Neural Computing and Applications (2021) 33:11233–11254
https://doi.org/10.1007/s00521-020-05570-7

ORIGINAL ARTICLE

A novel ensemble method for classification in imbalanced datasets using split balancing technique based on instance hardness (sBal_IH)

Halimu Chongomweru[1] · **Asem Kasem**[1]

Received: 27 May 2020 / Accepted: 1 December 2020 / Published online: 11 January 2021

Abstract
Classification tasks in datasets that suffer from high class imbalance pose challenge to machine learning algorithms and such datasets are prevalent in many real-world domains and applications. In machine learning research, ensemble methods for classification tasks in imbalanced datasets have attracted a lot of attention due to their ability to improve classification performance. However, these methods are still prone to the negative effects of noise in the training sets. Furthermore, many of them alter the original class distribution to create a sort of balance in the datasets through over-sampling or under-sampling techniques, which can lead to overfitting or discarding useful data, respectively, and thus may still hamper performance. In this work, we propose a novel ensemble method for classification that creates an arbitrary number of balanced splits (sBal) of data generated based on Instance Hardness as a weighting mechanism for creating balanced bags. Each of the generated bags will contain all the minority instances, and a mixture of majority instances with varying degrees of hardness (easy, normal, and hard), and we call this approach sBal_IH technique. This will enable base learners to train on different balanced bags comprising varied characteristics of the training data. We evaluated the performance of our proposed method on a total of 100 datasets that include 30 synthetic datasets with controlled levels of noise, 29 balanced and 41 imbalanced real-world datasets, and compared its performance with both traditional ensemble methods (Bagging, Wagging Random Forest, and AdaBoost), and those specialized for class imbalanced problems (Balanced Bagging, Balanced Random forest, RUSBoost, and Easy Ensemble). The results reveal that our proposed method brings a substantial improvement in classification performance relevant to the compared methods. For statistical significance analysis, we conducted Friedman's nonparametric statistical test with Bergman post hoc test. The analysis shows that our method performs significantly better than the compared traditional and specialized ensemble methods for imbalanced problems across many datasets.

Keywords Imbalanced data · Binary classification · Instance hardness · Data complexity · Ensemble · Split balancing · Balanced bagging · Balanced random forest · Easy ensemble · Wagging

1 Introduction

Pattern classification has attracted considerable attention in the field of machine learning with a wide area of application in contemporary real life such as face recognition, anomaly detection, image classification, cancer classification, medical diagnosis, among many others. The main challenge of pattern classification is the capability of the trained classifier to classify unseen patterns correctly. This challenge is most often associated with data complexities existing within the underlying datasets [1] such as class ambiguity, data sparsity and dimensionality, class boundary complexity, and class imbalance problem.

Chongomweru Halimu and Asem Kasem have contributed equally to the research works of this study and paper writing.

✉ Halimu Chongomweru
P20181008@student.utb.edu.bn

Asem Kasem
asem.kasem@utb.edu.bn

[1] School of Computing and Informatics, Universiti Teknologi Brunei, Bandar Seri Begawan, Brunei Darussalam

Class imbalance problem is a situation where a dataset is characterized by an uneven class distribution, i.e., the proportion of instances of one class is much smaller than those of other classes. In the real world, despite the vast availability of data, the class of interest (minority or positive class) in binary classification problems usually has fewer instances than the opposite class (majority or negative class).

One clear complication that arises as a result of the class imbalance problem is the performance of traditional machine learning algorithms and the effectiveness of their accuracy. Let's consider an example of a dataset with an imbalance ratio of 99:1 where the majority class is composed of 99% of the data instances. In such a situation, a naive classifier will have an accuracy score of 99% since it always predicts the class with majority instances. This limitation cuts across most traditional machine learning algorithms such as decision tree (DT), k-nearest neighbors (kNN) when faced with imbalanced datasets since most of them optimize accuracy-based loss metrics. Hence, producing similar models as to that of the naïve model in the example described before [2, 3].

Apparently, in most practical situations misclassifying the class of interest can result in a heavy cost. For example, in cancer detection, patients with tumors might emerge out of hundreds of records but failing to identify a malignant tumor (false negative) may cause a serious threat to a patient since they may miss out on treatment. It is therefore useless to have a model with a very high accuracy score but failing to detect the minority class (i.e., low sensitivity) which is considered as the class of interest. However, it is possible to get a low misclassification cost by using an effective traditional algorithm for solving such a problem. But the cost component has to be addressed during the classification process through cost-sensitive learning. Cost-sensitive learning, which aims at minimizing errors, takes into account the cost of prediction errors, and potentially other costs during the training process [4]. It is closely related to the field of imbalanced learning that is much concerned with the classification of imbalanced datasets. As a result, various cost-sensitive techniques have been proposed. Unfortunately, defining the costs is still a big challenge, since misclassification costs are often unknown [5].

As a result, the problem of class imbalance has emerged as one of the challenges in the machine learning community and it has attracted much attention of researchers in academia and industry to an extent that 2 special workshops devoted to addressing this problem were held in 2000, at AAAI 2000 Workshop on Learning from Imbalanced Datasets [6] and in 2003 at ICML 2003 Workshop on Learning from Imbalanced Datasets [7]. It is evident that researchers have extensively studied the problem of class imbalance and various methods have been proposed to overcome this problem. These methods fall into three categories: Data level, Algorithm level, and hybrid/Ensemble Level.

Data level methods, also referred to as external techniques, first preprocess the data by trying to rebalance the class distribution. The data level techniques can further be broken down into sampling techniques and feature selection techniques. For sampling techniques, instances from the minority class are replicated to balance the class distribution through oversampling [8], or instances of the majority class are discarded to balance the class distribution through undersampling [9]. Researchers in [10]–[13] have proposed various oversampling approaches for balancing the class distributions; furthermore, researchers in [14]–[16] have similarly presented several undersampling approaches for solving the class imbalance problem. On the other hand, the feature selection techniques try to neutralize the effects of class imbalance by selecting the most influential features that can produce exclusive knowledge that can easily discriminate between the classes. Researchers in [17] explored feature selection for the categorization of text with imbalanced data. Mladenic et al. [18], utilized feature subsets to develop a Naive Bayes (NB) classifier on imbalanced text data. It is important to note that feature selection techniques for addressing the class imbalance problem have not yet been fully explored, creating a research gap in this area. On the other hand, feature selection techniques have been widely used in class balanced problems to improve performance score [19]–[22].

Algorithm Level (Internal) techniques modify the learning algorithm to take into consideration the significance of the minority instances by biasing the algorithm to learn towards the minority class [23, 24]. The most common algorithm level technique is the cost-sensitive method [25], where the algorithm is modified to integrate in varying penalties for each of the selected groups of examples. The higher cost is assigned to fewer represented instances to boost their significance during the learning process. Various cost-sensitive classification techniques have been proposed in the literature [4, 26]. A common strategy of these techniques is to deliberately increase the weight of instances with higher misclassification costs during the boosting process. However, the challenge with cost-sensitive classification is that the misclassification costs are often unknown and difficult to estimate.

Finally, the Hybrid/Ensemble level techniques utilize the advantage of both data level and algorithm level techniques by combining them to handle the problem of class imbalance efficiently. For example, Wang et al. proposed the hybridization of both sampling and cost-sensitive learning in handling imbalanced data [27].

Furthermore, Zhang et al. in their work [28] proposed a method for handling imbalanced data by integrating ensemble and classification techniques in enhancing the performance of the ensemble. In another study [29], Paweł et al. proposed a hybrid ensemble method that combines the advantages of ensemble learning, deep learning, and evolutionary computation, to effectively classify cardiac arrhythmias using ECG signal segments.

Ensemble machine learning techniques combine more than one single learner (base learner) using a given combination rule to produce enhanced predictive models [30]. Base learners can be any machine learning algorithm (e.g., Decision Tree, Naïve Bayes, Artificial Neural Network, Linear Regression, etc.). The ensemble topology can be as simple as an independent collection of learners combined via a majority vote or using some other advanced mechanisms such as those indicated in [31], where ensembles consist of General Regression Neural Network and Geometric transformation model.

Ensemble techniques are among the most commonly used methods that utilize data level or algorithm level together with data resampling techniques in the classification of imbalanced data. This work is motivated by the growing trends in the use of ensemble techniques in imbalanced classification. This is because of their ability to improve classification performance, by leveraging the classification power of multiple base learners trained on different bootstraps of training data as compared to traditional classification algorithms. The main ground of the ensemble techniques is that by combining various classifiers, the error of one classifier will most likely be compensated by another classifier in the ensemble, and as a result, the general prediction score of the ensemble model would be more effective than that of a single classifier [32].

Different reasons have been discussed why ensemble methods most of the time perform better than single classifiers [33, 34]. One of the reasons is that the training data may not give adequate information for selecting a single best algorithm. For instance, there may be a range of algorithms that performs equally fine on the training data. Instead of choosing one of them, joining the predictions of these algorithms can be a much better choice. The other important reason is that, in some cases, the hypothesis space which is being searched may not include the real target function. In such a situation, combining various hypotheses can efficiently broaden the space and provide a better estimation for the unknown predictor function. For instance, the classification boundary of normal decision trees is hyperplanes that are parallel to the coordinate axes. In case the target classification boundary is of a different type, a single decision tree may not give a smooth estimation [35]. On the other hand, a combination of decision trees can give estimations to smooth boundaries of arbitrary shape.

In the literature, a good number of comprehensive surveys on ensemble techniques have been published [8, 36]–[44]. Bootstrap aggregating (Bagging) [45] and Boosting [46, 47] are considered the most widely used ensemble methods; for Bagging, Breiman in 1996 introduced the idea of bootstrap aggregation to build an ensemble where different base algorithms are trained on bootstrap samples randomly drawn from the original dataset based on uniform probability distribution with replacement. As for Boosting, it was introduced by Schapire in 1998 [46] as a technique for boosting the performance of weak learners.

Since then, many variations of bagging and boosting based ensembles have been proposed in the literature to address the problem of class imbalance. M. Galar et al. in their study [36] proposed a taxonomy for ensemble-based methods that utilize data preprocessing techniques to solve the problem of class imbalance. They clustered them into 3 groups which include: Boosting-based ensembles, Bagging-based and Hybrid ensembles as depicted in Fig. 1.

The majority of these ensemble techniques, despite reported improvement in their classification performance, it has been reported that they cannot completely survive the common problem of noise in machine learning [48]. The term noise applies to all types of anomalies in the training data, from errors to unusual cases of the observed domain, which make it harder to interpret the data. Noise in the training data can either be: attribute noise (errors or unusual instances) or class noise (incorrect class labels) or a mixture of both [49]. Therefore, those instances that complicate the learning process and degrade the performance of learning algorithms are referred to as noisy instances [50].

Most bagging-based ensemble methods utilize the existing data sampling techniques in the bootstrap aggregating process. However, they are still prone to the effects of noise, given that bootstrap instances are always selected based on uniform probability distribution with replacement, hence creating the possibilities of randomly

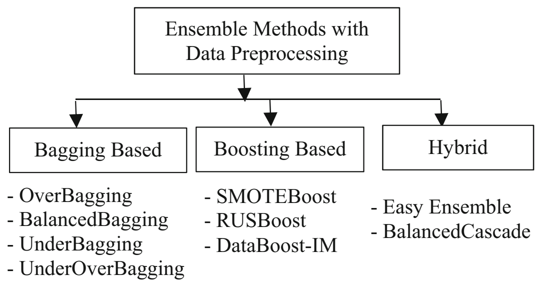

Fig. 1 Ensemble Methods utilizing Data Preprocessing

generating variety bootstraps with a high number of noisy instances that might be hard to classify, which might eventually affect the overall classification performance of an ensemble.

One of the key ideas in our proposed new method is not to make the sampling probability distribution uniform but instead a function of instance hardness. We aim to influence the process of picking instances, unlike most bagging methods that use a uniform probability to select instances. Our approach will select instances based on their level of hardness, i.e., the probability of an instance being selected will be equal to its hardness level; this will ensure that in each bootstrap/bag there is a representation of easy, normal, and hard instances. We discuss instance hardness in detail in Sect. 2.

Furthermore, there have been numerous studies pointing out defects of existing ensemble methods with their underlying data sampling techniques for handling class imbalance problems [13, 51]. Most of the proposed solutions alter the original dataset by either creating new data through oversampling methods that may increase the possibility of overfitting or by eliminating data from the majority class through undersampling techniques that may discard potentially useful data that might be important during the learning process. As discussed earlier, there are many widespread methods for building diverse ensembles classifiers, such as Bagging, AdaBoost, Random Forests, Random Subspaces, etc. [52], while each of these methods can be presented to datasets that have undergone sampling, this is not ideal as it disregards the power of joining the ensemble generation method and sampling to create a more structured approach. As an outcome, several ensemble techniques have been combined with sampling techniques to create ensemble methods that are more appropriate for handling class imbalance problem such as UnderBagging [53], OverBagging [54], SMOTE Bagging [54], Balanced Bagging [55], RUSBoost [12] among others. It is imperative to note that since most of these ensemble methods are based on sampling techniques that alter the original class distribution, they may, therefore, inherit the defects of sampling-based methods as earlier discussed [56].

