Neurocomputing 498 (2022) 75–88

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

PF-SMOTE: A novel parameter-free SMOTE for imbalanced datasets

Qiong Chen [a], Zhong-Liang Zhang [a,b,c,*], Wen-Po Huang [a], Jian Wu [a,c], Xing-Gang Luo [a]

[a] School of Management, Hangzhou Dianzi University, 310018 Hangzhou, China
[b] Antai College of Economics and Management, Shanghai Jiao Tong University, 200030 Shanghai, China
[c] Research Center for Youth Public Opinion of Zhejiang, Hangzhou Dianzi University, 310018 Hangzhou, China

ARTICLE INFO

Article history:
Received 31 December 2021
Revised 3 April 2022
Accepted 2 May 2022
Available online 11 May 2022

Keywords:
Imbalanced datasets
Data preprocessing
SMOTE
Gaussian process
Oversampling

ABSTRACT

Class imbalance learning is one of the most important topics in the field of machine learning and data mining, and the Synthetic Minority Oversampling Techniques (SMOTE) is the common method to handle this issue. The main shortcomings of the classic SMOTE and its variants is the interpolation of potential noise and unrepresentative examples. This paper is devoted to proposing a novel parameter-free SMOTE mechanism to produce sufficient representative synthetic examples while avoiding interpolating noisy examples. Specifically, two types of minority class examples are defined, namely boundary and safe minority examples. The synthetic examples generation procedure fully reflects the characteristics of the minority class examples with filling the region dominated by the minority class and expanding the margin of the minority class. To verify the effectiveness and robustness of the proposed method, a thorough experimental study on forty datasets selected from real-world applications is carried out. The experimental results indicate that our proposed method is competitive to the classic SMOTE and its state-of-the-art variants.

1. Introduction

Classification of imbalanced datasets is well-known as a challenging task in supervised learning [1,2]. Traditionally, the imbalanced datasets suffer from the problem of the skewed class distribution, i.e. some classes significantly outnumber the others. Many real-world applications face this issue, such as bioinformatics [3], fault diagnosis [4], fraud detection [5].

This study focuses on the binary imbalanced datasets, in which the relationship between classes is well-defined: one is considered as the majority class and the other as the minority one. Many previous works [6,7] have shown that the traditional classification algorithm is more inclined to majority class examples while performing poorly in identifying minority class examples.

The goal of these methods proposed to deal with binary class imbalance problems is to reduce the bias towards the majority class, and meanwhile, balance the performance on both classes. Basically, methods for addressing binary class imbalance problems could be roughly categorized into four groups: data-level solutions, algorithm-level strategies, cost-sensitive learning methods, and ensemble learning approaches [8].

Due to the simplicity and effectiveness, data-level solutions are the most active research area for addressing imbalance learning problems [9]. Roughly, the data-level solutions for resampling imbalanced datasets can be further divided into two groups: undersampling and oversampling [10]. The former removes the majority class examples to balance the class distribution, while the later duplicates the minority class examples to make the training set balanced. There are dozens of resampling techniques have been proposed in the previous works [11]. Among them, the Synthetic Minority Oversampling Technique (SMOTE) [12] proposed by Chawla et al. is the most famous over-sampling technique. Unlike random oversampling duplicating the minority examples, SMOTE produces synthetic minority examples by using k nearest neighbors of the minority example under consideration.

Over the past decade, a larger number of variants for SMOTE have been proposed [13]. According to the review presented in [14], these variants are namely (1) initial selection of examples to be oversampled, (2) integration with undersampling as step in the technique, (3) type of interpolation, (4) operation with dimensionality changes, (5) adaptive generation of synthetic examples, (6) possibility of relabeling, and (7) filtering of noisy generated examples. After deeply analyzing the characteristics of the classic SMOTE and its variants, we can find the following shortcomings:

* Corresponding author at: School of Management, Hangzhou Dianzi University, 310018 Hangzhou, China.
E-mail addresses: chenqiong2939@163.com (Q. Chen), zlzhang@hdu.edu.cn (Z.-L. Zhang), whuang@hdu.edu.cn (W.-P. Huang), hzieewujian@163.com (J. Wu), xgluo@mail.neu.edu.cn (X.-G. Luo).

https://doi.org/10.1016/j.neucom.2022.05.017
0925-2312/

- It is hard to decide the optimal value of k which is the key parameter in most variants. A large number of k might introduce noisy synthetic examples, while a small value of k tends to produce nearly duplicate examples.
- Most variants of SMOTE dedicate to interpolate safe synthetic minority examples, which might miss the opportunities to generate a wide variety of synthetic minority examples.

In order to overcome these issues of classic SMOTE and its variants mentioned above, we in this paper attempt to develop a more powerful variant for SMOTE. Concretely, we in this paper categorize the minority class into two groups: boundary and safe examples, which should be treated with totally different strategies. Furthermore, we propose a novel parameter-free variant of SMOTE, named as parameter-free SMOTE (PF-SMOTE), which fully reflects the characteristics of the minority class examples. Specifically, when the safe example is selected, the synthetic examples are interpolated into the region that dominated by the minority class. Meanwhile, to expand the margin of the minority class, a Gaussian-based method for generating synthetic examples is developed for the boundary minority class examples.

In order to verify the validity of our proposal, a thorough experimental study is carried out in this study. In particular, a set of forty datasets from the KEEL-dataset repository is selected for our experiments. The Area Under Curve (AUC) [15] is employed as the classification performance measure and the proper statistical tests suggested in [16] are used to study the significance of the results.

To sum up, the main contributions of this study with respect to these previous works are as follows:

(1) After deeply analyzing the main shortcomings of the classic SMOTE and its variants, we in this paper point out that the synthetic examples should be interpolated according to characteristics of the minority class examples.
(2) We propose a novel PF-SMOTE method, which attempts to fill the region dominated by the minority class and expanding the margin of the minority class.
(3) In order to demonstrate the robustness and effectiveness of our proposed method, a thorough experimental study on forty imbalanced datasets selected from the real-world applications is carried out.

The remainder of this paper is organized as follows. These works related to our study are provided in Section 2, including the review of imbalanced datasets and SMOTE-based methods. Next, Section 3 describes our proposed method in detail. Then, the experimental settings are presented in Section 4. Finally, the experimental results are analyzed in Section 5 and the conclusions are given in Section 6.

2. Related works

In this section, we attempt to recall the solutions for imbalanced datasets, as well as the most popular data preprocessing method, i.e. SMOTE, and its variants.

2.1. Solutions for imbalanced datasets

Suppose a given dataset TS with n examples (i.e. $|TS| = n$), which is formally defined as $TS = TS_1 \cup TS_2 \cup \ldots \cup TS_m$, where $TS_i = (x_j, \varphi(x_j)), j = 1, \ldots, n_i$, and $x_j \in X$ is the input feature vector of each example, $\varphi(x_j) \in Y = \{\omega_1, \ldots, \omega_m\}$ is the class label associated with the example x_j, m is the number of classes and n_i is the number of examples with the class ω_i. When several classes (minority classes) are under-represented with respect to the other classes (majority classes), i.e., if $|TS_i| \ll |TS_{i'}|, i \neq i'$ and $i, i' = 1, \ldots, m$, the classification task on TS is called class imbalance learning problem.

Learning from class imbalanced datasets is one of the most challenging tasks in machine learning and data mining, which always reduces the performance of classic classification algorithms [17], since standard classification algorithms are designed with the premise of a balanced training dataset [2], which makes it much more difficult for classic classification approaches to recognize the minority classes. However, on the other hand, the minority classes are usually the most important to identify, since the rare classes usually represent the interesting concepts [18]. In addition, it is expensive or hard to collect these minority class examples [19].

In this paper, we focus on the binary class imbalance problem (i.e. $m = 2$), in which the relationship between classes is well-defined: one is considered as the majority class and the other as the minority one. The goal of the method proposed to deal with binary class imbalance problems is to reduce the bias towards the majority class, and meanwhile balance the performance on both classes. Basically, methods for addressing binary class imbalance problems could be roughly categorized into four groups: data-level solutions, algorithm-level strategies, cost-sensitive learning methods and ensemble-based approaches [20].

- Data-level solutions. The origin of the class imbalance problem is the skewed distribution in the datasets. Thus, it is natural to consider of rebalancing by sampling the data space to reduce the impact of class imbalance, which is known as an external approach [21]. One of the advantages of such solution is independent from the classifier used, so they are also considered as pre-sampling method [22]. Under-sampling and over-sampling are the two basic ideas of resampling approaches. Random Under-Sampling (RUS) randomly removes the majority class examples to balance the class distribution while Random Over-Sampling (ROS) randomly duplicates the minority class examples to make the training set balanced. The most representative over-sampling technique is SMOTE [12], which has attracted a great attentions in the past decades.
- Algorithm-level strategies. These solutions attempt to adopt appropriate decision threshold to reinforce the learning towards the minority class examples. The proposed algorithms that take the class imbalance into consideration belong to such techniques. They are defined as internal approaches in some papers [23,24], since the effect depends on the problems and the classification algorithm [25]. For example, a variant of SVM for class imbalance problems called Near-Bayesian Support Vector Machines (NBSVMs) is presented in [26], in which the proposed method combines decision boundary shift with asymmetric regularization costs.
- Cost-sensitive learning methods. These methods consider higher costs for misclassifying the minority classes with respect to the majority classes, that is, misclassification of minority class is much more expensive [27]. The objective of the learning algorithm turns to minimize the cost errors instead of maximization of accuracy rate [28]. Cost-sensitive learning methods are less popular than resampling methods because misclassification costs cannot be identified from the data and there are difficulties in setting costs [29].
- Ensemble-based approaches. These methods integrate the powerful ensemble learning technique with one of the three previously mentioned strategies in order to create a balanced training dataset for the base classification algorithm used in ensemble learning and at the same time introduce diversity into

the pool of base classifiers [30]. These techniques proposed for handling class imbalance problems follow the architecture of Bagging [31,32] or Boosting [33].

2.2. A brief review for SMOTE methods

Unlike ROS duplicates the minority examples, SMOTE produces synthetic minority examples by using k nearest neighbors of the minority example under consideration. Concretely, the new synthetic examples are created by randomly selecting one of the k nearest neighbors and multiplying the corresponding feature vector by a random number as follows.

$$S_i = x_i + \Delta \times (x_i' - x_i) \tag{1}$$

where x_i is the minority example under consideration, x_i' is one of the k nearest neighbors of example x_i, Δ is a random value between 0 and 1. This procedure of SMOTE is illustrated in Fig. 1, where x_i is the selected minority example, x_{i1} to x_{i5} is the nearest neighbors, and S_{i1} to S_{i5} is the synthetic examples generated by randomized insertion.

