A Novel Oversampling Technique for Imbalanced Credit Scoring Datasets

Sudhansu Ranjan Lenka[1(✉)], Sukant Kishoro Bisoy[1], Rojalina Priyadarshini[1], and Jhalak Hota[2]

[1] Computer Science and Engineering, C.V. Raman Global University, Bhubaneswar, India
sudhansulenka2000@gmail.com
[2] KIIT University, Bhubaneswar, India
jhalak.hotafcs@kiit.ac.in

Abstract. The imbalanced class distribution of credit-scoring datasets typically makes the learning algorithms ineffective. In this study, NOSTE is proposed, a novel oversampling technique. It first identifies the informative minority instances by eliminating the noisy samples from the minority subset. Then, weight is assigned to the informative minority instances by considering the density and distance factors. Finally, new minority instances are created by determining the average of two different minority instances to make the dataset balanced. In the experimental study, NOSTE performance is validated by conducting an extensive comparison with four popular oversampling methods using three credit-scoring datasets from the UCI repository. The results confirmed that the proposed method brings significant improvement in the classification in terms of F-measure and AUC (Area under the Curve).

Keywords: Credit Scoring · Imbalance Class Distribution · Noisy Samples · Oversampling · Classification

1 Introduction

In the financial world, credit risk has gained tremendous importance for the growth and sustainability of organizations. The main task of credit risk is to identify the credit defaulters. Through the credit scoring model, banks access the creditworthiness of the applicants before approving their loan applications [1, 2]. Credit scoring is a binary classification problem where the applicants are classified as bad credit or good credit. In financial industries, most of the applicants are good as they repay the loan amount, but very few are bad as they are unable or unwilling to repay the loan amount [3]. Imbalanced data distribution is a condition where the number of positive (or minority) class instances is significantly lowered as compared to that of negative (or majority) class instances. The class distributions in the credit scoring problem are imbalanced, where the number of default applicants is significantly lower than that of non-defaulters [3]. Due to this significant difference in the class distribution, the classifiers are unable to learn

S. Nandan Mohanty et al. (Eds.): ICISML 2022, LNICST 471, pp. 147–157, 2023.
https://doi.org/10.1007/978-3-031-35081-8_12

the minority samples, which makes them more biased towards the majority samples. In such conditions, the overall accuracy of the model is high, but it often misclassifies the minority samples. This misclassification rate significantly increases the economic losses for the banks and financial industries [4].

The resampling technique is one of the common approaches widely used to alter the class distribution of the training set and then the balanced set is used to train the classification models. The resampling method either increases the minority class instances (oversampling) or decreases the majority class instances (under-sampling). Traditional oversampling methods have certain limitations, their approach is synthesized based [5–8]. The most popular oversampling is Synthetic Minority Oversampling Technique (SMOTE) [5]. It generates new samples along the line segment of two randomly selected neighboring minority instances. SMOTE method may pick the noisy samples during new instance generation as a result more noisy samples may get introduced into the dataset. Due to this limitation, different variants of SMOTE have been proposed, such as Borderline-SMOTE (BSMOTE) [7], Adaptive Synthetic Sampling (ADASYN) [6], and Safe-Level-SMOTE (SSMOTE) [9]. Each of these variations uses the k-nearest neighbor (KNN) algorithm to generate new samples by linear interpolation of neighboring samples, but in each method, different approaches are implemented to select the source minority samples. For SMOTE and their variants, we set the number of nearest neighbors, $k = 5$.

In this study, NOSTE a novel oversampling method has been proposed, whose goal is to make the imbalanced class distribution balanced by generating synthetic minority instances along the line segment of two dissimilar minority instances. The proposed method is very different from the conventional oversampling methods because it generates synthetic instances by computing the average of two dissimilar minority instances. As a result, the newly generated instances are very unique and evenly distributed within the region of minority class samples.

The remaining part of the paper is outlined as follows: Section 2 presents the related work of imbalanced learning approaches. The proposed work method and the algorithm are discussed in Sect. 3. Section 4 presents the experimental setup considering four factors: credit scoring datasets, classifiers, resampling methods, and evaluation metrics. Result analysis has been discussed in Sect. 5 and finally, the conclusion is presented in Sect. 6.