In our method, we propose to generate balanced bags by using the split Balancing (sBal) technique and instance hardness (IH) as a weighting mechanism during the sampling process. Each of the generated bags will contain all the minority instances, and a mixture of majority instances with varying degrees of hardness (easy, normal, and hard). This will ensure that base learners are trained on balanced bags, containing diverse data segments with different levels of hardness to learn various patterns over a different portion of the data.

We hypothesize that generating several balanced bags that contain instances with varying degrees of hardness, will induce a set of base algorithms that are able to learn patterns in the data induced from data points that represent the overall difficulty of dividing the input space into different classes. We expect that an ensemble constructed with such base algorithms should have a better overall classification performance as compared to an ensemble that is constructed with base algorithms trained on just balanced or imbalanced bags that are uniformly sampled, regardless of their importance in identifying the classes.

We carry out an extensive empirical study of our proposed method and compare its performance with existing widely used state-of-the-art ensemble methods. We have structured our experimental framework in a way that enables us to extract justified conclusions. We used three sets of datasets for our experiments, the first set is composed of 30 synthetic imbalanced datasets with controlled levels of noise obtained from KEEL repository [57], the second set is composed of 29 real-world balanced datasets and the final set is composed of 41 real-world imbalanced datasets obtained from KEEL and UCI repositories. We further, conducted a nonparametric Friedman test and Bergmann's post hoc statistical test, at a significant level of $p < 0.05$, to ascertain our findings.

The results reveal that training base algorithms on balanced bags with varying degrees of hardness can bring substantial improvement in the classification performance of an ensemble. The findings demonstrate that the proposed method performed significantly better than the regular ensemble methods (Bagging, Wagging, Random Forest, and Adaboost) on both synthetic and real-world balanced and imbalanced datasets except for Random Forest where performance was comparable when evaluated on real-world balanced datasets. Furthermore, the proposed method performed better than ensemble methods specialized for class imbalance problems (Balanced Bagging, Balanced Random Forest, RUSBoost, and Easy Ensemble) in the majority of both balanced and imbalanced datasets.

In summary, the main contribution of this paper is twofold. First, we propose an ensemble method based on a data-level sampling approach to balance ensemble bags (sBal_IH), which takes data complexity into account in order to find good representative points within the bags, to improve prediction accuracy in imbalanced datasets.

Secondly, we conducted extensive experiments on 100 datasets and evaluated the proposed method in comparison with state-of-the-art methods, both standard ensembles and those specialized for handling class imbalance problems. The corresponding results, validated by statistical significance tests, demonstrate that our innovative method of split-balancing based on data complexity, proxied by instance hardness (IH), significantly outperformed most of the other compared methods as measured by Area Under the Curve (AUC) performance. We believe that this study will significantly contribute to the efforts of the machine learning community on addressing the challenge of data imbalance.

The remainder of this paper is organized as follows. We present our proposed method in Sect. 2, followd by a detailed experimental design in Sect. 3. In Sect. 4, we report and discuss in detail experimental and statistical results. We finally make conclusions and propose future works in Sect. 5.

2 Proposed method

We base our proposed method on the idea of making balanced bags of instances, where each bag contains a mixture of varying degrees of data complexity. To achieve this, we utilize Instance Hardness as a measure of data complexity.

2.1 Instance hardness (IH)

It is well known in the machine learning community, that the performance of most classifiers is dependent on both their parameters and the underlying training dataset [14, 59]. However, most researchers mainly focus on model parameter tuning as a way of achieving better model performance while ignoring the understanding of the data that is being modeled by the classifier. As a result, it may be difficult to understand which instances are being misclassified and why they are being misclassified, on assumption that the right parameters and the right evaluation metric are being used. Instance Hardness (IH) is a measure that specifies the degree of complexity in classifying a given instance in a dataset [60]. This implies that each instance in a respective dataset has a property that suggests its probability of being classified incorrectly regardless of the choice of the classifier. For example, we anticipate having high IH among outliers and mislabeled instances since the classifier will most likely have to overfit in order to classify them correctly. IH examines classification problems at the instance level as compared to the majority of machine learning studies that are focused on dataset level and mostly concerned with maximizing $p(f|s)$, where $f : X \rightarrow Y$ is a function that maps input feature vector X in the input space to their corresponding label vectors Y, and $s = \{(x_i, y_i) : x_i \in X \wedge y_i \in Y\}$ is the training set. On assumption that the pairs in s are drawn independently and identically distributed (i.i.d).

M. Smith et al. in their study [60] presented the notion of IH through the decomposition of $p(f|s)$, while using the Bayes' theorem:

$$p(f|s) = \frac{p(s|f).P(f)}{p(s)}$$
$$= \frac{\prod_{i=1}^{|s|} p(x_i, y_i|f).p(f)}{p(s)}$$
$$= \frac{\prod_{i=1}^{|s|} p(y_i|x_i, f)p(x_i|f)p(f)}{p(s)} \tag{1}$$

Furthermore, for (x_i, y_i) as a training instance, $p(y_i|x_i, f)$ measures the probability that f will assign the label y_i to the input feature vector x_i. M. Smith et al. further state that the larger the $p(y_i|x_i, f)$ the more likely f will assign the right label to x_i; on the other hand, the smaller the $p(y_i|x_i, f)$ the less likely for f to produce the right label for x_i. Therefore, they define IH with respect to f. as

$$IH_f(x_i, y_i) = 1 - p(y_i|x_i, f)$$

Under normal practice, f is induced by an algorithm c being trained on the training set s. Hence, the hardness of an instance is reliant on the instances in the training set and the underlying algorithm used to produce f.

In the literature, M. Smith et al. in their study [60] proposed several IH measures such as k-Disagreeing Neighbors (kDN), Disjunct Size, Disjunct Class Percentage (DCP), Class Likelihood (CL), Class Balance (CB) among others. These measures measure several characteristics about the hardness level of a specific instance; they show why instances are misclassified hence giving an insight as to why specific instances are hard to classify and how best we can detect them. This has laid a base foundation for researchers on dealing with dataset complexities as a result, and many state-of-the-art machine learning classification studies have been proposed based on instance hardness.

Previous studies in the literature have utilized instance hardiness in different ways in order to improve classification performance, such as noise and outlier filtering [61, 62], boosting through weight adjustments, etc. A. Kabir et al. in their study [63] proposed a mixed bagging technique for non-class imbalance problems, that incorporates IH in the bootstrap aggregating process. Furthermore, researchers in [64] incorporated hardness ordering under the learning process using filtering and boosting; they significantly improved generalization accuracy.

Our proposed method utilizes the concept of IH in the classification process but in a basically different way. While bagging based methods have been reported to offer good classification performance, their bootstrap sampling process is still prone to the effects of outliers or noise, given that instances are randomly sampled based on uniform probability, thus creating the possibilities of having a high percentage of outliers or noisy instances in some bootstraps, which might eventually affect classification performance. For our case, we propose using IH as a weighting mechanism during the sampling process, we then incorporate IH information in the ensemble bootstrapping process, we ensure each bootstrap has a representation of instances with varying degree of hardness (we discuss in detail IH estimation and implementation in Sect.3) that will allow base algorithms to learn different patterns of the training data, and that an ensemble constructed with such base algorithms should have a better overall classification performance.

2.2 Split balancing (sBal)

Most popular machine learning binary classification algorithms are designed to perform better on balanced datasets [65], and under such circumstances, it is always easy to choose the right algorithm and the right performance evaluation metric that will truly represent your optimal model [66, 67]. But the challenges always start to emerge when faced with an imbalanced dataset since the majority of these algorithms tend to perform poorly where the cost of classifying the minority class is always much higher as compared to the cost of classifying the majority class [68]. Thus, several techniques that try to balance the imbalanced datasets have been proposed and given much attention. However, numerous studies have pointed out the defects of proposed solutions for handling the class imbalance problem [51], i.e., they alter the original class distribution of the dataset by either creating new data through oversampling that might likely lead to overfitting [15], or by discarding potentially useful data from the majority class through undersampling [69]. Moreover, ensemble techniques have been combined with sampling techniques to create ensemble methods that are more appropriate for handling the class imbalance problem [8]. There is a variety of widespread approaches for building ensembles that are diverse such as Random Forests [70], Random Spaces [71], Bagging [45], AdaBoost [72], and many more. Whereas each of these approaches can be applied to datasets that have gone through sampling, but in an actual sense this is not optimal as it disregards the combining power for generating ensemble methods and sampling to create a better-organized approach. Consequently, several ensemble methods have ended up being combined with sampling techniques to create suitable ensemble methods for dealing with a problem of class imbalance. Chawla et al. in their study [10] proposed a novel approach SMOTEBoost for addressing the class imbalance problem, their approach is based on Synthetic Minority Oversampling TEchnique (SMOTE) [73] and boosting techniques. In other studies, Seiffert et al. proposed RUSBoost [12], a hybrid ensemble-based method that combines the RUS approach with the boosting technique.

However, most of these ensemble-based methods, are based on data sampling techniques and therefore they may alter the class distribution of the original datasets by either eliminating the majority class samples (undersampling) or by increasing the minority class samples (oversampling). Furthermore, for the case of boosting and bagging ensemble-based methods, they might still suffer from a problem of class imbalance because for each iteration (for boosting and bagging methods) the class distribution in each sampled subset in a certain iteration is the same as that of the original dataset. It is, therefore, prudent to convert the imbalanced dataset into multiple balanced datasets (bags) without creating new extra data or discarding potentially useful original data.

Our proposed ensemble-based methods will try to tackle the possible shortfalls of the conventional methods stated above for handling class imbalance problems by first converting a problem of class imbalance into several balanced problems that do not suffer anymore from the challenge of class imbalance without creating new extra data or discarding potentially useful original data, hence making it unique from the conventional methods for handling class imbalance problem.

Our proposed method is based on a split balancing technique [74], dubbed as sBal, where we generate balanced bags by randomly splitting instances of the majority class into multiple bags and each of them containing all the minority instances and a sample of the majority instances. However, we introduce the additional constraint that sampled majority instances should have varying degrees of hardness (easy, normal, and hard). These majority instances will be sampled based on IH as a weighting mechanism. This will ensure that base learners are trained on data points with different levels of hardness that we believe will better represent the input space that the dataset represents. We then combine the binary classifiers into an ensemble to classify new unseen data. The overall method is shown in Fig. 2.

3 Experiment design

In this section, we present the details of the several experiments which we used to evaluate the effectiveness of our proposed sBal_IH ensemble method and compare its performance against existing regular ensemble methods and state-of-the-art ensemble methods specialized for solving class imbalance problems.

The proposed method is first compared with regular ensemble methods that include: Bagging, AdaBoost, Random Forest (RF), and Wagging [75], on both balanced and imbalanced datasets to assess the performance of sBal_IH in situations of balanced and imbalanced problems.

