpbc	199	34	3.2340
heart	300	13	2.1250
ionosphere	351	33	1.7937
iris	150	5	2.3333
bezediris	150	5	2.0000
raisin	600	8	2.3333

Tables 2, 3, 4, and 5 provide the content description of the datasets in groups 1 to 4, respectively. All datasets are divided into training and testing sets in the 7:3 ratios. The fourth column in the tables shows the imbalance rate (IR), which is defined by

$$\text{IR} = \frac{N_{majority}}{N_{minority}} \tag{14}$$

where $N_{majority}$, and $N_{minoirty}$ represent the sizes of the majority and minority classes in a dataset, respectively.

4.2. Evaluation Metrics

4.2.1. Confusion Matrix

The confusion matrix is a popular method for evaluating classifier performance (Gortler et al., 2022). Table 6 shows a typical binary confusion matrix setting. The rows represent the actual classes (positive or negative), while the columns represent the

Table 3: Description of Group 2 datasets.

Dataset	Instances	Attributes	IR
australia	697	15	1.2476
Obesity	2112	17	1.1728
Gender	4747	18	1.9631
approval	697	7	1.2476
spambase	4602	34	1.5383

Table 4: Description of Group 3 datasets.

Dataset	Instances	Attributes	IR
magic04	19020	11	1.8439
covertype	581012	12	1.2278
Purchase	12331	18	5.4628
Satisfied	103905	21	1.3077
AIDS_50000	50000	23	2.2246
smoke	62631	14	2.5040
vertebal	310	7	2.0794
parkinson	757	755	2.9223

Table 5: Description of Group 4 datasets.

Dataset	Instances	Instances	Attributes	IR
yeast	10	1485	9	77.3333
HR	3	1470	21	6.9333
employee	3	4654	7	1.9248
Drug200	4	413	5	3.8333
MEDV	9	512	13	7.8235

Table 6: Description of Group 5 datasets.

Dataset	Instances	Attributes	IR
Mushroom	61070	12	1.2238
Firewall	65533	11	1.3494
Cardiovascular	55772	10	1.6877
RT-IOT	123118	53	20.9775
Gender_by_name	146870	4	11.5712
Smartphone	2948	536	2.2183
Bankruptcy	6820	94	31.0000
darwin	129	423	1.9318
toxicity	172	1171	2.0714
cnae	1080	312	2.4951

predicted outcomes (positive or negative). Each cell in the matrix contains the number of samples for each combination of actual and predicted classes. The True Positive (TP) value indicates the number of correctly predicted positive cases, False Negative (FN) represents the number of actual positives incorrectly predicted as negative, False Positive (FP) is the count of actual negatives incorrectly predicted as positive, and True Negative (TN) indicates the number of correctly predicted negative cases.

4.2.2. Recall

Recall, also known as sensitivity or True Positive Rate (TPR), measures a model if it can correctly identify all positive samples. It gives the portion of true positive predictions to all actual positive cases, i.e.,

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{15}$$

Recall can help us understand how well the model captures the positive instances that may otherwise be missed (Kynkaanniemi et al., 2019).

4.2.3. F1-Score

The F1-score is a harmonic mean of Recall and Precision. The formulas of Precision and F1-score are

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{16}$$

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{17}$$

It is particularly suitable for imbalanced datasets, as it provides a comprehensive assessment of the performance of a model (Naidu et al., 2023).

4.2.4. Area Under Curve (AUC)

AUC measures the ability of a classifier in ranking positive and negative samples based on predicted scores (Shi et al., 2024). AUROC is the area under the ROC curve (receiver operating characteristic curve), which is a plot with the True Positive Rate (TPR) on the vertical axis against the False Positive Rate (FPR) on the horizontal axis. The ROC curve illustrates the performance of the classifier at different threshold levels. The expressions of TPR an FPR are:

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{18}$$

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}. \tag{19}$$

Generally, a higher AUROC indicates better performance of the classifier, meaning that the probability of positive instances being ranked above negative instances is greater (Yang & Ying., 2022).

4.2.5. G-Mean

G-mean is a commonly used metric for evaluating the performance of classifiers on imbalanced datasets. Similar to the ROC curve, which combines the TPR and FPR of a classifier, G-mean is the geometric mean of TPR and True Negative Rate (TNR). It provides a better balance between the classification results for both positive and negative instances in imbalanced datasets (Espindola & Ebecken, 2005). The expression of G-mean is:

$$\text{G-mean} = \sqrt{\text{TPR} \cdot \text{TNR}}. \tag{20}$$

4.3. Design and Environment for Experiments

To evaluate the performance and running duration of the Loyalty-SMOTE algorithm, we compared it with five other oversampling algorithms: SMOTE, SMOTE-ENN, SMOTE-Tomek Links, Borderline-SMOTE, and Safe-Level-SMOTE. Table 8 provides the specific parameter configurations for all algorithms involved in the experiments. In this study, the SVM (Support Vector Machine) (Hearst, 1998) classifier is used as the base learning model for all six algorithms.

Table 7: Confusion Matrix.

		Predicted Classes	
		Positive	**Negative**
Actual Classes	**Positive**	TP	FN
	Negative	FP	TN

To simplify the table, the six algorithms are abbreviated SMOTE, SMOTE-TL, BL-SMOTE, SMOTE-ENN, SL-SMOTE, and Loyalty-SMOTE.

All experiments conducted in this paper were performed on a computer with Windows 11 operating system, an AMD Ryzen 5 3600 processor, an NVIDIA GTX 3060 graphics card, and 16 GB of RAM. The programming language used for the experiments is Python v3.9, implemented using the PyCharm integrated development environment.

Table 8: Parameter settings of different algorithms.

Algorithm	**Parameter Settings**
Loyalty-SMOTE	Neighbors coefficient - Loyalty $k = 9$ Neighbors coefficient - SMOTE $k = 5$ Boundary threshold $m = 0.5$ Sampling ratio = 2
SMOTE	Neighbors coefficient $k = 5$ Sampling ratio = 2
SMOTE-ENN	Neighbors coefficient - SMOTE $k = 5$ Neighbors coefficient - ENN $k = 3$ Sampling ratio = 2
SMOTE-TL	Neighbors coefficient - SMOTE $k = 5$ Sampling ratio = 2
BL-SMOTE	Neighbors coefficient - SMOTE $k = 5$ Boundary coefficient $m = 10$ Sampling ratio = 2
SL-SMOTE	Neighbors coefficient - SMOTE $k = 5$ Safety coefficient $m = 10$ Sampling ratio = 2
SVM	Penalty parameter $C = 1.0$ Kernel function, kernel = rbf Gamma = 0.5

4.4. Experimental Results

We evaluate the system performance of Loyalty-SMOTE, the impact of key parameters on the performance of the Loyalty-SMOTE algorithm, the time efficiency of the Loyalty-SMOTE algorithm, and its performance in multiclass problems.

4.4.1. Classification Performance Comparisons

As listed from Table 9 to Table 12, the performance results of the six indicated algorithms (including F1-score, AUROC, Recall, and G-mean) on the Group 1 datasets are shown. The best scores for all the tests are put in bold types.

In Tables 9 and 10, among the 25 datasets in Group 1, our proposed Loyalty-SMOTE achieved the best F1-score in 21 datasets (84%) and the highest AUROC in 24 datasets (96%). As shown in Table 11, it also attained the highest Recall in 21 datasets (84%). Furthermore, Table 12 demonstrates that Loyalty-SMOTE achieved the best G-mean in 22 datasets (88%) across all test cases. Based on these measured experimental results, it can be concluded that the Loyalty-SMOTE algorithm performs better than the other algorithms in terms of Recall, F1-score, AUC, and G-mean across most datasets.