2 Literature Review

In many real-world applications, the uneven distribution of instances in each class is a common issue. The most common methods to deal with these imbalanced class distribution issues are the data level, the algorithm level, and the cost-sensitive methods. In this paper, we have applied the data-level method for oversampling. Therefore, we discuss only the data-level methods in this section.

a. **The Data-level Methods**

These methods implement different methods in the pre-processing step to make the training set balanced. Resamplings are the most common and effective methods used

to handle the imbalanced ratio either by generating synthetic samples (oversampling) [10], or by removing samples (under-sampling) [11] from the dataset. The resampling method balances the dataset, which improves the performance of the traditional classifiers [12]. The undersampling method is useful when the dataset is very large because the loss of information due to the elimination of majority class instances is marginal [13]. In this study, we employed small-sized imbalanced credit-scoring datasets, therefore only oversampling methods are discussed. Oversampling methods balance the class distributions by simply duplicating the existing minority instances, and it may lead to overfitting [14]. SMOTE is a popular oversampling technique, in which for each minority instance, K-nearest neighbors are identified to generate new samples through interpolation [5]. However, SMOTE may suffer from over-generalization. To overcome this over-generalization issue, different variants of SMOTE have been proposed, such as Borderline_SMOTE, Safe-level SMOTE, and ADASYN. Borderline-SMOTE generates new minority samples by interpolating the instances located near the decision boundary. Safe-level SMOTE defines the 'safe level' for each minority instance, and generates new instances closer to this safe level. The safe level of a minority instance is computed by finding the number of other minority instances in its neighborhood. ADASYN method oversamples the instances by assigning more weights to the minority instances that have more number of majority instances in their neighborhood, and thus instances are used in the oversampling. The synthetic instances generated by these methods may overlap with the majority class region, and this may lead to an increase in the misclassification rate of the majority class instances [15]. Another drawback of these methods is that they may generate new instances that are very similar to the existing instances. For example, when $k = 1$, then two very similar instances are used in the sample generation process, which results in the generation of duplicate instances. Additionally, the SMOTE-based methods may not be effective if noisy samples are not properly handled before training.

3 Proposed Model

The objective of the proposed oversampling method is to generate new minority instances by averaging two minority instances, not necessarily close to each other. The newly generated sample carries the properties of both instances, rather than simply duplicating the instances that are involved in the oversampling process. The proposed oversampling method, NOSTE involves three basic phases: elimination of noisy minority samples, determination of the weights of each minority instance, and generation of synthetic instances. Algorithm-1 presents the detailed process of NOSTE, in which the first phase is described in step 1, the second phase in steps 2 to 6, and Algorithm-2 describes the final phase. Each phase is discussed in the following sub-sections.

3.1 Elimination of Noise Points from Minority Instances

Let T_{min} and T_{maj} represents the minority and majority class subset, respectively. All the minority class instances do not carry equal importance because there may be some noisy points in T_{min}. It is, therefore, required to eliminate the noisy samples from T_{min}. A minority instance can be treated as a noisy point if it is located in the majority class

region, i.e. its nearest neighbor set contains only majority class instances. For each $x_i \in T_{min}$, compute its k1- nearest neighbor $NN(x_i)$ using Euclidean distance, and a point x_i is considered as noisy if $NN(x_i)$ includes only the majority class instances. The informative minority instance set, T_{imin} is defined as:

$$T_{imin} = T_{min} - \{x_i \in T_{min} \text{ and } NN(x_i) \text{ contains only the majority class instances}\} \tag{1}$$

3.2 Determine the Weights of the Informative Minority Instances

The instances that are selected in the previous sub-sections may not have equal importance to be used to generate the synthetic instances because some of them may have different discriminative capabilities in the classification process. These differences should be taken into consideration while performing oversampling. In this study, for each $x_i \in T_{imin}$, two factors: density and distance are considered for assigning weights to it. The density factor of x_i is computed by determining the proportion of majority instances present in its k2-nearest neighbors set, i.e.

$$C(x_i) = \frac{\left|NN_{maj}(x_i)\right|}{k2} \tag{2}$$

where $\left|NN_{maj}(x_i)\right|$ defines the number of majority instances in $NN(x_i)$.

The distance factor of x_i is computed by taking the ratio of the sum of its distances to the majority class instances in its nearest neighbor set to the sum of its distances to all the instances in its nearest neighbor set, i.e.

$$D(x_i) = \frac{\sum_{x_j \in NN_{maj}(x_i)} dist(x_i, x_j)}{\sum_{x_j \in NN(x_i)} dist(x_i, x_j)} \tag{3}$$

where $dist(x_i, x_j)$ defines the Euclidean distance between x_i and x_j.