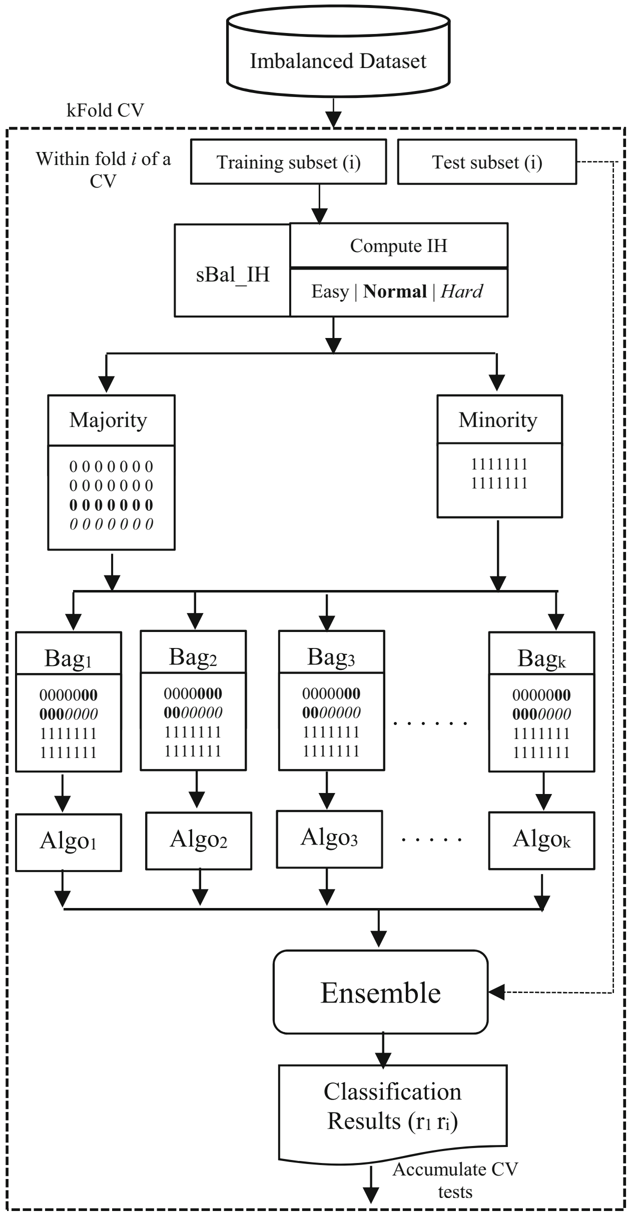

Fig. 2 Proposed Method—sBal_IH

Table 1 Synthetic Imbalanced Datasets with Controlled Noise

ID	Dataset	#Feat	#Inst	%Maj	%Min	IR
S1	03subcl5-600–5-0	2	600	83.3	16. 7	5
S2	03subcl5-600–5-30	2	600	83.3	16. 7	5
S3	03subcl5-600–5-50	2	600	83.3	16. 7	5
S4	03subcl5-600–5-60	2	600	83.3	16. 7	5
S5	03subcl5-600–5-70	2	600	83.3	16. 7	5
S6	03subcl5-800–7-0	2	800	87.5	12.5	7
S7	03subcl5-800–7-30	2	800	87.5	12.5	7
S8	03subcl5-800–7-50	2	800	87.5	12.5	7
S9	03subcl5-800–7-60	2	800	87.5	12.5	7
S10	03subcl5-800–7-70	2	800	87.5	12.5	7
S11	04clover5z-600–5-0	2	600	83.3	16. 7	5
S12	04clover5z-600–5-30	2	600	83.3	16. 7	5
S13	04clover5z-600–5-50	2	600	83.3	16. 7	5
S14	04clover5z-600–5-60	2	600	83.3	16. 7	5
S15	04clover5z-600–5-70	2	600	83.3	16. 7	5
S16	04clover5z-800–7-0	2	800	87.5	12.5	7
S17	04clover5z-800–7-30	2	800	87.5	12.5	7
S18	04clover5z-800–7-50	2	800	87.5	12.5	7
S19	04clover5z-800–7-60	2	800	87.5	12.5	7
S20	04clover5z-800–7-70	2	800	87.5	12.5	7
S21	paw02a-600–5-0	2	600	83.3	16. 7	5
S22	paw02a-600–5-30	2	600	83.3	16. 7	5
S23	paw02a-600–5-50	2	600	83.3	16. 7	5
S24	paw02a-600–5-60	2	600	83.3	16. 7	5
S25	paw02a-600–5-70	2	600	83.3	16. 7	5
S26	paw02a-800–7-0	2	800	87.5	12.5	7
S27	paw02a-800–7-30	2	800	87.5	12.5	7
S28	paw02a-800–7-50	2	800	87.5	12.5	7
S29	paw02a-800–7-60	2	800	87.5	12.5	7
S30	paw02a-800–7-70	2	800	87.5	12.5	7

Thereafter, we compare sBal_IH against ensemble methods specialized for class imbalanced problems, namely Balanced Bagging (BB), Balanced Random Forest (BRF), Easy Ensemble (EE), and Random Undersampling Boosting (RUSBoost), on both synthetic and real-life public datasets, balanced and imbalanced. In Sect. 3.1, we briefly introduce all the ensemble methods studied in this paper alongside our proposed method.

We organized our experiments and their analysis into 2 stages. In the first stage, we conduct experiments on different groups of datasets to facilitate the analysis of results. In one group, experiments are performed on synthetic imbalanced datasets (Table 1) with controlled levels of noise (disturbance ratio) to examine the influence of noise on the performance of sBal_IH alongside existing ensemble methods in terms of AUC. To further examine the performance of our proposed method in real-world situations against other ensemble methods, we performed experiments on another 2 groups of 29 balanced and 41 imbalanced real-world datasets. For evaluating the performance results of all the methods, we used a fivefold cross-validation technique repeated 5 times, where each fivefold cross-validation is calculated 5 different times with different random seeds. For fairness and uniformity, we used an ensemble size of 10 (n_estimators), and a Decision Tree algorithm with its default parameters as a base estimator for all the ensemble methods apart from Easy Ensemble, which uses AdaBoost as a base estimator. We considered the Easy ensemble method as part of our experiments to compare performance against a hybrid (ensemble of ensembles) method.

In the second stage, we performed a series of statistical comparisons of sBal_IH against existing ensemble techniques. The goal of this analysis is to validate whether there is a significant difference in the performance in terms of AUC, which will enable us to draw more confident conclusions.

All the experiments were carried out in Python 3.7 using scikit-learn library [76] version 0.21.3, and imbalanced-learn library [77] version 0.5. For statistical significance tests, we used KEEL's nonparametric statistical analysis software [57], version 3.0. The computer used for conducting the experiments was running Windows 10 64 bit, with an Intel Xeon processor (2.5 GHz) and 12 GB of RAM.

3.1 Featured ensemble methods

We briefly introduce the 8 ensemble methods evaluated in this study. Four of them are regular ensemble methods, namely Bagging, AdaBoost, Random Forest (RF), and Wagging, and the other four are ensemble methods specialized for class imbalance problems, namely Balanced Bagging (BB), Balanced Random Forest (BRF), Easy Ensemble (EE) and Random Undersampling Boosting (RUSBoost). The motivation for selecting these methods is that they are widely used and the majority of the studies on ensemble methods and class imbalance problems include one or more of these methods. They are all discussed comprehensively in the literature, such as in the studies of [36] and [39] mentioned above, and of [46] and [75] that will be referred afterward.

Bagging, also referred to as Bootstrap Aggregating, was first introduced by Breiman [45], and today is one of the most intuitive and possibly the simplest ensemble-based methods. It involves training different classifiers with bootstrapped copies of the data points randomly drawn with replacement from the original training dataset. The decisions of individual classifiers are then combined through majority voting.

Wagging (Weights Aggregation) is a variant of Bagging that was proposed by Bauer et al. [75], it needs a base algorithm that utilizes training instances with different weights. It randomly assigns weights to the instances that are available in each training set rather than using bootstrap samples to establish successive training set. It uses Gaussian noise to adjust the weight of instances, this might reduce the weights of some instances to zero, hence efficiently eliminating them from the training set.

Balanced Bagging [9], also referred to as Blagging, is an extension of bagging for solving class imbalance problems. The idea is to continuously undersample instances from the majority class in each of the bootstraps in order to get balanced bootstraps, on which individual decision trees are trained. This leads to an emphasis on the minority class and more balanced decisions.

Random forest [70] consists of a big number of individual decision trees that work as an ensemble. The random trees are generated using bootstrap samples of the training data and a set of selected random features during the tree induction process.

Balanced Random Forest (BRF) [78] is a variant of RF that was designed to deal with the problem of class imbalance. It utilizes the undersampling technique in each iteration of RF by drawing bootstrap instances from the minority class, and then drawing the same number of instances with replacement from the majority class, and then induces classification trees (CART algorithm) from the data in each iteration.

AdaBoost algorithm was developed by Schapire and Freund [46, 72]; it uses the whole training set to train multiple classifiers in a serial format. In each round, AdaBoost gives more attention to misclassified instances, with an aim to correctly classify them in subsequent iterations. This is achieved by maintaining a set of weights of the training instances, which are initially equal, and get updated according to correct or incorrect classification.

RUSBoost [79], a variation of the SMOTEBoost algorithm [10], works in a similar way to AdaBoost, but it discards instances from the majority class by Random Undersampling (RUS) in each iteration. In a previous study [80], it was shown that RUS often outperforms SMOTE, and thus RUSBoost being preferred as an alternative method to SMOTEBoost due to its simplicity and lower computational cost.

Easy Ensemble (EE) was proposed by Liu et al. [14]. It involves generating balanced samples of the training set by selecting all minority instances from the majority instances. Then boosted decision trees are induced on each balanced subset, particularly the AdaBoost algorithm.

3.2 Experiment datasets

As mentioned earlier (Sect. 3), we use three groups of datasets in our experiments, all of them prepared for binary classification tasks in mind. The first group is shown in Table 1, which was obtained from KEEL repository [57] and is composed of 30 synthetic imbalanced datasets with controlled levels of noise, i.e., disturbance ratio (the last number in dataset name) applied to create noisy examples in the dataset. This set was used in [81] to investigate the effect of noise and borderline instances from the minority class on the performance of the classifier. We chose to add this group to evaluate the effectiveness of our proposed method at handling imbalanced datasets in the presence of noise compared to other methods. The second group is composed of 29 real-world balanced datasets as indicated in Table 2. We included this group to examine in a fair way if our proposed method will show a distinctive effect if the dataset was not actually imbalanced. The final group is composed of 41 real-world imbalanced datasets as shown in Table 3. Both groups were obtained from KEEL [57] and UCI repositories. Tables 1, 2, and 3 summarize all the datasets used, highlighting some of their respective properties which include Dataset id (ID), Dataset name (Dataset), Total number of instances (#Inst), Percentage of Majority instances (%Maj), Percentage of minority instances (%Min) and Imbalance Ratio (IR). In this study, we consider IR as the ratio of percentages of Majority to Minority instances, i.e., $IR = \frac{\%Maj}{\%Min}$.

Table 2 Summary of Balanced Real-world Datasets Sorted by IR

ID	Datasets	#Feat	#Inst	%Maj	%Min	IR
B1	balance_scale_LR	4	576	50	50	1.0
B2	monks_prob_1	15	432	50	50	1.0
B3	teaching_assistant_LM	3	99	50.51	49.49	1.0
B4	teaching_assistant_MH	3	102	50.98	49.02	1.0
B5	teaching_assistant_LH	3	101	51.49	48.51	1.1
B6	kr-vs-kp	36	3196	52.22	47.78	1.1
B7	monks_prob_3	15	432	52.78	47.22	1.1
B8	diabetic_retinopathy	19	1151	53.08	46.92	1.1
B9	Sonar	60	208	53.37	46.63	1.1
B10	mammographic_mass	12	961	53.69	46.31	1.2
B11	arrhythmia_cfs	37	452	54.2	45.8	1.2
B12	heart_disease	13	303	54.46	45.54	1.2
B13	wine_c1c2	13	130	54.62	45.32	1.2
B14	contraceptive_NS	9	1140	55.18	44.82	1.2
B15	Phishing	30	2456	55.46	44.54	1.2
B16	credit-a	10	690	55.51	44.49	1.2
B17	banknote_authentication	4	1372	55.54	44.46	1.2
B18	wine_c2c3	13	119	59.66	40.34	1.5
B19	wine_c1c3	13	119	59.66	40.34	1.5
B20	contraceptive_LS	9	844	60.55	39.45	1.5
B21	voting_records	16	435	61.38	38.62	1.6
B22	Titanic	9	1309	61.8	38.2	1.6
B23	chronic_kidney	24	400	62.5	37.5	1.7
B24	horse_colic	18	368	63.04	36.96	1.7
B25	Ionosphere	34	351	64.1	35.9	1.8
B26	pima_indians_diabetes	8	768	65.1	34.9	1.9
B27	tic-tac-toe	9	958	65.34	34.66	1.9
B28	contraceptive_NL	9	962	65.38	34.62	1.9
B29	breast-w	9	699	65.52	34.48	1.9

3.3 Instance hardness estimation

As explained in Sect. 2, IH is the probability that an instance will be classified incorrectly by a classifier built from other instances of the same dataset. M. Smith et al. in their study [60] described various metrics for estimating IH; they empirically analyzed IH in over 190,000 instances of multiple datasets, using different learning algorithms. They established that there are always a good number of instances that are hard to be classified correctly.

The chosen metric used in our experiments estimates the hardness of a data instance as a percentage of a pre-selected set of classifiers that misclassify that instance. We considered the following algorithms for instance hardness estimation: Logistic Regression (LR), Decision Tree (DT), k-Nearest Neighbor (kNN), and Gaussian Naive Bayes (NB). This collection is a cocktail of both linear (LR) and nonlinear algorithms (kNN, DT, NB), and both parametric (NB, LR) and nonparametric algorithms (kNN, DT), as well as instance-based (kNN) and model-based (DT, NB, LR) ones. This enables a diverse and probably more reliable estimate of instance hardness.