4.4.2. Key Parameters on Algorithm Performance

As stated in Sections 3.1, 3.2, and 3.3, the number of neighbors k and the boundary threshold m are both key parameters in the Loyalty-SMOTE algorithm. Therefore, this section employs the method of controlling variables to conduct repeated experiments by varying the third variable while keeping the other two variables fixed, in order to explore the impact of these key parameters on the performance of the Loyalty-SMOTE algorithm.

To study the influence of key parameters on algorithm performance, the experiments utilize the Group 2 dataset mentioned in Section 4.1. The experimental results are shown in Figure 8 and 9. In Figure 8, it can be observed that during the increment of m from 0.1 to 0.9, the F1-Score, AUC, Recall, and G-mean of the *obesity* dataset in Group 2 experience a sharp decline at $m = 0.5$. The *Australia* dataset significantly increases at $m = 0.3$, while the other three datasets do not show drastic reactions to the changes in m. Thus, we conclude that the value of the boundary threshold m has a highly unstable effect on the algorithm. This instability is due to the distribution of Loyalty values in each dataset being related to the distribution of that particular data; therefore, different datasets have varying distributions of Loyalty values. As a result, when m takes on a specific value, the proportion of samples classified as boundary samples differs across datasets, leading to varying sensitivities to m. From Figure 9, aside from the *Australia* dataset which experiences significant changes at $k = 17$, the Loyalty-SMOTE algorithm maintains stable F1-score, AUC, Recall, and G-mean across the other datasets at any given time. In summary, the value of the number of neighbors k has a relatively minor impact on the performance of the Loyalty-SMOTE algorithm; conversely, the effect of the boundary threshold m on performance is related to the distribution of Loyalty across all samples. Therefore, it is recommended to try multiple parameter values for both k and m in practical applications of the Loyalty-SMOTE algorithm to achieve optimal results. From Figure 8 and 9, it can be seen that the Loyalty-SMOTE algorithm performs relatively stable around $k = 9$ and $m = 0.3$. Hence, it is suggested to set $k = 9$ and $m = 0.3$ in the algorithm.

4.4.3. Runtime Performance

The time overhead of algorithms in the context of big data problems is often enormous (Maillo et al., 2020). Currently, the issue of imbalanced big data classification is considered a significant challenge under the umbrella of big data classification problems (Kulkarni et al., 2020). This compels us to continuously develop more efficient algorithms to address these imbalanced data classification issues in big data. The time complexities of various algorithms are shown in Table 13.

Table 9: F1-scores on Group 1 datasets.

Dataset	SMOTE	SMOTE-TL	BL-SMOTE	SMOTE-ENN	SL-SMOTE	Loyalty-SMOTE
adult	0.8409	**0.8743**	0.8321	0.8403	0.8255	0.8232
ai4i2020	0.8219	0.5785	0.8571	0	0.8619	**0.9680**
banknote	1	1	1	1	0.9991	**1**
drug	0.7261	0.7620	0.5907	0.7444	0.6979	**0.9971**
AIDS_5000	0.8026	0.7221	0.6960	0.8061	0.7834	**0.8932**
Maternal	0.8786	0.7973	0.8015	0.8822	0.8741	**0.9010**
national	**0.9884**	0.9814	0.9837	0.9587	0.9875	0.9913
occupancy	0.9904	0.9893	0.9778	0.9904	0.9904	**0.9904**
online	0.4480	0.5376	0.2313	0.4449	0.3159	**0.5980**
Shoppers	0.7756	0.8233	0.6466	0.7814	0.7681	**0.8441**
Academic	0.8611	0.8717	0.8616	0.7814	0.8462	**0.9509**
AIDS_15000	0.8611	0.8717	0.8616	0.7814	0.7681	**0.8995**
german	0.7962	0.8098	0.8028	0.7886	0.7780	**0.9079**
statlog	0.8020	0.7861	0.7659	0.7920	0.7800	**0.8934**
Shuttle	0.9965	0.9965	0.9902	0.9979	**0.9986**	0.9807
Fault	0.8370	0.8137	0.8165	0.8222	0.7926	**0.9624**
ICR	0.9206	0.9128	0.9205	0.9197	0.9206	**0.9904**
breast	0.9793	0.9858	0.9272	0.9793	0.9793	**0.9920**
WDBC	0.9741	0.9817	0.9720	0.9741	0.9741	**0.9915**
WPBC	0.7674	0.8434	0.7813	0.8261	0.7381	**0.9010**
heart	0.8878	**0.8914**	0.7354	0.8499	0.8298	0.8515
ionosphere	0.9558	0.9633	0.9726	0.9652	0.9607	**0.9833**
iris	1	1	1	1	1	**1**
bezdeiris	1	1	1	1	1	**1**
raisin	0.8931	0.8950	0.8566	0.9038	0.8957	**0.9270**

Table 10: AUROCs on Group 1 datasets.

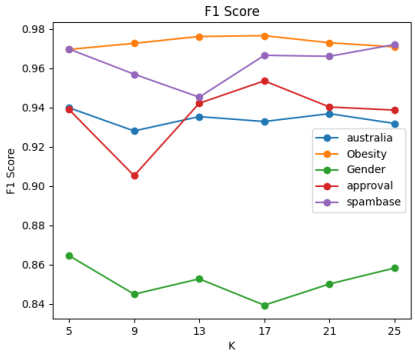
Dataset	SMOTE	SMOTE-TL	BL-SMOTE	SMOTE-ENN	SL-SMOTE	Loyalty-SMOTE
adult	0.8791	0.9135	0.8816	0.8762	0.8599	**0.9326**
ai4i2020	1	0.5223	1	0	1	**1**
banknote	1	1	1	1	1	**1**
drug	0.7429	0.7825	0.5738	0.7712	0.7095	**1**
AIDS_5000	0.8308	0.7845	0.7627	0.8289	0.8136	**0.9334**
Maternal	0.8684	0.7479	0.7678	0.8742	0.8652	**0.9347**
national	0.9857	0.9702	0.9811	0.9857	0.9875	**0.9999**
occupancy	1	1	1	1	1	**1**
online	0.3103	0.4094	0.1352	0.3084	0.1966	**0.5981**
Shoppers	0.7327	0.7713	0.5947	0.7410	0.7209	**0.8951**
Academic	0.8859	0.8951	0.9086	0.7410	0.8701	**0.9874**
AIDS_15000	0.8315	0.8348	0.7752	0.8294	0.8294	**0.9458**
german	0.7477	0.7867	0.8066	0.7477	0.7342	**0.9609**
statlog	0.7960	0.7596	0.7548	0.7861	0.7761	**0.9694**
Shuttle	1	0.9956	0.9805	**1**	1	0.9995
Fault	0.7902	0.7810	0.8074	0.7762	0.7483	**0.9892**
ICR	0.9925	0.9647	0.9794	0.9824	0.9833	**1**
breast	1	1	0.9211	1	1	**1**
WDBC	0.9576	0.9727	0.9720	0.9576	0.9576	**0.9996**
WPBC	0.7174	0.8537	0.7813	0.8261	0.6739	**0.9332**
heart	0.8922	0.9176	0.7589	0.9004	0.8966	**0.9627**
ionosphere	0.9391	0.9633	0.9595	0.9652	0.9565	**1**
iris	1	1	1	1	1	**1**
bezdeiris	1	1	1	1	1	**1**
raisin	0.7328	0.6947	0.8182	0.7328	0.7328	**0.9701**

Table 11: Recall results on Group 1 datasets.