Based on these two factors the weights are assigned to each informative minority instance. For each $x_i \in T_{imin}$ the weight is defined as:

$$W(x_i) = C(x_i) + D(x_i) \tag{4}$$

These weights are used in the next phase to generate the synthetic instances.

3.3 The Synthetic Instance Generation Phase

In this phase, the informative minority instances T_{imin} are sequentially arranged based on their weights in decreasing order. Then, the minority instances are partitioned into two halves, such that the first half includes the instances having weights greater than or equal to that of the middle instance, while the second part comprises the remaining instances. The two partitions are defined as:

$$\text{Partition 1} = (x_1, x_2, \cdots, x_{n/2}) \text{ and Partition 2} = (x_{n/2+1}, x_{n/2+2}, \cdots, x_n) \tag{5}$$

where $x_i \in T_{imin}$ and $n = |T_{imin}|$

The instances within the two groups are sequentially labeled, i.e.

$$\begin{aligned} &\text{partition 1}\{\text{label}(\mathrm{x_i}) = \mathrm{l_i},\ \text{for i} = 1, 2, \cdots \mathrm{n}/2\} \text{ and} \\ &\textit{partition 2}\left\{\textit{label}(x_i) = l_{i-n/2}, \textit{for } i = n/2 + 1, n/2 + 2, \cdots n\right\} \end{aligned} \tag{6}$$

Two instances x_a and x_b having the same label l_i are picked from partition1 and partition2, respectively. The new instance is generated by computing the mean of x_a and x_b, i.e.

$$\mathrm{x_{new}} = (\mathrm{x_a} + \mathrm{x_b})/2 \tag{7}$$

The same process is repeated for the remaining instances in sequential order and to each new instance, the same label is assigned as that of their parents. The new minority instances are added to the informative minority instance set, which can be further used to generate the new instances. Algorithm 2 presents the descriptions of the synthetic instance generation phase.

Figure 1 shows the detailed procedure of the synthetic instance generation of minority instances. Node M_1 and M_2 represents the partitions of the borderline minority instances and C_{li} represents the set of synthetic instances generated at each level. In level 0, the synthetic minority instance set C_{01} are generated by sequentially pairing the instances of M_1 and M_2. In the next level, new instance sets C_{11} and C_{12} are generated by first pairing the instances from the set M_1 and C_{01} and later from the set M_2 and C_{01}, respectively. Consequently, in each level, new instances are generated by pairing the instances of the previous two levels. If each set contains n instances, then in each level l_i it can generate maximum $2^{l_i} \times \mathrm{n}$ instances and depending on the imbalances ratio more instances can be generated by pairing the newly generated instances with the previous level instances and the process is repeated until the number of borderline minority instances becomes equal to the number of majority instances. For example in level 2, two sets C_{22} and C_{23} are generated by respectively pairing (C_{01}, C_{11}) and (C_{01}, C_{12}) sets, and if required more instances can be generated by pairing (C_{12}, M_1) and (C_{11}, M_2) sets.

This method of oversampling ensures that the new instances do not overlap the majority class region. Additionally, the new instances must reside within the boundary of the minority class and also helps to fill the space between the paired instances. The proposed oversampling method generates distinct and unique instances but is related to the paired instances of both partitions, unlike SMOTE algorithm, which oversamples the minority instances by using k-nearest neighbors and the new instances may overlap with majority instances if the neighbors are not closely located. Another advantage of the proposed method over the k-nearest neighbor methods is that the latter method tends to generate synthetic instances in a cluster within the minority class instances and may increase the length of the minority class boundary, thus unable to provide more

informative information to the classifier, but the former method generates new instances that are evenly distributed within the convex hull of the minority class instances, thus helps to train the classifier more effectively.

