International Journal of Machine Learning and Cybernetics (2023) 14:3393–3411
https://doi.org/10.1007/s13042-023-01839-0

ORIGINAL ARTICLE

OUBoost: boosting based over and under sampling technique for handling imbalanced data

Sahar Hassanzadeh Mostafaei[1] · **Jafar Tanha**[1]

Received: 5 February 2022 / Accepted: 17 April 2023 / Published online: 10 May 2023

Abstract
Most real-world datasets usually contain imbalanced data. Learning from datasets where the number of samples in one class (minority) is much smaller than in another class (majority) creates biased classifiers to the majority class. The overall prediction accuracy in imbalanced datasets is higher than 90%, while this accuracy is relatively lower for minority classes. In this paper, we first propose a new technique for under-sampling based on the Peak clustering method from the majority class on imbalanced datasets. We then propose a novel boosting-based algorithm for learning from imbalanced datasets, based on a combination of the proposed Peak under-sampling algorithm and over-sampling technique (SMOTE) in the boosting procedure, named OUBoost. In the proposed OUBoost algorithm, misclassified examples are not given equal weights. OUBoost selects useful examples from the majority class and creates synthetic examples for the minority class. In fact, it indirectly updates the weights of samples. We designed experiments using several evaluation metrics, such as Recall, MCC, Gmean, and F-score on 30 real-world imbalanced datasets. The results show improved prediction performance in the minority class in most used datasets using OUBoost. We further report time comparisons and statistical tests to analyze our proposed algorithm in more details.

Keywords Imbalanced data classification · Class imbalanced problem · Over-sampling · Under-sampling · Imbalance ratio · Boosting

1 Introduction

Classification is an essential task of knowledge discovery in data mining. This task can be a big challenge when the data is imbalanced. Most real-world datasets usually contain imbalanced data. For example, identifying rare diseases, such as cancer in medicine [1], or fraudulent transactions in banks [2]. Learning from datasets where the number of samples in one class (minority) is much smaller than in another class (majority) creates biased classifiers to the majority class. Therefore, traditional algorithms in the classification of imbalanced data are weak and classify most samples into the same as the majority class. The overall prediction accuracy in imbalanced datasets is higher than 90%, while this accuracy is relatively lower for minority classes. However, in classifying imbalanced data, the minority class is more valuable to us, and the goal is to accurately learn minority class examples. Hence, new reliable methods are needed so that models can identify these useful and rare examples.

Techniques to solve the class imbalance problem are divided into data-level and algorithm-level methods [3]. Data-level methods rebalance the class distribution by manipulating the data space. Over-sampling of the minority class and under-sampling of the majority class are among the data-level methods. Over-sampling balances class distribution in the imbalanced dataset by adding examples to the minority class, and under-sampling also tries to balance the dataset by removing samples from the majority class [4]. There are several techniques for under-sampling and over-sampling. The simplest way to rebalance the imbalanced dataset is random sampling, which is done in two ways: random over-sampling and random under-sampling. Random over-sampling balances the imbalanced dataset by repeating samples from the minority class until the desired class ratio is obtained [5]. In the same way, random under-sampling

✉ Jafar Tanha
Tanha@tabrizu.ac.ir

Sahar Hassanzadeh Mostafaei
S.h.mostafaei@tabrizu.ac.ir

[1] Faculty of Electrical and Computer Engineering, University of Tabriz, P.O. Box, Tabriz 51666-16471, Iran

randomly deletes samples of the majority class to achieve the desired ratio [5]. There are more advanced methods for over-sampling and under-sampling [6], and several combinations of these techniques also improved classification performance in imbalanced datasets [7–9]. Algorithm-level methods try to force algorithms to learn minority class samples by adding a penalty cost. Cost-sensitive learning [10] and learning based on recognition [11] are algorithm-level methods.

Under-sampling and over-sampling methods have the advantages and disadvantages described as below. The disadvantages of the under-sampling method are related to the loss of information that is deleted from the training data [12]. However, as the size of the training data set decreases, the execution speed increases. On the other hand, there is no problem with data loss in over-sampling, but as the size of the original training data set increases, the execution speed decreases. The over-sampling algorithm can include duplicate or new minority samples. In addition to the problem of increasing execution time, another problem in oversampling is the excessive repetition of examples, which can lead to over-fitting [13].

Boosting is another method that is used to improve classification performance in imbalanced data [4]. This technique can be used in balanced and imbalanced data and improve classification performance. AdaBoost [14] is one of the most common boosting algorithms that iteratively produces a set of models. This algorithm increases the weight of misclassified examples during each iteration, so in the next iterations, they have a better chance of being selected and learned better. At the end of the iterations, all classifiers participate in a poll to classify the unseen samples. In such a method, when dealing with class imbalances, minority instances that are misclassified are given more weight in the next iterations. As a result, the algorithm pays more attention to them and learns better. Boosting can be considered as a method of advanced data sampling [14] and can be used in two ways: re-weighting or re-sampling [15]. In re-weighted boosting, the weights of the modified examples are transferred directly to the base learner during each iteration. Since not all learning algorithms can use this weighted information [16], it is more appropriate to use re-sampled boosting. In re-sampled boosting, instead of transferring the sample weights to the learner, the training data can be re-sampled according to the weights of the samples. In this method, a new training dataset is created by sampling (replacement or new sample). In the new dataset, samples that have a higher weight are repeated several times, so the classifier is biased towards these examples and learns better. Recently several re-sampled boosting algorithms for imbalanced data have been proposed, such as RUSBoost [17] and SMOTEBoost [18]. The RUSBoost algorithm uses the undersampling during the boosting process and randomly removes the samples from the majority class. As mentioned, there is a problem with data loss by removing samples in the undersampling. The SMOTEBoost algorithm also uses oversampling in the boosting process to generate synthetic samples from the minority class, although it may lead to over-fitting.

To partially alleviate these problems, in this paper, we first propose a novel technique for undersampling of the majority class in imbalanced datasets. We then propose a new boosting-based algorithm for learning from imbalanced datasets based on a combination of the proposed Peak undersampling algorithm and oversampling technique (SMOTE) in the boosting procedure, named OUBoost. We represent a comprehensive description of the proposed OUBoost algorithm and compare its performance with other boosting-based algorithms, such as SMOTEBoost, RUSBoost, and state-of-the-art algorithms. We then evaluate the performance of algorithms using several evaluation metrics such as Recall, MCC, Gmean, and F-score on 30 imbalanced datasets. We also report time comparisons and statistical tests to analyze our proposed algorithm.