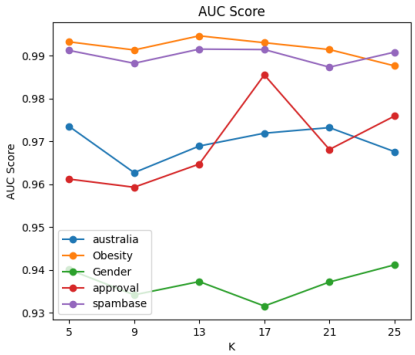
Dataset	SMOTE	SMOTE-TL	BL-SMOTE	SMOTE-ENN	SL-SMOTE	Loyalty-SMOTE
adult	0.8409	**0.8743**	0.8321	0.8403	0.8255	0.8571
ai4i2020	0.8219	0.5785	0.8571	0	0.8619	**0.9483**
banknote	1	1	1	1	0.9991	**1**
drug	0.7261	0.7620	0.5907	0.7444	0.6979	**1**
AIDS_5000	0.8418	0.7439	0.6882	0.8489	0.8335	**0.9296**
Maternal	0.8786	0.7973	0.8015	0.8822	0.8741	**0.9368**
national	**0.9884**	0.9814	0.9837	0.9587	0.9875	0.9846
occupancy	0.9904	0.9893	0.9778	0.9904	0.9904	**0.9928**
online	0.4480	0.5376	0.2313	0.4449	0.3159	**0.5980**
Shoppers	0.7756	0.8233	0.6466	0.7814	0.7681	**0.8889**
Academic	0.8611	0.8717	0.8616	0.7814	0.8462	**0.9620**
AIDS_15000	0.8041	0.7794	0.7062	0.8842	0.8065	**0.9130**
german	0.7962	0.8098	0.8028	0.7886	0.7780	**0.8698**
statlog	0.8020	0.7861	0.7659	0.7920	0.7800	**0.9226**
Shuttle	0.9965	0.9965	0.9902	0.9979	**0.9986**	0.9982
Fault	0.8370	0.8137	0.8165	0.8222	0.7926	**0.9884**
ICR	0.8969	0.9175	0.9177	0.8865	0.8969	**0.9811**
breast	0.9793	0.9858	0.9272	0.9793	0.9793	**0.9934**
WDBC	0.9741	0.9817	0.9720	0.9741	0.9741	**0.9915**
WPBC	0.7674	0.8434	0.7813	0.8261	0.7381	**0.9762**
heart	0.8878	**0.8914**	0.7354	0.8499	0.8298	0.7818
ionosphere	0.9558	0.9633	**0.9726**	0.9652	0.9607	0.9672
iris	1	1	1	1	1	**1**
bezdeiris	1	1	1	1	1	**1**
raisin	0.8931	0.8950	0.8566	0.9038	0.8957	**0.9153**

Table 12: G-mean results on Group 1 datasets.

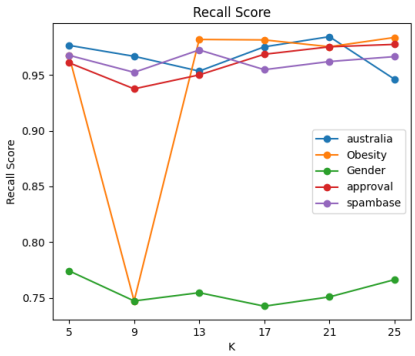
Dataset	SMOTE	SMOTE-TL	BL-SMOTE	SMOTE-ENN	SL-SMOTE	Loyalty-SMOTE
adult	0.8409	**0.8743**	0.8321	0.8403	0.8255	0.8485
ai4i2020	0.8219	0.5785	0.8571	0	0.8619	**0.9685**
banknote	1	1	1	1	0.9991	**1**
drug	0.7261	0.7620	0.5907	0.7444	0.6979	**0.9973**
AIDS_5000	0.7433	0.7164	0.6897	0.7459	0.7299	**0.8532**
Maternal	0.8786	0.7973	0.8015	0.8822	0.8741	**0.9368**
national	0.9884	0.9814	0.9837	0.9587	0.9875	**0.9914**
occupancy	**0.9904**	0.9893	0.9778	0.9904	0.9904	0.9846
online	0.4480	0.5376	0.2313	0.4449	0.3159	**0.5980**
Shoppers	0.7756	0.8233	0.6466	0.7814	0.7681	**0.8294**
Academic	0.8611	0.8717	0.8616	0.7814	0.8462	**0.9556**
AIDS_15000	0.7373	0.7432	0.7128	0.7335	0.7335	**0.8734**
german	0.7962	0.8098	0.8028	0.7886	0.7780	**0.8946**
statlog	0.8020	0.7861	0.7659	0.7920	0.7800	**9100**
Shuttle	0.9965	0.9965	0.9902	0.9979	**0.9986**	0.9965
Fault	0.8370	0.8137	0.8165	0.8222	0.7926	**0.9844**
ICR	0.9321	0.9233	0.9252	0.9302	0.9321	**0.9955**
breast	0.9793	0.9858	0.9272	0.9793	0.9793	**0.9882**
WDBC	0.9741	0.9817	0.9720	0.9741	0.9741	**0.9904**
WPBC	0.7674	0.8434	0.7813	0.8261	0.7381	**0.9011**
heart	0.8878	**0.8914**	0.7354	0.8499	0.8298	0.8663
ionosphere	0.9558	0.9633	0.9726	0.9652	0.9607	**0.9836**
iris	1	1	1	1	1	**1**
bezdeiris	1	1	1	1	1	**1**
raisin	0.8931	0.8950	0.8566	0.9038	0.8957	**0.9336**



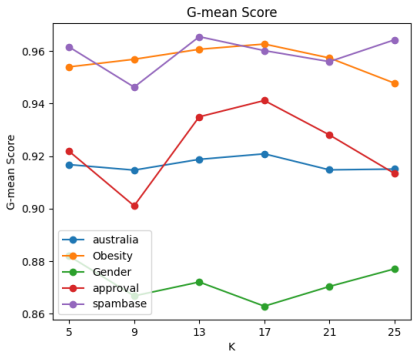
(a)



(b)

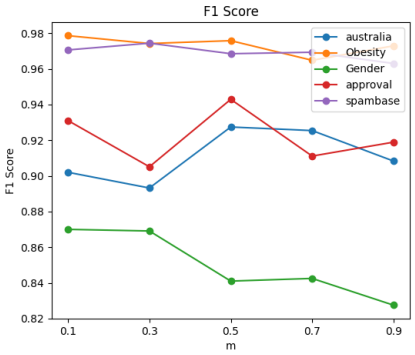


(c)

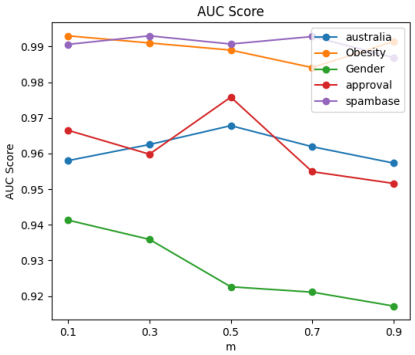


(d)

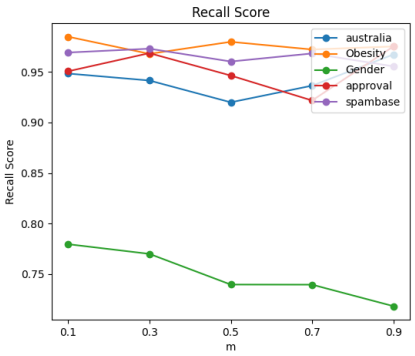
Figure 8: Boundary Threshold m on the performance of the Loyalty-SMOTE: (a) F1-score; (b) AUROC; (c) Recall; (d)G-mean.