We estimate Instance Hardness (IH) of each instance (data point) in a dataset as the percentage of incorrect classifications for that instance made by a pre-selected set of classifiers $C = \{c_1, c_2, .., c_m\}$ which are built from other instances in the dataset.

```
Algorithm 1: Computing Instance Hardness
Input: Dataset $T$ with n examples $[(x_i, y_i), .., (x_n, y_n)]$
       with labels $y_i \in Y = \{0, 1\}$
Input: Set of classification algorithms $C = \{c_1, c_2, .., c_m\}$
Input: $t_e$ and $t_h$ : IH thresholds for Easy & Hard instances
Output: Sets E, N, H for Easy, Normal, and Hard instances

1:  E, N, H ← Ø;
2:  for each instance $I_i$ in $T$ do
3:      $T_{ri} \leftarrow T - \{I_i\}$ ; /* training set without $I_i$ */
4:      $C'_i = 0$;
5:      for each classifier $C_j$ in $C$ do
6:          $M_{ij} \leftarrow$ use $C_j$ to build a classifier model $M_{ci}$
                using $T_{ri}$ ;
7:          $\hat{y}_{ij} \leftarrow$ label of $I_i$ as determined by $M_{ij}$ ;
8:          if $(\hat{y}_{ij} = y_i)$ then increment $C'_i$ ;
9:      end for
10:     $IH_i = \frac{C'_i}{m}$;
11:     if $(IH_i < t_e)$
12:         $E \leftarrow E \cup \{i\}$;
13:     else if $(IH > t_h)$
14:         $H \leftarrow H \cup \{i\}$;
15:     else
16:         $N \leftarrow N \cup \{i\}$;
17: end for
```

summarizes the steps that can be taken to calculate IH for every instance in a given dataset. The steps are laid out

Table 3 Summary of Imbalanced Real-world Datasets Sorted by IR

ID	Datasets	#Feat	#Inst	%Maj	%Min	IR
I-1	monks_prob_2	15	432	67.1	32.9	2.0
I-2	vertebral_column	6	310	67.7	32.3	2.1
I-3	credit-g	19	1000	70.0	30.0	2.3
I-4	car_evaluation	6	1728	70.0	30.0	2.3
I-5	breast_cancer	13	286	70.3	29.7	2.4
I-6	indian_liver_patients	10	583	71.4	28.6	2.5
I-7	haberman	3	306	73.5	26.5	2.8
I-8	page-blocks-1-3_vs_4	10	472	94.1	28.0	3.4
I-9	hepatitis	19	155	79.4	20.7	3.8
I-10	spect_heart	22	267	79.4	20.6	3.9
I-11	cleveland-0_vs_4	13	190	84.2	15.8	5.3
I-12	cardiotocography_c1c2	21	1950	84.9	15.1	5.6
I-13	balance_scale_BL	4	337	85.5	14.5	5.9
I-14	balance_scale_BR	4	337	85.5	14.5	5.9
I-15	internet_ad_cfs	24	3279	86.0	14.0	6.1
I-16	ecoli-0–3-4_vs_5	7	200	90.0	10.0	9.0
I-17	yeast-2_vs_4	8	514	90.1	9.9	9.1
I-18	ecoli-0–6-7_vs_3-5	7	200	90.1	9.9	9.1
I-19	yeast-0–3-5-9_vs_7-8	8	506	90.1	9.9	9.1
I-20	yeast-0–2-5–7-9_vs_3-6–8	8	1004	90.1	9.9	9.1
I-21	ecoli-0–4-6_vs_5	6	203	90.2	9.9	9.2
I-22	ecoli-0-1_vs_2-3–5	7	244	90.2	9.8	9.2
I-23	ecoli-0–3-4-6_vs_5	7	205	90.2	9.8	9.2
I-24	yeast-0–5-6–7-9_vs_4	8	528	90.3	9.7	9.4
I-25	cardiotocography_c1c3	21	1831	90.4	9.6	9.4
I-26	cardiotocography_c2c3	21	1831	90.4	9.6	9.4
I-27	vowel0	13	988	90.9	9.1	10.0
I-28	ecoli-0–1-4-7_vs_2-3–5-6	7	336	91.4	8.6	10.6
I-29	climate_model	18	540	91.5	8.5	10.7
I-30	led7digit-0–2-4–5-6–7-8-9_	7	443	91.7	8.4	11.0
I-31	ecoli-0–1-4-6_vs_5	6	280	92.9	7.1	13.0
I-32	shuttle-c0-vs-c4	9	1829	93.3	6.7	13.9
I-33	seismic-bumps	18	2584	93.4	6.6	14.2
I-34	yeast-1_vs_7	7	459	93.5	6.5	14.3
I-35	cervical_cancer_risk_facto	33	858	93.6	6.4	14.6
I-36	ecoli4	7	336	94.1	6.0	15.8
I-37	yeast-1–4-5-8_vs_7	8	693	95.7	4.3	22.1
I-38	yeast4	8	1484	96.6	3.4	28.1
I-39	yeast-1–2-8-9_vs_7	8	947	96.8	3.2	30.5
I-40	yeast5	8	1484	97.0	3.0	32.8
I-41	yeast6	8	1484	97.6	2.4	41.4

based on a Leave-One-Out (LOO) style to assess the hardness of each instance separately, using the remaining instances in the training set. Even though we use LOO in the layout of the algorithm for its simplicity, the calculation of IH can be made using a less compute-intensive approach, such as a less extreme style of cross-validation. In our implementation, we have used a variant based on fivefold cross-validation in order to save on computation times.

The hardness of an instance is represented by a real value between 0 and 1, where an instance with a hardness value close to 0 is more likely to be correctly classified, whereas the opposite is true for that instance with a hardness value close to 1. This implies that an instance with an estimated hardness value higher than a predefined threshold t_h is categorized as Hard (H), and with hardness value below the threshold t_e is categorized as Easy (E), or else as Normal (N) as shown in Fig. 3. In our study, we set the default IH thresholds for easy and hard instances as $t_e = 0.33$ and $t_h = 0.66$, respectively, to maintain equal ranges of hardness.

Even though we use the term "training dataset" in IH estimation above, we emphasize that this does not refer to the original imbalanced datasets in our study, but rather to the training subset in a cross-validation fold. In other words, for rigorous experiment design, IH estimation is performed multiple times, once in every fold, for each of the original imbalanced datasets. This is done to avoid data leakage from the test sets to the training phase as shown in Fig. 2.

3.4 Evaluation metric

An obvious challenge raised with evaluating classifiers on imbalanced datasets is choosing the right performance metric. Accuracy is a well-known metric frequently used for classification problems [82]; however, it might not be an appropriate metric as reported by many studies, for its ineffectiveness in situations of imbalanced datasets [67, 83, 84]. Instead of accuracy, researchers have adopted other evaluation metrics for imbalanced problems such as F-Measure, Recall, Precision, G-Means, Area Under Curve (AUC), and Mathew Correlation Coefficient (MCC). On precision and recall, it has been reported in [85] that precision is sensitive to data distribution while recall is not. With many metrics in place, researchers have encountered the challenge of choosing the right metric for imbalanced problems; most researchers in the machine learning community are preferring AUC [2, 3], and MCC [86] over other metrics. Researchers in [87] empirically compared and evaluated the performance of AUC against MCC on a series of real-world imbalanced datasets and established that AUC was statistically consistent and more discriminating than MCC; hence suggesting that AUC is a better measure than MCC to be used for evaluating binary classification with imbalanced datasets. In that regard, we

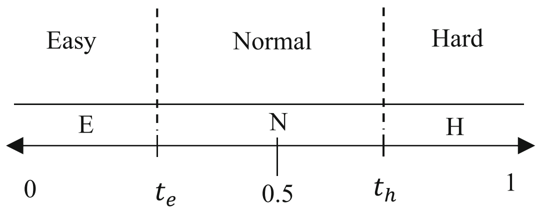

Fig. 3 Instance Hardness Threshold Ratio

chose to use AUC as an evaluation metric throughout our experiments.

4 Results and discussion

In this section, we present and discuss the outcomes of our experiments resulting from the fivefold cross-validation runs, and randomly recalculated with 5 different random seeds (as explained in Sect. 3.0). We tested our proposed method against regular ensemble methods; Bagging (Bag), Wagging (Wag), Random Forest (RF), AdaBoost (AdaB), and ensemble methods specialized for imbalanced problems; Easy Ensemble (EE), Balanced Bagging (BB), Balanced Random Forest (BRF) and RUSBoost (RUSB). These methods were evaluated on the 100 publicly available datasets: 30 synthetic imbalanced datasets (from Table 1), 29 balanced real-world datasets (from Table 2), and 41 imbalanced real-world datasets (from Table 3).

In all the tables, the results highlighted in bold indicate a higher AUC score as compared to the others in the same row for a given dataset. In case of a tie between methods, we consider it a win in its capacity.

4.1 Results for synthetic datasets

In this subsection, we present and discuss the performance of our proposed methods against regular and specialized ensembles methods for class imbalance problems when faced with imbalanced datasets perturbed with different levels of noise. This group of datasets will help us evaluate and understand the effectiveness of our proposed method, compared to the other methods, at handling noise in datasets as a proxy for underlying data complexities.

4.1.1 (a) sBal_IH Against Regular Ensemble Methods on Synthetic Datasets

We compared the proposed sBal_IH ensemble method with the regular ensemble methods in terms of AUC, evaluated on the 30 synthetic imbalanced datasets. Table 4 shows the mean AUC values of 5 repeated trials. The results highlighted in bold demonstrate the better performance of sBal_IH against all the other regular ensemble methods in 28 datasets out of 30. AdaBoost, despite its efforts of adding weight on difficult to classify instances, it ranked second, performing better in only 2 out of 30 datasets.

We can also observe that with the increase in noise level (disturbance ratio values increase every 5 rows in Table 4, from 0, 30, 50, 60, up to 70), performance starts to deteriorate proportionally for all ensemble methods; however, our proposed method struggles the least with this and still maintains higher performance. We believe that the reason

Table 4 AUC Score for sBal_IH Against Regular Ensemble Methods on 30 Synthetic Imbalance Datasets

Dataset ID	Bagging	RF	Wagging	AdabB	sBal_IH
S1	0.95	0.882	0.943	**0.969**	0.947
S2	0.788	0.749	0.789	0.802	**0.831**
S3	0.711	0.686	0.697	0.723	**0.787**
S4	0.668	0.666	0.681	0.696	**0.803**
S5	0.628	0.604	0.645	0.66	**0.777**
S6	0.952	0.865	0.968	**0.971**	0.96
S7	0.746	0.743	0.769	0.798	**0.836**
S8	0.68	0.634	0.674	0.714	**0.786**
S9	0.645	0.651	0.644	0.685	**0.777**
S10	0.601	0.569	0.619	0.628	**0.767**
S11	0.849	0.807	0.864	0.878	**0.931**
S12	0.737	0.716	0.758	0.783	**0.86**
S13	0.701	0.674	0.696	0.709	**0.832**
S14	0.677	0.666	0.678	0.726	**0.807**
S15	0.615	0.64	0.661	0.682	**0.804**
S16	0.794	0.774	0.836	0.843	**0.918**
S17	0.735	0.656	0.746	0.766	**0.844**
S18	0.658	0.622	0.66	0.673	**0.834**
S19	0.636	0.611	0.638	0.661	**0.813**
S20	0.609	0.578	0.61	0.661	**0.779**
S21	0.926	0.904	0.916	0.921	**0.94**
S22	0.803	0.792	0.789	0.805	**0.853**
S23	0.76	0.751	0.779	0.772	**0.852**
S24	0.711	0.71	0.714	0.729	**0.815**
S25	0.707	0.694	0.719	0.718	**0.84**
S26	0.906	0.902	0.898	0.911	**0.941**
S27	0.794	0.779	0.806	0.79	**0.833**
S28	0.732	0.718	0.736	0.757	**0.841**
S29	0.677	0.677	0.691	0.681	**0.812**
S30	0.641	0.635	0.668	0.682	**0.83**

for this is the ability of sBal_IH to still learn most of the underlying complexities existing within the dataset due to the use of learners trained on balanced bags which still have varying levels of hardness. This implies that our proposed method is more efficient in handling noise and class imbalance problems than the regular ensemble methods.

4.1.2 (b) sBal_IH against specialized ensemble methods for class imbalance problems on synthetic datasets.

Since the regular ensemble methods are not specially designed to handle class imbalance problems, we evaluate the effectiveness of our proposed method in comparison with the methods specialized for class imbalanced problems as mentioned in Sect.3. In Table 5, we present the performance of sBal_IH against ensemble methods in terms of AUC on the 30 synthetic imbalanced datasets.

The results highlighted in bold, show that out of the 30 datasets, sBal_IH outperformed the rest of the methods in 16 datasets, followed by BRF with only 6 wins. The results indicate that sBal_IH faced fair competition from state-of-the-art ensemble methods specialized for handling imbalanced datasets.

