electronics

MDPI

Article

How Far Have We Progressed in the Sampling Methods for Imbalanced Data Classification? An Empirical Study

Zhongbin Sun [1,2,*], Jingqi Zhang [3], Xiaoyan Zhu [3] and Donghong Xu [1,2]

[1] Mine Digitization Engineering Research Center of Ministry of Education, Xuzhou 221116, China; xudh123@cumt.edu.cn
[2] School of Computer Science and Technology, China University of Mining and Technology, Xuzhou 221116, China
[3] School of Computer Science and Technology, Xi'an Jiaotong University, Xi'an 710049, China; q871052938@stu.xjtu.edu.cn (J.Z.); zhu.xy@xjtu.edu.cn (X.Z.)
* Correspondence: zhongbin@cumt.edu.cn

Abstract: Imbalanced data are ubiquitous in many real-world applications, and they have drawn a significant amount of attention in the field of data mining. A variety of methods have been proposed for imbalanced data classification, and data sampling methods are more prevalent due to their independence from classification algorithms. However, due to the increasing number of sampling methods, there is no consensus about which sampling method performs best, and contradictory conclusions have been obtained. Therefore, in the present study, we conducted an extensive comparison of 16 different sampling methods with four popular classification algorithms, using 75 imbalanced binary datasets from several different application domains. In addition, four widely-used measures were employed to evaluate the corresponding classification performance. The experimental results showed that none of the employed sampling methods performed the best and stably across all the used classification algorithms and evaluation measures. Furthermore, we also found that the performance of the different sampling methods was usually affected by the classification algorithms employed. Therefore, it is important for practitioners and researchers to simultaneously select appropriate sampling methods and classification algorithms, for handling the imbalanced data problems at hand.

Keywords: imbalanced data; sampling methods; classification

Citation: Sun, Z.; Zhang, J.; Zhu, X.; Xu, D. How Far Have We Progressed in the Sampling Methods for Imbalanced Data Classification? An Empirical Study. *Electronics* **2023**, *12*, 4232. https://doi.org/10.3390/electronics12204232

Academic Editors: Agnieszka Konys and Agnieszka Nowak-Brzezińska

Received: 22 September 2023
Revised: 10 October 2023
Accepted: 11 October 2023
Published: 13 October 2023

1. Introduction

Over the last two decades, imbalanced data classification has drawn wide-spread attention in both the academic and industry communities [1,2]. To date, a large number of real-world classification tasks have been reported to suffer from imbalanced data, such as customer churn prediction [3], software defect prediction [4], wafer defect pattern identification [5] and financial fraud detection [6]. In particular, imbalanced data refers to a dataset with one or some of the classes having a much larger number of instances than others, and in which the rarest class is usually called the minority class. Many traditional classification algorithms often show poor performance on imbalanced data, as they ignore the interesting minority class, due to maximizing the overall classification accuracy [7].

Many different kinds of method have been proposed to tackle the class imbalance problem, such as data sampling methods [8,9], cost-sensitive learning [10,11], and ensemble learning [12,13]. Among these, sampling methods are more popular and prevalent, due to their independence from classification algorithms [14]. Undersampling and oversampling are two different strategies of sampling methods. In particular, undersampling focuses on eliminating majority instances, and due to a reduction in training samples, the model construction efficiency can be improved. However, oversampling allows increasing the

minority instances to rebalance the imbalanced data, and the model construction efficiency may decrease with more training samples.

However, there is not a consensus in the research community about which is the best sampling method [15,16], as each sampling method has pros and cons. As stated by Loyola et al. [16], no sampling methods can clearly obtain the best results, as there are certain factors which do not allow us to answer this question, including vastly different intrinsic data characteristics and various application domains. We argue that contradictory conclusions on the dominance of different sampling methods may lead to the use of discrepant experimental setups, including various imbalanced datasets, diverse classification algorithms, and different performance measures. Therefore, in the present study, we intend to provide a comprehensive comparison of different sampling methods under the circumstance of a large number of imbalanced datasets, as well as several widely-used classification algorithms and performance metrics.

To be specific, 16 different sampling methods, including seven oversampling methods and nine undersampling methods, have been carefully selected with four different criteria for the experimental study. In addition, 75 original binary datasets from four different application domains are used for the comparison study, including from physical sciences, life sciences, business, and software engineering. Moreover, we use four popular classification algorithms (Naive Bayes, KNN, C4.5, and Random Forest) and four widely-used performance measures (TPR, AUC, F-Measure, and G-Mean) in the experiment. The following three research questions are studied in detail:

RQ1: How effective are the employed sampling methods?

Most of the employed sampling methods can improve the classification performance of a naive classification algorithm, and thus they are effective in dealing with imbalanced data. In addition, the newly proposed and complex sampling methods may not perform better than the simple and classical sampling methods.

RQ2: How stable are the employed sampling methods?

None of the 16 employed sampling methods show stable classification results across all four classification algorithms and four evaluation measures used, which indicates that the selection of baseline sampling methods for validating the performance of new sampling methods is rather difficult.

RQ3: Is the performance of sampling methods affected by the learners?

Different sampling methods may behave very differently when employing different classification algorithms for building classification models. Therefore, we suggest that the sampling methods and classification algorithms should be taken into consideration simultaneously when handling the imbalanced data.

The main contributions of our work are summarized as follows: (1) We provide a comprehensive comparison of 16 sampling methods on 75 original imbalanced datasets, using four popular classification algorithms and four widely-used performance measures. (2) The experimental results demonstrated the effectiveness yet instability of the employed sampling methods. Furthermore, different sampling methods may behave very differently with different classification algorithms. (3) We advocate that researchers pay more attention to the selection of appropriate sampling methods for imbalanced data, while considering the classification algorithms used.

The reminder of this paper is organized as follows: Section 2 provides a detailed review of different sampling methods. The sampling methods employed in present study and the specific design of our experiment are described in Section 3. In Section 4, we introduce the corresponding experimental results and discussions. Finally, a brief summary of the present study and future work is presented in Section 5.

2. Related Work

In the field of machine learning, traditional classification algorithms usually assume that the training data employed to build classification models are balanced. However, this

is not always the case in practical applications, where one class might be represented by a large number of examples, while the other is represented by only a few [17–19], which is known as the imbalanced data classification problem [1,2]. To date, a variety of studies have focused on proposing a number of different methods to solve the class imbalance problem, such as sampling methods [16,20], cost-sensitive learning [21,22], and ensemble learning methods [12,23]. Among these, sampling methods are the more prevalent, as they are independent of the classification algorithms employed [14]. To be specific, sampling methods alter the distribution of the original data, either by increasing the minority class instances (oversampling) or by eliminating majority class instances (undersampling).

Oversampling methods balance the training data distribution by increasing the number of minority class instances [24–27]. As the most frequently used oversampling method, random oversampling (ROS) increases the minority class instances by random duplication of existing minority class instances. However, ROS usually suffers from overfitting, due to massive replication, and Chawla et al. [28] proposed a popular oversampling approach, SMOTE, for avoiding the overfitting problem. In SMOTE, minority class instances are created based on the neighbor instances, rather than copying existing instances. However, there are still some disadvantages with SMOTE and a number of SMOTE variations have been proposed in the past decade [29]. For example, focusing on the decision boundary, borderline-SMOTE (BSMOTE) was proposed to make an improvement of SMOTE by oversampling the minority class instances near the borderline [30]. He et al. [31] focused on the hard to classify minority instances and presented an adaptive synthetic sampling approach (ADASYN). In addition, MWMOTE was proposed by Sukarna et al. [32], which identifies hard-to-learn informative minority class samples and assigns corresponding weights, according to the nearest majority class samples. Recently, the k-means clustering method was combined with SMOTE (KSMOTE [9]) to avoid the generation of noise instances, which effectively overcame the imbalance between and within classes. In addition, there are also other oversampling methods focusing on solving the class imbalance problem. MAHAKIL [33] was proposed, to employ the chromosomal theory of inheritance to alleviate the class imbalance issue in software defect prediction. RACOG is a probabilistic oversampling technique, combining the joint probability distribution and Gibbs sampling [24]. Yan et al. [34] proposed a novel optimal transport-based oversampling method. In light of expanding class boundaries possibly influencing classification performance, RWO-Sampling [35] creates synthetic samples through randomly walking from the real data.

Undersampling methods rebalance imbalanced data by removing majority class instances, until the desired ratio is obtained [36–39], which can suffer from the problem of discarding potentially useful data. Random undersampling (RUS) is the most simple and representative undersampling approach, randomly eliminating instances from the majority class. Two popular data cleaning techniques, including condensed nearest neighbor (CNN) [40] and edited nearest neighbor (ENN) [41] have been used to remove majority instances by employing a nearest-neighbor-based classification algorithm. Moreover, Laurikkala [42] proposed the method of the neighborhood cleaning rule (NCL) by modifying ENN. Kubat et al. [43] propose one-sided selection (OSS), to undersample datasets by combining CNN and Tomek links [44]. Recently, clustering methods combined with undersampling have received attention from researchers [36,45,46]. In particular, a specific clustering method was first applied to different data, such as the total training data [45], the minority class instances [36], and the majority class instances [37,46]. Then, based on the results of clustering, different strategies have been adopted to delete the majority class instances to obtain relatively balanced training data. Furthermore, there are also other kinds of undersampling methods based on different learning algorithms, such as evolutionary algorithms [47], metric learning [48], and reinforcement learning [38].

A variety of studies have been conducted to compare the classification performance of different sampling methods, and some prior studies [15,49–52] showed inconsistent results regarding which kind of sampling method is best. Batista et al. [8] studied the behavior of

several sampling methods, and their experimental results showed that oversampling methods provided a better classification performance than undersampling methods in terms of AUC. What is more, they found that ROS was very competitive for more complex oversampling methods, such as SMOTE. Moreover, Hulse et al. [53] also experimentally determined that ROS and RUS usually perform better than the complex sampling methods, by using 11 learning algorithms with 35 real-world benchmark datasets. However, based on the contrast pattern classifier, SMOTE was the best oversampling method, while Tomek was the best undersampling method [16]. Furthermore, Bennin et al. [54] concluded that RUS and BSMOTE were more stable across several performance measures and prediction models.