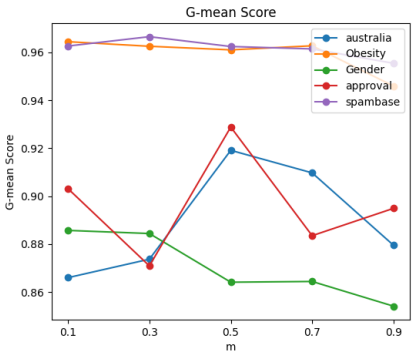
(a)



(b)



(c)



(d)

Figure 9: Neighboring coefficient k on the performance of the Loyalty-SMOTE: (a) F1-score; (b) AUROC; (c) Recall; (d)G-mean.

Table 13: Time Complexity.

Algorithm	Complexity
SMOTE	$O(nk + nm)$
SMOTE-TL	$O(nk + nm + n_{tl}k_{tl})$
BL-SMOTE	$O(nk + n_b m)$
SMOTE-ENN	$O(nk + nm + n_{enn}k_{enn})$
SL-SMOTE	$O(nk + nm + n_s k_s)$
Loyalty-SMOTE	$O(Nk + nk + nm)$

In Table 13, N represents the total number of samples, n is the number of samples in the minority class, k is the SMOTE neighbor coefficient, and m is the number of synthetic samples generated. n_b refers to the number of boundary samples, n_{enn} is the count of samples processed by the ENN algorithm, and k_{enn} is the number of k neighbors in the ENN stage. n_{tl} indicates the count of Tomek Links identified after SMOTE processing, while k_{tl} is the number of k neighbors during the processing of Tomek Links. n_s denotes the number of safe-level samples, and k_s refers to the k neighbors calculated for the safe level. k represents the number of k neighbors used for calculating loyalty. From Table 13, it can be seen that the time complexity of Loyalty-SMOTE is generally not the best among the six algorithms, as Loyalty-SMOTE requires traversing all samples during the Loyalty calculation phase, which impacts the performance of the Loyalty-SMOTE.

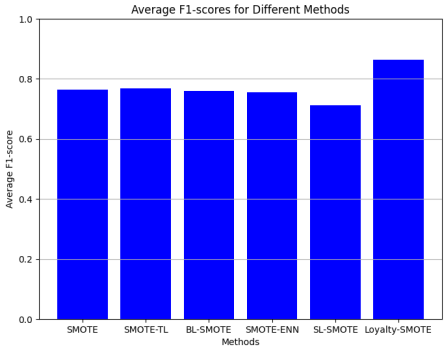
The running times of the six algorithms are shown in Table 14. The Loyalty-SMOTE algorithm demonstrates relatively stable time performance across the 9 datasets in Group 3. Its time performance ranks at a mid-level among the six algorithms. It is also evident that its time performance is significantly better than that of the Safe-Level-SMOTE algorithm and remains relatively close to the time performance of the SMOTE-ENN algorithm.

4.4.4. Multi-class Classification

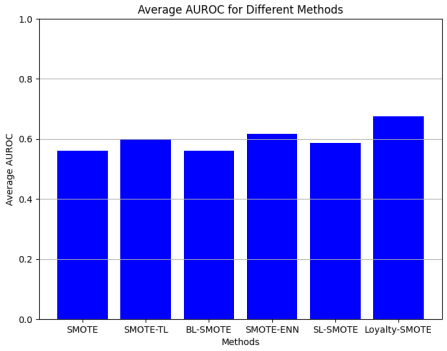
For multi-class imbalanced datasets, the class with the highest number of instances is considered the majority class, while the class with the fewest instances is considered the minority class. The experimental results of the multi-class dataset Group 4 are presented in Tables 15, 16, 17, and 18.

From Tables 15 to 18, it can be seen that Loyalty-SMOTE algorithm achieves the highest F1-Score, Recall, and G-mean across all five datasets in Group 4, and it also reaches the highest AUC value for three datasets. To visually demonstrate the performance of the Loyalty-SMOTE algorithm on multi-class classification, bar charts displaying the average results for F1-Score, AUC, Recall, and G-mean are presented in Figure 10.

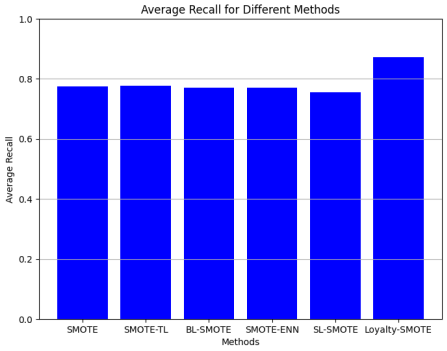
In Figure 10, it can be seen that the average values of F1-Score, Recall, G-mean, and AUC for the Loyalty-SMOTE algorithm are significantly the highest among the six algorithms, reaching 0.8317, 0.6153, 0.8537, and 0.6717, respectively. It can be concluded that, compared to the other five algorithms, the Loyalty-SMOTE algorithm demonstrates a considerable improvement in performance when handling multi-class imbalanced data problems.



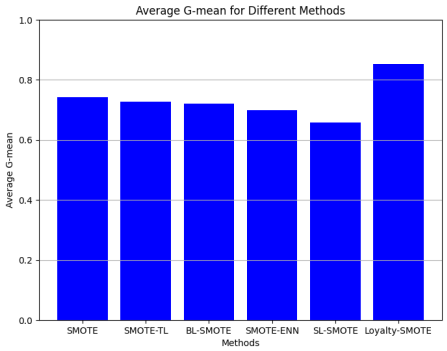
(a)



(b)



(c)



(d)

Figure 10: Average results for F1-score, AUC, Recall, and G-mean.

Table 14: Runtime Comparisons.

Dataset	SMOTE	SMOTE-TL	BL-SMOTE	SMOTE-ENN	SL-SMOTE	Loyalty-SMOTE
bank-full	0.1986	0.2313	0.1475	0.1808	1.5405	0.1994
magic04	0.1361	0.4506	0.3061	0.5904	5.3407	1.3548
covertype	2804.6083	9084.7474	6039.1104	10611.5414	50061.2422	8072.226
Purchase	0.0267	0.3215	0.0717	0.7179	0.6922	0.4057
Satisfied	9.5190	69.5993	13.1038	71.2169	402.7582	49.2664
AIDS_50000	0.4908	4.8425	1.3810	4.6529	21.7226	4.4664
smoke	0.2001	1.4463	0.4100	1.5201	7.3310	1.6729
vertebal	0.0001	0.0004	0.0004	0.0005	0.0337	0.0009

Table 15: F1-scores for Group 4 multi-class datasets.

Dataset	SMOTE	SMOTE-TL	BL-SMOTE	SMOTE-ENN	SL-SMOTE	Loyalty-SMOTE
yeast	0.6535	0.6552	0.6474	0.6386	0.6273	**0.7801**
HR	0.7150	0.7270	0.7066	0.7210	0.6118	**0.7417**
employee	0.6565	0.6744	0.6601	0.6703	0.6475	**0.9136**
Drug200	0.9705	0.9603	0.9779	0.9558	0.8863	**0.9919**
MEDV	0.8229	0.8223	0.8016	0.7916	0.7882	**0.8959**

Table 16: AUROC values for Group 4 multi-class datasets.