This is because most of these specialized methods try to take care of the class imbalance problem and the underlying complexities existing within the dataset. For instance, Balanced Random Forest (BRF) and Balanced Bagging (BB) are similar to our proposed method in trying to

Table 5 AUC Score for sBal_IH against ensemble methods specialized for class imbalanced problems on 30 synthetic imbalanced datasets

Dataset ID	EE	RUSB	BB	BRF	sBal_IH
S1	**0.95**	0.905	0.946	0.949	0.947
S2	0.825	**0.831**	0.82	0.816	**0.831**
S3	**0.816**	0.798	0.779	0.799	0.787
S4	**0.829**	0.803	0.792	0.82	0.803
S5	**0.805**	0.793	0.783	0.794	0.777
S6	**0.968**	0.933	0.948	0.939	0.96
S7	0.829	0.82	0.828	0.83	**0.836**
S8	0.801	**0.812**	0.791	0.797	0.786
S9	0.807	0.769	0.783	**0.819**	0.777
S10	**0.805**	0.763	0.766	0.788	0.767
S11	0.835	0.816	0.916	0.881	**0.931**
S12	0.795	0.765	0.842	0.844	**0.86**
S13	0.779	0.762	0.823	0.823	**0.832**
S14	0.799	0.782	0.802	**0.807**	**0.807**
S15	0.768	0.747	0.791	0.792	**0.804**
S16	0.819	0.805	0.908	0.881	**0.918**
S17	0.776	0.773	0.841	0.824	**0.844**
S18	0.771	0.785	0.816	0.811	**0.834**
S19	0.788	0.771	**0.823**	0.798	0.813
S20	0.768	0.745	0.79	**0.793**	0.779
S21	0.894	0.858	0.93	0.915	**0.94**
S22	0.819	0.797	0.834	0.844	**0.853**
S23	0.833	0.824	0.848	0.841	**0.852**
S24	0.816	0.798	0.804	**0.837**	0.815
S25	0.823	0.795	0.827	0.837	**0.84**
S26	0.905	0.867	0.94	0.925	**0.941**
S27	0.825	0.819	**0.861**	0.847	0.833
S28	0.816	0.799	0.822	**0.844**	0.841
S29	0.803	0.795	0.802	**0.826**	0.812
S30	0.806	0.798	0.828	0.821	**0.83**

balance bootstraps during the aggregating process. However, this happens in an interestingly different way; for BRF, they draw bootstrap instances from the minority class and then randomly draw the same number of instances, with replacement, from the majority class [78]. As for BB, in each bootstrap, they draw the same number of minority and majority instances based on the undersampling technique using a negative binomial distribution [55].

Nevertheless, for most of the specialized methods, as much as they try to balance the class distributions, there is a likelihood of drawing more complex or noisy instances into a bag or across many bags, since the sampling process is based on a uniform probability distribution, and this seems to have a negative effect on the overall classification performance of the ensemble as compared to what sBal_IH was able to achieve when the balanced bags were composed of a mixture of instances with varying degree of hardness.

4.2 Results for real-world datasets

We go ahead and compare the performance of sBal_IH against regular and specialized ensemble methods on real-world multi-domain datasets to further understand its performance behavior in the real-world setup.

4.2.1 (a) sBal_IH against regular ensemble methods on real-world datasets

Here, we compare sBal_IH with the regular ensemble methods in terms of AUC on the 29 balanced and 41 imbalanced real-world datasets. Tables 6 and 7 present the AUC performance scores on the mentioned datasets groups, respectively. In the experiments, we considered a group of balanced datasets in order to examine in a fair way if our proposed method will show a distinctive effect if the datasets were actually balanced. As highlighted in bold in Table 6, it is observed that sBal_IH outperformed the traditional ensemble methods in only 15 out of the 29 balanced datasets, followed by Random Forest with 13 wins, and the remaining methods far behind. The results show a comparable performance between sBal_IH and the other regular ensemble methods.

On the other hand, with the 41 real-world imbalanced datasets, the highlighted results in Table 7 show a similar pattern to that of the synthetic imbalanced datasets (subsection 4.1.a), where sBal_IH exhibits good performance in 36 out of the 41 real-world imbalanced datasets. This is an endorsement of the effectiveness of our proposed method in addressing the class imbalance problem as compared to the regular ensemble methods.

The observations above may indicate that when the data is already balanced, sBal_IH still benefited from the varying hardness in generated bags, but that benefit will magnify if the dataset is also imbalanced. This can be inferred from our findings of sBal_IH outperforming the regular ensemble methods when the datasets were imbalanced (synthetic and real-world) and is consistent with what other researchers have found [88], in that when the data is imbalanced, the negative effect of noise is magnified.

Table 6 AUC Score for sBal_IH Against Regular Ensemble Methods on 29 Balanced Datasets

Dataset ID	Bag	RF	Wag	AdaB	sBal_IH
B1	0.904	0.907	0.906	0.884	**0.931**
B2	0.993	0.91	**0.995**	0.921	**0.995**
B3	**0.616**	0.605	0.611	0.591	0.606
B4	0.665	0.663	0.65	**0.699**	0.654
B5	**0.636**	0.614	0.63	0.635	0.62
B6	0.993	0.979	0.993	**0.995**	0.994
B7	**1**	0.988	**1**	**1**	**1**
B8	0.634	0.656	0.629	0.612	**0.681**
B9	0.749	**0.79**	0.749	0.715	0.789
B10	0.766	**0.823**	0.77	0.77	0.803
B11	0.791	**0.827**	0.794	0.755	0.814
B12	0.758	**0.81**	0.766	0.753	0.787
B13	**0.977**	0.972	0.973	0.976	0.959
B14	0.631	0.658	0.633	0.61	**0.666**
B15	0.926	**0.938**	0.925	0.926	**0.938**
B16	0.843	0.847	0.825	0.811	**0.853**
B17	0.986	**0.99**	0.988	0.983	**0.99**
B18	0.939	**0.988**	0.96	0.951	0.986
B19	0.994	**1**	0.99	0.995	**1**
B20	0.575	0.61	0.575	0.585	**0.615**
B21	0.945	**0.96**	0.945	0.943	0.953
B22	0.757	**0.781**	0.754	0.749	0.766
B23	0.979	**0.996**	0.979	0.976	0.99
B24	0.789	0.81	0.801	0.776	**0.824**
B25	0.888	**0.913**	0.893	0.888	0.907
B26	0.688	0.721	0.697	0.654	**0.738**
B27	0.968	0.931	0.965	0.954	**0.981**
B28	0.672	0.69	0.664	0.69	**0.707**
B29	0.944	**0.957**	0.947	0.941	**0.957**

4.2.2 (b) sBal_IH against specialized ensemble methods for class imbalance problems on real-world datasets

Our proposed method was also compared against state-of-the-art ensemble methods specialized for handling imbalanced data on a group of 41 real-world multi-domain imbalanced datasets. The outcomes from this experiment,

Table 7 AUC Score for sBal_IH Against Regular Ensemble Methods on 41 Real-world Imbalanced Datasets

Dataset ID	Bagging	RF	Wagging	AdaB	sBal_IH
I-1	0.732	0.493	0.774	**0.944**	0.64
I-2	0.803	0.797	0.789	0.775	**0.849**
I-3	0.655	0.663	0.651	0.643	**0.705**
I-4	0.992	0.978	0.992	0.989	**0.997**
I-5	**0.6**	0.589	0.584	0.589	0.598
I-6	0.608	0.588	0.596	0.596	**0.683**
I-7	0.547	0.568	0.558	0.575	**0.645**
I-8	0.945	0.928	0.955	0.979	**0.992**
I-9	0.664	0.656	0.668	0.663	**0.752**
I-10	0.7	0.681	0.7	0.686	**0.753**
I-11	0.895	0.897	0.914	0.935	**0.948**
I-12	0.861	0.859	0.871	0.874	**0.903**
I-13	0.503	0.509	0.509	0.511	**0.722**
I-14	0.548	0.522	0.511	0.502	**0.709**
I-15	0.926	0.932	0.928	0.928	**0.933**
I-16	0.875	0.87	0.854	0.813	**0.879**
I-17	0.841	0.876	0.845	0.83	**0.917**
I-18	0.838	0.79	0.834	**0.861**	0.86
I-19	0.637	0.581	0.635	0.656	**0.672**
I-20	0.877	0.864	0.874	0.848	**0.913**
I-21	0.856	0.867	0.853	0.813	**0.889**
I-22	0.83	0.779	0.87	0.832	**0.874**
I-23	0.875	0.87	0.854	0.813	**0.879**
I-24	0.646	0.589	0.647	0.662	**0.787**
I-25	0.95	**0.953**	0.951	0.947	0.95
I-26	0.948	0.94	0.944	0.945	**0.958**
I-27	0.95	0.947	0.953	0.944	**0.967**
I-28	0.831	0.783	0.797	0.799	**0.853**
I-29	0.656	0.533	0.605	0.673	**0.883**
I-30	0.886	0.87	0.88	0.888	**0.895**
I-31	0.789	0.859	0.813	0.836	**0.873**
I-32	**1**	**1**	**1**	**1**	**1**
I-33	0.534	0.508	0.535	0.554	**0.653**
I-34	0.649	0.606	0.638	0.639	**0.684**
I-35	0.758	0.713	0.805	0.742	**0.921**
I-36	0.83	0.81	0.868	0.854	**0.898**
I-37	0.491	0.499	0.492	0.536	**0.598**
I-38	0.648	0.557	0.654	0.656	**0.813**
I-39	0.611	0.549	0.594	0.6	**0.614**
I-40	0.795	0.759	0.78	0.836	**0.967**
I-41	0.703	0.649	0.732	0.713	**0.867**

Table 8 AUC Score for sBal_IH Against Ensemble Methods Specialized for Class Imbalanced Problems on 41 Imbalanced Datasets

Dataset ID	EE	RUSBoost	BB	BRF	sBal_IH
I-1	**0.937**	0.51	0.733	0.532	0.64
I-2	0.797	0.818	0.815	0.828	**0.849**
I-3	0.682	0.688	0.693	0.697	**0.709**
I-4	0.994	0.952	0.995	0.972	**0.997**
I-5	0.581	**0.64**	0.6	0.604	0.598
I-6	0.675	0.655	0.675	0.676	**0.683**
I-7	0.62	0.605	0.587	**0.654**	0.645
I-8	**0.992**	0.977	0.989	0.951	**0.992**
I-9	0.739	0.731	0.704	**0.767**	0.752
I-10	0.718	0.748	0.749	0.744	**0.753**
I-11	0.951	0.94	**0.955**	0.926	0.948
I-12	**0.91**	0.877	0.896	**0.91**	0.903
I-13	**0.772**	0.717	0.736	0.717	0.722
I-14	**0.771**	0.768	0.737	0.698	0.709
I-15	0.934	0.864	0.931	0.932	**0.941**
I-16	0.85	0.906	0.865	**0.92**	0.879
I-17	**0.935**	0.833	0.907	0.894	0.917
I-18	0.849	0.819	0.849	0.826	**0.86**
I-19	**0.702**	0.675	0.671	0.681	0.672
I-20	0.896	0.858	0.909	0.887	**0.914**
I-21	0.854	0.863	0.852	0.878	**0.889**
I-22	0.871	0.85	**0.878**	0.856	0.874
I-23	0.857	0.841	0.865	**0.92**	0.879
I-24	0.78	0.708	0.751	0.783	**0.787**
I-25	0.958	0.891	0.956	0.958	**0.959**
I-26	0.941	**0.96**	0.952	0.942	0.958
I-27	**0.967**	0.917	0.966	0.965	**0.967**
I-28	0.835	0.823	**0.863**	0.852	0.853
I-29	0.873	0.807	0.871	0.842	**0.883**
I-30	0.866	0.861	0.883	0.852	**0.895**
I-31	0.854	0.875	0.854	**0.951**	0.873
I-32	**1**	**1**	**1**	0.998	**1**
I-33	0.665	**0.7**	0.652	0.69	0.653
I-34	0.688	0.612	**0.762**	0.74	0.684
I-35	0.904	0.831	0.915	0.896	**0.921**
I-36	0.916	0.894	0.904	**0.928**	0.898
I-37	**0.767**	0.603	0.63	0.625	0.598
I-38	**0.834**	0.804	0.798	0.833	0.813
I-39	0.651	0.593	0.636	**0.706**	0.614
I-40	0.967	0.904	0.96	0.948	**0.968**
I-41	0.855	0.84	0.854	**0.875**	0.867

if good enough, will position the sBal_IH method among the best alternative ensemble methods for addressing the problem of class imbalance.

In Table 8, we observe a better performance of the proposed method (sBal_IH) across 18 out of the 41 imbalanced real-world datasets. This is followed by Easy Ensemble (EE) with 11 wins and Balanced Random Forest coming third with only 9 wins. RUSBoost and Balanced Bagging were the least performing in terms of AUC.