To summarize, we found that the performance of different sampling methods in the existing studies has not reached a clear consensus. The use of different datasets may have resulted in such contradictory conclusions, as different sampling methods may be more suitable according to the internal characteristics of real-world datasets [55,56]. Furthermore, the classification algorithms and performance measures employed are also two important factors that may have had a great impact on the classification performance of the different sampling methods [3,57,58]. Such contradictory conclusions make it hard to derive practical guidelines for whether, and which, class rebalancing techniques should be applied in the context of different application domains. Therefore, in the present study, we performed a more extensive experimental comparison of sampling methods, with the use of 75 imbalanced binary datasets from four different application domains. To be specific, 16 popular sampling methods, four well-known classification algorithms, and four widely-used performance measures were investigated in the present study.

3. Experimental Study

In this section, the overall experimental design is provided, including the evaluated sampling methods, employed datasets, evaluation measures, experimental setup, and research questions. In particular, Sections 3.1 and 3.2, respectively, give a short overview of the employed sampling methods and experimental datasets. Then, we present some evaluation measures in Section 3.3, which are typically employed to assess the performance of sampling methods with a specific classification algorithm. Finally, the experimental setup and research questions are separately provided in detail in Sections 3.4 and 3.5.

3.1. Sampling Methods

In the present study, we employed the following criteria to select the sampling methods: (i) The simplest sampling methods, namely ROS and RUS; (ii) The classical sampling methods that have been widely used as baseline methods for comparison [8,16,54], such as SMOTE [28] and OSS [43]; (iii) The sampling methods that are proposed in recent years, yet with a certain number of citations, such as KSMOTE [9] and CentersNN [46].

Therefore, a total of 16 popular sampling methods were selected for conducting the following experiment, including seven oversampling methods and nine undersampling methods. The selected sampling methods are briefly introduced in the following. For more details about these sampling methods, please refer to the corresponding literature.

3.1.1. Oversampling Methods

Oversampling methods balance skewed datasets by increasing minority instances with a specific strategy. There are seven common oversampling methods in the present study, and a description of these methods is given below.

- ROS: ROS increases the number of minority instances by randomly replicating minority instances. In particular, for a given dataset, a instance from the minority class is randomly selected and then a copy of the selected instance is added to the dataset. By these means, the originally skewed dataset is balanced to the desired level with ROS;
- SMOTE [28]: SMOTE is a method for generating new minority class instances based on k-nearest neighbors, which aims to solve the overfitting problem in ROS. In particular,

one minority class instance is randomly selected and corresponding k nearest neighbor instances with the same class label are found. Then, the synthetic instance is randomly generated along the line segment, which joins the selected instance and its individual neighbor instance;

- BSMOTE [30]: BSMOTE makes a modification of SMOTE by focusing on the dangerous and borderline instances. It first finds the dangerous minority class instances that have more majority class instances as neighbors than minority class neighbors. The dangerous minority class instances are regarded as the borderline minority instances, and then they are fed into the SMOTE method for generating synthetic minority instances in the neighborhood of the borderline minority instances;
- ADASYN [31]: ADASYN is another modification of the SMOTE method and focuses on the hard to classify minority class instances. First, the learning difficulty of each minority instance is calculated as the ratio of instances belonging to the majority class in its neighborhood. Then, ADASYN assigns weights to the minority class instances, according to their level of learning difficulty. Finally, the weight distribution is employed to automatically generate the number of synthetic instances that need to be created with SMOTE method for all minority data;
- SLSMOTE [59]: Based on SMOTE, SLSMOTE assigns each minority class instance a safe level, before generating synthetic instances, which are calculated based on the number of minority class instances in the corresponding k nearest neighbor instances. In SLSMOTE, all synthetic instances are generated in safe regions, as each synthetic instance is positioned closer to the largest safe level;
- MWMOTE [32]: Based on the weaknesses of SMOTE, BSMOTE and ADASYN, MWMOTE first identifies the hard-to-learn informative minority class instances, which are then assigned weights according to their Euclidean distance from the nearest majority class instance. Then, a clustering method is employed to cluster these weighted informative minority class instances, and finally new minority class instances are generated inside each individual cluster;
- KSMOTE [9]: KSMOTE first clusters the data with k-means clustering method and then picks out the clusters with a number of minority class instances greater than majority class instances.For each selected cluster, the number of generated instances is obtained according to the sampling weight computed based on its minority density, and SMOTE is then applied to achieve the target ratio of minority and majority instances.

3.1.2. Undersampling Methods

Undersampling methods aim to decrease the majority instances to balance skewed datasets. The nine selected undersampling methods are briefly introduced as follows:

- RUS: RUS decreases the number of majority instances in a given dataset by deleting majority instances at random; namely, randomly selecting a majority class instances and removing them until the given dataset reaches the specified imbalance ratio;
- CNN [40]: CNN aims to find a subset that can correctly classify the original dataset using the 1-nearest neighbor method. For the purpose of deleting majority instances, all minority instances and a randomly selected majority instance are merged into an initial subset. Then, other majority instances are classified based on the 1-nearest neighbor method with the initial subset. The misclassified majority instances are put into a subset for obtaining the final training data;
- ENN [41]: ENN is a kind of data cleaning technique that can be used as a undersampling method [60]. In ENN, the KNN technique with $K = 3$ is first employed to classify each majority class instance in the original data using all the remaining instances. Finally, those misclassified majority class instances are removed from the original data and all residual instances are regarded training data;
- Tomek [44]: Tomek is also a data cleaning technique employed as a undersampling method. When the distance between a majority instance and a minority instance is smaller than the distances between any other instances and each of the two instances,

such a pair of instances is called a tomek link.When employing Tomek as a undersampling method, the majority instances belonging to all tomek links are deleted, to rebalance the original data;

- OSS [43]: For the purpose of removing redundant and noisy majority class instances, OSS combines the CNN and Tomek methods. To be specific, OSS first employs a CNN to remove redundant majority class instances and thus obtains a subset that can represent the original data. Then, the Tomek method is applied to the obtained subset, to delete the noisy majority class instances;
- NCL [42]: NCL makes improvements on ENN, as it deals with not only the majority class instances but also the minority class instances. In particular, for a majority class instance, it will be deleted from the original data if most of its neighbors are minority class instances (namely ENN). However, for a minority class instance, all the its neighbors belonging to the majority class are removed;
- NearMiss2 [61]: NearMiss2 applies the simple KNN approach to resolve the imbalanced class distribution, aiming to pick out majority class instances that are close to all minority class instances. In particular, in this method, majority class instances are selected based on their average distance to the three farthest minority class instances;
- USBC [45]: USBC first clusters all the training instances into certain clusters using the k-means clustering method. Based on the idea that each cluster seems to have distinct characteristics, USBC selects a suitable number of majority class instances from each cluster by considering the imbalanced ratio of the corresponding cluster. In other words, a cluster with more majority class instances will be sampled more;
- CentersNN [46]: CentersNN is another undersampling method based on the clustering technique. Differently from USBC, CentersNN uses the k-means method for the majority class instances, in which the number of clusters is set as the number of the minority class. Finally, the nearest neighbors of all the cluster centers are picked out to combine with all the minority class instances as the final training data.

3.2. Datasets

In the experiment, 75 imbalanced binary datasets from four different application domains were used, including physical sciences, business, life sciences, and software engineering. Differently from the previous studies [14,16,37] that used artificial datasets created from multi-class data (e.g., KEEL datasets [62]), all the datasets employed were originally binary. Table 1 provides a statistical summary of the datasets from the UCI repository employed, including the number of minority instances (# Minority), the number of total instances (# Total), and the imbalance ratio (IR) (IR = $\frac{The\ number\ of\ majority\ instances}{The\ number\ of\ minority\ instances}$).

Table 1. Statistical summary of the employed datasets.

ID	Data	#Min	#Total	IR	ID	Data	#Min	#Total	IR
1	cylinder	228	540	1.37	39	PC3	132	1073	7.13
2	htru	1639	17,898	9.92	40	PC4	176	1276	6.25
3	ionosphere	126	351	1.79	41	PC5	459	1679	2.66
4	ozone1hr	73	2536	33.74	42	ant1.7	166	724	3.36
5	ozone8hr	160	2534	14.84	43	arc	25	213	7.52
6	sonar	97	208	1.14	44	camel1.0	13	327	24.15
7	bankruptcy1	271	7027	24.93	45	camel1.6	181	878	3.85
8	bankruptcy2	400	10,173	24.43	46	ivy2.0	40	345	7.63
9	bankruptcy3	495	10,503	20.22	47	jedit3.2	90	268	1.98
10	bankruptcy4	515	9792	18.01	48	jedit4.2	48	363	6.56
11	bankruptcy5	410	5910	13.41	49	log4j1.1	37	109	1.95
12	credit-a	307	690	1.25	50	lucene2.0	90	191	1.12
13	creditC	6636	30,000	3.52	51	poi2.0	35	282	7.06
14	credit-g	300	1000	2.33	52	prop1	1536	8011	4.22
15	market	5289	45,211	7.55	53	prop6	32	377	10.78

Table 1. *Cont.*

ID	Data	#Min	#Total	IR	ID	Data	#Min	#Total	IR
16	alizadeh	87	303	2.48	54	redaktor	25	169	5.76
17	breast-c	85	286	2.36	55	synapse1.0	16	153	8.56
18	breast-w	241	699	1.9	56	synapse1.2	86	245	1.85
19	coimbra	52	116	1.23	57	tomcat	77	796	9.34
20	colic	136	368	1.71	58	velocity1.6	76	211	1.78
21	haberman	81	306	2.78	59	xalan2.4	110	694	5.31
22	heart	120	270	1.25	60	xercesInit	65	146	1.25
23	hepatitis	32	155	3.84	61	xerces1.3	68	362	4.32
24	ilpd	167	583	2.49	62	Clam_fit	93	1597	16.17
25	liver	145	345	1.38	63	Clam_test	56	8723	154.77
26	pima	268	768	1.87	64	eCos_fit	110	630	4.73
27	retinopathy	540	1151	1.13	65	eCos_test	67	3480	50.94
28	sick	231	3772	15.33	66	NetBSD_fit	546	6781	11.42
29	thoracic	70	470	5.71	67	NetBSD_test	295	10,960	36.15
30	CM1	42	327	6.79	68	OpenBSD_fit	275	1706	5.2
31	JM1	1612	7720	3.79	69	OpenBSD_test	158	5964	36.75
32	KC1	294	1162	2.95	70	OpenCms_fit	193	1727	7.95
33	KC3	36	194	4.39	71	OpenCms_test	93	2821	29.33
34	MC1	36	1847	50.31	72	Samba_fit	184	1623	7.82
35	MC2	44	125	1.84	73	Samba_test	223	2559	10.48
36	MW1	25	251	9.04	74	Scilab_fit	233	2636	10.31
37	PC1	55	696	11.65	75	Scilab_test	238	1248	4.24
38	PC2	16	734	44.88					

To be specific, 29 datasets from three domains of physical sciences (1–6), business (7–15) and life sciences (16–29) were freely extracted from the UCI repository [63]. In addition, 46 software engineering datasets were extracted from three different repositories for the task of software defect prediction [64–66], including NASA [67] (30–41), PROMISE [68] (42–61), and OSS [51] (62–75). The performance of software defect prediction has been hindered by the imbalanced nature of software defect data, in which the non-defective modules always outnumber the defective [69,70]. The main difference among the three repositories is that they gather different source code metrics as features in the final defect datasets.