Due to its simplicity and robustness, SMOTE is successful in dealing with imbalanced datasets. However, classic SMOTE has several shortcomings in some specific scenarios. Thus, a large number of variants for SMOTE have been proposed in the past decades, which can be roughly divided into seven groups [14] as follows:

- Initial selection of examples to be oversampled. This family of methods are to determine the best candidate examples to be oversampled before the synthetic examples are interpolated. The main purpose of these methods is to avoid the generation of noise and overlapping examples by screening suitable candidates to be synthesized. The most representative of such examples is Borderline-SMOTE [34], in which the core is to select boundary examples for over-sampling. In addition to selecting the boundary examples for oversampling, this algorithm also decides whether to select the examples for oversampling according to the number of minority examples in the nearby neighborhood. More recently, a novel variant for SMOTE is proposed, in which the selection of natural neighbors is used to find the initial examples [35].
- Integration with undersampling. When dealing with imbalanced data, not only the oversampling algorithm can be used to synthesize the data, but also the undersampling algorithm can be used to remove the data. Therefore, some algorithms integrate the oversampled algorithm and the undersampled algorithm. This integration falls into two categories: undersampling before oversampling and as an internal operation with oversampling.

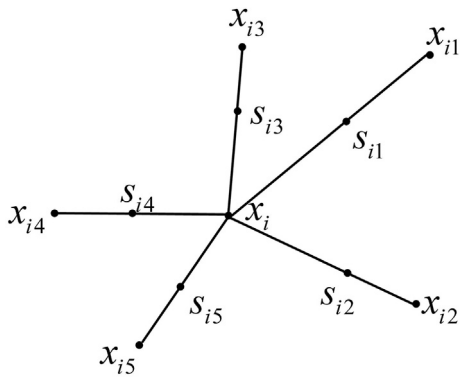

Fig. 1. An illustration of SMOTE on how to generate synthetic examples.

- Type of interpolation. This kind of method is generally by modifying the interposition mechanism of synthetic examples to improve the recognition accuracy of the minority examples. Among them, there are many original interpolation methods, such as multiple or limited interpolation, and there are also clustering based interpolation. There are even ways to synthesize examples without interpolation, such as by dithering [36].
- Operation with dimensionality changes. This kind of method usually attempts to change the dimensions of the data first, such as principal component analysis (PCA) [37], feature selection [38], using the kernel function [39,40], etc., and then synthesizes the example in the new dimensional space.
- Adaptive generation of synthetic examples. Adaptive Synthetic Sampling (ADASYN) [41], which is first proposed in this kind of method, assumes that the weight is determined according to the degree of learning difficulty of examples, and then determines the number of synthesized examples of each minority example. Later, inspired by ADASYN, some methods use a similar mechanism to determine the number of compositions for each minority examples [42].
- Relabeling. These methods usually provide the choice to change the label of some majority class examples to that of the minority class examples [43] or to replace the interpolation mechanism [44].
- Filtering of noisy generated examples. In this method, a noise filtering operation is added after synthesizing examples to remove the synthesized examples belonging to the noise. At present, many filters are proposed, such as greedy filtering strategy [45], ensembles-based filtering [46], filter based on rough set [47–49], etc.

3. Our proposed method

In this section, we aim to present the details of the proposed PF-SMOTE in this paper. In SubSection 3.1 we introduce several preliminary concepts and assumptions. In SubSection 3.2 the procedure of the proposed synthetic examples generation is described. In SubSection 3.3 the computational complexity of our PF-SMOTE is provided.

3.1. Preliminary concepts and assumptions

As stated above, the previous works greatly facilitate the research in the development of powerful classifiers for the imbalanced datasets. Although a large number of variants for SMOTE have been proposed, these methods may still encounter various problems [50] in some cases. The shortcomings of these variants can be summarized as follows:

Generation of noisy examples. To be our best knowledge, most variants for SMOTE involve the parameter k, which is the number of nearest neighbors that usually predetermined. In fact, an inappropriate setting for k might lead to introduce synthetic noisy examples. As shown in Fig. 2, the selected minority class example x_i is located in the small disjunct, and x_j is one of its k ($k = 5$) nearest neighbors. Thus, the synthetic example x_{syn} is located in the area of the majority class, which is called as noisy examples.

Aggravation of overlapping phenomenon. Usually, in the imbalance learning tasks, the overlapping problem which leads to an ambiguous decision boundary is the key factor. If the selected minority class example is located within the decision boundary, the interpolation of synthetic examples would be an inevitable consequence of aggravating the overlapping problem. As a brief illustration is shown in Fig. 3, in which the selected minority example x_i is located within the decision boundary. It can be easily find that the interpolated synthetic examples artificially generate an overlapping region.

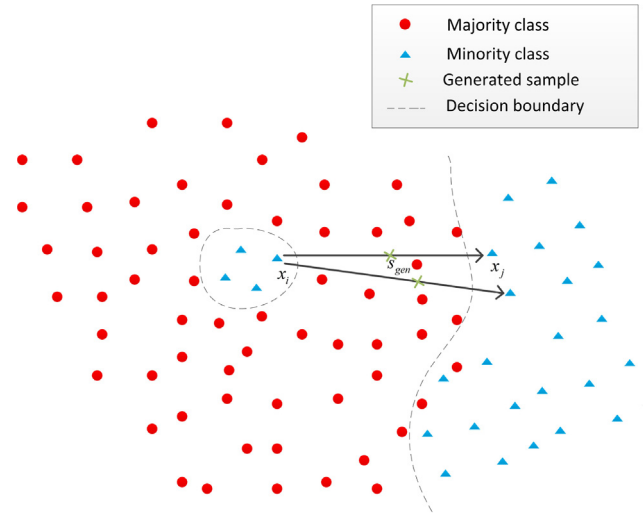

Fig. 2. An illustration of generating noisy synthetic examples due to the selection of k nearest neighbors.

Interpolation of nearly useless synthetic examples. Generally, the aim of imbalance learning is to increase the minority class prediction performance without heavily compromising the accuracy of the majority class. Thus, the purpose of classic SMOTE and its variants is to generate synthetic examples that can deeply affect the decision boundary. However, as shown in Fig. 4, the selected example x_i is located in a cluster of minority class, and then these synthesized examples derived from x_i are of little use because they do not provide additional information to the classification task.

Regarding the shortcomings of SMOTE and its variants explained above, a new strategy for generating synthetic minority examples requires to be developed. Thus, we in this paper aim to develop a variant for SMOTE with the following objectives:

- The new strategy is able to abandon the hyper-parameter of k nearest neighbors, which is used in most of variants for SMOTE.
- The new strategy can adaptively determine the generation strategy for each minority example according to its property.
- The new strategy can expand the space of minority class to the region which is dominated by no class.

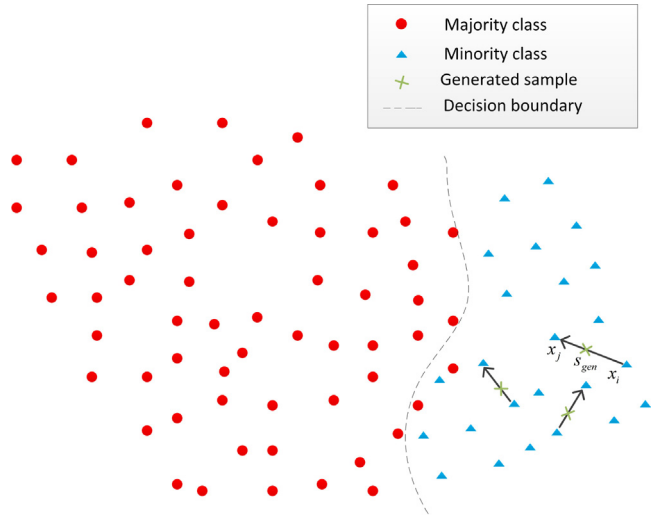

Fig. 4. An illustration of generating duplicate synthetic examples.

3.2. PF-SMOTE

Afterhere, we aim to describe our proposed PF-SMOTE in detail. Given a dataset $TS \in R^d$, where TS consists of two classes of examples: minority examples TS^+ and majority examples TS^-, for $x_p \in TS, x_q \in TS, d(x_p, x_q)$ represents the distance of x_p and x_q, and d_{min} represents the smallest distance among x_p and its nearest neighbors in TS. The nearest neighbor of x_p in TS are denoted as x_{nn}, where $x_{nn} = \{x_q \in TS | d(x_p, x_q) = d_{min}\}$. Firstly, we categorize the minority examples into two groups: boundary minority examples and safe minority examples, which are defined as follows:

Definition 1 (*Boundary minority example*). If an example $x \in TS^+$ and its $x_{nn} \in TS^-$, then x is a boundary minority example. An example of illustration for the boundary minority examples can be found in Fig. 5(a).

Definition 2 (*Safe minority example*). If an example $x \in TS^+$ and its $x_{nn} \in TS^+$, then x is a safe minority example. An example of illustration of the safe minority examples can be found in Fig. 5(b).

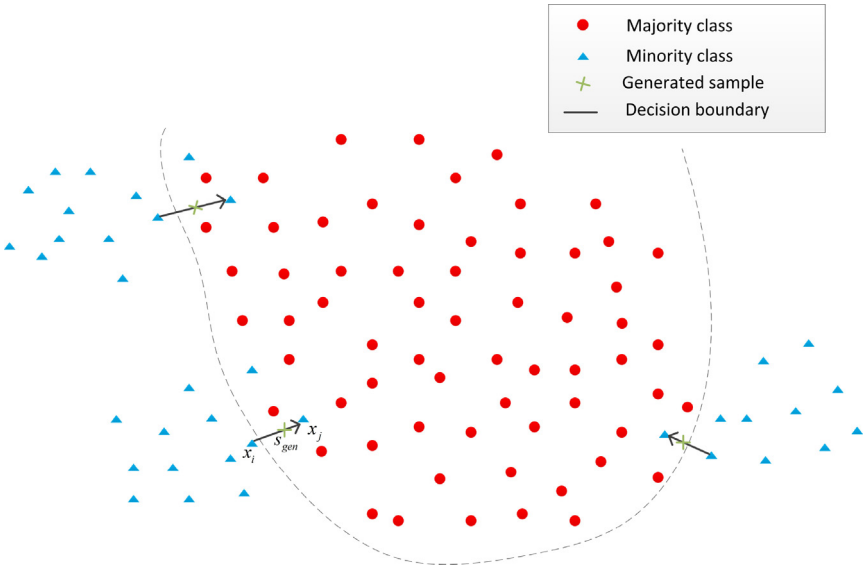

Fig. 3. An illustration of aggravating the overlapping problem.