We note that even though there might be a reasonable competition in performance between sBal_IH and EE method, sBal_IH still outperformed EE even though EE is considered a hybrid ensemble (an ensemble of ensembles), since it uses AdaBoost ensemble as its default base algorithm, hence having a higher advantage over sBal_IH.

Lastly, we curiously assessed the performance of sBal_IH against the specialized ensemble methods in situations where the data is actually balanced, i.e., on the 29 balanced real-world datasets.

The results in Table 9 show a remarkable performance of sBal_IH against the other methods, with 16 wins out of 29 balanced datasets, followed by Easy Ensemble with 6 wins, and RUSBoost registering the least performance. The findings suggest that sBal_IH significantly outperforms specialized ensemble methods when there is no class imbalance problem.

Table 9 AUC Score for sBal_IH Against Specialized Methods for Class Imbalanced Problems on 29 Balanced Datasets

Dataset ID	EE	RUSBoost	BB	BRF	sBal_IH
B1	0.884	0.928	0.924	0.929	**0.938**
B2	0.945	0.75	**0.992**	0.973	0.978
B3	**0.611**	0.574	0.594	0.564	0.552
B4	**0.66**	0.586	**0.66**	0.611	0.615
B5	**0.655**	0.593	0.619	0.601	0.632
B6	**0.995**	0.95	0.994	0.979	0.982
B7	**1**	**1**	**1**	0.997	0.996
B8	0.661	0.646	0.656	0.657	**0.683**
B9	0.782	0.766	0.778	0.761	**0.786**
B10	0.771	**0.826**	0.778	0.819	0.822
B11	0.806	0.772	**0.834**	0.805	0.816
B12	0.754	0.795	0.784	0.8	**0.815**
B13	0.97	0.982	0.969	**0.99**	0.982
B14	0.99	0.995	**1**	**1**	**1**
B15	0.631	0.682	0.639	0.667	**0.687**
B16	0.921	0.922	0.929	0.944	**0.945**
B17	0.83	0.849	0.84	0.858	**0.868**
B18	0.986	0.962	0.988	0.991	**0.994**
B19	0.973	0.968	0.979	0.981	**0.986**
B20	0.573	**0.644**	0.619	0.612	**0.644**
B21	0.948	0.951	0.947	**0.957**	0.955
B22	0.759	0.766	0.775	0.783	**0.786**
B23	0.984	0.996	0.986	0.994	**0.997**
B24	0.791	0.78	0.801	**0.822**	0.81
B25	0.88	0.835	0.893	0.907	**0.918**
B26	0.723	0.714	0.73	0.73	**0.745**
B27	**0.976**	0.689	0.972	0.942	0.965
B28	0.683	0.715	0.692	0.72	**0.733**
B29	0.954	0.942	0.957	**0.974**	0.969

Throughout the experiments, we observe a consistent and good performance of the proposed method across the majority of the datasets. The intuitive explanation for this is that when a classifier is trained on a section of instances from a dataset with varying degrees of hardness, the classifier will have a better chance to learn the underlying pattern to classify the data while maintaining two properties: learn from hard instances as well as easy ones, and at the same time not overfit to those that are easy or hard. Hence, producing a consistent and noise-tolerant (robust) model that can better deal with both balance and class imbalanced problems.

In order to draw more reliable conclusions, in the next subsection, we discuss Friedman's nonparametric statistical test used to ascertain if there exists any significant difference in performance between sBal_IH and the other ensemble methods.

4.3 Statistical tests

Throughout our experiments, we observed a consistent trend in performance advantage that cuts across most of the studied methods. However, in order to draw reliable conclusions, it is important to determine if there exists a statistically significant difference in the classification performance as measured by AUC. For that purpose, we utilized the standard methodology proposed by Demšar [89] for testing statistical significance among multiple methods across various datasets. We carried out a nonparametric (distribution-free) statistical test on all mean AUC results. We chose to use [90] Friedman's test because we have no knowledge of the distribution of the values in our analyzed data.

Friedman test is a nonparametric statistical test equivalent to the test of repeated ANOVA. It computes and ranks the algorithms for each dataset separately, where the best performing algorithm is assigned rank 1, second with rank 2, and so on. In case there is a tie, it assigns an average rank to the affected algorithm. Suppose, r_i^j is the rank of the $j-th$ algorithm (of k algorithms) on the $i-th$ dataset (of N datasets). The Friedman test calculates the average rank of an algorithm j, $R_j = \frac{1}{N}\sum_i r_i^j$.

The Friedman statistic, χ_F^2, is calculated with $k-1$ degrees of freedom, F, as shown in Eq. 2:

$$\chi_F^2 = \frac{12N}{k(k+1)}\left[\sum_j R_j^2 - \frac{k(k+1)^2}{4}\right] \qquad (2)$$

For this test, the Null Hypothesis (H_0) states that all the algorithms are comparable and the observed differences in their ranks might be random. Whereas the Alternative Hypothesis (H_1) states that, there is a statistically significant difference among the algorithms.

Our interest is to reject the null hypothesis; in case the $H_0$is rejected, we go-ahead to carry out a post hoc test in order to establish specific pairs of algorithms that produce differences. Many post hoc tests have been proposed in the literature [91], such as Nemenyi's, Holm's, Bergmann's, and Shaffer's tests, among others, as shown in Fig. 4. A detailed discussion of nonparametric tests and their corresponding post hoc tests is presented in [89] and [91].

For our study, we used Friedman's N $\times$ N statistical test to first determine if there exists any statistically significant difference among methods being studied across the datasets, followed by a post hoc Bergman procedure to compute a probability value (p value) for the test on each pair of methods. We choose to Bergman post hoc procedure because of its high statistical power devoted to multiple comparisons. Bergman's procedure has been recommended in the literature for this kind of study since it exhaustively finds all the possible sets of hypotheses for given comparisons and all those elementary hypotheses that cannot be rejected [91].

We carried out the statistical tests using KEEL's nonparametric statistical analysis tool [57]. We present the outcome of the statistical analysis in the next subsection.

4.4 Statistical analysis results.

As an outcome of the statistical analysis of the performance of our proposed method (sBal_IH) against existing popular ensemble methods when using Friedman/Bergman methods with a significance level of $\alpha = 0.05$, we present our results in Tables 10–15. We highlight with bold all pairs that comprise the sBal_IH method. In the results column, we indicate whether the null hypothesis is rejected or accepted, and we are much interested in determining statistically significant performance differences between

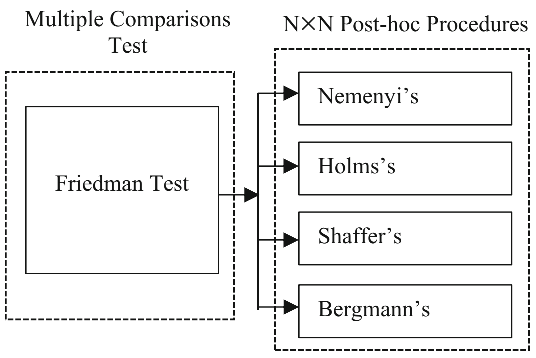

Fig. 4 Nonparametric Tests and Post hoc Procedures for N × N Comparisons

Table 10 Statistical Results for Data from Table 4—sBal_IH vs Regular Ensemble Method on Synthetic Imbalanced Datasets

Comparison	$z = (R_0 - R_1)/SE$	p value	Result
sBal_IH vs RF	**9.02229**	**0**	**H0 rejected**
RF vs AdabB	6.5728	0	H0 rejected
sBal_IH vs Bag	**6.32785**	**0**	**H0 rejected**
sBal_IH vs Wag	**5.06228**	**0**	**H0 rejected**
Wag vs RF	3.96001	0.000075	H0 rejected
Bag vs AdabB	3.87836	0.000105	H0 rejected
Bag vs RF	2.69444	0.007051	H0 rejected
Wag vs AdabB	2.61279	0.008981	H0 rejected
sBal_IH vs AdabB	**2.44949**	**0.014306**	**H0 rejected**
Bag vs Wag	1.26557	0.205667	H0 accepted

sBal_IH and other ensemble methods across a series of datasets.

For the data resulting from Tables 4 and 5, where we studied sBal_IH against regular and specialized ensemble methods, respectively, on a series of synthetic imbalanced datasets with controlled levels of noise, we include the statistical analysis results in Tables 10 and 11.

It is observed in Table 10 that H_0 is rejected across all pairs of algorithms involving sBal_IH. This shows that there is a significant difference in performance between sBal_IH and all traditional ensemble methods on the synthetic imbalanced datasets.

When it comes to specialized ensemble methods for class imbalance problems, we observe in Table 11, a significant difference in performance between sBal_IH and the other specialized ensemble methods except for sBal_IH vs Balanced Random Forest (BRF).

Table 11 Statistical Results for Data from Table 5—sBal_IH vs Specialized Ensemble Methods on Real-world Balanced Datasets

Comparison	$z = (R_0 - R_1)/SE$	p value	Result
sBal_IH vs RUSB	**4.733592**	**0.000002**	**H0 rejected**
EE vs RUSB	0.373705	0.708624	H0 accepted
BRF vs RUSB	2.615933	0.008898	H0 accepted
BB vs RUSB	1.827001	0.0677	H0 accepted
sBal_IH vs BB	**2.906592**	**0.003654**	**H0 rejected**
sBal_IH vs BRF	**2.11766**	**0.034204**	**H0 accepted**
sBal_IH vs EE	**4.359888**	**0.000013**	**H0 rejected**
EE vs BB	1.453296	0.146142	H0 accepted
BRF vs BB	0.788932	0.430152	H0 accepted
EE vs BRF	2.242228	0.024947	H0 accepted

In Table 12, sBal_IH is statistically analyzed against regular ensemble methods when faced with balanced datasets using results obtained from Table 6. For all the pairs of comparisons containing sBal_IH, Bergman's procedure rejects the null hypothesis except for RF vs sBal_IH.

In Table 13, which shows the analysis of the results from Table 7, we observe that $H_0$is rejected for all pairs involving sBal_IH vs the regular ensemble methods. This confirms that there exists a statistically significant difference in performance between sBal_IH and all regular ensemble methods on imbalanced real-world datasets (in line with the same observation above on imbalanced synthetic datasets).

Finally, in Tables 14 and 15, we analyze the differences between sBal_IH and ensemble methods specialized for class imbalance problems on a series of imbalanced and balanced datasets, using results from Tables 8 and 9, respectively.

In Table 14, despite sBal_IH being ranked first above all other methods by average rankings of Friedman's test, we observe that Bergman's procedure only rejects the H_0 hypothesis for sBal_IH vs RUSBoost. However, it is important to note that, at times, the Friedman test might report a significant difference in performance between algorithms but the post hoc test fails to detect it. This might be due to the used power of the post hoc test.

Interestingly, when it comes to balanced datasets, as indicated in Table 15, sBal_IH registered a significant difference in performance with Easy Ensemble, Balanced Bagging, and RUSBoost. This shows that our proposed method is comparable to the state-of-the-art specialized ensemble methods in situations of class imbalance problem, and more superior in situations where the datasets are balanced.