3.3. Evaluation Measures

Different imbalanced data handling methods may show diverse behaviors with various evaluation measures [3], indicating that the performance of sampling methods is affected by the evaluation metrics employed.Guo et al. [2] reported the top-five most widely used evaluation measures for imbalanced data learning, including AUC, accuracy, G-Mean, F-Measure, and TPR (true positive rate). However, accuracy is ineffective in the imbalanced learning scenario, as it has been shown to be biased towards the majority class [1]. Therefore, TPR, AUC, F-Measure, and G-Mean were individually employed as the performance metrics in the present study. We obtained these four evaluation measures based on the confusion matrix shown in Table 2.

Table 2. Confusion Matrix.

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

In Table 2, TP (true positive) represents the number of positive instances correctly classified as positive and FP (false positive) is the number of negative instances wrongly classified as positive. Moreover, TN (true negative) stands for the number of negative instances correctly classified as negative and FN (false negative) is the number of positive

instances wrongly classified as negative. Based on the confusion matrix, TPR, AUC, F-Measure, and G-Mean were obtained as follows:

- TPR: TPR measures the proportion of correctly classified positive instances, which is also called recall or sensitivity. The equation for TPR is as follows:

$$TPR = \frac{TP}{TP + FN} \tag{1}$$

- AUC: AUC is obtained by calculating the area under the ROC curve, in which the horizontal axis is the FPR (false positive rate) and the vertical axis is the TPR. Furthermore, AUC ranges from 0 to 1 with a larger AUC value indicating a better classification performance;
- F-Measure: F-Measure computes the harmonic mean between precision and recall (namely TPR), and the equation of F-Measure is shown as follows:

$$F - Measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{2}$$

where

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

- G-Mean: G-Mean was proposed as a compromise between TPR and TNR (true negative rate), and the corresponding calculation is shown as follows:

$$G - Mean = \sqrt{TPR \times TNR} \tag{4}$$

where

$$TNR = \frac{TN}{TN + FP} \tag{5}$$

3.4. Experimental Setup

In the present study, an experimental study was conducted using a 10-fold cross-validation strategy. That is, each dataset was divided into ten folds, and each fold had a similar number of instances. Then, for each fold, a learning algorithm was trained on the remaining nine folds and then tested on the current fold. Furthermore, it should be noted that some of the sampling methods allowed resampling to any desired ratio, such as RUS and SMOTE. In the current study, we tried to obtain a level of balance in the training data near to a 50:50 distribution, as suggested in [54]. Moreover, we also evaluated the classification algorithms directly applied to the original binary data without any sampling methods, which will be represented as the "None" method in the following discussion.

In addition, four different types of classification algorithm were selected, including naive Bayes (NB), k-nearest neighbors (KNN), C4.5, and random forest (RF). In particular, NB is based on the Bayes theory, and KNN is a kind of lazy learning method. Moreover, C4.5 is an implementation of the decision tree algorithm, while RF is an ensemble learning method based on C4.5. These four classification algorithms were implemented in the open source machine learning tool Weka [71], which has been widely used in imbalanced data classification, as well as other data mining tasks. Note that classification performance can be affected by the hyperparameters of classification algorithms, which is beyond the scope of the present study; thus, the default hyperparameters in the Weka software were used in our present study for all the four classification algorithms, except KNN. Whereby, K was set to 5 for the KNN classification algorithm.

3.5. Research Questions

The present study concentrated on the following three questions:

RQ1: *How effective are the employed sampling methods?*

In previous studies, a conclusion of the sampling methods performing better than the "None" method may have been widely confirmed. However, we questioned whether such a conclusion may be limited by the number of employed datasets, classification algorithms, and evaluation measures. For example, Zhu et al. [3] empirically determined that the applied evaluation metrics had a great impact on the performance of techniques. Therefore, in the present study, we first researched the effectiveness of the employed sampling methods by comparing them with "None" across all the employed datasets, learners, and performance metrics.

RQ2: *How stable are the employed sampling methods?*

Each sampling method has its pros and cons. As stated by Loyola et al. [16], no sampling method can clearly obtain the best results, as there are some factors which do not allow us to answer this question, including vastly different intrinsic data characteristics and various application domains. Therefore, there is no consensus in the research community about which is the best sampling method for all datasets. However, some sampling methods may perform relatively stably and may be selected as baseline methods for future studies. Therefore, with this question we intended to investigate the stability of the employed sampling methods.

RQ3: *Is the performance of the sampling methods affected by the learners?*

Imbalanced datasets are often first preprocessed using sampling methods and then fed to different classification algorithms (learners) to build classification models. Sun et al. [4] empirically showed that different learners may benefit differently from their proposed method of handling imbalanced data. Therefore, in the present study, we focused on investigating whether the selection of different learners may have an impact on the sampling methods with a specific performance measure.

4. Results and Discussion

4.1. Results of RQ1

In this section, we investigated the effectiveness of the employed sampling methods by individually comparing them with the None method. In particular, for each classification algorithm, the percentage performance improvement of each sampling method over None was first calculated in terms of each evaluation measure employed. Then, a boxplot was used to present the distribution of the percentage improvement for the 75 imbalanced datasets studied. To be specific, Figures 1–4 show the corresponding boxplots of the percentage performance improvement for the classification algorithms NB, C4.5, KNN, and RF, respectively. Note that in each boxplot, for a specific sampling method, the percentage improvements of all the four evaluation metrics are summarized in an individual boxplot.

Figure 1 provides the boxplots of percentage performance improvement for the 16 sampling methods over None with the classification algorithm NB. From Figure 1, it can be observed that for 12 of the employed 16 sampling methods, the median was (i) larger than zero (SMOTE, BSMOTE, ADASYN, SLSMOTE, MWMOTE, RUS, ENN, NCL, and USBC) or (ii) closer to the lower quartile if the mean was nearly equal to 0 (ROS, Tomek, and CentersNN). This indicates that for these 12 sampling methods, they usually performed better than None on most of the employed datasets. However, for the remaining four sampling methods of KSMOTE, CNN, OSS, and NearMiss2, there were still some datasets on which the four sampling methods performed better than None.

The boxplots of the percentage performance improvement for sampling methods over None with C4.5 is demonstrated in Figure 2. We can observe from Figure 2 that, except KSMOTE, the 15 sampling methods usually performed better than None, as their corresponding median values are larger than zero. In addition, for KSMOTE, the median is close to zero, and this indicates that nearly half of the employed datasets may benefit from KSMOTE when using C4.5 as the classification algorithm.

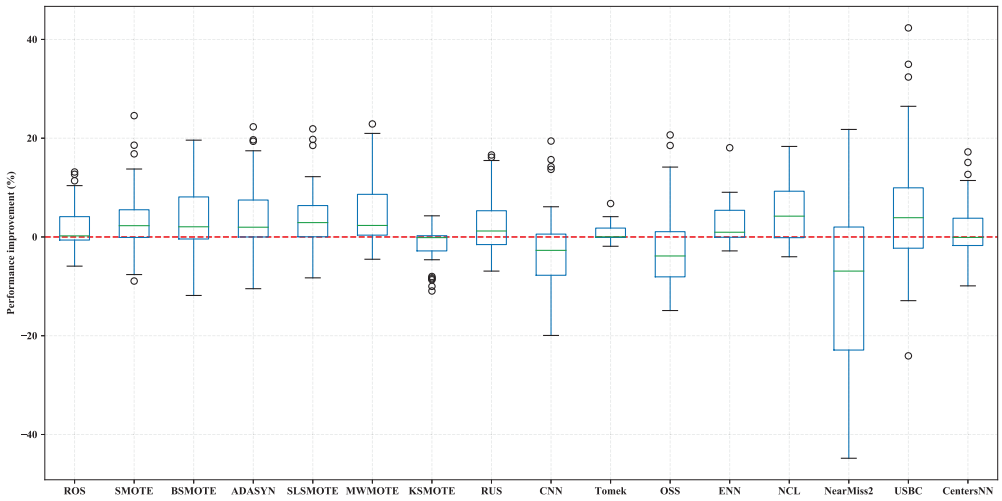

Figure 1. The percentage improvement of the sampling methods over None for NB.

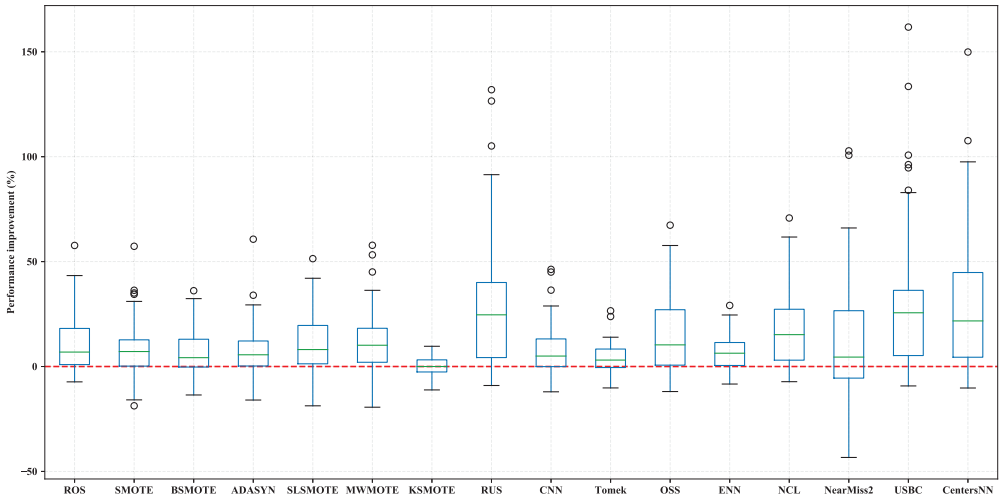

Figure 2. The percentage improvement of the sampling methods over None for C4.5.