Dataset	SMOTE	SMOTE-TL	BL-SMOTE	SMOTE-ENN	SL-SMOTE	Loyalty-SMOTE
yeast	0.5776	0.5848	0.5703	0.5674	0.5835	**0.6147**
HR	0.2730	**0.4437**	0.2672	0.4221	0.3935	0.2428
employee	0.5667	0.6115	0.5867	0.7741	0.6036	**0.8173**
Drug200	0.5835	0.5693	0.5749	0.5386	0.5443	**0.6179**
MEDV	0.7981	0.7917	0.8003	0.7853	**0.8036**	0.6836

Table 17: Recall values for Group 4 multi-class datasets.

Dataset	SMOTE	SMOTE-TL	BL-SMOTE	SMOTE-ENN	SL-SMOTE	Loyalty-SMOTE
yeast	0.6610	0.6633	0.6564	0.6429	0.7748	**0.7889**
HR	0.7293	0.7357	0.7244	0.7524	0.6606	**0.7889**
employee	0.6805	0.6939	0.6836	0.6930	0.6724	**0.9136**
Drug200	0.9708	0.9602	0.9781	0.9560	0.8833	**0.9919**
MEDV	0.8283	0.8311	0.8065	0.8082	0.7913	**0.8967**

Table 18: G-mean values for Group 4 multi-class datasets.

Dataset	SMOTE	SMOTE-TL	BL-SMOTE	SMOTE-ENN	SL-SMOTE	Loyalty-SMOTE
yeast	0.6361	0.6427	0.6196	0.5796	0.6043	**0.7592**
HR	**0.7369**	0.7067	0.7330	0.6515	0.5934	0.6946
employee	0.6543	0.6385	0.6277	0.6378	0.6146	**0.9136**
Drug200	0.9697	0.9680	0.9771	0.9569	**0.8274**	0.9935
MEDV	0.7098	0.6854	0.6491	0.6681	0.6515	**0.8986**

4.4.5. Classification of Large-Scale and High-Dimensional Datasets

To evaluate the performance of the Loyalty-SMOTE algorithm on large-scale and high-dimensional datasets, this paper compares the performance of six algorithms on Group 5. The experimental results are presented in Tables 19, 20, 21, and 22.

From the experimental results, it can be seen that the Loyalty-SMOTE algorithm achieves the best performance on five large-scale and five high-dimensional datasets in Group 5. It can be concluded that Loyalty-SMOTE is proficient in handling large-scale and high-dimensional datasets.

5. Discussion

Traditional methods for handling imbalanced data, such as the SMOTE algorithm, are effective in improving the performance of classifiers, especially in terms of classification accuracy for minority class samples. However, SMOTE is sensitive to noise (Cheng et al., 2019). A common approach is to detect and remove noise before oversampling, which alters the original sample distribution and can lead to newly generated samples that do not accurately represent the original minority class (Nugroho et al., 2021). In the Loyalty-SMOTE algorithm proposed in this paper, the concept of Loyalty is introduced to determine whether a sample is loyal to its class, thereby inferring whether it is noise. During the assimilation process, signed all unloyal samples for subsequent data cleaning, and based on this, generate new samples to reduce the impact of noise in the original samples on the algorithm's results.

Boundary samples are typically the samples that classifiers find most difficult to classify correctly. By oversampling around these boundary samples, the classifier's learning ability concerning these samples can be enhanced, thereby improving overall classification performance. This paper adopts this approach; by using Loyalty to identify the boundaries of the minority class, SMOTE is then applied to oversample these samples, achieving the goal of data balancing (Li et al., 2022). To avoid generating new noise with SMOTE, Loyalty-SMOTE performs denoising on the data to enhance the robustness of the algorithm.

Conventional algorithms for handling imbalanced data, such as SMOTE and Borderline-SMOTE, perform well in binary classification problems but are less effective in multi-class imbalanced problems (Xu et al., 2023). This is mainly because the SMOTE algorithm tends to generate erroneous instances in majority class regions (Naglik & Lango, 2023), which can lead to overfitting of the classifier (Montesinos Lopez et al., 2022). On the other hand, Borderline-SMOTE performs oversampling by locating boundary samples. Unfortunately, determining boundary instances can be challenging in multi-class data, leading to the generation of overlapping data and ultimately resulting in overfitting. However, the Loyalty-SMOTE algorithm addresses these issues by preserving the original data distribution before oversampling and then generating new data on this foundation, ensuring that the generated data adequately reflects the distribution of the entire multiclass dataset.

This paper conducted five sets of experiments on the Loyalty-SMOTE algorithm. In the performance comparison experiment, we compared Loyalty-SMOTE with five mainstream algorithms for handling imbalanced data, including SMOTE. The results show that Loyalty-SMOTE achieved the highest F1-score in 21 datasets (84%) and the highest AUROC in 24 datasets (96%). it also attained the highest Recall in 21 datasets (84%). Furthermore, Loyalty-SMOTE achieved the highest G-mean in 22 datasets (88%). We then discussed the impact of key parameters on the performance of Loyalty-SMOTE, concluding that the algorithm's sensitivity to the boundary threshold m is related to the Loyalty distribution of all samples in the dataset, while the size of the neighbor coefficient k has a relatively minor impact on the Loyalty-SMOTE algorithm. This paper recommends setting the parameters as $k = 9$ and $m = 0.3$.

Next, we explored the time efficiency of Loyalty-SMOTE, analyzing its time complexity. It was found that the Loyalty-SMOTE algorithm exhibits good time performance, significantly outperforming the Safe-Level-SMOTE algorithm, and its time performance is relatively close to that of the SMOTE-ENN algorithm. Then, we investigated the performance of the Loyalty-SMOTE algorithm on the five multi-class datasets from group4 obtained from UCI, revealing that Loyalty-SMOTE outperforms the other five mainstream algorithms for handling imbalanced data in tackling multi-class problems. It achieved the highest F1-Score, Recall, and G-mean on all five datasets, with the highest AUC on three of them. Finally, we explored the performance of the Loyalty-SMOTE algorithm on large-scale and high-dimensional datasets, and we found that the Loyalty-SMOTE algorithm performs best on all large-scale and high-dimensional datasets in Group 5. This demonstrates that the Loyalty-SMOTE algorithm is proficient in handling large-scale and high-dimensional datasets.

In our experiments, we observed that the Loyalty-SMOTE algorithm does not consistently perform best on all datasets, such as 'adult' and 'shuttle'. To discuss potential reasons, we selected datasets where Loyalty-SMOTE performed poorly ('adult' and 'shuttle') and datasets where it performed well ('faults' and 'statlog'). Principal Component Analysis (PCA) was used to reduce the dimensionality of these datasets. The resulting data distributions are shown in Figure 11, where subfigures (a), (b), (c), and (d) represent the reduced dimensionality results for the four datasets, respectively.

From Figure 11, we can see that adult and shuttle exhibit clear class boundaries with compact clusters, while faults and statlog have complex, overlapping distributions. We infer that Loyalty-SMOTE performs relatively worse on datasets with clear, simple boundaries but excels on complex, irregular distributions. We believe this is because the Loyalty-SMOTE algorithm, compared to traditional imbalanced data handling algorithms (such as SMOTE), possesses unique boundary oversampling and outlier cleaning mechanisms. It makes it more suitable for datasets with complex distributions or fuzzy boundaries (like faults and statlog datasets) than traditional imbalanced data handling algorithms. For datasets with simple distributions and clear boundaries, the boundary oversampling and outlier cleaning mechanisms of Loyalty-SMOTE are less effec-

Table 19: F1-Scores for Group 5 datasets.