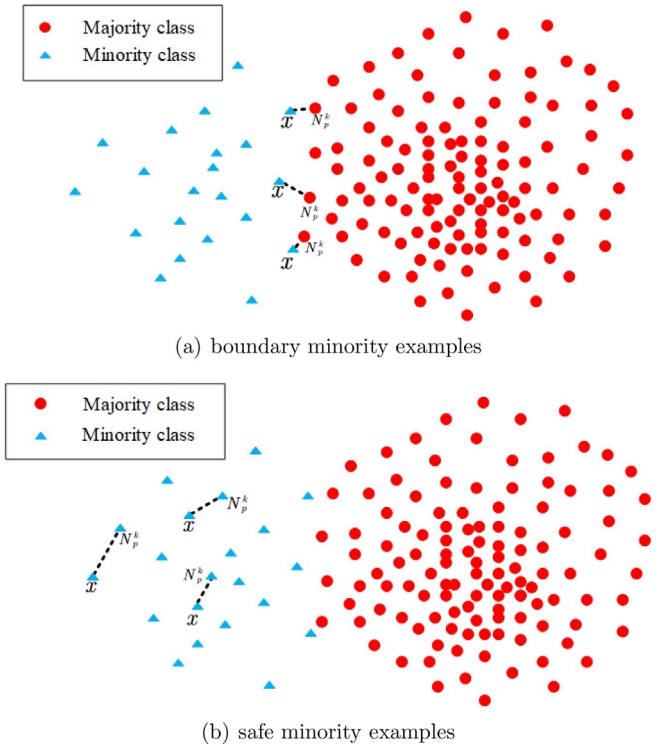

Fig. 5. An illustration for the categorization of the minority examples.

Obviously, different types of minority examples have completely different properties, which leads to require different strategies for generating synthetic examples. That is, the generation of synthetic examples should reflect the characteristics of the minority class examples. If the safe minority examples are selected to generate synthetic examples, which might be more likely to be safe ones. In this case, we should interpolate the synthetic examples with the aim to fully cover the region that dominated by the minority class. On the contrary, when the boundary minority examples are selected, the classic SMOTE is more likely to produce noisy examples or overlapping issue. In this scenario, the aim of generating synthetic examples is to expand the margin of the minority class. Thus, different types of minority examples have different requirements for the synthesis of examples.

With respect to safe minority examples, on one hand, the local region needs to be as large as possible in order to avoid generating duplicate examples. However, on the other hand, an overlarge local region is easy to generate noisy examples. Therefore, it must be careful to determine the size of local region. In this paper, we define the local region for each safe minority example using the closest majority example. An illustration of determining the local region for safe minority examples is shown in Fig. 6. We first find the nearest majority example for the selected minority example x_i, and then calculate the their distance which is used as radius. Finally, the minority examples in the radius are defined as the local region for generating synthetic examples.

Regarding boundary minority examples, the objective of generating the synthetic examples is to expand the boundary toward the majority class, meanwhile to avoid generate noisy examples. That is, the location of synthetic examples are interpolated biased toward the majority class. To implement this, a gaussian process is proposed to generate the synthetic examples. Concretely, once the selected minority example is a kind of boundary one, its nearest majority class example is found. Then, the location of a synthetic candidate is interpolated along the line between the selected minority example and majority example. Finally, in order to introduce the diversity of the synthetic examples, a gauss process is employed.

Thus, no parameters are required in our novel PF-SMOTE. The pseudocode of our PF-SMOTE is provided in Algorithm 1. Given a dataset TS, which consists of minority class TS^+ and majority class TS^-, the detailed procedure of generating synthetic examples in PF-SMOTE is as follows:

Step 1: For each minority example $x_i, x_i \in TS^+$, we obtain the nearest neighbor x_{nn}. If $x_{nn} \in TS^+$, we put x_i into a new set TS^+_{safe}, the set of safe examples. If $x_{nn} \in TS^-$, we put x_i into a new set $TS^+_{boundary}$, the set of boundary examples.

Step 2: Different over-sampling methods are used for TS^+_{safe} and $TS^+_{boundary}$. The process is as follows: a. For safe example $x_i \in TS^+_{safe}$, the synthetic example is generated according to Eq. 2:

$$s^{safe}_{gen} = x_i + gap \times (x_{safe} - x_i) \tag{2}$$

where $gap \sim U(0,1)$, and x_{safe} is an example randomly selected from the local safe region of x_i according to the following Eq. 3:

$$LSR(x_i, TS^+_{safe}, Rad) = \{x_{safe} \in TS^+_{safe}, d(x_i, x_{safe}) < Rad\} \tag{3}$$

where $d(\cdot)$ is the Euclidean distance between two examples. Rad is the distance between x_i and its nearest majority class example. b. For each boundary example $x_i \in TS^+_{boundary}$, the synthetic example is generated according to Eq. 4:

$$s^{boundary}_{gen} = N((x_i + gap \times (x^-_{nn} - x_i)), \sigma^2) \tag{4}$$

where x^-_{nn} is the closest majority example of $x_i, gap \sim U(0,1)$, and σ is the standard deviation estimated using the following manner:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{|TS^+|}(x_i - \bar{x})^2}{|TS^+| - 1}} \tag{5}$$

Algorithm 1 Our proposed PF-SMOTE

Input minority class examples TS^+, majority class examples TS^-
Output synthetic examples S_{gen}
$TS^+_{safe}, TS^+_{boundary} \leftarrow \varnothing$
for all $x_i \in TS^+$ **do**
  Find its nearest neighbor x_{nn}
  if $x_{nn} \in TS^+$ **then**
    $TS^+_{safe} \leftarrow TS^+_{safe} \cup x_i$
  else if $x_{nn} \in TS^-$ **then**
    $TS^+_{boundary} \leftarrow TS^+_{boundary} \cup x_i$
  end if
end for
$S^{safe}_{gen} \leftarrow \varnothing$
for all $x_i \in TS^+_{safe}$ **do**
  Find the local safe region LSR for x_i according to Eq. 3
  Randomly select an example $x_{safe} \in LSR$
  generate synthetic samples s^{safe}_{gen} according to Eq. 2
  $S^{safe}_{gen} \leftarrow S^{safe}_{gen} \cup s^{safe}_{gen}$
end for
$S^{boundary}_{gen} \leftarrow \varnothing$
for all $x_i \in TS^+_{boundary}$ **do**
  Find its nearest majority class example $x^-_{nn} \in TS^-$

(*continued on next page*)

20: generate synthetic samples $s_{gen}^{boundary}$ according to Eq. 4
21: $S_{gen}^{boundary} \leftarrow S_{gen}^{boundary} \cup s_{gen}^{boundary}$
22: **end for**
23: **return** $S_{gen} \leftarrow S_{gen}^{safe} \cup S_{gen}^{boundary}$

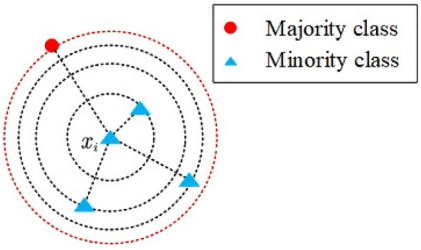

Fig. 6. An illustration of determining the local region for safe minority examples.

3.3. Computational complexity analysis for PF-SMOTE

Let us define the imbalanced training examples by TS, which contains $|TS|$ examples with d dimensions, and the minority example number $TS^+ < \frac{1}{2}|TS|$.

In the stage of dividing minority examples into two groups, i.e. boundary and safe minority examples, it requires $|TS|-1$ distance calculations, each with a complexity of $\mathcal{O}(d)$, then sorting the distance with a complexity equal to $\mathcal{O}((|TS|-1)log(|TS|-1))$. A total number of minority examples is equal to $|TS^+|$. Combining the above, the computational complexity for the first step is equal to: $\mathcal{O}(|TS^+| \cdot (((|TS|-1) \cdot d) + (|TS|-1)log(|TS|-1)))$.

In the stage of generating synthetic examples for each $x_i \in TS_{boundary}^+$, it requires to find its nearest majority class example with a complexity of $\mathcal{O}(|TS^-| \cdot d + |TS^-|log|TS^-|)$. Thus, the total computational complexity of the second step is equal to: $\mathcal{O}(|TS_{boundary}^+| \cdot (|TS^-| \cdot d + |TS^-|log|TS^-|))$.

Finally, for each $x_i \in TS_{safe}^+$, finding its local safe region leads to a complexity of $\mathcal{O}(((|TS|-1) \cdot d + (|TS|-1)log(|TS|-1))$. As a result, the computational complexity for this stage is equal to: $\mathcal{O}(|TS_{safe}^+| \cdot (((|TS|-1) \cdot d + (|TS|-1)log(|TS|-1)))$.

As we can see, $|TS| = |TS^-| + |TS^+|$ and $|TS^+| = |TS_{boundary}^+| + |TS_{safe}^+|$, the computational complexity of our PF-SMOTE can be simplified as $\mathcal{O}(|TS^+| \cdot (|TS| \cdot d + |TS|log|TS|))$.

4. Experimental Framework

4.1. Datasets

In this paper, we both use artificial datasets and real-world datasets in the experiments. Firstly, we use the Python library scikit-learn to produce five artificial datasets for the visible analysis. Then, forty imbalanced datasets from the KEEL-dataset repository, which is available on the corresponding webpage [1], are selected to evaluate the performance of tested methods. The detailed information of the datasets is provided in Table 1, in which it includes the number of attributes (#Attr.), the number of examples (#NE), the number of each class (%Class (min,maj)), and imbalance ratio (#IR).

The experimental results are obtained via a ten times 10-fold cross-validation method. That is, the dataset is stratified sampling into ten folds, and each fold contains 10% of examples. For each experiment, one of the folds is selected as the test set, and the remaining nine folds are used as the training dataset. Then, the average is the result for one time experiment. The procedure is repeated for ten times, and the average of each experimental result is the final performance.

[1] https://sci2s.ugr.es/keel/datasets.php

4.2. Performance measure

To evaluate the performance of classification algorithms, it is necessary to find the appropriate measure metrics. Although classification accuracy is widely used in assessing the performance of classification algorithms, it is not a proper choice in the scenario of imbalanced datasets due to the skew distribution. Thus, we in this paper employ the well-known performance measure, i.e. AUC [15], which is defined as the area under the Receiver Operating Characteristic (ROC) curve.

In order to clearer explain the calculation of AUC, a confusion matrix for a binary classification problem is shown in Table 2.

Based on the confusion matrix, the ROC curve can be drawn based on a series of different thresholds with the true positive rate(TPR) on the ordinate and the false positive rate(FPR) on the abscissa. The horizontal axis of the ROC curve is the False Positive Rate (FPR), and the vertical axis is True Positive Rate (TPR), where

$$TPR = \frac{TP}{(TP+FN)}, \tag{6}$$

$$FPR = \frac{FP}{(FP+TN)}. \tag{7}$$

By setting different parameter thresholds in the classifier, multiple different coordinate points can be obtained, and these coordinate points are connected to form an ROC curve. Because the ROC curve cannot quantitatively evaluate the performance of the classifier, we often use the area under the ROC curve AUC as the evaluation standard. And the larger the AUC value, the better the classification effect. AUC can be obtained by summing the area under the ROC curve. If the ROC curve is formed by sequentially connecting points with coordinates $(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m), (x_1 = 0, x_m = 1)$ AUC can be estimated as follows:

$$AUC = \frac{1}{2}\sum_{i=1}^{m-1}(x_{i+1} - x_i) \cdot (y_i + y_{i+1}). \tag{8}$$

In order to assess whether there are significant differences between experimental algorithms, it is necessary to use statistical tests. Hence, we follow the recommendations presented in [16] to consider the use of non-parametric hypothesis testing techniques for comparing the tested methods in this study. As regards the pairwise comparison between our method and other methods, the Wilcoxon signed-rank test [51] is employed.