The rest of this paper is formed as follows: Sect. 2 shows related works, and Sect. 3 explains the proposed algorithms. Sections 4 and 5 provide the details of the experiments and the experimental setup. We finally give the conclusion in Sect. 6.

2 Related work

Recently, several researches have been performed on imbalanced data, and various techniques have been proposed to deal with the class imbalance problem [19]. Methods for dealing with class imbalance include data level and algorithm level. Data level methods include sampling the minority or majority class, which is used to reduce the problem of data imbalance.

Chawla et al. [18] propose the SMOTEBoost algorithm, which tries to solve the imbalance problem during the boosting process by creating synthetic examples from the minority class. Based on SMOTEBoost, RUSBoost, (Random Under Sampling) is introduced to manage the imbalance problem by deleting samples from the majority class in the boosting process [17].

In [20], a Random Hybrid Sampling based on Boosting (RHSBoost) is used to handle the imbalance problem. This algorithm employs under-sampling and random over-sampling in the boosting algorithm. The RHS algorithm allows each base classifier to focus on minority class examples and uses a new balanced training data that contains characteristics of original data. This technique has stable and impressive classification performance in real-world data.

Pope et al. [21] introduce Hybrid Under-Sampling based on Boosting (HUSBoost) approach to manage imbalanced data. This algorithm needs three main steps: clearing, data

balancing, and classification. HUSBoost uses Tomek-link to clean up data to remove noise or overlapping data. Then, the dataset is divided into a balanced number of subsets by random sampling without replacement, and then classification is performed. The purpose of these methods is to optimize overall accuracy, while traditional algorithms often ignore the minority class samples.

In [22], LIUBoost combines a sampling technique and cost-sensitive learning in boosting process. This algorithm divides majority and minority samples into categories and gives the high cost to difficult samples in imbalanced datasets. However, in cost-sensitive methods, there is a problem of allocating domain-specific costs, and over-sampling methods lead to over-fitting with increased execution time; the results of LIUBoost are significant.

In [23] a novel ensemble method is proposed for classifying imbalanced data. The base classifier of this algorithm is based on reduced kernelized WELM that handles the class imbalance problem more efficiently. This algorithm generates balanced training subsets using random undersampling that serves as the centroid of the reduced kernelized WELM classifier. In this method, base classifiers are generated in a sequential manner. Both misclassified samples of the majority class in the first base classifier and all samples of the minority class were selected as the centroids of the next base classifier. Therefore, selected samples of the centroids are different in the classifiers.

MTSbag is a method that combines the Mahalanobis–Taguchi system (MTS) and bagging-based ensemble learning to increase the ability of conventional MTS in managing imbalanced data classification [24]. MTS is strong in addressing class imbalance problems and bagging can reduce the learning bias of classification algorithms. Therefore, MTSbag can be a useful method, especially for datasets with high imbalance levels.

SMOTECSELM is a novel SMOTE-based class-specific extreme learning machine introduced to manage imbalanced data classification [25]. This algorithm is a variant of class-specific extreme learning machine (CS-ELM) that gains the advantages of both minority oversampling and class-specific regularization. SMOTECSELM uses the synthetic minority oversampling technique (SMOTE) for minority oversampling that increases the significance of the minority class samples for determining the decision region of the classifiers, but it has fluctuation problem in random initialization of weights between the input and the hidden layer.

In [26] a new technique is developed to handle the problem of the SMOTE-CSELM. This technique uses SMOTE based class-specific kernelized extreme learning machine (SMOTE-CSKELM) with the Gaussian kernel function to map the input data to the feature space. The advantages of SMOTE-CSKELM are minority oversampling and class-specific regularization coefficients. The Gaussian kernel function of this technique can handle the non-optimal hidden node problem. The SMOTE method is used to generate synthetic samples of the minority class to balance the training dataset.

Jiang et al. [27] propose a boosting-based algorithm to handle imbalanced data. This algorithm combines static and dynamic re-sampling techniques and uses fuzzy entropy and fuzzy support in the boosting-based random forest (FES-Boost). In this algorithm, static re-sampling is for decreasing the ratio of an imbalanced dataset, and dynamic re-sampling is for updating training data. FESBoost uses the density peak clustering algorithm (DPCA) to select representative samples that are effective in training.

DBRF is a Density-Based Random Forest algorithm developed to improve prediction performance on imbalanced datasets [28]. This algorithm detects borderline samples using a density-based method to augment the samples. DBRF uses two different random forest classifiers to model the augmented boundary samples and the original dataset. This algorithm determines the final output using a bagging technique. DBRF algorithm can solve the problem of classifying minority samples located on the class boundary.

In [29] a novel ensemble classification method based on kernel density estimate (KDE) is proposed to handle imbalanced data. This method trains each tree in the ensemble using uniquely generated synthetically balanced data. KDE offers a natural and effective approach to generating new minority samples to balance subsets by estimating the underlying distribution of the data.

K-Means-SMOTE–ENN is a new method based on hybrid bag-boost is introduced to improve the acts of resampling techniques in noisy imbalanced datasets [30]. This algorithm combines a hybrid bag-boost model of decision tree and hybrid K-Means SMOTE–edited nearest neighbor (ENN) resampling technique to address the noisy class imbalanced problems. K-Means-SMOTE–ENN uses the edited nearest neighbor as an undersampling method to remove samples that create noise.

In [31] a new classification algorithm based on diverse sample generation and classifier fusion are proposed to address the drawbacks of SMOTE, such as lack of diversity and strong overlap of generated minority samples. This algorithm uses a generative adversarial network (GAN)-based framework that includes an oversampling method and a two-class imbalanced data classification approach. In this algorithm, the oversampling method is based on an improved GAN model, and the classification approach is based on classifier fusion through fuzzy integral that can model the interactions between the base classifiers with different balanced subsets.

Most of these aforementioned methods suggest that the use of hybrid methods is useful for learning samples of minority class and also can be used in the boosting process

for imbalanced data classification. In this paper, we propose a novel boosting-based method using a new under-sampling technique based on the Peak clustering algorithm.