Table 12 Statistical Results for Data from Table 6—sBal_IH vs Specialized Ensemble Methods on Synthetic Imbalanced Datasets

Comparison	$z = (R_0 - R_1)/SE$	p value	Result
sBal_IH vs RUSB	**6.0829**	**0**	**H0 rejected**
RUSB vs BRF	5.388877	0	H0 rejected
RUSB vs BB	3.878359	0.00011	H0 rejected
EE vs RUSB	3.429286	0.00061	H0 rejected
sBal_IH vs EE	**2.653614**	**0.00796**	**H0 rejected**
sBal_IH vs BB	**2.204541**	**0.02749**	**H0 rejected**
EE vs BRF	1.959592	0.05004	H0 accepted
BB vs BRF	1.510519	0.13091	H0 accepted
sBal_IH vs BRF	**0.694022**	**0.48767**	**H0 accepted**
EE vs BB	0.449073	0.65338	H0 accepted

Table 13 Statistical Results for Table 7—sBal_IH vs Regular Ensemble Methods on Imbalanced Datasets

Comparison	$z = (R_0 - R_1)/SE$	p value	Result
sBal_IH vs RF	**8.06687**	**0**	**H0 rejected**
sBal_IH vs Wag	**5.936658**	**0**	**H0 rejected**
sBal_IH vs Bag	**5.657285**	**0**	**H0 rejected**
sBal_IH vs AdaB	**5.657285**	**0**	**H0 rejected**
Bag vs RF	2.409585	0.015971	H0 rejected
RF vs AdaB	2.409585	0.015971	H0 rejected
Wag vs RF	2.130212	0.033154	H0 rejected
Wag vs Bag	0.279372	0.779959	H0 accepted
Wag vs AdaB	0.279372	0.779959	H0 accepted
Bag vs AdaB	0	1	H0 accepted

Table 14 Statistical Results for Data from Table 8—sBal_IH vs Regular Ensemble Methods on Balanced Datasets

Comparison	$z = (R_0 - R_1)/SE$	p value	Result
sBal_IH vs AdaB	**5.024252**	**0.000001**	**H0 rejected**
RF vs AdaB	3.986183	0.000067	H0 rejected
sBal_IH vs Wag	**3.654001**	**0.000258**	**H0 rejected**
sBal_IH vs Bag	**3.570956**	**0.000356**	**H0 rejected**
Wag vs RF	2.615933	0.008898	H0 rejected
Bag vs RF	2.532887	0.011313	H0 rejected
Bag vs AdaB	1.453296	0.146142	H0 accepted
Wag vs AdaB	1.37025	0.170609	H0 accepted
sBal_IH vs RF	**1.038068**	**0.299238**	**H0 accepted**
Wag vs Bag	0.083045	0.933815	H0 accepted

Table 15 Statistical Results for Data from Table 9—sBal_IH vs Specialized Ensemble Methods on Real-world Imbalanced Datasets

Comparison	$z = (R_0 - R_1)/SE$	p value	Result
sBal_IH vs RUSB	**5.203306**	**0**	**H0 rejected**
EE vs RUSB	3.561995	0.000368	H0 rejected
BRF vs RUSB	3.247701	0.001163	H0 rejected
BB vs RUSB	2.654035	0.007954	H0 rejected
sBal_IH vs BB	**2.549271**	**0.010795**	**H0 accepted**
sBal_IH vs BRF	**1.955605**	**0.050512**	**H0 accepted**
sBal_IH vs EE	**1.641311**	**0.100733**	**H0 accepted**
EE vs BB	0.907959	0.3639	H0 accepted
BRF vs BB	0.593666	0.552736	H0 accepted
EE vs BRF	0.314294	0.753298	H0 accepted

4.5 Computational Cost

From the experimental results, despite the promising performance of the proposed method (sBal_IH), we do note that this comes at the computational cost of pre-calculating the Instance Hardness of a given dataset. As observed in Table 16, the proposed method (sBal_IH) registered a high computational time compared to all studied methods (2 runs with different random seeds). Due to space limitations, we only reported computational times for 15 datasets, selecting 5 from each group (Synthetic, Real World Balanced, and Imbalanced), composed of different datasets and features sizes.

While we recognize that the computation times for the sBal_IH method are much higher than those of other methods, we believe that for many applications that use small and medium-size datasets, this can still be an acceptable compromise in exchange for performance improvement. We do also draw attention that these higher computation times are due to the IH estimation step. Table 17 demonstrates that clearly, by showing (in the first 4 columns) how much time it took to estimate IH versus the time for the remaining steps of sBal_IH.

Based on that observation, we emphasize that the IH estimation process is highly parallelizable since it consists of multiple independent tasks, performed within a cross-validation scheme (which can easily run in parallel for each fold). To demonstrate expected efficiency improvements, we ran some experiments that estimate performance gains expected if we were to run the IH process with parallelization. This is shown in Table 17 (last 4 columns) with the following logic:

- Currently, IH estimation was calculated using a fivefold cross-validation (XV) scheme, and therefore running this in 5 parallel processes would cost approximately 20% of sequential execution of such scheme.
- In each of the above folds, we have used 4 different learners to estimate IH. Our experiments show that KNN had the highest cost across all datasets, and thus if we were to parallelize each learner to a separate process, IH estimation has to wait for the longest leaner to finish (i.e., KNN).
- Based on the highest learner's cost, running each of the 4 learners in parallel would give an estimated improvement as shown in Table 17 (2nd last column).
- Lastly, the total running time of sBal_IH under such a parallelization approach is estimated to be as shown in the last column.

As demonstrated by the above discussion, computation time for the sBal_IH method can be squashed by a magnitude of $\sim$ 10 times by a simple parallelization scheme. We do however recognize that future work is required to address the issue of computational cost; maybe via faster alternatives for IH estimates. But even without this, the sBal_IH method still offers an acceptable compromise in exchange for performance improvement.

5 Conclusion and future works

In this study, we have addressed the problem of class imbalance from the perspective of improving the performance of ensemble-based classifiers. Though many state-of-the-art ensemble-based methods have been proposed for the problem, we have introduced a new ensemble method,

Table 16 Computational Time (in seconds) for all Ensemble Methods on Selected Datasets

Data Set ID	Bag	RF	Wag	AdaB	EE	RUB	BB	BRF	sBal_IH
S1	5.65	5.87	5.72	5.68	13.7	5.89	5.97	5.96	**132.6**
S10	6.07	6.11	6.16	5.91	14.1	6.25	6.32	6.37	**153**
S19	6.05	6.12	6.16	6.01	14.1	6.23	6.21	6.25	**154.8**
B17	7.13	7.05	6.97	7.03	18.9	7.33	7.45	7.42	**266.4**
I-4	7.87	8.01	7.9	7.75	18.3	8.01	8.15	7.92	**265.8**
I-7	5.26	5.37	5.28	5.43	13.1	5.55	5.51	5.52	**88.2**
I-9	5.14	5.19	5.12	5.07	12.8	5.37	5.37	5.35	**81**
I-15	11.7	11.6	11.6	12.5	22.6	11.85	11.9	11.6	**790.8**
I-17	5.64	5.75	5.62	5.6	13.5	5.83	5.91	5.89	**132**
S23	5.73	5.77	5.71	5.63	14	5.92	5.99	5.93	**135.6**
S27	6	6.07	6.11	5.9	13.9	6.25	6.37	6.32	**150**
B15	14	14	13.9	14.2	28.8	14.1	14.6	14.3	**611.4**
B9	5.76	5.75	5.68	5.54	19.6	6	6.12	5.83	**207**
B3	5.26	5.075	5.04	5.14	12.75	5.21	5.26	5.21	**70.8**
B19	4.99	5.08	4.98	4.97	5.36	5.03	5.25	5.22	**72.6**

Table 17 Computational Time (in seconds) subtasks of IH estimation and expected parallelization gains

Datasets	Before Parallelization			After Parallelization			
	IH Estimation	Remaining Steps	Total Sequential sBal_IH	IH Cost Per XV-Fold (20% of Sequential fivefold)	Learner with Highest Cost in Each Fold		Total Parallel sBal_IH
					%	Sec	
S1	129.24	3.36	**132.6**	25.85	57%	14.63	**17.99**
S10	149.67	3.33	**153**	29.93	58%	17.36	**20.69**
S19	151.49	3.31	**154.8**	30.30	58%	17.63	**20.94**
B17	262.88	3.52	**266.4**	52.58	60%	31.55	**35.07**
I-4	262.21	3.59	**265.8**	52.44	65%	34.22	**37.81**
I-7	84.85	3.35	**88.2**	16.97	52%	8.81	**12.16**
I-9	77.73	3.27	**81**	15.55	42%	6.52	**9.79**
I-15	786.46	4.34	**790.8**	157.29	65%	101.53	**105.87**
I-17	128.7	3.3	**132**	25.74	25%	6.49	**9.79**
S23	132.31	3.29	**135.6**	26.46	56%	14.89	**18.18**
S27	146.65	3.35	**150**	29.33	58%	17.01	**20.36**
B15	607.6	3.8	**611.4**	121.52	70%	85.12	**88.92**
B9	203.51	3.49	**207**	40.70	35%	14.38	**17.87**
B3	67.56	3.24	**70.8**	13.51	47%	6.32	**9.56**
B19	69.35	3.25	**72.6**	13.87	42%	5.77	**9.02**

called sBal_IH, with a new bootstrapping approach that balances the class distributions of the bags based on instance hardness. In the proposed method, each bag contains all of the minority instances, and we influence the process of picking majority instances using pre-calculated instance hardness categories (easy, normal, and hard), unlike the common way of most of the bagging based methods that use uniform probability to select instances. This ensures that base learners are trained on balanced bags, and at the same time contain diverse levels of hardness, allowing a good representation of the input space that reflected well on overall performance, as demonstrated by the experimental results.

The proposed method is evaluated on 100 datasets, which include 30 synthetic imbalanced datasets with controlled levels of noise, 29 balanced, and 41 imbalanced real-world datasets for binary classification. We used the nonparametric Friedman test and Bergmann's post hoc test, at a significant level of $p < 0.05$, to statistically ascertain our findings.

The findings demonstrate that our proposed sBal_IH method performs statistically and significantly better than the regular ensemble methods (Bagging, Wagging, Random Forest, and AdaBoost) on both synthetic and real-world imbalanced datasets, and better than these methods on balanced datasets (except for Random Forest, where performance was comparable). Furthermore, sBal_IH performed better than ensemble methods specialized for class imbalance problems (Balanced Bagging, Balanced Random Forest, RUSBoost, and Easy Ensemble) in the majority of both balanced and imbalanced datasets. The analysis showed a statistically significant difference in performance between sBal_IH and the specialized methods on the balanced datasets (except for Balanced Random Forest, where performance was comparable), and comparable difference on the imbalanced datasets.

These findings suggest that the approach followed by the sBal_IH method is significantly better than the compared methods, in the majority of the cases, for both balanced and imbalanced datasets. This paves the way for similar extensions based on data complexity and instance hardness to boost performance further. We do note that this improvement comes at the expense of computational cost to pre-calculate the Instance Hardness of a given dataset. However, for many applications that use small and medium-size datasets, this might be acceptable in exchange for promising improvement in performance.

In future works, we plan to investigate how to improve the computational cost of sBal_IH by studying other faster alternative approaches for estimating IH. We also propose to extend our experiments to cover multi-class imbalanced datasets by expanding the sBal_IH approach to take into consideration, multiple classes. Finally, we also plan to study the performance of sBal_IH when induced using heterogeneous base learners, since in current experiments we only considered a homogenous one for building the ensembles. We, therefore, anticipate that this study opens a few research opportunities that utilize data balancing techniques based on instance hardness and data complexity, to improve classification performance in the situation of class imbalance problems, and maybe balanced ones as well.