Dataset	SMOTE	SMOTE-TL	BL-SMOTE	SMOTE-ENN	SL-SMOTE	Loyalty-SMOTE
Mushroom	0.8834	0.8816	0.9186	0.8842	0.8839	**0.9917**
Firewall	0.9970	0.9973	0.9976	0.9970	0.9970	**0.9988**
Cardiovascular	0.8323	0.7347	0.7025	0.8994	0.8263	**0.9135**
RT-IOT	0.9236	0.9268	0.9779	0.9237	0.9235	**0.9935**
Gender_by_name	0.7941	0.7928	0.0223	0.7941	0.6122	**0.7685**
Smartphone	0.9987	0.9987	0.9956	0.9987	0.9987	**1**
Bankruptcy	0.6064	0.6454	0.5768	0.6006	0.7926	**0.9979**
Darwin	0.8889	0.9803	0.9090	0.8889	0.8636	**1**
toxicity	0.8363	0.7809	0.7346	0.8392	0.8173	**0.9356**
cnae	0.7507	0.6949	0.6550	0.7397	0.7510	**0.8149**

Table 20: AUROC for Group 5 datasets.

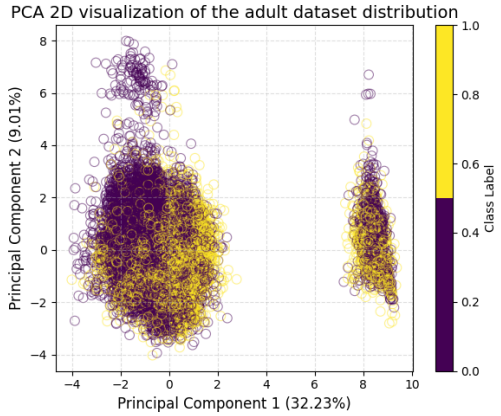
Dataset	SMOTE	SMOTE-TL	BL-SMOTE	SMOTE-ENN	SL-SMOTE	Loyalty-SMOTE
Mushroom	0.8110	0.8096	0.9161	0.8130	0.8128	**0.9996**
Firewall	0.9960	0.9956	0.9967	0.9960	0.9960	**0.9987**
Cardiovascular	0.8539	0.8163	0.7609	0.9550	0.8460	**0.9566**
RT-IOT	0.9937	0.9958	0.5749	0.9940	0.9938	**0.9989**
Gender_by_name	0.5113	0.5157	0.5000	0.4879	0.5001	**0.7853**
Smartphone	0.9999	1	1	1	0.9999	**1**
Bankruptcy	0.4815	0.5127	0.4402	0.4769	0.7483	**0.9997**
Darwin	0.9970	1	1	0.9920	1	**1**
toxicity	0.8804	0.8449	0.7947	0.8852	0.8566	**0.9559**
cnae	0.7011	0.6641	0.6870	0.6771	0.6776	**0.7580**

Table 21: Recall for Group 5 datasets.

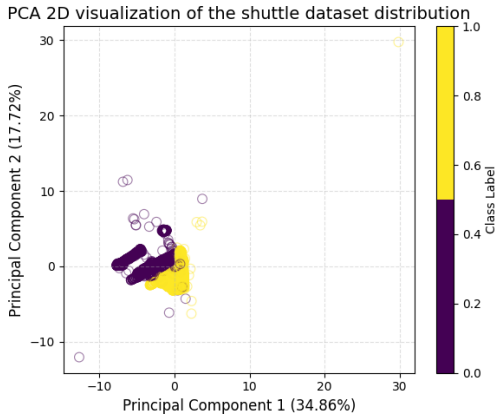
Dataset	SMOTE	SMOTE-TL	BL-SMOTE	SMOTE-ENN	SL-SMOTE	Loyalty-SMOTE
Mushroom	0.8834	0.8816	0.9186	0.8842	0.8839	**0.9918**
Firewall	0.9970	0.9973	0.9976	0.9970	0.9970	**0.9987**
Cardiovascular	0.8481	0.7205	0.7143	0.8847	0.8439	**0.8973**
RT-IOT	0.9909	0.9925	0.9781	0.9911	0.9909	**0.9890**
Gender_by_name	0.9999	0.0107	0.0013	0.9996	0.9997	**1**
Smartphone	1	1	1	1	1	**1**
Bankruptcy	0.6064	0.6454	0.5768	0.6006	0.7926	**0.9959**
Darwin	0.8000	0.9615	0.8333	0.8000	0.7600	**1**
toxicity	0.9787	0.8723	0.8181	1	1	**1**
cnae	0.9522	0.8838	0.7318	0.9301	0.9705	**0.9056**

Table 22: G-mean for Group 5 datasets.

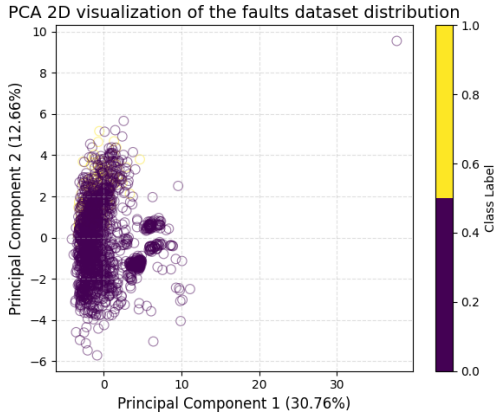
Dataset	SMOTE	SMOTE-TL	BL-SMOTE	SMOTE-ENN	SL-SMOTE	Loyalty-SMOTE
Mushroom	0.8834	0.8816	0.9186	0.8842	0.8839	**0.9917**
Firewall	0.9970	0.9973	0.9976	0.9970	0.9970	**0.9988**
Cardiovascular	0.7552	0.7414	0.6960	0.8925	0.7453	**0.9026**
RT-IOT	0.9843	0.9854	0.9771	0.9844	0.9843	**0.9935**
Gender_by_name	0.4999	0.5000	0.5007	0.4999	0.5001	**0.5000**
Smartphone	0.9984	0.9984	0.9979	0.9984	0.9984	**1**
Bankruptcy	0.6064	0.6454	0.5768	0.6006	0.7926	**0.9979**
Darwin	0.9000	0.9807	0.9167	0.9000	0.8800	**1**
toxicity	0.7656	0.7124	0.7727	0.7631	0.7236	**0.9431**
cnae	0.6420	0.6133	0.6278	0.6301	0.6344	**0.7168**



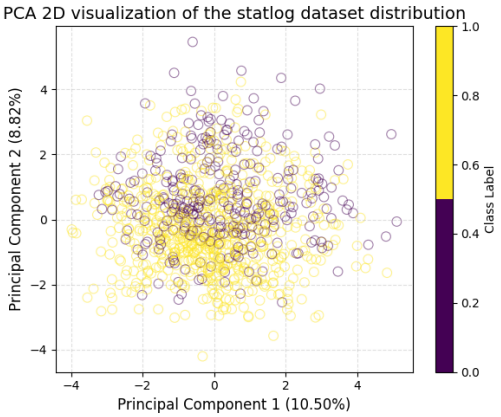
(a)



(b)



(c)



(d)

Figure 11: Dimensionality-reduced adult, shuttle, faults, and statlog datasets.

tive, resulting in no significant performance improvement over other methods.

In the study of the Loyalty-SMOTE algorithm, it was found that the algorithm sufficiently considers the distribution characteristics of the samples, retains their original distribution as much as possible, and then applies a more efficient oversampling strategy, which is key to improving the efficiency of imbalanced data processing algorithms.