4.3. Classification algorithms and parameter settings

After resampling the datasets, a classification method should be employed to construct the classifier using the balanced datasets. In order to obtain the influence of classification algorithm used in PF-SMOTE, three well-known classification algorithms are selected in our experiments, which are described in detail as follows:

- Decision Trees (DT) [52,53]: DT is a simple but powerful method of induction from labeled samples, which is constructed in a top-down manner. It starts a root node containing all the training data and then the internal nodes are constantly refined using a certain splitting criterion until the stopping condition is met. Thus, the classification rules can be induced from the final decision tree.

Table 1
Summary descriptions of the datasets

id	Datasets	#Attr.	#NE	%Class (maj,min)	#IR	id	Datasets	#Attr.	#NE	%Class (maj,min)	#IR
D1	new-thyroid1	5	215	(180,35)	5.14	D21	glass4	9	214	(201,13)	15.47
D2	new-thyroid2	5	215	(180,35)	5.14	D22	ecoli4	7	336	(316,20)	15.80
D3	ecoli2	7	336	(284,52)	5.46	D23	abalone9-18	8	731	(689,42)	16.40
D4	segment0	19	2308	(1979,329)	6.02	D24	dermatology-6	34	358	(338,20)	16.90
D5	yeast3	8	1484	(1321,163)	8.10	D25	zoo3	16	101	(96,5)	19.20
D6	ecoli3	7	336	(301,35)	8.60	D26	glass-0-1-6_vs_5	9	184	(175,9)	19.44
D7	page-blocks0	10	5472	(4913,559)	8.79	D27	shuttle-c2_vs_c4	9	129	(123,6)	20.50
D8	yeast-2_vs_4	8	514	(463,51)	9.08	D28	shuttle-6_vs_2-3	9	230	(220,10)	22.00
D9	yeast-0-3-5-9_vs_7-8	8	506	(456,50)	9.12	D29	yeast-2_vs_8	8	482	(462,20)	23.10
D10	yeast-0-2-5-7-9_vs_3-6-8	8	1004	(905,99)	9.14	D30	winequality-red-4	11	1599	(1546,53)	29.17
D11	glass-0-4_vs_5	9	92	(83,9)	9.22	D31	poker-9_vs_7	10	244	(236,8)	29.50
D12	vowel0	13	988	(898,90)	9.98	D32	yeast5	8	1484	(1440,44)	32.73
D13	glass-0-1-6_vs_2	9	192	(175,17)	10.29	D33	winequality-red-8_vs_6	11	656	(638,18)	35.44
D14	glass-0-6_vs_5	9	108	(99,9)	11.00	D34	yeast6	8	1484	(1449,35)	41.40
D15	glass-0-1-4-6_vs_2	9	205	(188,17)	11.06	D35	winequality-red-8_vs_6-7	11	855	(837,18)	46.50
D16	glass2	9	214	(197,17)	11.59	D36	winequality-white-3-9_vs_5	11	1482	(1457,25)	58.28
D17	ecoli-0-1-4-7_vs_5-6	6	332	(307,25)	12.28	D37	shuttle-2_vs_5	9	3316	(3267,49)	66.67
D18	ecoli-0-1-4-6_vs_5	6	280	(260,20)	13.00	D38	winequality-red-3_vs_5	11	691	(681,10)	68.10
D19	shuttle-c0_vs_c4	9	1829	(1706,123)	13.87	D39	poker-8_vs_6	10	1477	(1460,17)	85.88
D20	yeast-1_vs_7	7	459	(429,30)	14.30	D40	abalone19	8	4174	(4142,32)	129.44

Table 2
Confusion matrix.

	Positive prediction	**Negative prediction**
Positive class	TP	FN
Negative class	FP	TN

- Support Vector Machine (SVM) [54,55]: SVM is an effective machine learning algorithm, which attempts to minimize the Vapnik–Chervonenkis structural risk instead of the empirical risk. It constructs an optimal separating hyperplane with maximal margin by mapping the original input feature space into a high dimensional feature space.
- K-Nearest Neighbor (KNN) [56]: KNN is a well-known lazy learning method, which classifies the query sample as the majority class of its k nearest neighbors. Thus, the measure of distance and the number of nearest neighbors are the key factors in the KNN algorithm.

In this study, the parameter settings for each learning algorithm are shown in Table 3. One should note that it is not necessarily optimal to set the parameters for each learning algorithm, since the purpose of our experiments are to study the robustness of PF-SMOTE with respect to different classifiers.

With the aim to verify the property of our PF-SMOTE, in addition to the ROS and original SMOTE, we also select seven well-known variants of SMOTE as the benchmark. A brief description of each selected method is as follows:

- Borderline-SMOTE[34]: This method focuses on learning examples that are at the boundary of the class because they are prone to be misclassified. It divides the examples into NOISE, DANGER and SAFE according to the number of surrounding minority neighbors, and only uses examples belonging to DANGER to generate new synthetic data.
- ADASYN[41]: This method aims to balance the deviation between the minority examples. It assigns different weights to the learning difficulty of different examples so that the easy-to-learn examples generate less synthetic data to shift the decision boundary to the difficult-to-learn examples.

Table 3
Parameters setting for base learners used in the experiments.

Abbr.	Base learner	Parameters
DT	Decision Trees	criterion = entropy, minimum examples split = 10, minimum examples leaf = 2
SVM	Support Vector Machines	C = 1, kernel = rbf, γ=0.5
KNN	k-Nearest Neighbors	k = 3, distance metric = Euclidean distance

Table 4
Parameters setting for variants of SMOTE used in the experiments.

Abbr.	Parameters
SMOTE	$k = 5$
Borderline-SMOTE	$k = 5$, $n_{neighbors}$=5
ADASYN	$k = 5$, d_{th}=0.75, β=1
Safe-Level-SMOTE	$k = 5$
MWMOTE	k_1=5, k_2=3, k_3=5, $M = 3$, C_p=3, $C_f(th)$=5.0, $CMAX = 2.0$
k-means-SMOTE	$k = 5$, $n_{clusters}$=10, $irt = 2.0$
NRAS	$k = 5$, $t = 2.5$
SOMO	n_{grid}=10, σ=0.2, $learning_{rate}$=0.5, n_{iter}=100

- Safe-Level-SMOTE[57]: This method attempts to generate examples in a safe area to avoid overlapping with the majority class of examples. It assigns a safety factor to each minority example, and the newly synthesized example is closer to the example with a high safety factor, thereby ensuring that the new example is distributed in a safe area.
- MWMOTE [58]: This method finds the decision boundary and reduces the probability of generating wrong examples. It first finds the boundary examples that are difficult to learn and calculates the corresponding weights based on their closeness and density and generates new examples in the minority example cluster where they are located.
- k-means-SMOTE[59]: This method avoids the generation of noise and overcomes the imbalance within the class. It first uses the k-means algorithm to cluster all examples into k clusters, removes the clusters with a large proportion of most classes, and then allocates the larger number of synthesized clusters

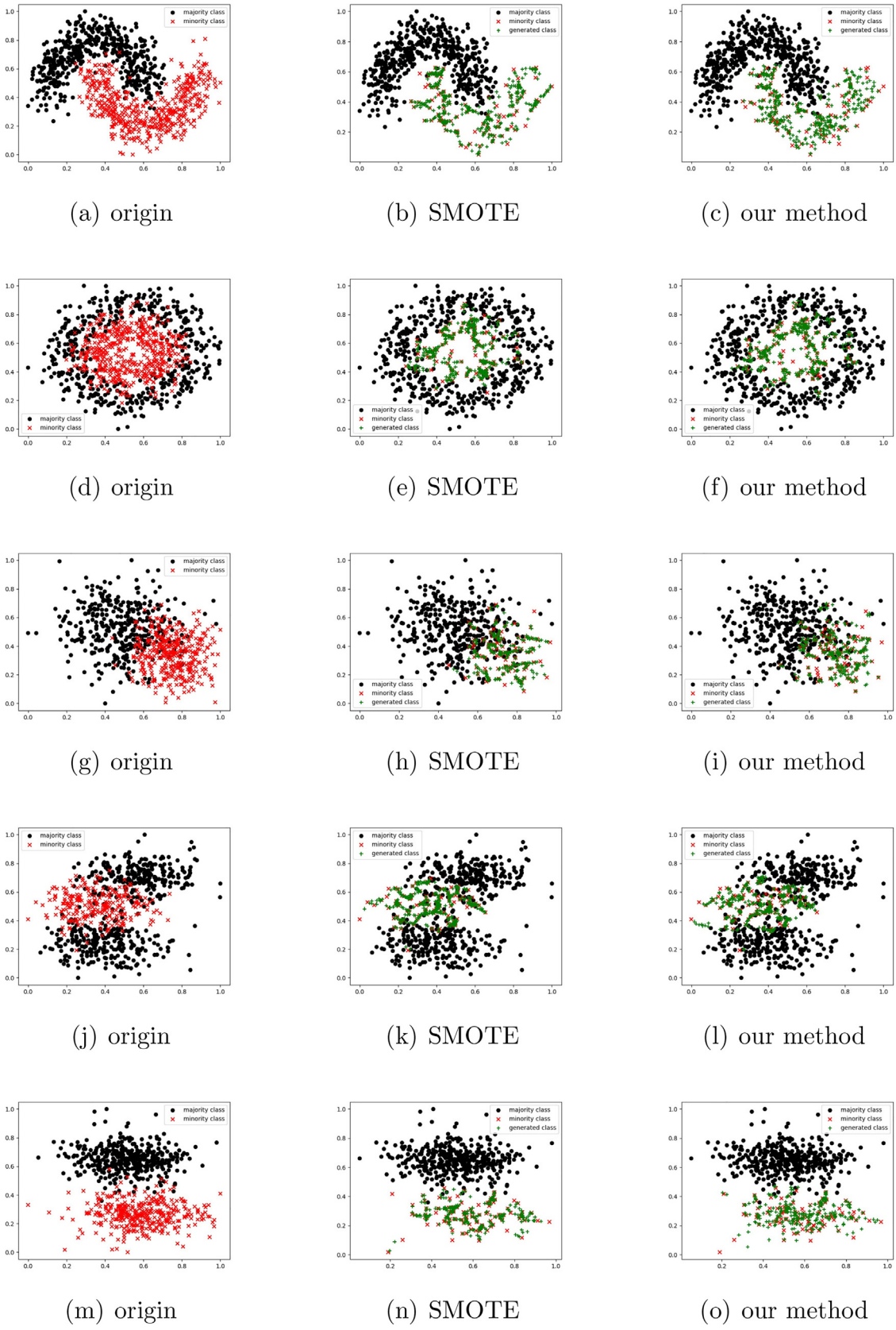

Fig. 7. Our method visually compares with SMOTE on 5 artificial datasets.

to the sparser clusters according to the density within the cluster. Finally, synthetic data are generated in each selected cluster.