References

1. Ho TK, Basu M (2002) Complexity measures of supervised classification problems. IEEE Trans Pattern Anal Mach Intell 24(3):289–300
2. Huang J, Ling CX (2005) Using AUC and accuracy in evaluating learning algorithms. IEEE Trans Knowl Data Eng 17(3):299–310
3. Ling CX and Zhang H 2003 AUC: a statistically consistent and more discriminating measure than accuracy
4. Tapkan P, Özbakir L, Kulluk S, Baykasoτlu A (2016) A cost-sensitive classification algorithm: BEE-Miner. Knowledge-Based Syst 95:99–113
5. Weiss G, McCarthy K, Zabar B (2007) Cost-sensitive learning vs. sampling: which is best for handling unbalanced classes with unequal error costs? Dmin, no pp 1–7
6. Japkowicz N, Proc AAAI 2000 Workshop on learning from imbalanced data sets, in Proc AAAI 2000 workshop on learning from imbalanced data sets, 2000
7. Chawla NV, Japkowicz N, Kotcz A (2004) Editorial: Special Issue on Learning from Imbalanced Data Sets. ACM SIGKDD Explor. Newsl. 6(1):1–6
8. Leevy JL, Khoshgoftaar TM, Bauder RA, Seliya N (2018) A survey on addressing high-class imbalance in big data. J Big Data 5(1):42
9. Liu X-Y, Jianxin Wu, Zhou Z-H (2009) exploratory undersampling for class-imbalance learning. IEEE Trans Syst Man, Cybern Part B Cybern 39(2):539–550
10. Chawla NV, Lazarevic A, Hall LO, Bowyer KW (2003) SMOTEBoost: improving prediction of the minority class in boosting, pp 107–119
11. Shahrabi J, Hadaegh F, Ramezankhani A, Azizi F, Khalili D, Pournik O (2014) The Impact of oversampling with SMOTE on the performance of 3 classifiers in prediction of type 2 diabetes. Med Decis Mak 36(1):137–144
12. Seiffert C, Khoshgoftaar TM, Van Hulse J, Napolitano A (2010) RUSBoost: a hybrid approach to alleviating class imbalance. IEEE Trans Syst Man Cybern Part A Syst Humans. 40(1):185–197
13. Zheng Z, Yunpeng Cai YL (2015) Oversampling Method for Imbalanced Classification. Comput. Inform 34:1017–1037
14. Liu XY, Wu J, Zhou ZH (2006) Exploratory under-sampling for class-imbalance learning. Proc IEEE Int Conf Data Mining, ICDM, pp 965–969
15. Barandela R, Valdovinos RM, Salvador Sánchez J, Ferri FJ (2004) The imbalanced training sample problem: under or over sampling? Lect Notes Comput Sci (including Subser Lect Notes Artif Intell Lect Notes Bioinformatics). 3138:806–814
16. Hoens TR, Chawla NV (2013). Imbalanced datasets: from sampling to classifiers. Imbalanced learn Algorithms Appl, pp 43–59
17. Zheng Z, Wu X, Srihari R (2004) Feature selection for text categorization on imbalanced data. ACM SIGKDD Explor Newsl 6(1):80
18. Mladenić D, Grobelnik M (1999) Feature selection for unbalanced class distribution and Naive Bayes. Proc Sixt Int Conf Mach Learn, pp 258–267
19. Khalid S, Khalil T, Nasreen S (2014) A survey of feature selection and feature extraction techniques in machine learning. Proc 2014 Sci Inf Conf SAI 2014, no. July, pp 372–378
20. Tan J, Zhang Z, Zhen L, Zhang C, Deng N (2013) Adaptive feature selection via a new version of support vector machine. Neural Comput Appl 23(3–4):937–945
21. Pes B (2019) Ensemble feature selection for high-dimensional data: a stability analysis across multiple domains. Neural Comput. Appl. 3:9951–9973
22. Chandrashekar G, Sahin F (2014) A survey on feature selection methods. Comput Electr Eng 40(1):16–28
23. Liu B, Ma Y, Wong CK (2000) Improving an association rule based classifier. In: Proceedings of the 4th european conference on principles and practice of knowledge discovery. pp 504–509
24. Sanchez JS, Barandela R, Rangel E, Garcia V (2003) Strategies for learning in class imbalance problems. Pattern Recognit 36(3):849–851
25. Zhou ZH, Liu XY (2010) On multi-class cost-sensitive learning. Comput Intell 26(3):232–257
26. Siers MJ, Islam MZ (2018) Novel algorithms for cost-sensitive classification and knowledge discovery in class imbalanced datasets with an application to NASA software defects. Inf Sci (Ny) 459:53–70
27. Wang S, Li Z, Chao W, Cao Q (2012) Applying adaptive oversampling technique based on data density and cost-sensitive SVM to imbalanced learning. Proc Int Jt Conf Neural Networks, pp 10–15
28. Zhang D, Ma J, Yi J, Niu X, Xu X (2016) An ensemble method for unbalanced sentiment classification. Proc Int Conf Nat Comput vol 2016-Janua, no 61170052, pp 440–445
29. Pławiak P, Acharya UR (2020) Novel deep genetic ensemble of classifiers for arrhythmia detection using ECG signals. Neural Comput Appl 32(15):11137–11161
30. Rokach L (2010) Ensemble-based classifiers. Artif Intell Rev 33(1–2):1–39
31. Tkachenko R, Izonin I, Kryvinska N, Dronyuk I, Zub K (2020) An approach towards increasing prediction accuracy for the recovery of missing IoT data based on the GRNN SGTM ensemble. Sensors. 20(9):2625
32. Zhang C, Ma Y (2012) Ensemble machine learning-methods and applications. Springer New, New York Dordrecht Heidelberg London
33. Wintner S (2000) Dietterich TG: an experimental comparison of three methods for constructing ensembles of decision trees. En Sci commons Org. 40(2):139–157
34. Polikar R (2006) Ensemble based systems in decision making. IEEE Circuits Syst Mag 6(3):21–44
35. Kotsiantis SB (2013) Decision trees: a recent overview. Artif Intell Rev 39(4):261–283
36. Galar M, Fernandez A, Barrenechea E, Bustince H, Herrera F (2012) A review on ensembles for the class imbalance problem: bagging-boosting and hybrid-based approaches. IEEE Trans Syst Man Cybern Part C Appl Rev. 42(4):463–484
37. Abd Elrahman SM, Abraham A (2013) A review of class imbalance problem. J. Netw. Innov. Comput. 1:332–340
38. Bolón-Canedo V, Alonso-Betanzos A (2019) Ensembles for feature selection: a review and future trends. Inf. Fusion 52:1–12
39. Sagi O, Rokach L (2018) Ensemble learning: a survey. Wiley Interdiscip Rev Data Min Knowl Discov 8(4):1–18
40. Krawczyk B, Minku LL, Gama J, Stefanowski J, Woźniak M (2017) Ensemble learning for data stream analysis: a survey. Inf Fusion 37:132–156

41. Bhatt J (2014) A survey on one class classification using ensembles method. Int J Innov Res Sci Technol 1(7):19–23
42. Jurek A, Bi Y, Wu S, Nugent C (2013) A survey of commonly used ensemble-based classification techniques. Knowl Eng Rev 29(5):551–581
43. Gomes HM, Barddal JP, Enembreck AF, Bifet A (2017) A survey on ensemble learning for data stream classification. ACM Comput. Surv. 50(2):1–36
44. Moyano JM, Gibaja EL, Cios KJ, Ventura S (2018) Review of ensembles of multi-label classifiers: models, experimental study and prospects. Inf. Fusion 44:33–45
45. L. Breiman (1994) Bagging predictors: technical report No 421, Mach Learn no 2, pp 19
46. Freund Y, Schapire RE (1999) A short introduction to boosting. J Jpn Soc Artif Intell 14(5):771–780
47. Schapire RE (1999) A brief introduction to boosting. IJCAI Int Joint Conf Artif Intell 2:1401–1406
48. Walmsley FN, Cavalcanti GDC, Oliveira DVR, Cruz RMO, Sabourin R (2018) An ensemble generation method based on instance hardness, Proc Int Jt Conf Neural Networks vol 2018-July
49. Zhu X, Wu X (2004) Class noise vs. attribute noise: a quantitative study. Artif Intell Rev 22(3):177–210
50. Frénay B, Verleysen M (2014) Classification in the presence of label noise: a survey. IEEE Trans Neural Netw Learn Syst 25(5):845–869
51. Rahman MM, Davis DN (2013) Addressing the class imbalance problem in medical datasets. Int J Mach Learn Comput 3(2):224–228
52. Ali A, Shamsuddin SM, Ralescu AL (2015) Classification with class imbalance problem: a review. Int J Adv Soft Comput its Appl 7(3):176–204
53. Barandela R, Sánchez JS, Valdovinos RM (2003) New applications of ensembles of classifiers. Pattern Anal Appl 6(3):245–256
54. Orriols-Puig A, Bernadó-Mansilla E (2009) Evolutionary rule-based systems for imbalanced data sets. Soft Comput 13(3):213–225
55. Hido S, Kashima H, Takahashi Y (2009) Roughly balanced Bagging for Imbalanced data. Stat Anal Data Min 2(5–6):412–426
56. Kasem A, Ghaibeh AA, Moriguchi H (2016) Empirical study of sampling methods for classification in imbalanced clinical datasets. In: International conference on computational intelligence in information system, pp 152–162
57. Alcalá-Fdez J et al (2011) KEEL data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. J Mult Log Soft Comput 17(2–3):255–287
58. Lavesson N, Davidsson P (2006) Quantifying the impact of learning algorithm parameter tuning. Proc Natl Conf Artif Intell 1(1): 395–400
59. Kwon O, Sim JM (2013) Effects of data set features on the performances of classification algorithms. Expert Syst Appl 40(5):1847–1857
60. Smith MR, Martinez T, Giraud-Carrier C (2014) An instance level analysis of data complexity. Mach Learn 95(2):225–256
61. Liu H, Shah S, Jiang W (2004) On-line outlier detection and data cleaning. Comput Chem Eng 28(9):1635–1647
62. Gamberger D, Lavrac N, Dzeroski S (2000) Noise detection and elimination in data preprocessing: experiments in medical domains. Appl Artif Intell 14(2):205–223
63. Kabir A, Ruiz C, Alvarez SA (2018) Mixed Bagging: a novel ensemble learning framework for supervised classification based on instance hardness. Proc IEEE Int Conf Data Mining, ICDM, vol 2018-Novem, pp 1073–1078
64. Smith MR, Martinez T (2016) A comparative evaluation of curriculum learning with filtering and boosting in supervised classification problems. Comput Intell 32(2):167–195
65. Wei Q, Dunbrack RL (2013) The role of balanced training and testing data sets for binary classifiers in bioinformatics. PLoS ONE 8(7):67863
66. Gu Q, Zhu L, Cai Z (2009) Evaluation measures of the classification performance of imbalanced data sets. Comput Intell Intell Syst 51(51):461–471
67. Pereira L, Nunes N (2018) A comparison of performance metrics for event classification in non-intrusive load monitoring. 2017 IEEE Int Conf Smart Grid Commun Smart GridComm 2017 vol 2018-Janua, no October, pp 159–164
68. Bi J, Zhang C (2018) An empirical comparison on state-of-the-art multi-class imbalance learning algorithms and a new diversified ensemble learning scheme. Knowl-Based Syst 158(May):81–93
69. Liu L, Ghosh J, Martin CE (2007) Generative oversampling for mining imbalanced datasets. Int Conf data Min, no May, pp 66–72
70. Breiman L (2001) Random forests. Mach Learn 45(1):5–32
71. Ho TK (1998) The random subspace method for constructing decision forests. IEEE Trans Pattern Anal Mach Intell 20(8):832–844
72. Freund Y, Schapire RRE (1996) Experiments with a new boosting algorithm. Int Conf Mach Learn, pp 148–156
73. Chawla KWPNV, Bowyer KW, Hall LO (2002) SMOTE Synthetic Minority Over Sampling Technique. J Artif Intell Res 16:321–357
74. Halimu C, Kasem A (2020) Split balancing (sBal)—a data preprocessing sampling technique for ensemble methods for binary classification in imbalanced datasets. In: Computational science and technology. Springer, Singapore
75. Bauer E, Kohavi R (1999) Empirical comparison of voting classification algorithms: bagging, boosting, and variants. Mach Learn 36(1):105–139
76. Varoquaux G, Buitinck L, Louppe G, Grisel O, Pedregosa F, Mueller A (2011) Scikit-learn: machine learning in python. J Mach Learn Res 12(1):2825–2830
77. Lemaître G, Nogueira F, Aridas CK (2017) Imbalanced-learn: a python toolbox to tackle the curse of imbalanced datasets in machine learning. J Mach Learn Res 18:1–5
78. Chen C, Liaw A, Breiman L Using random forest to learn imbalanced data, Discovery no 1999, pp 1–12
79. Seiffert C, Khoshgoftaar TM, Van Hulse J, Napolitano A (2008) RUSBoost: improving classification performance when training data is skewed. Proc Int Conf Pattern Recognit, no December, 2008
80. Van Hulse J, Khoshgoftaar TM, Napolitano A (2007) Experimental perspectives on learning from imbalanced data. Proc 24th Int Conf Mach Learn, pp 935–942
81. Napierała K, Stefanowski J, Wilk S (2010) Learning from imbalanced data in presence of noisy and borderline examples. Lect Notes Comput Sci (including Subser Lect Notes Artif Intell Lect Notes Bioinformatics). 6086:158–167
82. Koyejo OO, Natarajan N, Ravikumar PK, Dhillon IS (2014) Consistent binary classification with generalized performance metrics. Adv Neural Inf Process Syst 27 Annu Conf Neural Inf Process Syst 2014, December 8–13 2014, Montr Quebec, Canada, pp 2744–2752
83. Öztürk MM (2017) Which type of metrics are useful to deal with class imbalance in software defect prediction? Inf Softw Technol 92:17–29
84. Folleco A, Khoshgoftaar TM, Napolitano A (2008) Comparison of four performance metrics for evaluating sampling techniques for low quality class-imbalanced data. Proc 7th Int Conf Mach Learn Appl ICMLA, pp 153–158

85. Guo H, Viktor HL (2007) Learning from imbalanced data sets with boosting and data generation. ACM SIGKDD Explor Newsl 6(1):30
86. Boughorbel S, Jarray F, El-Anbari M (2017) Optimal classifier for imbalanced data using matthews correlation coefficient metric. PLoS ONE 12(6):1–17
87. Halimu C, Kasem A, Shah N (2019) Empirical comparison of area under roc curve (AUC) and mathew correlation coefficient (MCC) for evaluating machine learning algorithms on imbalanced datasets for binary classification. Int Conf Mach Learn Soft Comput no Mcc, pp 10–15
88. Van Hulse J, Khoshgoftaar T (2009) Knowledge discovery from imbalanced and noisy data. Data Knowl Eng 68(12):1513–1542
89. Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30
90. Friedman M (1937) The use of ranks to avoid the assumption of normality implicit in the analysis of variance. J Am Stat Assoc 32(200):675–701
91. Garcia S, Herrera F (2008) An extension on 'statistical comparisons of classifiers over multiple data sets' for all pairwise comparisons. J Mach Learn Res 9:2677–2694