We present boxplots of the percentage performance improvement for the sampling methods over None with the lazy learning algorithm KNN in Figure 3. To be specific, Figure 3 shows that among all the employed 16 sampling methods, 13 methods usually obtained a better classification performance than None, as their corresponding median values are larger than zero. For the remaining three sampling methods ROS, KSMOTE, and NearMiss2, KNN was affected little with ROS and KSMOTE, while NearMiss2 performed worse than None on most of the employed datasets when using KNN as the classification algorithm.

Figure 4 shows boxplots of the percentage performance improvement for all sampling methods over None with the ensemble learner RF. From Figure 4, it can be observed that all of the used sampling methods obtained median values larger than zero, which means that these 16 sampling methods usually performed better than None with RF as the classification algorithm.

To conclude, we found that (i) most of the employed sampling methods usually performed better than None on most of the employed datasets; (ii) the newly proposed and complex sampling methods may not perform better than the simple and classical sampling methods, such as KSMOTE [9] vs. SMOTE [28]. Therefore, these sampling methods are effective for dealing with imbalanced data, and for specific imbalanced data, we argue that the selection of appropriate sampling methods may be more practical than designing a new sampling method.

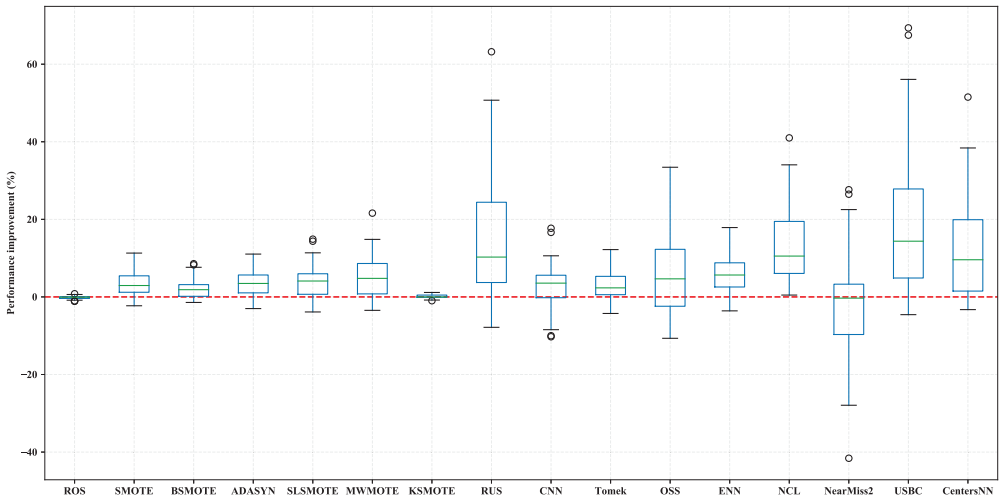

Figure 3. The percentage improvement of the sampling methods over None for KNN.

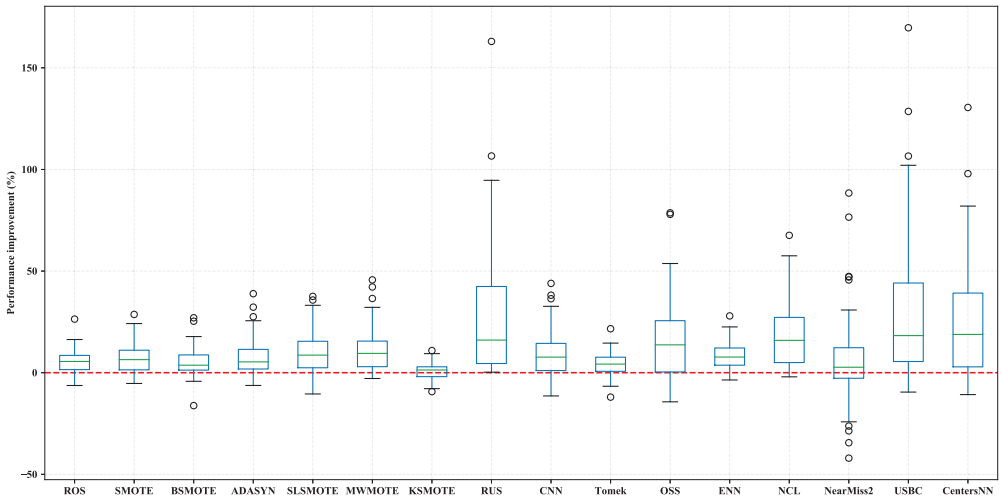

Figure 4. The percentage improvement of the sampling methods over None for RF.

4.2. Results of RQ2

In this section, we intend to provide a detailed analysis of the stability of the employed sampling methods when handling imbalanced data with various characteristics. To this end, for each of the employed datasets, the 16 sampling methods are first ranked according to their performance indicators within each individual classification algorithm. Then, the average rank and variance of rank are calculated, respectively. Finally, for each employed classification algorithm and performance measure, the average rank and variance are summarized in a scatter diagram, in which the horizontal axis represents the average rank, while the vertical axis is used to stand for variance of rank. Note that, in each diagram, a scatter located in the lower left quarter indicates that the corresponding sampling method was better, as it obtained a higher rank and lower variance. For simplicity, the sampling methods whose average rank and variance were both ranked as the top-three among all the employed sampling methods were regarded as stable. In the following, Figures 5–8 provide detailed scatter diagrams for each employed learner, respectively.

Figure 5 shows four scatter diagrams of the average rank vs. variance of rank for the classification algorithm NB, each corresponding to an individual performance measure. From Figure 5, it can be observed that when using AUC as the performance measure, Tomek and ENN were stable. Furthermore, SLSMOTE was stable when employing F-Measure and

G-Mean as the evaluation measures. However, for the measure TPR, no sampling method was stable within the current analysis setting.

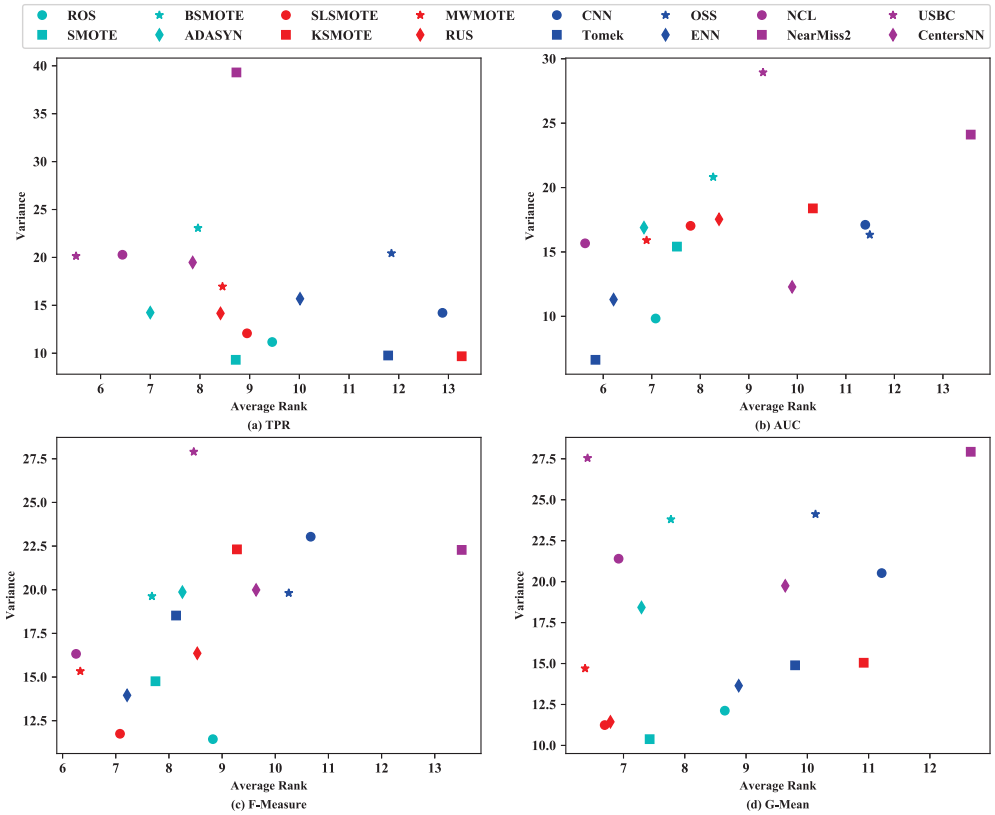

Figure 5. The scatter diagram of average rank vs. variance for NB.

We provide a scatter diagram for each performance measure when using C4.5 in Figure 6. As shown in Figure 6, USBC was stable for TPR and SLSMOTE was stable for F-Measure. However, for the two remaining performance measures, no sampling methods could obtain both the top-three average rank and variance of rank, which indicates that no sampling methods were stable for the C4.5 classification algorithm in terms of AUC and G-Mean.

When using KNN for imbalanced data classification, the corresponding scatter diagrams were as presented in Figure 7. From Figure 7, it can be observed that when using TPR to evaluate the classification performance for imbalanced data, USBC was stable. However, for the other three measures, no sampling methods were stable according to the current analysis setting, namely a stable sampling method should rank in top three of both the average rank and variance of rank.

Figure 8 demonstrates four scatter diagrams for the employed evaluation measures with RF as the classification algorithm. We can observe from Figure 8 that USBC was stable for TPR, MWMOTE was stable for AUC and F-Measure, and RUS and CentersNN were stable for G-Mean.

To conclude from the previous analysis, none of the employed sampling methods showed stable results across all the used classification algorithms and evaluation measures, which indicates the instability of the employed sampling methods. Therefore, when aiming at proposing a new sampling method for solving the imbalanced data problem, the selection of baseline sampling methods for comparative study is rather difficult. As a result, an unreasonable selection of sampling methods may give rise to inaccurate conclusions from the newly proposed sampling method. Therefore, we again advocate that the selection of appropriate sampling methods for the real-life imbalanced data problem may be more practical.