3 Boosting-based approach to imbalanced data

As mentioned earlier, the number of examples in different classes differs in many application domains. This leads to the imbalance problem in machine learning. Traditional classifiers cannot properly classify the minority class examples in the imbalanced datasets. Although, the accuracy of these datasets is high, however, most minority instances are misclassified. Therefore, accuracy cannot be used as a proper evaluation metric. To handle this issue, we focus on the boosting algorithm in this paper. A boosting algorithm increases the weights of misclassified examples in the boosting process. In fact, boosting leads to learn more from the hard examples through this training procedure. This is why boosting is a useful approach to classify imbalanced data. As mentioned, SMOTEBoost and RUSBoost are popular boosting-based methods for imbalanced data. In the RUSBoost algorithm, which tries to balance the data by randomly deleting samples, the minority class recall is improved, but many majority class samples are lost. The results mentioned in the article [18] show that using SMOTEBoost is an effective approach.

In this paper, we propose a new boosting-based algorithm along with a novel under-sampling approach using the Peak clustering method to handle the imbalanced data. The peak under-sampling algorithm performs the under-sampling by detecting clusters and selecting useful samples with the maximum density and distance from the minority class.

3.1 Density-peak clustering

In [32], a new clustering algorithm named Density-peak clustering (DPC) is introduced to identify clusters in datasets with complex structures. The DPC algorithm finds the correct number of clusters and has two suppositions, the center of the clusters is among the samples with lower local density, which are approximately far from any samples with higher local density. In this algorithm, two measures are computed for each sample j in the dataset: the local density $\boldsymbol{\eta_j}$ of the sample and its distances $\boldsymbol{\mu_i}$ from the higher density samples. The density $\boldsymbol{\eta_j}$ is then defined as follows:

$$\boldsymbol{\eta_j} = \boldsymbol{\Sigma}_{k\in S,\quad k\neq j}\Omega\left(d(\boldsymbol{j},\boldsymbol{k}) - \boldsymbol{t_r}\right) \tag{1}$$

here $\boldsymbol{d}(\boldsymbol{j},\boldsymbol{k})$ is the distance of sample $\boldsymbol{j}$ to sample $\boldsymbol{k}$, S is the dataset, and $\boldsymbol{t_r}$ is a tuning parameter. $\boldsymbol{t_r}$ is chosen so that the average number of neighbors is about 2/m that m is the number of all samples in the dataset.

$$\Omega(\mathrm{y}) = \begin{cases} 1 y < 0 \\ 0 otherwis \end{cases} \tag{2}$$

In general, η_j is the number of samples that are the vicinity of sample j with the radius t_r [33]. While the μ_j of sample j is the minimum distance from sample j to any other sample with the higher density that is defined in (3). For the sample with the maximum local density, μ_j is the maximum distance between point j and other points, which is computed as follows:

$$\mu_j = \begin{cases} maximum\{d(j,k)|k \in S\} & if \forall k \in \quad S, \eta_j \geq \eta_k \\ minimum\{d(j,k)|\eta_j < \eta_k, k \in S\} & otherwise \end{cases} \tag{3}$$

Then, to determine the center of clusters based on the density and distance values of each sample, a new factor λ_j is defined [32]. Therefore, the center of clusters is the samples with maximum μ and η that computed as:

$$\lambda_j = \mu_j \times \eta_j \tag{4}$$

Next, the λ-values, are sorted in descending, and samples with abnormal high λ-values are selected as center of clusters. Finally, all the residual samples are assigned to the clusters using the measure defined in (5). For a sample with the maximum density, Ψ_j is set to j and computed as:

$$\Psi_j = \begin{cases} jif \forall k \in S, \eta_j \geq \eta_k \\ argmin\left\{d(j,k)|\eta_j < \eta_k\right\} \quad otherwise \end{cases}$$

3.2 Proposed Peak-based undersampling algorithm

As mentioned earlier, under-sampling and over-sampling methods can be used in the boosting process to improve the performance of classifiers in case of imbalanced datasets. Inspired by our previous work [34], which has become developed for intrusion detection, we here propose a new general format for under-sampling in imbalanced datasets. In the proposed under-sampling technique, we use the following steps:

1. In the main dataset, separate the majority class and the minority class. Suppose S_{maj} be the majority class with size E and, S_{min} be the minority class with size F.
2. Use the proposed under-sampling algorithm in S_{maj} to generate e clusters of the majority class.
3. Then we select some effective samples of S_{maj} with high density and distance from S_{min}.

To implant the proposed approach, we determine two measures: $denc_i$ and $dist_i$, where $dens_i$ is the density of the

$cluster_i$ ($i \in \{1, 2, \ldots, e\}$), which is the summation of the local density (η_j) of each sample in the $cluster_i$ as:

$$dens_i = \Sigma_{j \in cluster_i} \eta_j \tag{6}$$

We then define $dist_i$ which is the distance between the center of $cluster_i$ and the minority class that is calculated as:

$$dist_i = \sum_{\substack{q \in min_{class} \\ p = \text{centroid of cluster}_i}} d(p, q) \tag{7}$$

By obtaining the distance and density values for the clusters, we select the most effective samples with the new measurement as H_i. The H_i specifies samples with maximum density and distance from the S_{min}. For each sample in the S_{maj}, H_i is computed as:

$$H_i = u \times dens_i v \times dist_i \tag{8}$$

where u + v = 1. Finally, we select samples with maximum H -values that are not close to each other. To select the samples from H, the new quantity L is computed as follows:

$$L = \{x_i | x_i, x_j \in H \quad \& \quad \& d(i,j) > \theta\} \tag{9}$$

where θ is a tuning parameter and D samples with $d(i,j) > \theta$ are selected as $S_{effective\_maj}$. At the end step, $S_{effective\_maj}$ and S_{min} are merged to generate a modified new dataset. This new dataset can be imbalanced, but it has a lower imbalance ratio than the original dataset. The imbalance ratio is selected based on the minority class size. Figure 1 Shows the proposed peak-based undersampling algorithm.

3.3 Proposed OUBoost algorithm

In this paper, we propose a combination of under-sampling and over-sampling methods within the boosting process. We first divide the given dataset into the minority class and the majority class. We then use over-sampling to generate synthetic data from the minority class. The SMOTE algorithm is used to generate synthetic data. This generated data is then added to the original dataset. In the next step, an under-sampling algorithm is applied to the majority class. We use the proposed peak-based under-sampling algorithm. In the peak under-sampling algorithm, according to the samples ratio of the minority class, useful and reliable examples of the majority class that have the maximum distance from the border are selected. In fact, in this under-sampling method, instead of removing samples from the majority class, useful and reliable samples are selected and placed in a temporary new dataset along with all minority class samples. This new dataset is then passed to the weak learners to learn. At each iteration, a new temporary dataset is generated based on the original dataset to learn, and at the end of the iteration is discarded. This process is repeated until the imbalance ratio in the original dataset reaches the desired adjustable value. Finally, the final model is made by voting from the classifiers.