- Noise reduction a priori synthetic oversampler (NRAS)[60]: This method is dedicated to removing noise from minority groups to shift boundaries. It first removes noise based on the propensity score of each minority example and then generates synthetic data.
- Self-Organizing Map Oversampler (SOMO) [61]: This method improves the selection criteria of the minority class examples to avoid the generation of noises. It first divides all examples into clusters by the Self-organizing Map (SOM) algorithm, and then uses the minority examples within a cluster and from adjacent clusters to synthesize examples.

The corresponding parameter configuration for each method is set up according to the original research, which is shown in Table 4.

5. Experimental results and discussion

The purpose of this section is to demonstrate the effectiveness and robustness of the proposed PF-SMOTE. With this objective, we carry out two aspects of experiments: (1) we provide visible results of the synthetic examples generation of PF-SMOTE in SubSection 5.1, and (2) we develop the comparison of our proposed PF-SMOTE and the well-known variants for SMOTE in SubSection 5.2.

5.1. Visualization of the synthetic examples

In order to view the ability of the proposed PF-SMOTE from the perspective of visualization, we carry out the experiments on the synthetic two-dimensional datasets. Concretely, we first produce five synthetic two-dimensional datasets, i.e. toy, moons, and circles, by using the Python library scikit-learn. Then, we randomly remove most of the examples from one of classes, which is considered as the minority class. Finally, we generate examples by using the classic SMOTE and our PF-SMOTE, respectively. In this way, it can intuitively exhibit the different ways to generate examples of classic SMOTE and our proposed PF-SMOTE.

The results of applying PF-SMOTE and classic SMOTE to the two-dimensional datasets are shown in Fig. 7, where black • represents the majority examples, red × represents the minority examples, and green + represents the generated synthetic minority examples, respectively. Fig. 7 illustrates how different our PF-SMOTE generates new minority examples with respect to classic SMOTE. Fig. 7(a), (d), (g), (j), and (m) present the original distribution of the synthetic datasets. SMOTE randomly selects a minority example and interpolates a point along the line between the

Table 5
The completed results with DT as the base learner. The best result within each dataset is highlighted in **bold-face**.

Dataset	Baseline	ROS	SMOTE	Borderline-SMOTE	ADASYN	Safe-Level-SMOTE	MWMOTE	*k*-means-SMOTE	NRAS	SOMO	PF-SMOTE
D1	0.9701	0.9557	0.9852	0.9542	0.9850	0.9748	**0.9894**	0.9641	0.9750	0.9653	0.9845
D2	0.9607	0.9787	0.9791	0.9654	0.9716	0.9621	0.9788	0.9504	0.9737	0.9486	**0.9826**
D3	0.8301	0.8301	0.8501	0.8515	0.8259	0.8639	0.8524	0.8647	0.8402	0.8669	**0.8783**
D4	0.9926	0.9911	0.9915	0.9875	0.9927	**0.9954**	0.9917	0.9916	0.9914	0.9935	0.9893
D5	0.9055	0.8626	0.9094	0.9118	0.9188	0.9153	**0.9262**	0.8953	0.9064	0.9044	0.9168
D6	0.8574	0.7989	0.8472	0.8528	0.8432	**0.8853**	0.8798	0.8222	0.8228	0.8569	0.8777
D7	0.9451	0.9260	0.9507	0.9436	0.9487	0.9522	**0.9570**	0.9427	0.9512	0.9419	0.9511
D8	0.9008	0.8438	0.9054	0.8976	0.9028	0.8846	**0.9243**	0.8970	0.9120	0.9134	0.9110
D9	**0.7295**	0.6184	0.6081	0.6076	0.5999	0.6716	0.6816	0.7127	0.6406	0.7277	0.6676
D10	0.9145	0.8893	0.8862	0.8786	0.9064	0.9137	0.9156	0.8828	0.9025	0.8899	**0.9161**
D11	0.9833	0.9900	0.9933	**0.9946**	0.9922	0.9923	0.9944	0.9878	0.9744	0.9900	0.9593
D12	0.9275	0.9253	0.9284	**0.9596**	0.9217	0.9293	0.9565	0.9414	0.9579	0.9289	0.9290
D13	0.8040	0.7594	0.7782	0.6980	0.7738	0.7676	0.7707	0.8072	0.8062	**0.8120**	0.7682
D14	**0.9950**	**0.9950**	**0.9950**	0.9462	**0.9950**	**0.9950**	**0.9950**	0.9813	0.9437	0.9570	0.9895
D15	0.7943	0.7718	0.8186	0.7072	**0.8296**	0.7233	0.7826	0.8034	0.7930	0.8190	0.7677
D16	0.6246	0.5891	0.6446	0.6101	0.6495	0.7255	0.6413	0.6249	0.5986	0.6221	**0.7668**
D17	0.8472	0.8103	0.8751	0.8114	0.8775	0.8744	0.8695	0.8597	0.8781	0.8637	**0.8823**
D18	0.9139	0.9062	0.9229	0.8500	0.9206	**0.9584**	0.9005	0.9405	0.9307	0.9116	0.9536
D19	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9997
D20	0.6118	0.6227	0.6546	0.6611	0.6327	**0.6973**	0.6602	0.6248	0.6790	0.6232	0.6462
D21	0.9161	0.8824	0.8898	0.9187	0.8839	0.9524	0.8855	0.8632	0.9275	0.8677	**0.9544**
D22	0.8845	0.9171	0.9101	0.9028	0.9023	0.9308	0.9153	0.9061	0.9145	0.8853	**0.9571**
D23	0.6390	0.6465	0.7230	0.6885	0.7128	0.7198	**0.7330**	0.6371	0.6283	0.6377	0.7118
D24	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
D25	0.7197	0.6135	0.5000	0.5310	0.5040	0.7312	0.7635	0.7216	0.7237	0.7137	**0.7643**
D26	0.8054	0.7502	0.7021	0.7377	0.7127	**0.8713**	0.7252	0.7331	0.6812	0.6270	0.7801
D27	0.9825	0.9800	0.9750	**0.9996**	0.9825	0.9825	0.9800	0.9825	0.9850	0.9800	0.9737
D28	**1.0000**	**1.0000**	**1.0000**	0.9977	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9993	**1.0000**	**1.0000**
D29	0.8942	0.8787	0.8666	0.6661	0.8382	0.8857	0.8610	**0.8944**	0.8942	0.8939	0.8708
D30	0.6219	0.5552	0.5843	0.5923	0.5923	0.5750	0.6166	**0.6255**	0.6072	0.6242	0.6040
D31	0.5924	0.5419	0.6207	0.5554	0.6012	0.5725	0.5514	0.6213	0.6428	0.5722	**0.6470**
D32	0.9146	0.8687	0.9074	0.8814	0.9038	0.8950	0.9058	0.9068	0.9166	0.9126	**0.9401**
D33	0.7604	0.5793	0.5995	0.5579	0.5665	0.5337	0.7325	0.7653	0.7607	**0.7667**	0.7162
D34	0.7687	0.7186	0.8291	0.7807	0.8295	0.8224	**0.8364**	0.7748	0.7916	0.7698	0.8075
D35	0.5159	0.5594	0.5612	0.5241	0.5607	0.5315	0.5902	0.5258	0.5388	0.5345	**0.7194**
D36	0.5468	0.4980	0.5588	0.5990	0.5460	0.5702	0.5476	0.5502	0.5562	0.5434	**0.7288**
D37	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
D38	0.5442	0.4968	0.4741	0.5321	0.4767	0.5586	0.5059	0.5446	0.5447	0.5443	**0.6331**
D39	0.7187	0.6514	0.8719	0.7763	0.8687	0.7298	0.8303	0.7183	0.5921	0.6653	**0.9494**
D40	0.5050	0.5168	**0.6215**	0.4964	0.6138	0.5589	0.5199	0.5089	0.5114	0.5077	0.5267
Avg.	0.8209	0.7930	0.8180	0.7957	0.8146	0.8276	0.8292	0.8193	0.8173	0.8145	**0.8505**

selected minority example and its nearest neighbors. From all datasets, we can observe that SMOTE generates synthetic minority examples near a line between two selected minority examples, which leads the generated synthetic closer to a convergence than the original distribution. Fig. 7(b), (e), (h), (k), and (n) show the generated examples by the classic SMOTE. Fig. 7(c), (f), (i), (l), and (o) show the examples generated by our PF-SMOTE. In contrast, the proposed PF-SMOTE allows to produce more diversified examples. In addition, PF-SMOTE tends to explore the area that no class examples are dominant, which might intensify the classification in favour of minority class. Overall, it can conclude that PF-SMOTE is effective in generating examples.

5.2. PF-SMOTE vs. well-known variants of SMOTE

The complete results are presented in Table 5, Table 7 and Table 9 for each base learner, respectively. Looking at these tables, we can observe that the average result of our proposed PF-SMOTE

Table 6
Results of Wilcoxon signed-rank tests for comparing the proposed PF-SMOTE and the well-known variants of SMOTE when DT is used as the base learner.

Comparison	R^+	R^-	Hypothesis	p-value
PF-SMOTE vs. Baseline	608.00	212.00	**Rejected for PF-SMOTE at 5%**	0.007779**
PF-SMOTE vs. ROS	751.00	69.00	**Rejected for PF-SMOTE at 5%**	0.000005**
PF-SMOTE vs. SMOTE	616.00	204.00	**Rejected for PF-SMOTE at 5%**	0.005622**
PF-SMOTE vs. borderline-SMOTE	752.50	67.50	**Rejected for PF-SMOTE at 5%**	0.000004**
PF-SMOTE vs. ADASYN	643.00	177.00	**Rejected for PF-SMOTE at 5%**	0.001736**
PF-SMOTE vs. Safe-Level-SMOTE	534.00	286.00	**Rejected for PF-SMOTE at 10%**	0.095555*
PF-SMOTE vs. MWMOTE	485.00	335.00	Not rejected	0.313386
PF-SMOTE vs. k-means-SMOTE	621.00	199.00	**Rejected for PF-SMOTE at 5%**	0.004565**
PF-SMOTE vs. NRAS	609.50	210.50	**Rejected for PF-SMOTE at 5%**	0.007328**
PF-SMOTE vs. SOMO	625.00	195.00	**Rejected for PF-SMOTE at 5%**	0.003852**

* Close to the p-value means that statistical differences are found with $\alpha = 0.1$ (90% confidence).
**Close to the p-value means that statistical differences are found with $\alpha = 0.05$ (95% confidence).