6. Conclusion

In this paper, we have presented a novel oversampling technique, Loyalty-SMOTE, which addresses the imbalanced data classification issue. In the design of Loyalty-SMOTE, we introduce the concepts of Loyalty and Attraction. The Loyalty identifies any unloyal points within a dataset. Then, upon recognizing these unloyal points, an assimilation process is executed to denoise the dataset based on the calculated Loyalty values. For denoising in multi-class datasets, values of Attraction are computed and used during the assimilation process. Indeed, the computation step of Loyalty assists the identification of sample boundaries that permit oversampling properly at the boundaries. Finally, Loyalty-SMOTE applies a second denoising process to clean up any new noise that may be introduced by SMOTE. Experimental results show that Loyalty-SMOTE achieved the highest F1-score in 21 datasets (84%), the highest AUROC in 24 datasets (96%), the highest Recall in 21 datasets (84%), and the highest G-mean in 22 datasets (88%) among the 25 imbalanced datasets in Group 1 using SVM classifiers. Our Loyalty-SMOTE outperforms all other five algorithms. For experiments on multi-class problems, Loyalty-SMOTE again demonstrates strong performance with average values of F1-score, AUC, Recall, and G-mean reaching 0.8317, 0.6153, 0.8537, and 0.6717, respectively. It scores higher than those of the other algorithms. The current known drawback of Loyalty-SMOTE is that it takes longer runtime with higher time complexity than other traditional imbalanced data processing algorithms. Our future work will focus on diminishing the time complexity of the Loyalty-SMOTE algorithm.

References

Abd El-Naby, A., Hemdan, E. E.-D., El-Sayed, A., *An efficient fraud detection framework with credit card imbalanced data in financial services*, Multimedia Tools and Applications, 82(3), 4139–4160. 2023, `https://doi.org/10.1007/s11042-022-13434-6` .

Chatterjee, S., Breitkopf, M., Sarasaen, C., Yassin, H., Rose, G., Nurnberger, A., Speck, O., *ReconResNet: Regularised residual learning for MR image reconstruction of Undersampled Cartesian and Radial data*, Computers in Biology and Medicine, 143, 105321, Apr. 2022, Elsevier, `https://doi.org/10.1016/j.compbiomed.2022.105321` .

Cheng, K., Zhang, C., Yu, H., Yang, X., Zou, H., Gao, S., *Grouped SMOTE With Noise Filtering Mechanism for Classifying Imbalanced Data*, IEEE Access, 7, 170668–170681, 2019, IEEE, `https://doi.org/10.1109/ACCESS.2019.2955086` .

Boonchuay, K., Sinapiromsaran, K., Lursinsap, C., *Boundary expansion algorithm of a decision tree induction for an imbalanced dataset*, Songklanakarin Journal of Science & Technology, 39(5), 2017, `https://www.thaiscience.info/Journals/Article/SONG/10988018.pdf` .

Chawla, N. V., Bowyer, K. W., Hall, L. O., Kegelmeyer, W. P. *SMOTE: Synthetic Minority Over-sampling Technique*, J. Artificial Intelligence Research, 16, 321–357. 2002, https://doi.org/10.1613/jair.953 .

Chen, Y., Chang, H., Meng, J., Zhang, D., *Ensemble Neural Networks (ENN): A gradient-free stochastic method*, Neural Networks. 110, 170-185, 2019, https://doi.org/10.1016/j.neunet.2018.11.009 .

Dolo, K. M., Mnkandla, E., *Modifying the SMOTE and Safe-Level SMOTE Oversampling Method to Improve Performance*, 4th Internat. Conf. Wireless, Intelligent and Distributed Environment for Communication, 47–59, Springer International Publishing, https://doi.org/10.1007/978-3-030-89776-5_4 .

Espindola, R. P., Ebecken, N. F. F., *On Extending F-measure and G-mean Metrics to Multi-class Problems*, WIT Trans. Information and Communication Technologies, 35, 25–34, WIT Press.

Gortler, J., Hohman, F., Moritz, D., Wongsuphasawat, K., Ren, D., Nair, R., Kirchner, M., Patel, K., *Generalizing confusion matrix visualization to hierarchical and multi-output labels*, Proc. Conf. Human Factors in Computing Systems (CHI 2022), 1–13, https://doi.org/10.1145/3491102.3501823 .

Han, H., Wang, W.-Y., Mao, B.-H., *Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning*, Internat. Conf. Intelligent Computing (ICIC 2005), 878–887, Hefei, China, 23-26 Aug., 2005, https://doi.org/10.1007/11538059_91 .

Hariri, S., Kind, M. C., Brunner, R. J., *Extended Isolation Forest*, IEEE Trans. Knowledge & Data Engineering, 33(4), 1479–1489, 2021, https://doi.org/10.1109/TKDE.2019.2947676 .

Hasanin, T., Khoshgoftaar, T. M., Leevy, J. L., Bauder, R. A., *Investigating class rarity in big data*, J. Big Data, 7(23), 2020, https://doi.org/10.1186/s40537-020-00301-0 .

Hearst, M. A., *Support Vector Machines*, IEEE Intelligent Systems Magazine, Jul/Aug. 1998, 18–28.

Krasic, I., Celar, S., *Telecom Fraud Detection with Machine Learning on Imbalanced Dataset*, 2022 Internat. Conf. Software, Telecommunications & Computer Networks (SoftCOM 2022), 1–6, https://doi.org/10.23919/SoftCOM55329.2022.9911518 .

Kulkarni, A., Chong, D., Batarseh, F. A., *Foundations of data imbalance and solutions for a data democracy*, Data Democracy, 83–106, 2020, Academic Press, https://doi.org/10.1016/B978-0-12-818366-3.00005-8 .

Li, Y., Hsu, W.-W., Initiative for the A. D. N., *A classification for complex imbalanced data in disease screening and early diagnosis*, Statistics in Medicine, 41(19), 3679–3695, 2022, Wiley, https://doi.org/10.1002/sim.9442 .

Kynkaanniemi, T., Karras, T., Laine, S., Lehtinen, J., Aila, T., *Improved Precision and Recall Metric for Assessing Generative Models*, Proc. Advances in Neural Information Processing Systems 32 (NeuIPS 2019), https://proceedings.neurips.cc/paper/2019/hash/0234c510bc6d908b28c70ff313743079-Abstract.html .

Li, D.-C., Shi, Q.-S., Lin, Y.-S., Lin, L.-S., *A Boundary-Information-Based Oversampling Approach to Improve Learning Performance for Imbalanced Datasets*, Entropy, 24(3), 322, 2022, https://doi.org/10.3390/e24030322 .

Maheshwari, D., *Imbalanced Classification: Challenges and Approaches to Handle*, In: Reddy, V. S., Prasad, V. K., Wang, J., Rao Dasari, N. M. (Eds.), Intelligent Systems and Sustainable Computing, 533–543, Springer Nature, https://doi.org/10.1007/978-981-99-4717-1_50 .

Maillo, J., Triguero, I., Herrera, F., *Redundancy and Complexity Metrics for Big Data Classification: Towards Smart Data*, IEEE Access, 8, 87918–87928, 2020, https://doi.org/10.1109/ACCESS.2020.2991800 .

Majzoub, H., Elgedawy, I., *AB-SMOTE: An Affinitive Borderline SMOTE Approach for Imbalanced Data Binary Classification*, Internat. J. Machine Learning & Computing, 10, 31–37, 2020, https://doi.org/10.18178/ijmlc.2020.10.1.894 .

Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., Galstyan, A., *A Survey on Bias and Fairness in Machine Learning*, ACM Computing Surveys, 54(6), 115, 1–35, 2021, https://doi.org/10.1145/3457607 .

Husain, G., Nasef, D., Jose, R., Mayer, J., Bekbolatova, M., Devine, T., Toma, M., *SMOTE vs. SMOTEENN: A Study on the Performance of Resampling Algorithms for Addressing Class Imbalance in Regression Models*, Algorithms, 18(1), 37, 2025, https://doi.org/10.3390/a18010037 .

Swana, E. F., Doorsamy, W., Bokoro, P., *Tomek Link and SMOTE Approaches for Machine Fault Classification with an Imbalanced Dataset*, Sensors, 22(9), 3246, 2022, MDPI, https://doi.org/10.3390/s22093246 .

Montesinos Lopez, O. A., Montesinos Lopez, A., Crossa, J., *Multivariate Statistical Machine Learning Methods for Genomic Prediction*, 2022, Springer International Publishing, https://doi.org/10.1007/978-3-030-89010-0_4 .

Naglik, I., Lango, M., *GMMSampling: A new model-based, data difficulty-driven resampling method for multi-class imbalanced data*, Machine Learning, 113, 5183–5202, 2023, Springer, https://doi.org/10.1007/s10994-023-06416-8 .

Naidu, G., Zuva, T., Sibanda, E. M., *A Review of Evaluation Metrics in Machine Learning Algorithms*, In: Silhavy, R., Silhavy, P. (Eds.), Artificial Intelligence Application in Networks and Systems, 15–25, Springer International Publishing, https://doi.org/10.1007/978-3-031-35314-7_2 .

Naim, F. A., Hannan, U. H., Humayun Kabir, M., *Effective Rate of Minority Class Over-Sampling for Maximizing the Imbalanced Dataset Model Performance*, In: Gupta, D., Polkowski, Z., Khanna, A., Bhattacharyya, S., Castillo, O. (Eds.), Proc. Data Analytics and Management, 9–20, Springer, https://doi.org/10.1007/978-981-16-6285-0_2 .

Najman, K., Zielinski, K., *Outlier Detection with the Use of Isolation Forests*, In: Jajuga, K., Najman, K., Walesiak, M. (Eds.), Data Analysis and Classification, 65–79. Springer International Publishing, https://doi.org/10.1007/978-3-030-75190-6_5 .

Nugroho, H., Utama, N. P., Surendro, K., *Normalization and outlier removal in class center-based firefly algorithm for missing value imputation*, J. Big Data, 8(1), 129, Springer, https://doi.org/10.1186/s40537-021-00518-7 .

Ou, G., *Analysis of the Research Status of SMOTE Algorithm in the Last Three Years-Statistical Analysis of Core Literature Based on CNKI 2022-2024*, Academic J. Mathematical Sciences, 5(3), Francis Press, https://doi.org/10.25236/AJMS.2024.050301 .

Rekha, G., Tyagi, A. K., Krishna Reddy, V., *A Novel Approach to Solve Class Imbalance Problem Using Noise Filter Method*, In: Abraham, A., Cherukuri, A. K., Melin, P., Gandhi, N. (Eds.), Intelligent Systems Design and Applications, 486–496, Springer International Publishing, https://doi.org/10.1007/978-3-030-16657-1_45 .

Riston, T., Suherman, S. N., Yonnatan, Y., Indrayatna, F., Pravitasari, A. A., Sari, E. N., Herawan, T., *Oversampling Methods for Handling Imbalance Data in Binary Classification*, In: Gervasi, O., Murgante, B., Rocha, A. M. A. C., Garau, C., Scorza, F., Karaca, Y., Torre, C. M. (Eds.), Computational Science and Its Applications - ICCSA 2023 Workshops, 3–23, Springer Nature Switzerland, https://doi.org/10.1007/978-3-031-37108-0_1.

Shi, W., Wang, C., Feng, F., Zhang, Y., Wang, W., Wu, J., He, X., *Lower-Left Partial AUC: An Effective and Efficient Optimization Metric for Recommendation*, Proc. ACM Web Conference 2024, 3253–3264, ACM, https://doi.org/10.1145/3589334.3645371 .

Shwartz-Ziv, R., Goldblum, M., Li, Y., Bruss, C. B., Wilson, A. G., *Simplifying Neural Network Training Under Class Imbalance*, Proc. Advances in Neural Information Processing Systems 36 (NeuIPS 2023), 35218–35245, https://proceedings.neurips.cc/paper_files/paper/2023/hash/6ea69f8116b7c01e3c3e43b62e6868fc-Abstract-Conference.html .

Silva Filho, T., Song, H., Perello-Nieto, M., Santos-Rodriguez, R., Kull, M., Flach, P., *Classifier calibration: A survey on how to assess and improve predicted class probabilities*, Machine Learning, 112(9), 3211–3260, Springer, https://doi.org/10.1007/s10994-023-06336-7 .

Sun, Y., Que, H., Cai, Q., Zhao, J., Li, J., Kong, Z., Wang, S., *Borderline SMOTE Algorithm and Feature Selection-Based Network Anomalies Detection Strategy*, Energies, 15(13), 4751, 2022, MDPI, https://doi.org/10.3390/en15134751 .

Swana, E. F., Doorsamy, W., Bokoro, P., *Tomek Link and SMOTE Approaches for Machine Fault Classification with an Imbalanced Dataset*, Sensors, 22(9), 3246, 2022, MDPI, https://doi.org/10.3390/s22093246 .

Talukder, Md. A., Islam, Md. M., Uddin, M. A., Hasan, K. F., Sharmin, S., Alyami, S. A., Moni, M. A., *Machine learning-based network intrusion detection for big and imbalanced data using oversampling, stacking feature embedding and feature extraction*, J. Big Data, 11, 33, 2024, Springer, https://doi.org/10.1186/s40537-024-00886-w .

Vashisht, R., & Rizvi, S. A. M., *Addressing Noise and Class Imbalance Problems in Heterogeneous Cross-Project Defect Prediction: An Empirical Study*, Internat. J. e-Collaboration, 19(1), 1–27, 2023, IGI Global,

https://doi.org/10.4018/IJeC.315777 .

Vinutha, H. P., Poornima, B., Sagar, B. M., *Detection of Outliers Using Interquartile Range Technique from Intrusion Dataset*, In: Satapathy, S. C., Tavares, J. M. R. S., Bhateja, V., Mohanty, J. R. (Eds.), Information and Decision Sciences, 511–518, 2018, Springer, https://doi.org/10.1007/978-981-10-7563-6_53 .

Wu, T., Ding, X., Zhang, H., Gao, J., Tang, M., Du, L., Qin, B., Liu, T., *DiscrimLoss: A Universal Loss for Hard Samples and Incorrect Samples Discrimination*, IEEE Trans. Multimedia, 26, 1957–1968. 2024, IEEE, https://doi.org/10.1109/TMM.2023.3290477 .

Xu, Y., Zhao, Y., Ke, W., He, Y.-L., Zhu, Q.-X., Zhang, Y., Cheng, X., *A multi-fault diagnosis method based on improved SMOTE for class-imbalanced data*, The Canadian J. Chemical Engineering, 101(4), 1986–2001, Wiley, https://doi.org/10.1002/cjce.24610 .

Yang, T., Ying, Y., *AUC Maximization in the Era of Big Data and AI: A Survey*, ACM Computing Surveys, 55(8), 172, 1–37, ACM, https://doi.org/10.1145/3554729 .