The main idea of our proposed approach, which differs from the state-of-the-art algorithms, is that each constructed model through the boosting procedure uses its own generated dataset from S based on the proposed peak-based under-sampling and oversampling in the current iteration. However, the proposed approach only updates the weights of the original dataset S at each iteration of the boosting procedure. The overview of the proposed algorithm is depicted in Fig. 2.

The proposed OUBoost algorithm is presented in Fig. 3. We now explain the proposed algorithm in more detail. Suppose S be the original dataset and D be the weights of examples related to S. Let define T as the number of iterations, $S_t^{'}$ is the temporary dataset, and $D_t^{'}$ be the weights of the examples on iteration t.

The original dataset S includes examples $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ where $x_i \in X$, $y_i \in Y = \{0,1\}$, CP, corresponds to minority (positive) class, and CN majority (negative) class, (CP < CN). At first, we assign 1/m as the weight to all samples, where m is the total number of examples.

We now address our proposed OUBoost algorithm based on the boosting formulation. We first generate P samples from the CP minority class and add them to the original S dataset. We then select N useful samples from the majority class using the proposed peak-based undersampling algorithm. We next create a new $S_t^{'}$ dataset using all the minority class examples and the selected examples by the proposed Peak-based undersampling algorithm. Now, the weak learner is trained using $S_t^{'}$, and the weak hypothesis h_t is calculated as:

$$h_t :: X * Y \rightarrow [0, 1] \tag{10}$$

Next, the weighted error rate ϵ_t for the original dataset S and weight distribution D is computed as follows:

$$\epsilon_t = \sum_{(i,y): y_i \neq y} D_t(i)\left(1 - h_t(x_i, y_i) + h_t(x_i, y)\right) \tag{11}$$

Then, the weight update parameter α is calculated based on ϵ_t as:

$$\alpha_t = \frac{\epsilon_t}{1 - \epsilon_t} \tag{12}$$

Finally, the weights of the samples are updated and normalized in D as:

$$D_{t+1}(i) = D_t(i)\alpha_t^{\frac{1}{2}\left(1 + h_t(x_i, y_i) - h_t(x_i, y: y \neq y_i)\right)} \tag{13}$$

$$D_{t+1}(i) = \frac{D_{t+1}(i)}{Z_t Z_t} = \sum_i D_{t+1}(i) \tag{14}$$

At the end of T iterations, the final output of model $H(x)$ is obtained as:

$$H(x) = \underset{y \in Y}{\operatorname{argmax}} \sum_{t=1}^{T} \quad h_t(x, y) \log \frac{1}{\alpha_t} \tag{15}$$

Note that when generating a temporary new dataset for training in each iteration, the number of samples selected from the majority and minority classes is not necessarily equal. Because if the number of samples selected from the majority class is equal to the number of samples from the minority class, we may lose a lot of data from the majority class.

In fact, the temporary new dataset generated in each iteration can be imbalanced, except that the new data set imbalance ratio is less than the original data set imbalance ratio. This means that the imbalance ratio can be modified depending on the nature of the data set. For example, if the imbalance ratio of the original dataset is 20, a temporary new dataset can be generated in each iteration with an imbalance ratio of less than 20. Reducing the imbalance ratio enables classifiers to learn properly from modified datasets. This is a hyper-parameter that can be tuned depending on the nature of the data. The generation of synthetic samples through over-sampling increases the number of samples in the minority class, so the number of samples selected from the majority class also increases to create a temporary new dataset. By integrating these samples into the newly generated data set, the classifiers are trained with different data in each iteration, which may lead to improve the classification performance.

4 Experiments

In the experiments, we compare the performance of the proposed OUBoost algorithm with several other boosting-based algorithms. In this section, we first introduce the used datasets in the experiments. We then address the experimental setup, baseline methods, and evaluation metrics.

4.1 Datasets

The performance of different methods for imbalanced data is measured using various datasets in terms of size and imbalance ratio and as well as a synthetic dataset.

4.1.1 Real-world datasets

We use 22 real-world datasets that contain imbalanced data. Seven of them are from UCI machine learning repository [35], ten datasets are from KEEL-dataset repository [36], and five are from Machine Learning Mastery repository [37], which have been commonly used in related studies. Table 1 provides the specifications of these datasets, including the number of samples, the number of attributes, the number of majority class samples, the number of minority class samples, the minority class name, the imbalance ratio, and the number of classes in the datasets. These datasets present a wide variety of dataset sizes, imbalance ratios, and application domains. Since in this article, our focus is on two-class datasets and binary classification; we convert multiclass datasets that contain more than two classes into two classes. So we converted Glass, Wine, Wheat-seeds, Satimage, Iris, Abalone, and Yeast datasets into two classes using the one-versus-all method, labeling the smallest class as a minority class and the rest as the majority class [38].

4.1.2 Largescale datasets

In Table 2, we introduce three large Imbalance datasets in our experiments. The CICIDS2017 dataset was developed by the Canadian Institute for Cyber Security at the University of New Brunswick. This dataset includes a variety of network traffic, consisting of benign and malignant. The total number of samples in this data set is 283,0743 samples, of which 172,848 samples belong to the minority class [39].

The mammography dataset includes 40,000 mammograms performed between January 2005 and December 2008, from women in the Breast Cancer Surveillance Consortium [40]. In this dataset from 40,000 samples, there are 259 positive examples of breast cancer.

The credit card fraud dataset includes transactions crated by credit cards in September 2013 by European cardholders [41]. This dataset shows transactions that occurred in two days, where there are 492 frauds out of 284,807 transactions. The dataset is highly imbalanced, and the minority class (frauds) size is 0.172% of all transactions.

4.1.3 Synthetic datasets

In this section, we generate five synthetic datasets with various known effects such as dataset size, noisy samples, outliers, and imbalanced ratios. We use these synthetic datasets to demonstrate the properties of the proposed algorithm. The characteristics of the synthetic datasets are summarized in Table 3. Figure 4 shows these synthetic datasets, where the red points represent the samples of the majority class and the blue points represent the samples of the minority class. In this figure, (a) corresponds to the original synthetic datasets and (b) depicts the first temporary new datasets created in the first iteration of the proposed algorithm. These temporary new datasets contain a combination of minority class and majority class samples.