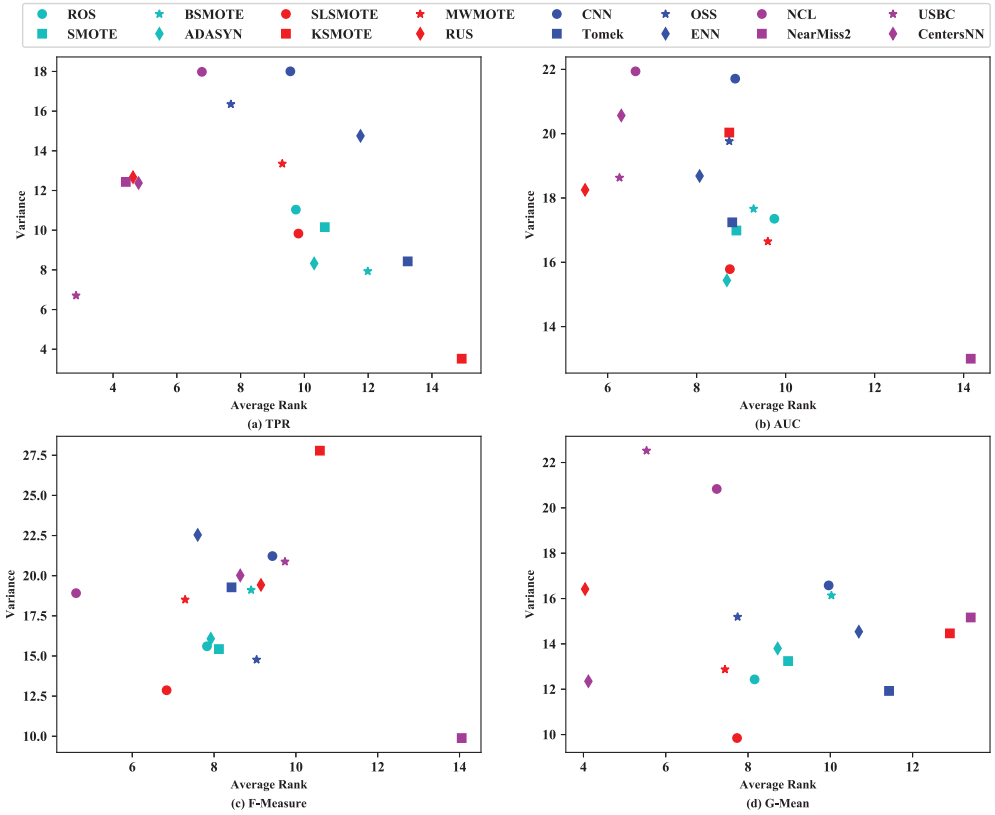

Figure 6. Scatter diagram of average rank vs. variance for C4.5.

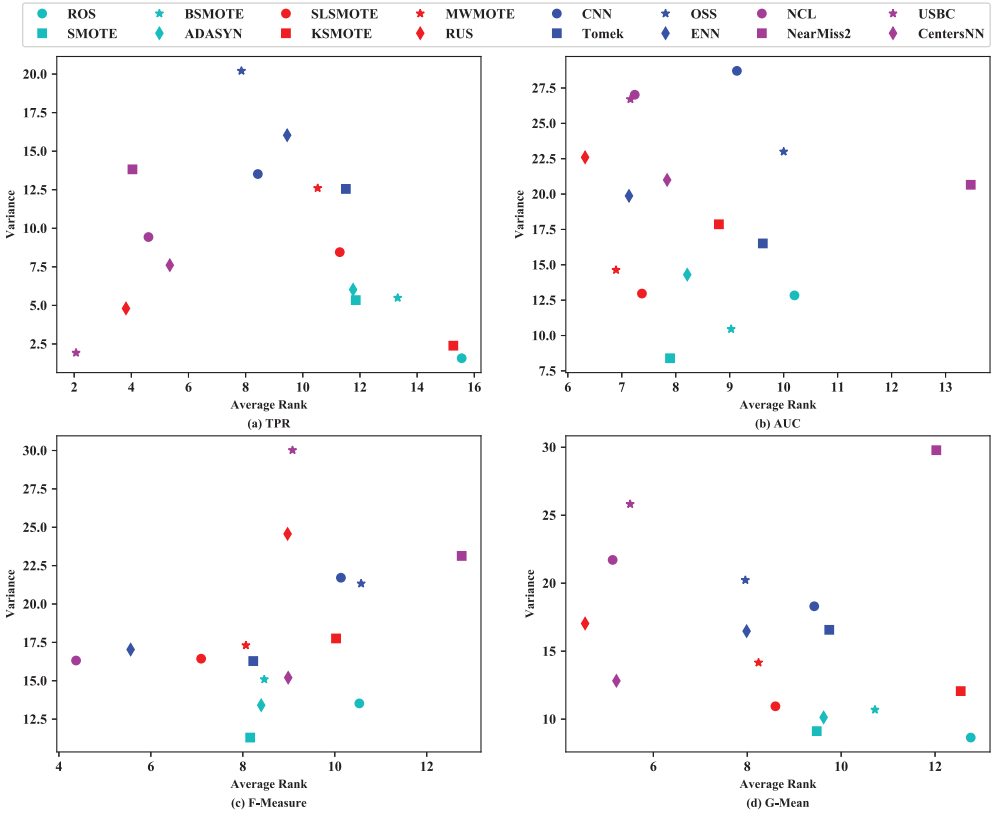

Figure 7. Scatter diagram of average rank vs. variance for KNN.

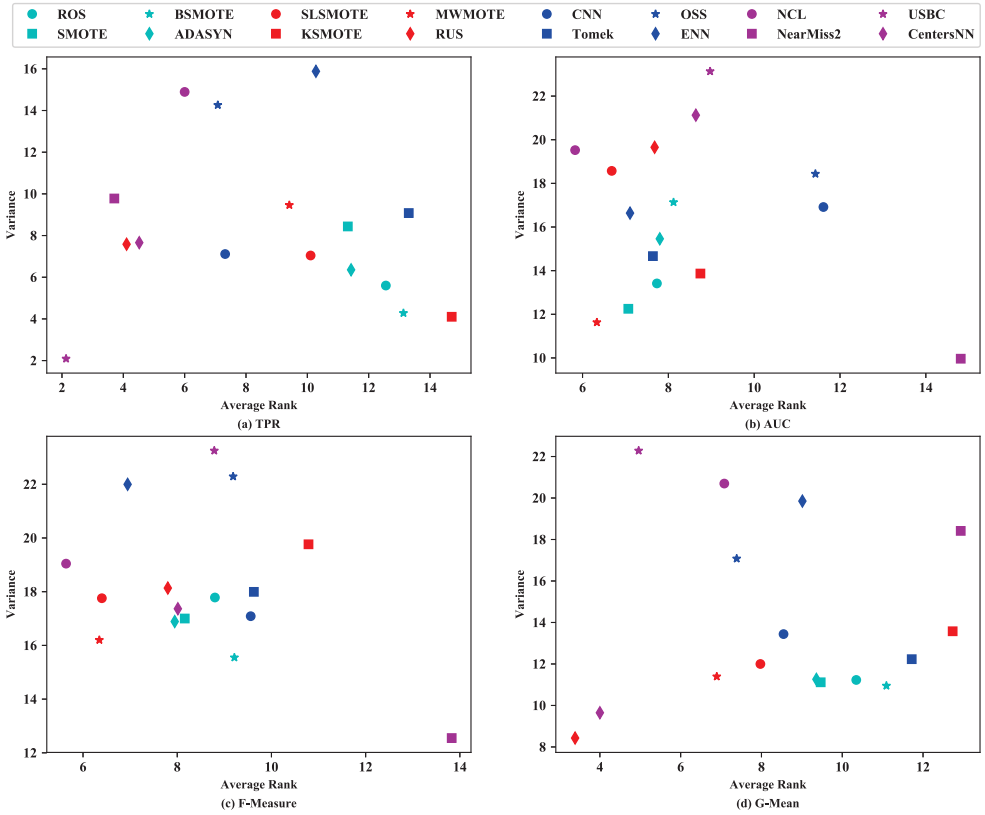

Figure 8. Scatter diagram of average rank vs. variance for RF.

4.3. Results of RQ3

In order to study the effect of different classification algorithms on the sampling methods, we first used the Friedman test [72] to compare the employed sampling methods over multiple datasets. To be specific, for each classification algorithm with an individual evaluation measure, a Friedman test with a significance level of 0.05 was conducted. As a result, the corresponding p-values are all less than 0.05, which indicates that the performance differences among the employed methods were not random and therefore confirms that the differences of the corresponding performance measures were significant. After that, as suggested by Demsar [73], a Nemenyi test [74] at a significance level $\alpha = 0.05$ was conducted, to show which particular methods performed significantly better.

Figures 9–12 provide detailed Nemenyi test results with a significance level of 0.05 for each individual performance measure. Note that in each figure, four sub-figures are included and each represents the corresponding Nemenyi test results for each classification algorithm employed. In addition, each sub-figure plots the employed methods against average performance ranks, where all methods are sorted according to their ranks. The '$*$' denotes the respective average rank of each method and the line segment to the right of each method represents its critical difference, which means the methods whose '$*$' is at the right end of the line were outperformed significantly. The critical difference is highlighted by a vertical dotted line in two cases. The left vertical line is associated with the best method, and all methods to the right of this line performed significantly worse than this method. The right vertical line is associated with the worst method, and all methods to the left of this line performed significantly better than it.

Figure 9 provides the Nemenyi test results at a 0.05 significance level with TPR as the performance measure. We can observe from Figure 9 that the different sampling methods usually ranked differently with different classification algorithms. For example, ADASYN and BSMOTE obtained a higher ranking for NB, while having a lower ranking for the other three classification algorithms. In Figure 9, USBC obtained the best TPR performance

rank among all four employed classification algorithms. In particular, USBC performed significantly better than eight sampling methods for NB and the 12 different sampling methods for the other three classification algorithms. In addition, KSMOTE performed worst with three classification algorithms, including NB, C4.5, and RF. To be specific, 12 sampling methods performed significantly better than KSMOTE for NB and RF, while for C4.5 the number was 14. Moreover, ROS performed the worst for the classification algorithm KNN, with 13 sampling methods outperforming ROS significantly.