Table 7
The completed results with SVM as the base learner. The best result within each dataset is highlighted in **bold-face**.

Dataset	Baseline	ROS	SMOTE	Borderline-SMOTE	ADASYN	Safe-Level-SMOTE	MWMOTE	k-means-SMOTE	NRAS	SOMO	PF-SMOTE
D1	**0.9995**	0.9557	0.9981	0.9961	0.9970	0.9991	0.9970	0.9961	0.9981	0.9993	0.9988
D2	**1.0000**	0.9534	**1.0000**	0.9983	0.9996	0.9998	0.9999	0.9975	1.0000	**1.0000**	0.9999
D3	0.9401	0.8301	0.9669	0.9555	0.9666	0.9446	0.9459	**0.9678**	0.9668	0.9661	0.9641
D4	0.9991	0.9911	0.9995	0.9980	0.9994	0.9995	**0.9998**	0.9991	0.9992	0.9994	0.9996
D5	0.9719	0.8626	0.9747	0.9755	0.9769	0.9755	0.9738	0.9696	0.9703	0.9725	**0.9773**
D6	0.9238	0.7989	0.9143	0.9203	0.9114	0.9169	0.9166	0.9213	0.9295	0.9238	**0.9481**
D7	0.9561	0.9260	0.9635	0.9585	**0.9648**	0.9614	0.9501	0.9555	0.9505	0.9563	0.9626
D8	0.9213	0.8438	0.9548	0.9611	0.9551	0.9482	0.9461	0.9604	0.9622	0.9655	**0.9672**
D9	0.6746	0.6184	0.7683	0.7447	0.7997	0.8118	0.8174	0.6758	**0.8470**	0.6813	0.8314
D10	0.9287	0.8893	0.9466	0.9344	**0.9483**	0.9463	0.9477	0.9455	0.9452	0.9448	0.9427
D11	**1.0000**	0.9900	0.9900	0.9989	0.9917	0.9850	0.9856	0.9878	0.9806	**1.0000**	0.9972
D12	0.9528	0.9253	0.9999	0.9890	**0.9999**	0.9998	0.9992	0.9992	0.9997	0.9994	0.9999
D13	0.8129	0.7594	0.7283	0.6415	0.7534	0.5355	0.6901	**0.8684**	0.8684	0.8683	0.6487
D14	**1.0000**	0.9950	**1.0000**	0.9789	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
D15	0.8054	0.7261	0.8022	0.7511	0.7908	0.5723	0.6941	0.8874	0.8874	**0.8876**	0.6036
D16	**0.8415**	0.5891	0.7127	0.5934	0.7070	0.5721	0.6515	0.8412	0.8291	0.8412	0.5603
D17	0.9037	0.8103	0.9089	0.9227	0.9053	0.9081	0.9083	0.9064	0.9085	0.9046	0.9220
D18	0.9766	0.9062	0.9256	0.9946	0.8977	0.9866	0.9846	0.9906	0.9927	0.9863	0.9832
D19	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
D20	0.6750	0.6227	0.6856	0.7165	0.6867	0.6971	0.7056	0.7265	0.6320	0.6590	0.6853
D21	0.9500	0.8824	0.9223	0.9883	0.9223	0.9463	0.9652	0.9624	0.9581	0.8845	0.9629
D22	0.9956	0.9171	0.9963	0.9964	0.9948	0.9905	0.9948	0.9963	0.9968	0.9956	0.9946
D23	0.7916	0.6465	0.8717	0.8724	0.8707	0.8818	0.8523	0.8086	0.8243	0.8085	0.8900
D24	0.9873	**1.0000**	0.9981	0.9725	0.9981	0.9981	0.9981	0.9982	0.9691	0.9875	0.9967
D25	0.9600	0.6135	0.9900	0.9633	0.9900	0.9900	1.0000	0.9700	0.9900	0.9900	0.9765
D26	0.9917	0.7502	**1.0000**	0.9720	**1.0000**	0.9800	0.9744	0.9786	0.9725	0.9306	**1.0000**
D27	0.9962	0.9800	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
D28	**1.0000**	**1.0000**	**1.0000**	0.9745	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
D29	0.9003	0.8787	0.9249	0.5319	0.8562	0.9010	0.8663	0.8887	0.9343	0.8895	**0.9477**
D30	0.6815	0.5552	0.7776	0.7801	0.7778	0.7774	0.7463	0.7037	0.6845	0.7038	**0.8009**
D31	0.9617	0.5419	0.9593	0.8790	0.9582	0.9616	0.9707	0.9546	0.9462	**0.9775**	0.9720
D32	0.9893	0.8687	0.9885	0.9819	0.9888	0.9856	0.9881	0.9887	0.9877	0.9895	0.9885
D33	0.7552	0.5562	0.8890	0.7846	**0.8989**	0.8801	0.8690	0.7555	0.7552	0.7552	0.8845
D34	0.9275	0.7186	0.9274	0.9204	**0.9312**	0.9350	0.9430	0.9275	0.9387	0.9275	0.9415
D35	0.6576	0.5764	0.8725	0.8669	0.8730	**0.9045**	0.8768	0.7591	0.7593	0.7590	0.8934
D36	0.7541	0.4980	0.6524	0.7374	0.6447	0.7046	0.5918	0.7541	0.6613	0.7541	**0.7411**
D37	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
D38	0.7535	0.4968	0.7266	0.6794	0.7260	0.6788	0.7023	0.7407	0.7467	0.7419	**0.8303**
D39	0.9735	0.6514	0.9984	0.8288	0.9984	0.9859	0.9818	0.9925	0.7105	0.9925	0.9955
D40	0.6651	0.5168	0.8395	0.6320	**0.8412**	0.8146	0.6957	0.6651	0.6651	0.6651	0.7936
Avg.	0.8994	0.7910	0.9144	0.8848	0.9130	0.9019	0.9033	0.9110	0.9042	0.9077	**0.9150**

is the highest in all cases. In addition, the number of datasets in which PF-SMOTE outperforms is also remarkable. Specifically, our proposed PF-SMOTE obtains the best results in 17 (DT), 12 (SVM), and 17 (KNN) out of 40 cases. In other words, our proposed PF-SMOTE has good property for generating synthetic examples regardless of the base learner used.

Although we can find the superiority of the proposed PF-SMOTE with respect to the selected methods from experimental results, we still cannot draw a meaningful conclusion without proper statistical analysis. Thus, we employ the Wilcoxon signed-rank tests in each scenario of base learner to check whether significant differences exist between the proposed PF-SMOTE and the well-known variants of SMOTE. The results of statistical tests are shown in Table 6, Table 8, and Table 10, for DT, SVM, and KNN, respectively.

Regarding DT as the base learner, we can observe the statistical results from Table 6 in two aspects. First, the value of R^+ for PF-

Table 8
Results of Wilcoxon signed-rank tests for comparing the proposed PF-SMOTE and the well-known variants of SMOTE when SVM is used as the base learner.

Comparison	R^+	R^-	Hypothesis	p-value
PF-SMOTE vs. Baseline	641.00	179.00	**Rejected for PF-SMOTE at 5%**	0.001901**
PF-SMOTE vs. ROS	760.00	60.00	**Rejected for PF-SMOTE at 5%**	0.000003**
PF-SMOTE vs. SMOTE	493.50	326.50	Not rejected	0.261526
PF-SMOTE vs. borderline-SMOTE	652.00	168.00	**Rejected for PF-SMOTE at 5%**	0.001142**
PF-SMOTE vs. ADASYN	489.50	330.50	Not rejected	0.285067
PF-SMOTE vs. Safe-Level-SMOTE	635.50	184.50	**Rejected for PF-SMOTE at 5%**	0.002432**
PF-SMOTE vs. MWMOTE	559.50	260.50	**Rejected for PF-SMOTE at 5%**	0.044403**
PF-SMOTE vs. k-means-SMOTE	566.50	253.50	**Rejected for PF-SMOTE at 5%**	0.035375**
PF-SMOTE vs. NRAS	594.50	225.50	**Rejected for PF-SMOTE at 5%**	0.013121**
PF-SMOTE vs. SOMO	537.50	282.50	**Rejected for PF-SMOTE at 10%**	0.086501*

* Close to the p-value means that statistical differences are found with $\alpha = 0.1$ (90% confidence).
** Close to the p-value means that statistical differences are found with $\alpha = 0.05$ (95% confidence).

Table 9
The completed results with KNN as the base learner. The best result within each dataset is highlighted in **bold-face**.