Table 1 Real-world datasets

	Dataset	Samples	Attributes	Minority class name	Minority samples	Majority samples	Imbalance Ratio	Classes
1	Ionosphere	351	34	bad	126	225	1.78	2
2	Ecoli_4	336	7	Imu	35	301	8.6	2
3	Glass_3	214	9	vehic wind float	17	197	11.5	6
4	PC_1	1107	16	Negative	76	1032	13.57	2
5	Vehicle	846	18	Negative	212	634	3	2
6	German	1000	20	Positive	300	700	2.33	2
7	Heberman	306	3	Positive	81	225	2.77	2
8	Oil Spill	937	49	oil slick	41	896	21.8	2
9	Pima Diabetic	768	8	Positive	268	500	1.86	2
10	Wine_3	178	13	2	49	129	1.51	3
11	Breast cancer Wisconsin	699	9	Malignant	241	458	1.99	2
12	Wheat-seeds	210	7	3	70	140	2	3
13	Satimage	6435	36	4	626	5809	9.27	7
14	IRIS	150	4	Iris versicolor	50	100	2	3
15	Bupa	345	6	Sick	145	200	1.38	2
16	Heart	270	13	Positive	120	150	1.25	2
17	Transfusion	748	4	Donating blood	178	570	3.20	2
18	ILPD	579	10	2	167	416	2.49	2
19	Mammography	11,183	6	2	260	10,923	42.01	2
20	Abalone	4177	8	5	115	4061	35.32	28
21	Yeast	1484	8	ME2	51	1433	28.9	10
22	Banknote authentication	1372	4	1	610	762	1.24	2

Peak Under Sampling Algorithm

Input: S $\{(x_1, y_1),\ldots, (x_m, y_m)\}$ $x_i \in X$, $y_i \in Y = \{0,1\}$

$IR = \{1, 2\ldots, N\}$, Imbalance Ratio, we select one of them based on our priority.

Procedure:

1. Divide S into S_{maj} = E and S_{min} = F
2. Cluster S_{maj} to e clusters: cluster_1, cluster_2,…, cluster_e.
3. Compute $dens_i$ for each $\text{cluster}_i \in \{1,2. \ldots, e\}$
4. Compute $distance_j$ for each $\text{cluster}_j \in \{1,2. \ldots, e\}$
5. Compute $H_i = u \times dens_i + v \times dist_i$
6. Choose D samples with maximum H-values from L (According to IR)
7. Merge $S_{effective\_maj}$ with S_{min}

Output: Modified S to S_{ir} $\forall ir \in IR$

Fig. 1 The proposed Peak-based undersampling algorithm

To generate temporary new datasets in each boosting iteration, first, synthetic samples of the minority class are generated by the SMOTE over-sampling technique, and then the majority-class samples are selected by the proposed undersampling method. The peak-based undersampling method selects samples from the majority class that have the maximum density and distance from the minority class samples. Figure 4c shows the final temporary new dataset created by the proposed algorithm, which has a significant boundary between the minority and majority classes. As can be seen in all datasets, the proposed algorithm tries to separate minority and majority class samples with a safe margin. This is the advantage of the proposed OUBoost algorithm that helps classifiers to learn properly rare samples of the minority class while preserving the initial information of the majority class samples in a reduced size. Comparing Fig. 4a, c shows that the OUBoost algorithm selects samples from the majority class that contain useful information instead of using all samples from this class.

4.2 Experimental setup

In this paper, all experiments are performed using ten-fold cross validation. Ten-fold cross-validation is a method that divides a dataset into ten parts, in which each part is used as test data. Nine of them are used as training sets, and the rest for testing. This method performs the fitting process ten times so that each part acts as test data. The learning rate is 0.3 in all experiments, and the used base learner in all methods is Decision Tree.

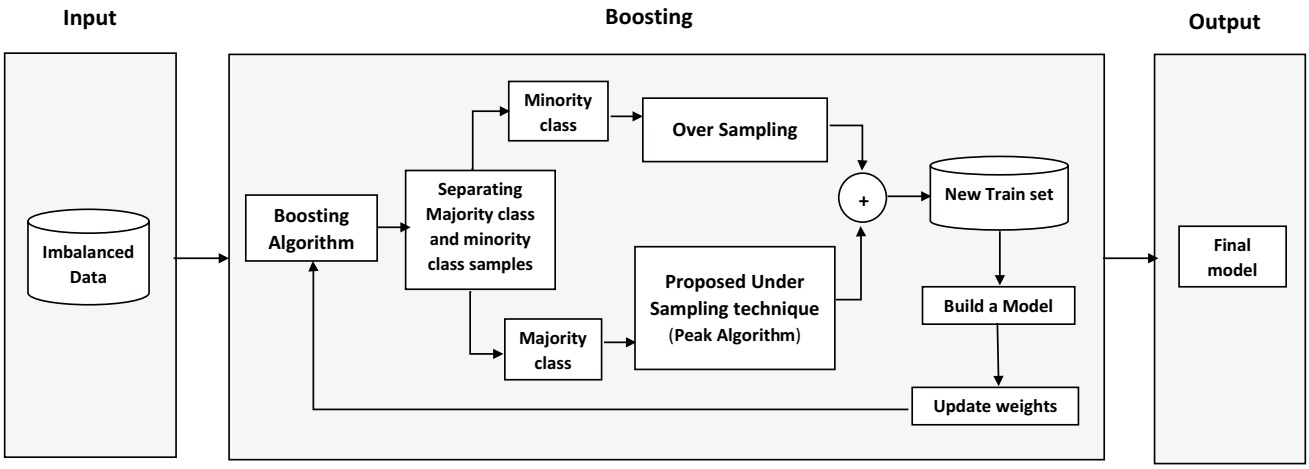

Fig. 2 The general overview of the proposed boosting-based algorithm

OUBoost Algorithm

Given: S $\{(x_1, y_1),\ldots, (x_m, y_m)\}$ $x_i \in X$, $y_i \in Y = \{0,1\}$ CP, corresponds to minority (positive) class and CN majority (negative) class, (CP < CN), Number of iterations, T

A. Initialize the weights D_1 on the samples, such that $D_1(i) = 1/m$.

B. While t < T

1. Create P synthetic examples from CP.
2. Select N examples from CN using the proposed Peak-based undersampling algorithm.
3. Create temporary training dataset S'_t with distribution D'_t using all examples from CP and N selected examples from CN.
4. Train a weak learner based on S'_t and their weights D'_t.
5. Compute weak hypothesis by eq. (10)
6. Compute the pseudo-loss of hypothesis (weighted error rate) for original dataset S by eq. (11)
7. Compute the alpa_t by eq. (12)
8. Update the weights D_t by eq. (13)
9. Normalize D_t+1 by eq. (14)
10. t = t + 1;