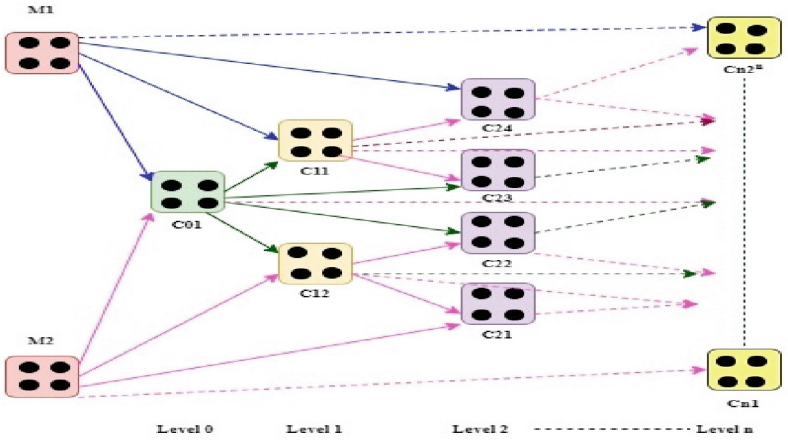

Fig. 1. Illustrates the synthetic instance generation process by pairing the previous levels samples

Algorithm 1: The proposed NOSTE algorithm

Input: Imbalanced training set, $T = T_{maj} \cup T_{min}, k1, k2$

Output: Balanced training set, T_{bal}

Procedure Begin

1. Select the informative minority instances, T_{imin} using Eq.1
2. for each $x_i \in T_{imin}$ do
3. Compute the density factor, $C(x_i)$ using Eq.2
4. Compute the distance factor, $D(x_i)$ using Eq.3
5. Compute the weights, $W(x_i)$ using Eq.4
6. end for
7. Generate synthetic minority instances, T_{gmin} using **Algorithm 2**
8. $T_{bal} = T_{maj} \cup T_{gmin}$
9. return T_{bal}

End

Algorithm 2: Generation of synthetic minority instances

Input: $W(\cdot)$ and T_{imin}

Output: Synthetic minority instances, T_{gmin}

Procedure Begin

1. Sort the weights of each n informative minority instance in decreasing order.
2. Find the middle instance, $mid = n/2$
3. Divide T_{imin} into partition1 and partition2 using Eq.5
4. To each instance of both the partitions, sequentially unique labels are assigned using Eq. 6.
5. Determine the oversampling size, $T = |T_{maj}| - |T_{imin}|$
6. $C = 0$
7. $T_{new} = \{\}$
8. for $i = 1, \cdots n$ do
9. Select x_a and x_b from parition1 and partition2, respectively, such that $label(x_a) = label(x_b)$.
10. Generate new minority instance x using Eq.7
11. $T_{new} = T_{new} \cup \{x\}$
12. $C = C + 1$
    end for
13. If $C < T$, pair the instances of the new set T_{new} with the instances of both the partitions and repeat steps 8-12. If still $C < T$, then pair the instances of the current T_{new} with the immediate previous level and later with the predecessor levels and for each subsequent level repeat steps 8- 12.
14. If $C \geq T$, then $T_{gmin} = T_{imin} \cup \{T_{new}\}$
15. return T_{gmin}

End

4 Experimental Study

The main objective of our work is to improve the performance level of the learning algorithms on imbalanced datasets by implementing the proposed oversampling method. To validate the effectiveness of NOSTE, a comparative analysis is performed with different popular resampling methods, such as SMOTE, BSMOTE, ADASYN, Random oversampling (ROS), and no oversampling (NONE).

4.1 Datasets

For experimental comparisons, three credit-scoring datasets are used. Table 1 presents brief descriptions of the datasets. The imbalanced credit scoring datasets are collected from two sources. German and Australian credit datasets are sourced from the UCI repository and the Giveme credit dataset is obtained from the Kaggle competition named "Give me some credits".

Table 1. Brief descriptions of credit scoring datasets

Datasets	No. of instances	Imbalanced Ratio
German	1000	2.33
Australian	690	1.25
Giveme	150000	13.9

4.2 Experimental Design

Classifier
In the experiment, four classification algorithms were implemented to show the validity of the results, which are CART Decision Tree (DT), Naïve Bayes (NB), Support Vector Machine (SVM), and Logistic regression (LR). All the algorithms with their default parameters were implemented using the Scikit-learn. The experiments were conducted using 5-fold cross-validation and the process is repeated three times, a total of 15 experiments were conducted and the final result is obtained by averaging each of them.

Performance Metrics
In the experiment, F-measure and AUC metrics are used to evaluate the performance of the methods. These two metrics are widely used in the field of binary imbalanced datasets [13]. The performance metrics are defined using four types of classifications, i.e. true positive (TP), true negative (TN), false positive (FP), and false negative (FN). For the definition of these metrics refer to the paper [13].