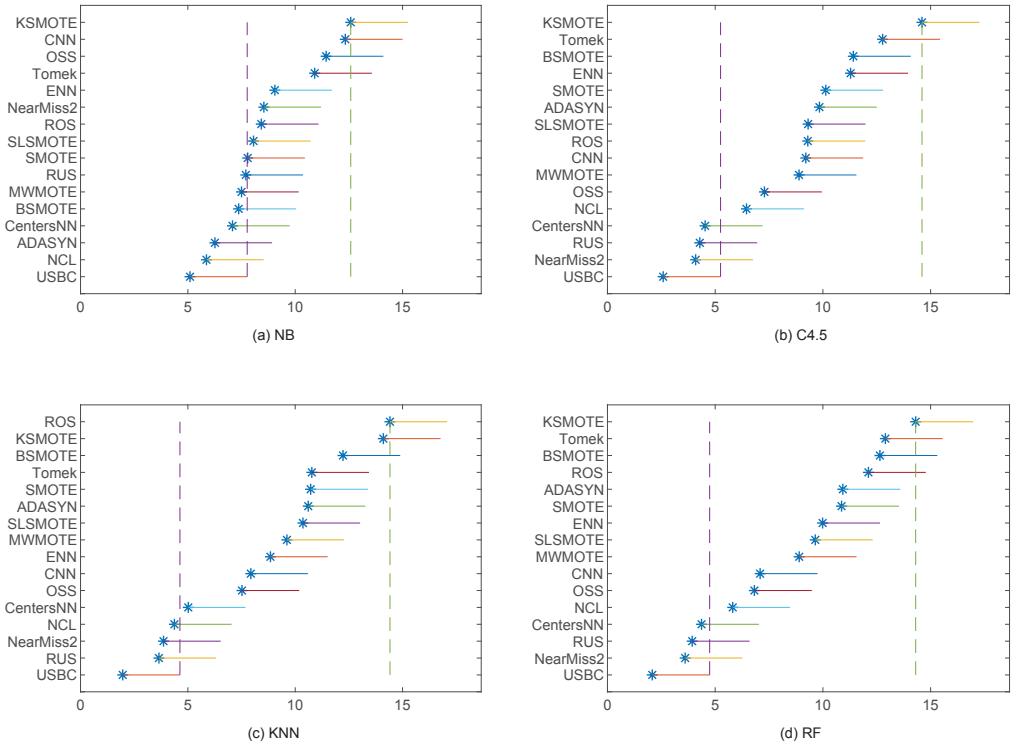

Figure 9. The results of the Nemenyi test at a 0.05 significance level for TPR.

Figure 10 presents the detailed Nemenyi test results at a 0.05 significance level for AUC. It can be observed from Figure 10 that the different sampling methods may rank differently when using different classification algorithms for model construction. For example, MWMOTE ranked second for KNN and RF, while it ranked among the last three for C4.5. In Figure 10, NearMiss2 obtained the worst AUC performance for all four employed classification algorithms. Specifically, for the three classification algorithms C4.5, KNN and RF, all the other sampling methods employed performed significantly better than NearMiss2. However, for NB, 13 of the 15 employed sampling methods significantly outperformed NearMiss2. In addition, NCL obtained the best rank for NB and RF, while RUS performed best for C4.5 and KNN. In particular, for NB and RF, NCL performed significantly better than 6 different sampling methods. However, for RUS, it performed significantly better than 11 sampling methods for C4.5 and 6 sampling methods for KNN.

In Figure 11, the corresponding Nemenyi test results with a 0.05 significance level for the F-Measure are provided in detail for each employed classification algorithm. It can be observed from Figure 11 that NCL ranked first for the three classification algorithms, while MWMOTE performed best for the remaining classification algorithm. To be specific, NCL performed significantly better than 14 sampling methods for C4.5, 13 sampling methods for KNN, and 8 sampling methods for RF. However, for NB, MWMOTE only significantly outperformed five sampling methods. Furthermore, for all the classification algorithms employed, NearMiss2 always obtained the worst F-Measure performance. Almost all the other 13 employed sampling methods performed significantly better than NearMiss2,

except for OSS with KNN. What is more, the corresponding ranks of the different sampling methods usually varied with different classification algorithms.

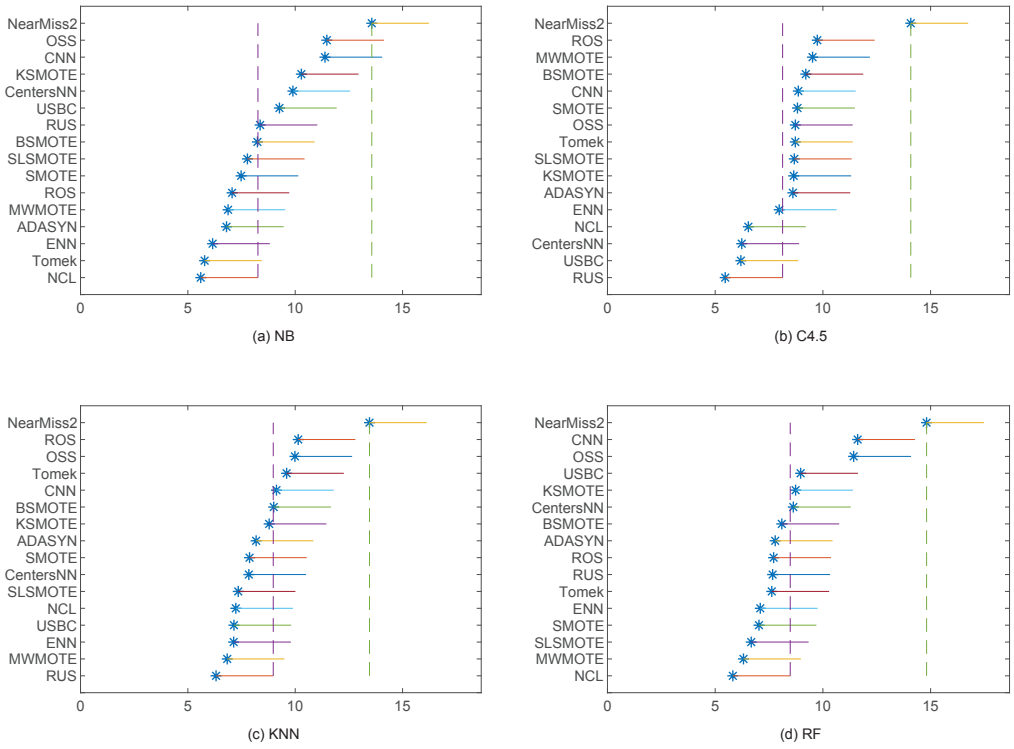

Figure 10. The results of the Nemenyi test at a 0.05 significance level for AUC.

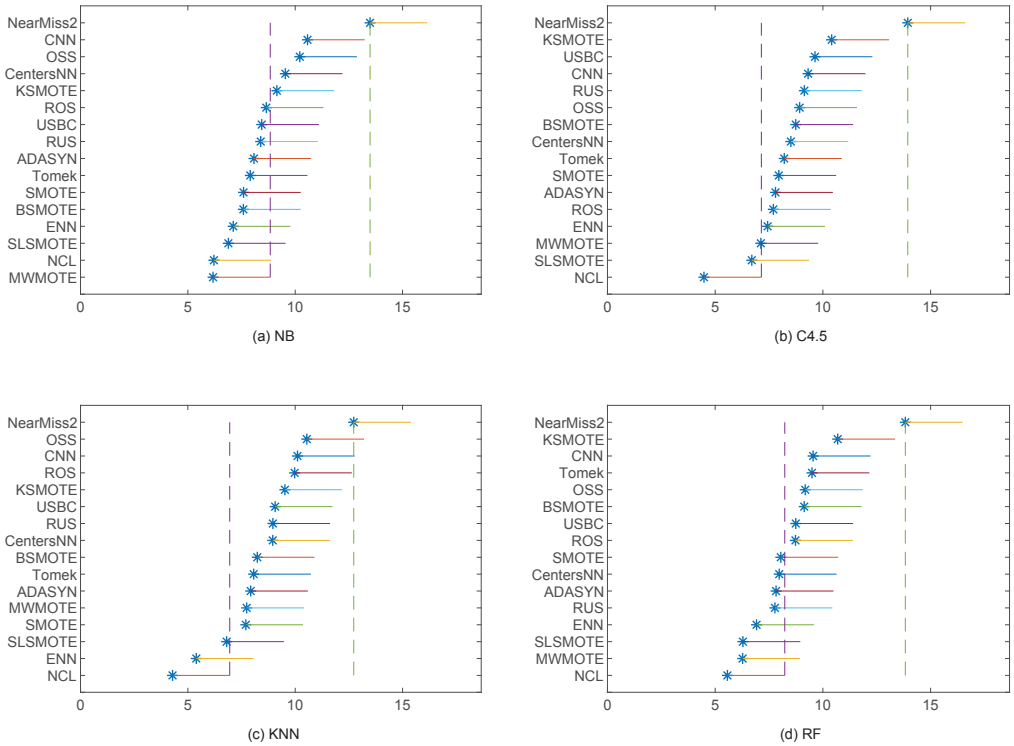

Figure 11. The results of the Nemenyi test at a 0.05 significance level for F-Measure.

Figure 12 shows the results of the Nemenyi test at a 0.05 significance level with G-Mean as the classification performance measure. We can observe from Figure 12 that RUS

obtained the first rank for the two classification algorithms KNN and RF, which significantly outperformed 12 and 13 sampling methods, respectively. However, MWMOTE and CentersNN performed best for NB and C4.5, respectively. To be specific, MWMOTE performed significantly better than seven sampling methods for NB, while CentersNN performs significantly better than 13 sampling methods for C4.5. Furthermore, for the three classification algorithms NB, C4.5, and RF, NearMiss2 obtained the worst G-Mean performance and at least 12 sampling methods performed significantly better than NearMiss2. For KNN, ROS was the worst, with 12 sampling methods performing significantly better than ROS. As a matter of fact, in Figure 12, the corresponding ranks of the different sampling methods generally vary with the different classification algorithms.