End While

C. Output the final hypothesis by eq. (15)

Fig. 3 The proposed OUBoost

Table 2 Large-scale datasets

	Dataset	Samples	Attributes	Minority class name	Minority samples	Majority samples	Imbalance Ratio	Classes
23	CICIDS2017	2,830,743	30	benign	172,848	2,657,895	16.37	15
24	Mammography	40,000	11	2	259	39,741	153.44	2
25	Credit card fraud	284,807	30	2	492	284,315	578	2

Table 3 Synthetic datasets

	Dataset	Samples	Attributes	Minority class name	Minority samples	Majority samples	Imbalance Ratio	Classes
26	Synthetic data 1	600	2	1	540	60	9	2
27	Synthetic data 2	600	2	1	570	30	19	2
28	Synthetic data 3	1000	2	1	960	40	24	2
29	Synthetic data 4	2000	2	1	1950	50	39	2
30	Synthetic data 5	5000	2	1	4900	100	49	2

4.3 Baseline methods

Since our proposed method is based on boosting, we use the state-of-the-art boosting-based methods to compare the results. For comparison, we use SMOTEBoost, RUSBoost, RHSBoost, and FESBoost algorithms. We also use the standard boosting model in the comparisons to show the performance of the proposed algorithm.

4.4 The used evaluation metrics

As mentioned earlier, when dealing with imbalanced datasets, traditional classifiers cannot learn properly minority class. This is because the number of minority class samples is much smaller than the majority class samples. Therefore, the accuracy cannot be used to evaluate the performance of the models. In this paper, we use several evaluation metrics to measure classification performance of algorithms. These metrics include Recall, MCC, Gmean, and F-score. We also report time comparisons and use statistical tests to provide a statistical basis for the main comparisons.

4.4.1 Recall

Recall measures the proportion of correctly classified examples as:

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

The best value is 1, and the worst value is 0.

4.4.2 F-Score

F-measure [42] metric is the harmonic mean of the precision and recall and computed as:

$$F - measure = \frac{(1 + \beta)\ \text{PrecisionRecall}}{\beta^2\ \text{Precision} + \text{Recall}} \tag{16}$$

This measure is an appropriate evaluation metric for the imbalanced datasets. The range of F1 is in [0, 1], where 1 is the correct classification and 0 is the total failure.

4.4.3 G-mean

Geometric Mean (G-mean) is a good choice for imbalanced classification that measures the balance between classification performance in the majority and minority classes. A low G-mean value indicates poor performance in classifying positive samples even if the negative samples are classified correctly. This metric is important in preventing the over-fitting of the majority class and the under-fitting of the minority class. G-mean [43] is defined as:

$$G - mean = \sqrt{\frac{TP}{TP + FN}\frac{TN}{TN + FP}} \tag{17}$$

4.4.4 MCC

Matthews Correlation Coefficient (MCC) is a measure to evaluate the performance of boosting algorithms that is especially utilized in imbalanced data classification [42]. It ranges between − 1 and 1, where 1 score presents a good prediction, 0 equals the random prediction, and − 1 shows the total difference between predicted scores and the true labels [44]. This metric can be obtained from the confusion matrix defined as follows:

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{18}$$

4.4.5 Statistical tests

We use statistical tests to evaluate the performance of the proposed algorithm [45]. First, we use the Friedman test, which is a non-parametric test equivalent to ANOVA with repeated measures. In the null hypothesis, the Friedman test states that all algorithms are equal, and the rejection of this hypothesis indicates the difference in the performance of the algorithms. It ranks the algorithms based on their performance for each dataset, then assigns rank 1 to the best algorithm, rank 2 to the second best, and so on. In this test, the value of the significance level is set to 0.05. Next, we apply