5 Results Analysis

The comparison of NOSTE with other resampling methods is discussed in this section. The average values of F-measure and AUC over all three credit scoring datasets and implementing all four classification algorithms are presented in Table 2 and Table 3. The results and the ranks within the parenthesis of all four resampling methods and without resampling are presented. Additionally, the mean ranks of each method are shown in the last row. The bold fonts indicate the method that performs best for a particular dataset. The results exhibit that NOSTE performs best values in terms of both F-measure and AUC with an average rank of 1.33 and 1.75, respectively.

Table 2. Performance of F-measure for different resampling methods

Dataset	Classification Models	NONE	ROS	SMOTE	ADASYN	BSMOTE	NOSTE
German	LR	58.41(6)	73.64(2)	72.84(4)	72.39(5)	73.62(3)	**74.54(1)**
	SVM	48.27(6)	63.57(4)	65.27(2)	64.86(3)	63.49(5)	**66.71(1)**
	DT	51.12(5)	**53.33(1)**	52.48(3)	48.61(6)	52.77(2)	52.28(4)
	NB	69.38(2)	67.87(3)	65.08(5)	64.12(6)	65.11(4)	**70.24(1)**
Australian	LR	83.48(3)	82.76(5)	83.33(4)	84.29(2)	82.70(6)	**85.64(1)**
	SVM	80.89(3)	79.93(6)	80.76(4)	81.32(2)	80.33(5)	**82.75(1)**
	DT	77.47(5)	78.84(4)	**83.56(1)**	78.99(3)	75.64(6)	79.84(2)
	NB	74.51(3)	73.26(6)	74.88(2)	73.52(4)	73.33(5)	**75.89(1)**
Giveme	LR	20.76(6)	35.82(3)	35.46(4)	30.75(5)	36.51(2)	**37.24(1)**
	SVM	17.12(6)	33.52(4)	34.13(3)	32.72(5)	35.17(2)	**34.58(1)**
	DT	23.01(6)	24.89(3)	24.12(4)	23.18(5)	25.91(2)	**32.25(1)**
	NB	34.43(3)	36.85(2)	24.72(4)	19.36(5)	17.85(6)	**42.25(1)**
Mean Ranking		4.5	3.58	3.33	4.25	4.00	**1.33**

Table 3. Performance of AUC for different resampling methods

Dataset	Classification Models	NONE	ROS	SMOTE	ADASYN	BSMOTE	NOSTE
German	LR	70.09(6)	74.86(4)	78.59(2)	78.38(3)	73.62(5)	**80.12(1)**
	SVM	65.33(6)	70.52(5)	72.74(2)	72.25(3)	71.52(4)	**73.58(1)**
	DT	62.47(4)	64.41(2)	63.14(3)	48.61(6)	52.77(5)	**66.24(1)**
	NB	72.22(5)	70.23(6)	77.53(2)	74.24(3)	73.42(4)	**78.27(1)**
Australian	LR	85.75(3)	84.64(5)	85.76(2)	**86.48(1)**	83.51(6)	85.47(4)
	SVM	83.41(4)	82.01(6)	83.68(2)	83.51(3)	82.86(5)	**84.78(1)**
	DT	80.89(5)	83.29(2)	82.55(3)	81.76(4)	78.79(6)	**85.26(1)**
	NB	**79.01(1)**	78.14(5)	78.05(6)	78.34(4)	78.91(2)	78.59(3)
Giveme	LR	56.01(6)	74.19(3)	74.51(2)	73.91(4)	73.84(5)	**75.89(1)**
	SVM	54.75(6)	**74.75(1)**	74.44(2)	74.12(4)	74.39(3)	73.28(5)
	DT	62.16(5)	59.94(6)	63.39(2)	62.38(3)	62.23(4)	**64.74(1)**
	NB	69.42(6)	70.65(3)	71.26(2)	69.45(4)	68.38(5)	**74.56(1)**
Mean Ranking		4.75	4	2.5	3.5	4.5	**1.75**

6 Conclusion and Future Work

Analyzing the drawbacks of the SMOTE-based oversampling techniques, a novel oversampling technique NOSTE is proposed to handle the highly-imbalanced credit scoring datasets. The main advantages of the proposed method as compared to the SMOTE-based methods are that: 1) using k-nearest neighbors algorithms the noisy samples are removed from the minority subset, 2) generate diverged and unique minority instances that are evenly distributed within the convex hull of the minority instances, and 3) new minority instances do not overlap the majority class region.

In the experiments, we compare the performance of NOSTE with other popular oversampling techniques. The obtained results show that NOSTE has better ranks than other methods in terms of F_measure and AUC.