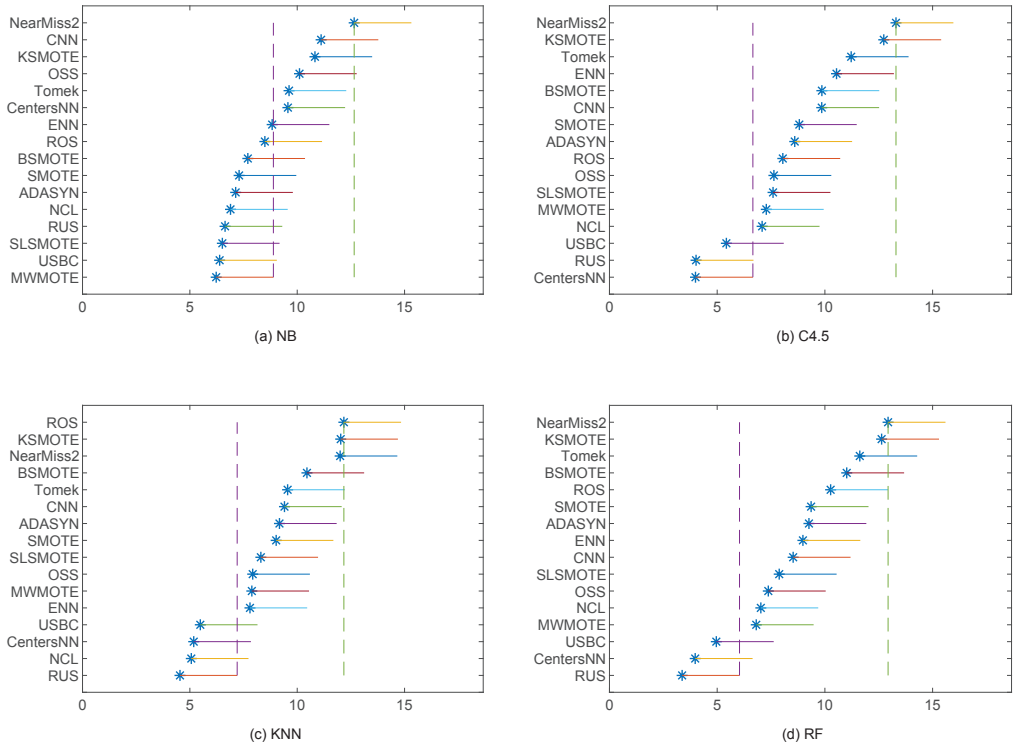

Figure 12. The results of the Nemenyi test at a 0.05 significance level for G-Mean.

In summary, when employing different machine learning algorithms for learning imbalanced data, different sampling methods may behave very differently. In Sections 4.1 and 4.2, the selection of appropriate sampling methods was shown to be very important for specific imbalanced data. Therefore, combining with the conclusion of the present section, it is suggested that sampling methods should be considered simultaneously with the used classification algorithms.

5. Conclusions

In present study, we present an extensive experimental comparison of different sampling methods for imbalanced data classification. In the experiment, 75 originally imbalanced binary datasets from four different application domains were employed. In particular, these 75 datasets originated from the following four application domains: physical sciences, life sciences, business, and software engineering. In addition, seven oversampling methods and nine undersampling methods were selected for comparison according to several designed criteria. Moreover, four popular classification algorithms and four widely-used performance measures were used in the current experiment.

We first researched the effectiveness and stability of the employed sampling methods. The corresponding experimental results showed that most sampling methods were effective in tackling the imbalanced data problem. However, we unexpectedly found that some

new and complex sampling methods may not perform better than the simple and classical sampling methods. Furthermore, none of the employed sampling methods showed consistently stable results across all the used classification algorithms and evaluation measures. Thereafter, the impact of different classification algorithms was studied, and the corresponding experimental results demonstrated that different sampling methods may behave very differently when employing different machine learning algorithms to build classification models.

Summarizing from previous findings, we advocate that, rather than proposing new sampling methods to handle imbalanced data, researchers should pay more attention to selecting applicable sampling methods for given imbalanced data, while taking into consideration the used classification algorithms. For such problems, there may be two feasible solutions. One is to research recommended sampling methods for a specific classification algorithm according to the meta features of datasets. The other is to research how to integrate the selection of appropriate sampling methods into automated machine learning for imbalanced data.

There are still some shortcomings in the current research. For example, the generalization of the obtained conclusions may have been affected by the limited datasets, sampling methods, classification algorithms, and performance measures. Moreover, in the present study, we only paid attention to the classification performance of the different sampling methods, while ignoring the corresponding efficiency. What is more, our present study focused on the traditional classification scenes, with a major assumption that training and test data are in the same feature space and follow the same distribution. However, in many real-word applications, this assumption may not hold, and transfer learning [75] has emerged as a new learning framework to address this problem. Therefore, in the future, we intend to employ further datasets from different fields and more classification algorithms, sampling methods, and performance measures to improve the generalization of the conclusions of the present study. In addition, we plan to research the effect of different sampling methods on efficiency and validate whether the sampling methods have similar effects with transfer learning.

Author Contributions: Funding acquisition, Z.S.; Investigation, Z.S., J.Z. and X.Z.; Methodology, Z.S. and J.Z.; Supervision, Z.S. and X.Z.; Validation, Z.S. and J.Z.; Writing—original draft, Z.S. and J.Z.; Writing—review and editing, X.Z. and D.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Fundamental Research Funds for the Central Universities under Grant No. 2021QN1075.

Data Availability Statement: This study used open data sources.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. He, H.; Garcia, E.A. Learning from Imbalanced Data. *IEEE Trans. Knowl. Data Eng.* **2009**, *21*, 1263–1284.
2. Guo, G.; Li, Y.; Shang, J.; Gu, M.; Huang, Y.; Gong, B. Learning from class-imbalanced data: Review of methods and applications. *Expert Syst. Appl.* **2017**, *73*, 220–239.
3. Zhu, B.; Baesens, B.; Broucke, S.K.L.M.V. An empirical comparison of techniques for the class imbalance problem in churn prediction. *Inf. Sci.* **2017**, *408*, 84–99. [CrossRef]
4. Sun, Z.; Song, Q.; Zhu, X. Using coding-based ensemble learning to improve software defect prediction. *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.* **2012**, *42*, 1806–1817. [CrossRef]
5. Xie, Y.; Li, S.; Wu, C.T.; Lai, Z.; Su, M. A novel hypergraph convolution network for wafer defect patterns identification based on an unbalanced dataset. *J. Intell. Manuf.* **2022**, 1–14. [CrossRef]
6. Wei, W.; Li, J.; Cao, L.; Ou, Y.; Chen, J. Effective detection of sophisticated online banking fraud on extremely imbalanced data. *World Wide Web* **2013**, *16*, 449–475. [CrossRef]
7. Chawla, N.V.; Japkowicz, N.; Kotcz, A. Special issue on learning from imbalanced data sets. *ACM SIGKDD Explor. Newsl.* **2004**, *6*, 1–6. [CrossRef]
8. Batista, G.E.; Prati, R.C.; Monard, M.C. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD Explor. Newsl.* **2004**, *6*, 20–29. [CrossRef]

9. Douzas, G.; Bacao, F.; Last, F. Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE. *Inf. Sci.* **2018**, *465*, 1–20. [CrossRef]
10. López, V.; Fernández, A.; Moreno-Torres, J.G.; Herrera, F. Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. Open problems on intrinsic data characteristics. *Expert Syst. Appl.* **2012**, *39*, 6585–6608. [CrossRef]
11. Yu, H.; Sun, C.; Yang, X.; Yang, W.; Shen, J.; Qi, Y. ODOC-ELM: Optimal decision outputs compensation-based extreme learning machine for classifying imbalanced data. *Knowl.-Based Syst.* **2016**, *92*, 55–70. [CrossRef]
12. Galar, M.; Fernandez, A.; Barrenechea, E.; Bustince, H.; Herrera, F. A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches. *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.* **2011**, *42*, 463–484. [CrossRef]
13. Sun, Z.; Song, Q.; Zhu, X.; Sun, H.; Xu, B.; Zhou, Y. A novel ensemble method for classifying imbalanced data. *Pattern Recognit.* **2015**, *48*, 1623–1637. [CrossRef]
14. López, V.; Fernández, A.; García, S.; Palade, V.; Herrera, F. An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Inf. Sci.* **2013**, *250*, 113–141. [CrossRef]
15. Estabrooks, A.; Jo, T.; Japkowicz, N. A Multiple Resampling Method for Learning from Imbalanced Data Sets. *Comput. Intell.* **2004**, *20*, 18–36. [CrossRef]
16. Loyola-González, O.; Martínez-Trinidad, J.F.; Carrasco-Ochoa, J.A.; García-Borroto, M. Study of the impact of resampling methods for contrast pattern based classifiers in imbalanced databases. *Neurocomputing* **2016**, *175*, 935–947. [CrossRef]
17. López, V.; Fernández, A.; Herrera, F. On the importance of the validation technique for classification with imbalanced datasets: Addressing covariate shift when data is skewed. *Inf. Sci.* **2014**, *257*, 1–13. [CrossRef]
18. Chen, C.P.; Zhang, C.Y. Data-intensive applications, challenges, techniques and technologies: A survey on Big Data. *Inf. Sci.* **2014**, *275*, 314–347. [CrossRef]
19. Maldonado, S.; Weber, R.; Famili, F. Feature selection for high-dimensional class-imbalanced data sets using Support Vector Machines. *Inf. Sci.* **2014**, *286*, 228–246. [CrossRef]
20. Sáez, J.A.; Luengo, J.; Stefanowski, J.; Herrera, F. SMOTE–IPF: Addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering. *Inf. Sci.* **2015**, *291*, 184–203. [CrossRef]
21. Krawczyk, B.; Woźniak, M.; Schaefer, G. Cost-sensitive decision tree ensembles for effective imbalanced classification. *Appl. Soft Comput.* **2014**, *14*, 554–562. [CrossRef]
22. Tao, X.; Li, Q.; Guo, W.; Ren, C.; Li, C.; Liu, R.; Zou, J. Self-adaptive cost weights-based support vector machine cost-sensitive ensemble for imbalanced data classification. *Inf. Sci.* **2019**, *487*, 31–56. [CrossRef]
23. 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.* **2018**, *425*, 76–91. [CrossRef]
24. Das, B.; Krishnan, N.C.; Cook, D.J. RACOG and wRACOG: Two Probabilistic Oversampling Techniques. *IEEE Trans. Knowl. Data Eng.* **2015**, *27*, 222–234. [CrossRef] [PubMed]
25. Abdi, L.; Hashemi, S. To Combat Multi-Class Imbalanced Problems by Means of Over-Sampling Techniques. *IEEE Trans. Knowl. Data Eng.* **2016**, *28*, 238–251. [CrossRef]
26. Yang, X.; Kuang, Q.; Zhang, W.; Zhang, G. AMDO: An Over-Sampling Technique for Multi-Class Imbalanced Problems. *IEEE Trans. Knowl. Data Eng.* **2018**, *30*, 1672–1685. [CrossRef]
27. Li, L.; He, H.; Li, J. Entropy-based Sampling Approaches for Multi-class Imbalanced Problems. *IEEE Trans. Knowl. Data Eng.* **2019**, *32*, 2159–2170. [CrossRef]
28. Chawla, N.V.; Bowyer, K.W.; Hall, L.O.; Kegelmeyer, W.P. SMOTE: Synthetic minority over-sampling technique. *J. Artif. Intell. Res.* **2002**, *16*, 321–357. [CrossRef]
29. Fernandez, A.; Garcia, S.; Herrera, F.; Chawla, N.V. SMOTE for Learning from Imbalanced Data: Progress and Challenges, Marking the 15-year Anniversary. *J. Artif. Intell. Res.* **2018**, *61*, 863–905. [CrossRef]
30. Han, H.; Wang, W.Y.; Mao, B.H. Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning. In Proceedings of the International Conference on Intelligent Computing, Hefei, China, 23–26 August 2005; pp. 878–887.
31. He, H.; Bai, Y.; Garcia, E.A.; Li, S. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In Proceedings of the 2008 IEEE International Joint Conference on Neural Networks, Hong Kong, China, 1–6 June 2008; pp. 1322–1328.
32. 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.* **2012**, *26*, 405–425. [CrossRef]
33. 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. Softw. Eng.* **2017**, *44*, 534–550. [CrossRef]
34. Yan, Y.; Tan, M.; Xu, Y.; Cao, J.; Ng, M.; Min, H.; Wu, Q. Oversampling for Imbalanced Data via Optimal Transport. In Proceedings of the AAAI Conference on Artificial Intelligence 2019, Honolulu, HI, USA, 27 January–1 February 2019; pp. 5605–5612.
35. Zhang, H.; Li, M. RWO-Sampling: A random walk over-sampling approach to imbalanced data classification. *Inf. Fusion* **2014**, *20*, 99–116. [CrossRef]
36. Ofek, N.; Rokach, L.; Stern, R.; Shabtai, A. Fast-CBUS: A fast clustering-based undersampling method for addressing the class imbalance problem. *Neurocomputing* **2017**, *243*, 88–102. [CrossRef]
37. Tsai, C.F.; Lin, W.C.; Hu, Y.H.; Yao, G.T. Under-sampling class imbalanced datasets by combining clustering analysis and instance selection. *Inf. Sci.* **2019**, *477*, 47–54. [CrossRef]