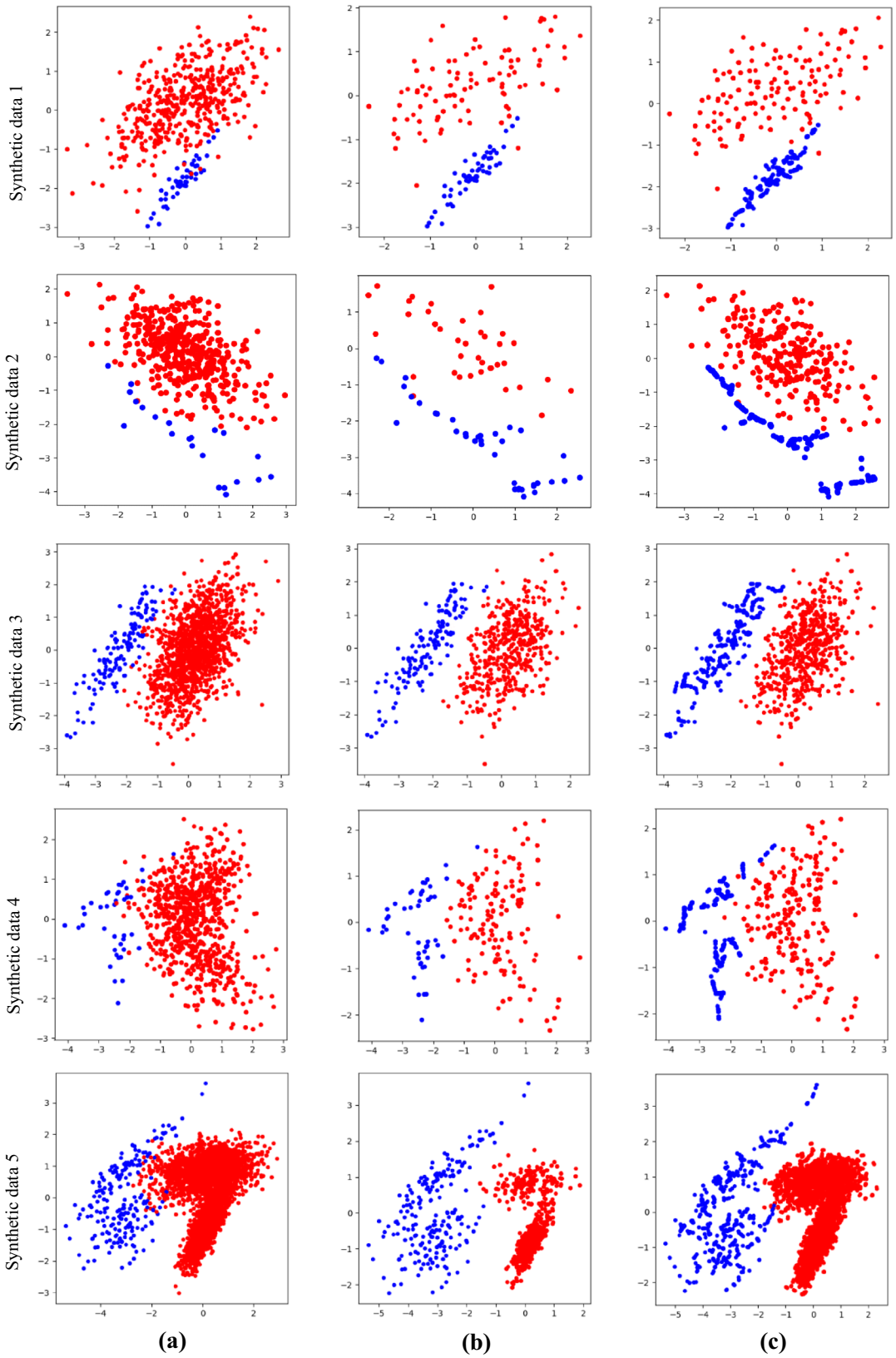

Fig. 4 The used synthetic datasets in our experiments. **a** ''The original synthetic dataset'', **b** ''The first new dataset'', and **c** the final new dataset created by OUBoost algorithm

a post-hoc test using Holm's method for comparing the performance of the algorithms with each other [46]. This test compares two algorithms and checks the hypothesis ordered by their performance as p-values.

5 Results

This section consists of three parts. In the first part, we analyze the performance of the proposed algorithm on several synthetic datasets with known effects such as noise samples, outliers, different sizes, and different imbalance ratios. In the second part, the performance of the algorithms is compared on 22 Real-world datasets through various evaluation metrics such as MCC, Gmean, and F-score. Finally, in the third part, the performance of the algorithms is analyzed on 3 large imbalanced datasets with different imbalance ratios. The best results are boldfaced in the Tables.

5.1 The results of synthetic datasets

In the first part of the experiments, we use the five synthetic datasets introduced in Sect. 4.1. These datasets are generated with known effects such as noise samples, outliers, different sizes, and different imbalance ratios. To analyze the performance of the proposed algorithm, we report the MCC, Gmean, and F-score values. Then we compare them with other algorithms in classifying imbalanced synthetic datasets. Tables 4, 5, 6 show the results of algorithms based on MCC, Gmean, and F-score metrics respectively. As can be seen, by increasing the size and imbalance ratio of the dataset, the proposed method can achieve accurate classification and better performance than others in most cases. Also despite the noise samples and outliers in synthetic data 1, 4, and 5, the OUBoost algorithm is not affected and outperforms the other algorithms.

5.2 The results of Real-world datasets

In the second part, we report the classification performance of different algorithms on Real-world datasets using several evaluation metrics, including MCC, G-mean, and F-score. We also compare execution time and analyze statistical test results of algorithms. Tables 7, 8, 9, 10, 11, 12, 13, 14, 15, 16 give the results of these experiments. In Tables 7, 8, 9, 10, the first column shows the name of the dataset. The second

Table 4 MCC averages of synthetic datasets

Dataset	AdaBoost	SMOTEBoost	RUSBoost	FESBoost	RHSBoost	OUBoost
Synthetic data 1	0.7099	0.8528	0.7810	0.8744	0.8531	**0.8901**
Synthetic data 2	0.8036	0.8554	0.8294	0.8555	0.8467	**0.8682**
Synthetic data 3	0.7955	0.8549	0.8253	**0.8726**	0.8588	0.8246
Synthetic data 4	0.9325	0.9491	0.9311	0.9501	0.9497	**0.9673**
Synthetic data 5	0.7214	0. 8378	0.8227	0.8465	0.8114	**0.8620**

The best results are boldfaced in the Tables

Table 5 Gmean averages of synthetic datasets

Dataset	AdaBoost	SMOTEBoost	RUSBoost	FESBoost	RHSBoost	OUBoost
Synthetic data 1	0.8344	0.9410	0.9543	0.9470	0.9441	**0.9605**
Synthetic data 2	0.8969	0.9526	0.9270	0.9590	0.9325	**0.9613**
Synthetic data 3	0.8864	0.9609	0.9445	**0.9702**	0.9666	0.9541
Synthetic data 4	0.9625	0.9785	0.9730	0.9755	0.9750	**0.9861**
Synthetic data 5	0.7670	0.8768	0.8689	0.8804	0.8723	**0.8847**

The best results are boldfaced in the Tables

Table 6 Fscore averages of synthetic datasets

Dataset	AdaBoost	SMOTEBoost	RUSBoost	FESBoost	RHSBoost	OUBoost
Synthetic data 1	0.7161	0.8557	0.7791	0.8697	0.8631	**0.8930**
Synthetic data 2	0.8173	0.8642	0.8413	0.8759	0.8541	**0.8860**
Synthetic data 3	0.7900	0.8507	0.8233	**0.8641**	0.8566	0.8404
Synthetic data 4	0.9372	0.9532	0.9370	0.9588	0.9554	**0.9617**
Synthetic data 5	0.7201	0.8365	0.8224	0.8370	0.8267	**0.8438**