Dataset	Baseline	ROS	SMOTE	Borderline-SMOTE	ADASYN	Safe-Level-SMOTE	MWMOTE	k-means-SMOTE	NRAS	SOMO	PF-SMOTE
D1	**0.9986**	0.9557	0.9851	0.9858	0.9889	0.9833	0.9867	0.9973	0.9914	0.9984	0.9919
D2	0.9928	0.9787	0.9937	0.9910	0.9907	0.9747	**0.9952**	0.9942	0.9934	0.9930	0.9938
D3	0.9566	0.8301	0.9482	0.9609	0.9260	0.9572	0.9516	0.9589	0.9561	**0.9611**	0.9521
D4	**0.9984**	0.9911	0.9949	0.9975	0.9924	0.9979	0.9945	0.9977	0.9978	0.9984	0.9972
D5	0.9149	0.8626	0.9183	0.9105	0.9180	0.9114	0.9229	0.9180	0.9292	0.9149	**0.9334**
D6	0.8665	0.7989	0.8992	0.8930	0.9000	0.8997	0.9076	0.9089	0.8897	0.8665	**0.9157**
D7	0.9246	0.9260	0.9239	0.9359	0.9489	0.9193	**0.9611**	0.9267	0.9339	0.9259	0.9306
D8	0.8733	0.8438	0.8818	**0.9007**	0.8805	0.8847	0.8946	0.8724	0.8794	0.8814	0.8890
D9	0.7894	0.6184	0.7994	0.7778	0.7928	0.8108	0.7861	0.7894	0.7864	0.7894	**0.8015**
D10	0.9343	0.8893	0.9295	0.9153	0.9214	0.9293	0.9230	0.9337	**0.9369**	0.9331	0.9328
D11	0.8556	**0.9900**	0.8950	0.9537	0.8797	0.9660	0.9000	0.8528	0.9129	0.8556	0.9375
D12	0.9999	0.9253	**1.0000**	0.9976	1.0000	0.9988	**1.0000**	1.0000	0.9956	0.9999	**1.0000**
D13	0.7715	0.7594	0.8017	0.7143	0.7683	0.8127	**0.8357**	0.7715	0.7715	0.7715	0.7376
D14	**1.0000**	0.9950	**1.0000**	0.9873	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9950	**1.0000**	**1.0000**
D15	0.7270	0.7718	0.8073	0.6379	**0.8157**	0.7303	0.7996	0.7270	0.7244	0.7270	0.7206
D16	0.7297	0.5891	**0.8369**	0.6810	0.8277	0.7913	0.8262	0.7297	0.7344	0.7297	0.7831
D17	0.9211	0.8103	0.9146	0.9034	0.9127	0.9136	0.9176	0.9207	0.9215	0.9211	**0.9359**
D18	0.9871	0.9062	0.9871	0.9869	0.9853	0.9845	0.9863	0.9871	0.9871	0.9871	**0.9884**
D19	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
D20	0.6923	0.6227	0.7091	0.7464	0.7081	**0.7565**	0.6959	0.6906	0.7023	0.6923	0.6623
D21	**0.9875**	0.8824	0.9788	0.9795	0.9732	0.9651	0.9725	0.9798	0.9700	0.9851	0.9718
D22	0.9348	0.9171	0.9339	0.9276	0.9335	0.9308	0.9340	0.9343	0.9380	0.9348	**0.9730**
D23	0.6675	0.6465	0.7070	**0.7391**	0.7089	0.6779	0.7139	0.6675	0.6784	0.6675	0.6800
D24	**1.0000**	**1.0000**	**1.0000**	0.9973	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9988	**1.0000**	**1.0000**
D25	0.9900	0.6135	0.9900	0.9930	0.9900	**0.9935**	0.9900	0.9900	0.9900	0.9900	0.9782
D26	0.7922	0.7502	0.9499	**0.9514**	0.9480	0.9493	0.9268	0.8242	0.9187	0.7922	0.9497
D27	0.9750	0.9800	0.9750	**1.0000**	0.9750	0.9750	0.9750	0.9750	0.9750	0.9750	0.9898
D28	0.7250	**1.0000**	0.9825	0.9432	0.9850	0.9625	0.9000	0.8975	0.9995	0.7250	0.9500
D29	0.8935	0.8787	**0.9315**	0.9148	0.9304	0.8909	0.9140	0.8935	0.8916	0.8935	0.9263
D30	0.5851	0.5552	0.5688	0.5779	0.5624	0.5551	0.6146	0.5851	0.5833	0.5851	**0.6278**
D31	0.9750	0.5419	0.9733	0.9633	0.9733	0.9724	0.9734	0.9750	0.9735	**0.9750**	0.9710
D32	0.9286	0.8687	0.9258	0.9200	0.9677	0.9491	0.9353	0.9276	0.9368	0.9286	**0.9726**
D33	0.5882	0.5793	0.8336	0.6010	0.8331	0.8157	0.8450	0.5882	0.5882	0.5882	**0.8542**
D34	0.8606	0.7186	0.8765	0.8549	0.8713	0.8629	**0.9092**	0.8606	0.8910	0.8606	0.9032
D35	0.5219	0.5594	0.5030	0.5025	0.5027	0.5308	0.6534	0.5219	0.5219	0.5219	**0.6794**
D36	0.5081	0.4980	0.5132	0.5530	0.5130	0.5464	0.5411	0.5081	0.5081	0.5081	**0.5952**
D37	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
D38	0.4942	0.4968	0.5285	0.5387	0.5280	0.5570	0.5314	0.4942	0.4942	0.4942	**0.6758**
D39	0.9619	0.6514	0.9819	0.9468	**0.9833**	0.9587	0.9618	0.9619	0.9556	0.9619	0.9819
D40	0.4983	0.5168	**0.6003**	0.5217	0.6003	0.4997	0.4912	0.4983	0.4983	0.4983	0.4958
Avg.	0.8455	0.7930	0.8745	0.8576	0.8734	0.8704	0.8767	0.8515	0.8587	0.8458	**0.8820**

Table 10
Results of Wilcoxon signed-rank tests for comparing the proposed PF-SMOTE and the well-known variants of SMOTE when KNN is used as the base learner.

Comparison	R^+	R^-	Hypothesis	p-value
PF-SMOTE vs. Baseline	648.00	172.00	**Rejected for PF-SMOTE at 5%**	0.001377**
PF-SMOTE vs. ROS	744.00	76.00	**Rejected for PF-SMOTE at 5%**	0.000007**
PF-SMOTE vs. SMOTE	504.50	315.50	Not rejected	0.203910
PF-SMOTE vs. borderline-SMOTE	608.50	211.50	**Rejected for PF-SMOTE at 5%**	0.007628**
PF-SMOTE vs. ADASYN	513.00	307.00	Not rejected	0.166172
PF-SMOTE vs. Safe-Level-SMOTE	573.00	247.00	**Rejected for PF-SMOTE at 5%**	0.028439**
PF-SMOTE vs. MWMOTE	536.50	283.50	**Rejected for PF-SMOTE at 10%**	0.088998*
PF-SMOTE vs. k-means-SMOTE	641.00	179.00	**Rejected for PF-SMOTE at 5%**	0.001901**
PF-SMOTE vs. NRAS	620.50	199.50	**Rejected for PF-SMOTE at 5%**	0.004663**
PF-SMOTE vs. SOMO	648.00	172.00	**Rejected for PF-SMOTE at 5%**	0.001377**

* Close to the p-value means that statistical differences are found with $\alpha = 0.1$ (90% confidence).
** Close to the p-value means that statistical differences are found with $\alpha = 0.05$ (95% confidence).

SMOTE are considerably greater than that of R^- for the compared methods in all cases. Second, all equivalence hypotheses are rejected for PF-SMOTE except for the case of PF-SMOTE vs. MWMOTE (a small p-value 0.313386). That is, the statistical results support the superiority of our PF-SMOTE when comparing the well-known variants of SMOTE in the scenario of DT as the base learner.

With respect to SVM (see Table 8), in general, the results are less significant than in the case of DT. The Wilcoxon tests reject all the null hypotheses of the equivalence between the methods except for the case of the comparison of PF-SMOTE vs. SMOTE (a small p-value 0.261526) and ADASYN (a small p-value 0.285067). Despite no significant differences being found in the case of Safe-Level-SMOTE, we can still highlight the superiority of the proposed PF-SMOTE, because the R^+ of PF-SMOTE is slightly larger than the R^- of SMOTE and ADASYN.

Finally, in the case of KNN as the base learner (see Table 10), the statistical results are similar to SVM. Concretely, except for the case of PF-SMOTE vs. SMOTE and ADASYN, in which the obtained p-values (0.203910 and 0.166172, respectively) are lower than our α-value (0.1), all the hypothesis of equivalence are rejected for PF-SMOTE by the Wilcoxon test with a very low p-value in the other cases. Considering the ranks obtained from the cases SMOTE and ADASYN, we can also point out that our PF-SMOTE is better than the ADASYN due to the greater value of R^+ for PF-SMOTE.

Summarizing the results obtained from the statistical analysis, we have demonstrated our proposed PF-SMOTE is the most robust and effective variant for the family of SMOTE with respect to the selected methods.

6. Concluding Remarks

To deal with binary imbalanced datasets, we in this paper develop a novel variant for SMOTE, in which the minority examples are categorized into two groups: boundary and safe examples. Considering the definitely different characteristics of the two kinds of minority examples, different procedure for generating synthetic examples are developed with the aim to fill the margin between two classes. In addition, our new variant for SMOTE is a kind of parameter free method due to no specific parameters need to be fixed in our method.

In order to demonstrate the effectiveness and robustness of our proposed method, a thorough experimental study based on a large number of real-world applications has been provided. The visualization of the synthetic examples produced by the proposed method is provided in marked contrast to the classic SMOTE. From the experimental results of the comparison of our method and the state-of-the-art SMOTE-based methods, we also can conclude that the proposed variant of SMOTE significantly outperforms the previous techniques. Furthermore, we must stress that the superiority of the proposed method received in this study is due to the interpolation in the space that no class is dominant.

Regarding to the robust behavior of our proposed variant for SMOTE, we believe that this study opens up a new direction for developing SMOTE-based method. With respect to our further work, we intend to introduce our strategy in the multi-class imbalanced datasets that is considerably more complex than binary scenario. Furthermore, it will be interesting to investigate different categorizations of minority examples and develop corresponding strategy for each categorization. Finally, we provide one future work is to consider the negative effect of noise examples in our method.

CRediT authorship contribution statement

Qiong Chen: Methodology, Software. **Zhong-Liang Zhang:** Conceptualization, Writing - original draft, Writing - review & editing, Supervision. **Wen-Po Huang:** Investigation. **Jian Wu:** Writing - review & editing. **Xing-Gang Luo:** Supervision, Funding acquisition.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The authors would like to thank the (anonymous) reviewers for their constructive comments. This work was supported by the National Science Foundation of China under Grant No. 71801065, 72171065 and 71831006, Zhejiang Provincial Natural Science Foundation of China under Grant No. LZ20G010001 and LY20G020013, as well as the Fundamental Research Funds for the Provincial Universities of Zhejiang under Grant No. GK209907299001-202.

References

[1] A. Guzmán-Ponce, J.S. Sánchez, R.M. Valdovinos, J.R. Marcial-Romero, DBIG-US: A two-stage under-sampling algorithm to face the class imbalance problem, Expert Systems with Applications 168 (2021) 114301.
[2] F. Thabtah, S. Hammoud, F. Kamalov, A. Gonsalves, Data imbalance in classification: Experimental evaluation, Information Sciences 513 (2020) 429–441.
[3] S. Liang, H. Yu, Revealing new therapeutic opportunities through drug target prediction: A class imbalance-tolerant machine learning approach, Bioinformatics 36 (16) (2020) 4490–4497.