38. Peng, M.; Zhang, Q.; Xing, X.; Gui, T.; Huang, X.; Jiang, Y.G.; Ding, K.; Chen, Z. Trainable Undersampling for Class-Imbalance Learning. In Proceedings of the AAAI Conference on Artificial Intelligence 2019, Honolulu, HI, USA, 27 January–1 February 2019; pp. 4707–4714.
39. Vuttipittayamongkol, P.; Elyan, E. Neighbourhood-based undersampling approach for handling imbalanced and overlapped data. *Inf. Sci.* **2020**, *509*, 47–70. [CrossRef]
40. Hart, P. The condensed nearest neighbor rule (Corresp.). *IEEE Trans. Inf. Theory* **1968**, *14*, 515–516. [CrossRef]
41. Wilson, D.L. Asymptotic properties of nearest neighbor rules using edited data. *IEEE Trans. Syst. Man Cybern.* **1972**, *2*, 408–421. [CrossRef]
42. Laurikkala, J. Improving identification of difficult small classes by balancing class distribution. In Proceedings of the Conference on Artificial Intelligence in Medicine in Europe 2001, Cascais, Portugal, 1–4 July 2001; pp. 63–66.
43. Kubat, M.; Matwin, S. Addressing the curse of imbalanced training sets: One-sided selection. In Proceedings of the International Conference on Machine Learning 1997, Nashville, TN, USA, 8–12 July 1997; pp. 179–186.
44. Tomek, I. Two modifications of CNN. *IEEE Trans. Syst. Man Cybern.* **1976**, *6*, 769–772.
45. Yen, S.J.; Lee, Y.S. Cluster-based under-sampling approaches for imbalanced data distributions. *Expert Syst. Appl.* **2009**, *36*, 5718–5727. [CrossRef]
46. Lin, W.C.; Tsai, C.F.; Hu, Y.H.; Jhang, J.S. Clustering-based undersampling in class-imbalanced data. *Inf. Sci.* **2017**, *409*, 17–26. [CrossRef]
47. García, S.; Herrera, F. Evolutionary undersampling for classification with imbalanced datasets: Proposals and taxonomy. *Evol. Comput.* **2009**, *17*, 275–306. [CrossRef]
48. Wang, N.; Zhao, X.; Jiang, Y.; Gao, Y. Iterative metric learning for imbalance data classification. In Proceedings of the 27th International Joint Conference on Artificial Intelligence 2018, Stockholm, Sweden, 13–19 July 2018; pp. 2805–2811.
49. Japkowicz, N. Learning from imbalanced data sets: A comparison of various strategies. In Proceedings of the AAAI Workshop on Learning from Imbalanced Data Sets 2000, Austin, TX, USA, 31 July 2000; pp. 10–15.
50. Drummond, C.; Holte, R.C. C4. 5, class imbalance, and cost sensitivity: Why under-sampling beats over-sampling. In Proceedings of the Workshop on Learning from Imbalanced Datasets II 2003, Washington, DC, USA, 1 July 2003; pp. 1–8.
51. Bennin, K.E.; Keung, J.; Monden, A.; Kamei, Y.; Ubayashi, N. Investigating the effects of balanced training and testing datasets on effort-aware fault prediction models. In Proceedings of the 2016 IEEE 40th Annual Computer Software and Applications Conference, Atlanta, GA, USA, 10–14 June 2016; pp. 154–163.
52. García, V.; Sánchez, J.S.; Mollineda, R.A. On the effectiveness of preprocessing methods when dealing with different levels of class imbalance. *Knowl.-Based Syst.* **2012**, *25*, 13–21. [CrossRef]
53. Hulse, J.V.; Khoshgoftaar, T.M.; Napolitano, A. Experimental perspectives on learning from imbalanced data. In Proceedings of the 24th International Conference Machine Learning 2007, Corvalis, OR, USA, 20–24 June 2007; pp. 935–942.
54. Bennin, K.E.; Keung, J.W.; Monden, A. On the relative value of data resampling approaches for software defect prediction. *Empir. Softw. Eng.* **2019**, *24*, 602–636. [CrossRef]
55. Zhou, L. Performance of corporate bankruptcy prediction models on imbalanced dataset: The effect of sampling methods. *Knowl.-Based Syst.* **2013**, *41*, 16–25. [CrossRef]
56. Napierala, K.; Stefanowski, J. Types of minority class examples and their influence on learning classifiers from imbalanced data. *J. Intell. Inf. Syst.* **2016**, *46*, 563–597. [CrossRef]
57. Kamei, Y.; Monden, A.; Matsumoto, S.; Kakimoto, T.; Matsumoto, K. The effects of over and under sampling on fault-prone module detection. In Proceedings of the International Symposium on Empirical Software Engineering and Measurement 2007, Madrid, Spain, 20–21 September 2007; pp. 196–204.
58. Bennin, K.E.; Keung, J.; Monden, A.; Phannachitta, P.; Mensah, S. The significant effects of data sampling approaches on software defect prioritization and classification. In Proceedings of the ACM/IEEE International Symposium on Empirical Software Engineering and Measurement 2017, Toronto, ON, Canada, 9–10 November 2017; pp. 364–373.
59. Bunkhumpornpat, C.; Sinapiromsaran, K.; Lursinsap, C. Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. In Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining 2009, Bangkok, Thailand, 27–30 April 2009; pp. 475–482.
60. Barandela, R.; Valdovinos, R.M.; Sánchez, J.S.; Ferri, F.J. The imbalanced training sample problem: Under or over sampling? In Proceedings of the Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition and Structural and Syntactic Pattern Recognition 2004, Lisbon, Portugal, 18–20 August 2004; pp. 806–814.
61. Mani, I.; Zhang, I. kNN approach to unbalanced data distributions: A case study involving information extraction. In Proceedings of the Workshop on Learning from Imbalanced Datasets 2003, Washington, DC, USA, 1 July 2003; pp. 1–7.
62. Alcalá, J.; Fernández, A.; Luengo, J.; Derrac, J.; García, S.; Sánchez, L.; Herrera, F. Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. *J. Mult.-Valued Log. Soft Comput.* **2011**, *17*, 255–287.
63. Dua, D.; Graff, C. UCI Machine Learning Repository. 2017. Available online: http://archive.ics.uci.edu/ml (accessed on 21 September 2023).
64. Jing, X.Y.; Wu, F.; Dong, X.; Xu, B. An improved SDA based defect prediction framework for both within-project and cross-project class-imbalance problems. *IEEE Trans. Softw. Eng.* **2017**, *43*, 321–339. [CrossRef]

65. Czibula, G.; Marian, Z.; Czibula, I.G. Software defect prediction using relational association rule mining. *Inf. Sci.* **2014**, *264*, 260–278. [CrossRef]
66. Park, B.J.; Oh, S.K.; Pedrycz, W. The design of polynomial function-based neural network predictors for detection of software defects. *Inf. Sci.* **2013**, *229*, 40–57. [CrossRef]
67. Shepperd, M.; Song, Q.; Sun, Z.; Mair, C. Data quality: Some comments on the NASA software defect datasets. *IEEE Trans. Softw. Eng.* **2013**, *39*, 1208–1215. [CrossRef]
68. Jureczko, M.; Madeyski, L. Towards identifying software project clusters with regard to defect prediction. In Proceedings of the 6th International Conference on Predictive Models in Software Engineering 2010, Timisoara, Romania, 12–13 September 2010; pp. 9–18.
69. Catal, C.; Diri, B. Investigating the effect of dataset size, metrics sets, and feature selection techniques on software fault prediction problem. *Inf. Sci.* **2009**, *179*, 1040–1058. [CrossRef]
70. Seiffert, C.; Khoshgoftaar, T.M.; Van Hulse, J.; Folleco, A. An empirical study of the classification performance of learners on imbalanced and noisy software quality data. *Inf. Sci.* **2014**, *259*, 571–595. [CrossRef]
71. Hall, M.; Frank, E.; Holmes, G.; Pfahringer, B.; Reutemann, P.; Witten, I.H. The WEKA data mining software: An update. *ACM SIGKDD Explor. Newsl.* **2009**, *11*, 10–18. [CrossRef]
72. Friedman, M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *J. Am. Stat. Assoc.* **1937**, *32*, 675–701. [CrossRef]
73. Demšar, J. Statistical Comparisons of Classifiers over Multiple Data Sets. *J. Mach. Learn. Res.* **2006**, *7*, 1–30.
74. Nemenyi, P. Distribution-Free Multiple Comparisons. Ph.D. Thesis, Princeton University, Princeton, NJ, USA, 1963.
75. Pan, S.J.; Yang, Q. A Survey on Transfer Learning. *IEEE Trans. Knowl. Data Eng.* **2010**, *22*, 1345–1359. [CrossRef]