In the future study, we intend to apply fuzzy c-means clustering and select the representative minority instances from each sub-cluster to be used in the oversampling process. We are also interested to implement our work in multi-class problems.

References

1. Luo, C., Wu, D., Wu, D.: A deep learning approach for credit scoring using credit default swaps. Eng. Appl. Artif. Intell. **65**(December), 465–470 (2017). https://doi.org/10.1016/j.engappai.2016.12.002
2. Sun, J., Lang, J., Fujita, H., Li, H.: Imbalanced enterprise credit evaluation with DTE-SBD: decision tree ensemble based on SMOTE and bagging with differentiated sampling rates. Inf. Sci. (N Y) **425**, 76–91 (2018). https://doi.org/10.1016/j.ins.2017.10.017
3. Brown, I., Mues, C.: An experimental comparison of classification algorithms for imbalanced credit scoring data sets. Expert Syst. Appl. **39**(3), 3446–3453 (2012). https://doi.org/10.1016/j.eswa.2011.09.033
4. Xiao, J., Cao, H., Jiang, X., Gu, X., Xie, L.: GMDH-based semi-supervised feature selection for customer classification. Knowl. Based Syst. **132**, 236–248 (2017). https://doi.org/10.1016/j.knosys.2017.06.018
5. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: synthetic minority over-sampling technique. J. Artif. Intell. Res. **16**, 321–357 (2002). https://doi.org/10.1613/jair.953
6. He, H., Bai, Y., Garcia, E.A., Li, S.: ADASYN: adaptive synthetic sampling approach for imbalanced learning. In: Proceedings of the International Joint Conference on Neural Networks, no. 3, pp. 1322–1328 (2008). https://doi.org/10.1109/IJCNN.2008.4633969
7. Han, H., Wang, W.-Y., Mao, B.-H.: Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning. In: Huang, D.-S., Zhang, X.-P., Huang, G.-B. (eds.) ICIC 2005. LNCS, vol. 3644, pp. 878–887. Springer, Heidelberg (2005). https://doi.org/10.1007/11538059_91
8. Barua, S., Islam, M.M., Yao, X., Murase, K.: MWMOTE - majority weighted minority oversampling technique for imbalanced data set learning. IEEE Trans. Knowl. Data Eng. **26**(2), 405–425 (2014). https://doi.org/10.1109/TKDE.2012.232
9. Bunkhumpornpat, C., Sinapiromsaran, K., Lursinsap, C.: Safe-Level-SMOTE: safe-level-synthetic minority over-sampling TEchnique for handling the class imbalanced problem. In: Theeramunkong, T., Kijsirikul, B., Cercone, N., Ho, T.-B. (eds.) PAKDD 2009. LNCS (LNAI), vol. 5476, pp. 475–482. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01307-2_43

10. Douzas, G., Bacao, F.: Self-organizing map oversampling (SOMO) for imbalanced data set learning. Expert Syst. Appl. **82**, 40–52 (2017). https://doi.org/10.1016/j.eswa.2017.03.073
11. Lin, W.C., Tsai, C.F., Hu, Y.H., Jhang, J.S.: Clustering-based undersampling in class-imbalanced data. Inf. Sci. (N Y) **409–410**, 17–26 (2017). https://doi.org/10.1016/j.ins.2017.05.008
12. Krawczyk, B., Galar, M., Jeleń, Ł, Herrera, F.: Evolutionary undersampling boosting for imbalanced classification of breast cancer malignancy. Appl. Soft Comput. J. **38**, 714–726 (2016). https://doi.org/10.1016/j.asoc.2015.08.060
13. He, H., Garcia, E.A.: Learning from imbalanced data. IEEE Trans. Knowl. Data Eng. **21**(9), 1263–1284 (2009). https://doi.org/10.1109/TKDE.2008.239
14. Nekooeimehr, I., Lai-Yuen, S.K.: Adaptive semi-unsupervised weighted oversampling (A-SUWO) for imbalanced datasets. Expert Syst. Appl. **46**, 405–416 (2016). https://doi.org/10.1016/j.eswa.2015.10.031
15. Bennin, K.E., Keung, J., Phannachitta, P., Monden, A., Mensah, S.: MAHAKIL: diversity based oversampling approach to alleviate the class imbalance issue in software defect prediction. IEEE Trans. Software Eng. **44**(6), 534–550 (2018). https://doi.org/10.1109/TSE.2017.2731766