[4] B. Zhao, X. Zhang, H. Li, Z. Yang, Intelligent fault diagnosis of rolling bearings based on normalized CNN considering data imbalance and variable working conditions, Knowledge-Based Systems 199 (2020) 105971.
[5] Z. Li, M. Huang, G. Liu, C. Jiang, A hybrid method with dynamic weighted entropy for handling the problem of class imbalance with overlap in credit card fraud detection, Expert Systems with Applications 175 (2021) 114750.
[6] G. Lemaître, F. Nogueira, C.K. Aridas, Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning, The Journal of Machine Learning Research 18 (1) (2017) 559–563.
[7] R.C. Prati, G.E. Batista, D.F. Silva, Class imbalance revisited: a new experimental setup to assess the performance of treatment methods, Knowledge and Information Systems 45 (1) (2015) 247–270.
[8] Z.-L. Zhang, X.-G. Luo, S. González, S. García, F. Herrera, DRCW-ASEG: One-versus-one distance-based relative competence weighting with adaptive synthetic example generation for multi-class imbalanced datasets, Neurocomputing 285 (2018) 176–187.
[9] M. Koziarski, B. Krawczyk, M. Woźniak, Radial-based oversampling for noisy imbalanced data classification, Neurocomputing 343 (2019) 19–33.
[10] C. Jia, M. Zhang, C. Fan, F. Li, J. Song, Formator: predicting lysine formylation sites based on the most distant undersampling and safe-level synthetic minority oversampling, IEEE/ACM Transactions on Computational Biology and Bioinformatics doi: 10.1109/TCBB.2019.2957758 doi:10.1109/TCBB.2019.2957758. .
[11] S.A. Alasadi, W.S. Bhaya, Review of data preprocessing techniques in data mining, Journal of Engineering and Applied Sciences 12 (16) (2017) 4102–4107.
[12] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, SMOTE: Synthetic minority over-sampling technique, Journal of Artificial Intelligence Rresearch 16 (2002) 321–357.
[13] G. Kovács, Smote-variants: A python implementation of 85 minority oversampling techniques, Neurocomputing 366 (2019) 352–354.
[14] A. Fernández, S. Garcia, F. Herrera, N.V. Chawla, SMOTE for learning from imbalanced data: Progress and challenges, marking the 15-year anniversary, Journal of Artificial Intelligence Research 61 (2018) 863–905.
[15] A.P. Bradley, The use of the area under the ROC curve in the evaluation of machine learning algorithms, Pattern Recognition 30 (7) (1997) 1145–1159.
[16] J. Demšar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine Learning Research 7 (2006) 1–30.
[17] S. Wang, X. Yao, Multiclass imbalance problems: Analysis and potential solutions, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 42 (4) (2012) 1119–1130.
[18] G.M. Weiss, Mining with rarity: a unifying framework, ACM Sigkdd Explorations Newsletter 6 (1) (2004) 7–19.
[19] G.M. Weiss, Y. Tian, Maximizing classifier utility when there are data acquisition and modeling costs, Data Mining and Knowledge Discovery 17 (2) (2008) 253–282.
[20] J.J. Rodríguez, J.-F. Diez-Pastor, A. Arnaiz-Gonzalez, L.I. Kuncheva, Random balance ensembles for multiclass imbalance learning, Knowledge-Based Systems 193 (2020) 105434.
[21] A. Fernández, V. LóPez, M. Galar, M.J. Del Jesus, F. Herrera, Analysing the classification of imbalanced data-sets with multiple classes: Binarization techniques and ad-hoc approaches, Knowledge-Based Systems 42 (2013) 97–110.
[22] G.E. Batista, R.C. Prati, M.C. Monard, A study of the behavior of several methods for balancing machine learning training data, ACM SIGKDD Explorations Newsletter 6 (1) (2004) 20–29.
[23] D.A. Cieslak, T.R. Hoens, N.V. Chawla, W.P. Kegelmeyer, Hellinger distance decision trees are robust and skew-insensitive, Data Mining and Knowledge Discovery 24 (1) (2012) 136–158.
[24] Y. Tang, Y.-Q. Zhang, N.V. Chawla, S. Krasser, SVMs modeling for highly imbalanced classification, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 39 (1) (2008) 281–288.
[25] W.M. Czarnecki, J. Tabor, Multithreshold entropy linear classifier: Theory and applications, Expert Systems with Applications 42 (13) (2015) 5591–5606.
[26] S. Datta, S. Das, Near-bayesian support vector machines for imbalanced data classification with equal or unequal misclassification costs, Neural Networks 70 (2015) 39–52.
[27] S. Höppner, B. Baesens, W. Verbeke, T. Verdonck, Instance-dependent cost-sensitive learning for detecting transfer fraud, European Journal of Operational Research 297 (1) (2022) 291–300.
[28] A. Telikani, A.H. Gandomi, K.-K.R. Choo, J. Shen, A cost-sensitive deep learning based approach for network traffic classification, IEEE Transactions on Network and Service Management 19 (1) (2022) 661–670.
[29] B. Krawczyk, M. Woźniak, G. Schaefer, Cost-sensitive decision tree ensembles for effective imbalanced classification, Applied Soft Computing 14 (2014) 554–562.
[30] G. Haixiang, L. Yijing, J. Shang, G. Mingyun, H. Yuanyue, G. Bing, Learning from class-imbalanced data: Review of methods and applications, Expert Systems with Applications 73 (2017) 220–239.
[31] L. Breiman, Bagging predictors, Machine Learning 24 (2) (1996) 123–140.
[32] J.H.Y.F. Qi Wang, ZhiHao Luo, Z. Liu, A novel ensemble method for imbalanced data learning: bagging of extrapolation-SMOTE SVM, Computational Intelligence and Neuroscience (2017) 1–11, https://doi.org/10.1155/2017/1827016.
[33] Y. Freund, R.E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting, Journal of Computer and System Sciences 55 (1) (1997) 119–139.
[34] H. Han, W.-Y. Wang, B.-H. Mao, Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning, in: International Conference on Intelligent Computing, 2005, pp. 878–887. .
[35] J. Li, Q. Zhu, Q. Wu, Z. Fan, A novel oversampling technique for class-imbalanced learning based on SMOTE and natural neighbors, Information Sciences 565 (2021) 438–455.
[36] D. Mease, A.J. Wyner, A. Buja, Boosted classification trees and class probability/quantile estimation, Journal of Machine Learning Research 8 (2007) 409–439.
[37] L. Abdi, S. Hashemi, To combat multi-class imbalanced problems by means of over-sampling techniques, IEEE Transactions on Knowledge and Data Engineering 28 (1) (2015) 238–251.
[38] F. Koto, SMOTE-Out, SMOTE-Cosine, and Selected-SMOTE: An enhancement strategy to handle imbalance in data level, in: 2014 International Conference on Advanced Computer Science and Information System, 2014, pp. 280–284. .
[39] J. Mathew, M. Luo, C.K. Pang, H.L. Chan, Kernel-based SMOTE for SVM classification of imbalanced datasets, in: IECON 2015-41st Annual Conference of the IEEE Industrial Electronics Society, 2015, pp. 1127–1132. .
[40] M.K. Hamdan, D.T. Rover, M.J. Darr, J. Just, Generalizable semi-supervised learning method to estimate mass from sparsely annotated images, Computers and Electronics in Agriculture 175 (2020) 105533.
[41] H. He, Y. Bai, E.A. Garcia, S. Li, ADASYN: Adaptive synthetic sampling approach for imbalanced learning, in: 2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence), 2008, pp. 1322–1328. .
[42] R. Alejo, V. García, J.H. Pacheco-Sánchez, An efficient over-sampling approach based on mean square error back-propagation for dealing with the multi-class imbalance problem, Neural Processing Letters 42 (3) (2015) 603–617.
[43] X.T. Dang, D.H. Tran, O. Hirose, K. Satou, SPY: A novel resampling method for improving classification performance in imbalanced data, in: 2015 Seventh International Conference on Knowledge and Systems Engineering (KSE), 2015, pp. 280–285. .
[44] J. Błaszczyński, M. Deckert, J. Stefanowski, S. Wilk, IIvotes ensemble for imbalanced data, Intelligent Data Analysis 16 (5) (2012) 777–801.
[45] K. Sriwanna, K. Puntumapon, K. Waiyamai, An enhanced class-attribute interdependence maximization discretization algorithm, in: International Conference on Advanced Data Mining and Applications, 2012, pp. 465–476. .
[46] J.A. Sáez, J. Luengo, J. Stefanowski, F. Herrera, SMOTE-IPF: Addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering, Information Sciences 291 (2015) 184–203.
[47] E. Ramentol, Y. Caballero, R. Bello, F. Herrera, SMOTE-RSB*: A hybrid preprocessing approach based on oversampling and undersampling for high imbalanced data-sets using smote and rough sets theory, Knowledge and Information Systems 33 (2) (2012) 245–265.
[48] L. Suo, Y.-S. Hu, H. Li, M. Armand, L. Chen, A new class of solvent-in-salt electrolyte for high-energy rechargeable metallic lithium batteries, Nature Communications 4 (1) (2013) 1–9.
[49] E. Ramentol, I. Gondres, S. Lajes, R. Bello, Y. Caballero, C. Cornelis, F. Herrera, Fuzzy-rough imbalanced learning for the diagnosis of high voltage circuit breaker maintenance: The smote-frst-2t algorithm, Engineering Applications of Artificial Intelligence 48 (2016) 134–139.
[50] G. Douzas, F. Bacao, Geometric SMOTE a geometrically enhanced drop-in replacement for SMOTE, Information Sciences 501 (2019) 118–135.
[51] F. Wilcoxon, Individual comparisons by ranking methods, Biometrics Bulletin 1 (6) (1945) 80–83.
[52] L. Breiman, J. Friedman, C.J. Stone, R.A. Olshen, Classification and regression trees, CRC Press, 1984.
[53] J.R. Quinlan, C4.5: Programs for machine learning, 2014. .
[54] V.N. Vapnik, V. Vapnik, Statistical learning theory, Vol. 1, Wiley, New York, 1998.
[55] V. Vapnik, The support vector method of function estimation, Nonlinear Modeling: Advanced Black-box Techniques 55 (1998) 86.
[56] T. Cover, P. Hart, Nearest neighbor pattern classification, IEEE Transactions on Information Theory 13 (1) (1967) 21–27.
[57] C. Bunkhumpornpat, K. Sinapiromsaran, 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, 2009, pp. 475–482. .
[58] S. Barua, M.M. Islam, X. Yao, K. Murase, MWMOTE–majority weighted minority oversampling technique for imbalanced data set learning, IEEE Transactions on Knowledge and Data Engineering 26 (2) (2012) 405–425.
[59] G. Douzas, F. Bacao, F. Last, Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE, Information Sciences 465 (2018) 1–20.
[60] W.A. Rivera, Noise reduction a priori synthetic over-sampling for class imbalanced data sets, Information Sciences 408 (2017) 146–161.
[61] G. Douzas, F. Bacao, Self-Organizing Map Oversampling (SOMO) for imbalanced data set learning, Expert Systems with Applications 82 (2017) 40–52.

Qiong Chen is currently an undergraduate student from Honors College of Hangzhou Dianzi University. She is interested specifically in data mining and knowledge engineering.

Zhong-Liang Zhang received his Ph.D. degree in system engineering from the Northeastern University, China. He is currently an associate professor in the Department of Information Management and Information System, Hangzhou Dianzi University, China. His current research interests include classification algorithms, imbalance learning and ensemble learning.

Wenpo Huang is currently an Associate Professor in the School of Management at Hangzhou Dianzi University. He received his B.Sc. and M.Sc. degrees in Mathematics from Xi'an Jiaotong University and his Ph.D. in Business Administration from the University of Macau. His current research interests include statistical process control, supply chain management and data mining.

Jian Wu received his Ph.D. degree in technical economics and management from Tongji University, China. He is currently a professor in Hangzhou Dianzi University, China. His current research interests include digital innovation and management of public opinion.

Xing-Gang Luo received a M.Sc. degree in mechanical design and manufacturing from the Northeastern University, China, and a Ph.D. degree in system engineering from the Northeastern University. His current research interests include product planning, data mining and service operation management.