The best results are boldfaced in the Tables

Table 7 MCC averages of Real-world datasets under 6 algorithms

Dataset	AdaBoost	SMOTEBoost	RUSBoost	FESBoost	RHSBoost	OUBoost
Ionosphere	0.8175	0.7845	0.7457	0.8231	0.8198	**0.8399**
Ecoli_4	0.4247	0.5314	0.4965	0.4536	0.5529	**0.6162**
Glass_3	0.2972	0.2964	0.2771	0.5123	0.4555	**0.5547**
PC_1	0.3125	0.2859	0.3027	0.3023	0.2987	**0.3291**
Vehicle	0.3965	0.4092	0.3084	0.4247	0.4025	**0.4365**
German	0.4108	0.4449	0.3028	0.4366	0.4508	**0.4600**
Heberman	0.1414	0.3025	0.2433	0.2565	**0.3272**	0.2829
Oil Spill	0.4444	0.4506	0.4131	0.4665	0.4587	**0.4768**
Pima Diabetic	0.4451	0.5000	0.4470	0.4906	0.4956	**0.5058**
Wine	0.8816	**0.9612**	0.8880	0.9160	0.9303	0.9401
Breast cancer Wisconsin	0.8988	0.9036	0.9012	0.8904	0.9020	**0.9062**
Wheat-seeds	0.8765	0.8861	0.8385	0.8941	0.8788	**0.9110**
Satimage	0.5517	0.5495	0.5249	0.5533	0.5470	**0.5610**
IRIS_3	0.8787	0.8929	0.8556	**0.9024**	0.8832	0.8978
Bupa	0.3215	0.4261	0.3653	0.4270	0.4121	**0.4301**
Heart	0.6051	0.6321	0.6157	0.6522	**0.3272**	0.6325
Transfusion	0.2912	0.3561	0.2614	0.3354	0.3258	**0.3591**
ILPD	0.1358	0.2349	0.2681	0.2841	**0.2901**	0.2860
Mammography	0.6861	0.7136	0.6962	0.7243	0.7122	**0.7427**
Abalone	0.1983	0.4355	0.3745	0.4411	0.4281	**0.4586**
Yeast	0.2684	0.3286	0.3018	0.3467	0.3183	**0.3696**
Banknote authentication	0.9773	0.9623	0.9519	0.9681	0.9641	**0.9782**

The best results are boldfaced in the Tables

Table 8 G-mean averages of Real-world datasets under 6 algorithms

Dataset	AdaBoost	SMOTEBoost	RUSBoost	FESBoost	RHSBoost	OUBoost
Ionosphere	0.8982	0.8907	0.8656	0.9027	0.8974	**0.9161**
Ecoli_4	0.6336	0.8275	0.6449	0.8088	0.8363	**0.8685**
Glass_3	0.4772	0.5659	0.5564	0.6587	0.6055	**0.7464**
PC_1	0.4445	0.5833	0.5720	0.5874	0.5760	**0.6080**
Vehicle	0.6253	0.7053	0.6179	0.6973	0.6888	**0.7189**
German	0.6593	0.7042	0.6178	0.7144	0.7337	**0.7871**
Heberman	0.5049	0.6004	0.5633	0.5984	**0.6184**	0.6108
Oil Spill	0.4896	0.6882	0.6490	0.6927	0.6830	**0.7130**
Pima Diabetic	0.7035	0.7455	0.7110	0.7189	0.7334	**0.7496**
Wine	0.9366	**0.9814**	0.9539	0.9597	0.9612	0.9630
Breast cancer Wisconsin	0.9502	0.9539	0.9505	0.9536	0.9547	**0.9563**
Wheat-seeds	0.9324	0.9390	0.9263	0.9397	0.9408	**0.9570**
Satimage	0.7171	0.7646	0.7389	0.7754	0.7666	**0.7883**
IRIS_3	0.9280	0.9484	0.9498	**0.9614**	0.9573	0.9573
Bupa	0.6406	0.6926	0.6644	0.6873	0.6787	**0.6984**
Heart	0.7835	0.8122	0.7920	0.8266	**0.8333**	0.8075
Transfusion	0.5353	0.6331	0.5555	0.6351	0.6025	**0.6538**
ILPD	0.4908	0.5411	0.5830	0.5994	0.6270	**0.6352**
Mammography	0.7671	0.7873	0.7788	0.7918	0.7801	**0.8311**
Abalone	0.3937	0.8058	0.6739	0.7866	0.7611	**0.8176**
Yeast	0.3567	0.5966	0.4611	0.6041	0.5973	**0.6747**
Banknote authentication	0.9806	0.9821	0.9519	0.9784	0.9681	**0.9899**

The best results are boldfaced in the Tables

Table 9 F-Score averages of Real-world datasets under 6 algorithms

Dataset	AdaBoost	SMOTEBoost	RUSBoost	FESBoost	RHSBoost	OUBoost
Ionosphere	0.8714	0.8805	0.8756	0.8856	0.8832	**0.9015**
Ecoli_4	0.4614	0.5539	0.5250	0.5175	0.5714	**0.6336**
Glass_3	0.3031	0.3343	0.3233	0.5217	0.4635	**0.5666**
PC_1	0.3250	0.3331	0.3380	0.3357	0.3344	**0.3745**
Vehicle	0.5023	0.5632	0.4655	0.5712	0.5382	**0.5896**
German	0.5541	0.6021	0.4956	0.6045	0.6070	**0.6178**
Heberman	0.3401	0.4569	0.4126	0.4302	**0.4753**	0.4712
Oil Spill	0.4323	0.4636	0.4398	0.4674	0.4666	**0.4977**
Pima Diabetic	0.6244	0.6711	0.6388	0.6451	0.6547	**0.6986**
Wine	0.9021	0.9614	0.9142	0.9247	0.9331	**0.9708**
Breast cancer Wisconsin	0.9333	0.9361	0.9336	0.9151	0.9352	**0.9580**
Wheat-seeds	0.9012	0.9146	0.9079	0.9283	0.9254	**0.9457**
Satimage	0.5805	0.5995	0.5889	0.5998	0.5990	**0.6598**
IRIS_3	0.9111	0.9246	0.9147	**0.9278**	0.9160	0.9251
Bupa	0.5847	0.6413	0.6188	0.6456	0.6267	**0.6698**
Heart	0.7532	0.7782	0.7763	0.7888	**0.9035**	0.7862
Transfusion	0.3946	0.4836	0.4087	0.4566	0.4307	**0.5374**
ILPD	0.3455	0.4023	0.4566	0.4733	0.4987	**0.5101**
Mammography	0.6824	0.7039	0.6987	0.7054	0.6993	**0.7589**
Abalone	0.2086	0.4145	0.3751	0.3958	0.3886	**0.4487**
Yeast	0.2756	0.3445	0.3012	0.3555	0.3127	**0.3863**
Banknote authentication	0.9808	0.9858	0.9832	0.9869	0.9880	**0.9947**

The best results are boldfaced in the Tables

Table 10 Execution time of Real-world datasets under 6 algorithms

Dataset	AdaBoost	SMOTEBoost	RUSBoost	FESBoost	RHSBoost	OUBoost
Ionosphere	234 ms	39,013 ms	46 ms	5366 ms	21,786 ms	4046 ms
Ecoli_4	62 ms	296 ms	15 ms	256 ms	246 ms	249 ms
Glass_3	31 m	281 ms	15 ms	300 ms	278 ms	171 ms
PC_1	244 ms	39,628 ms	81 ms	38,365 ms	38,657 ms	37,955 ms
Vehicle	175 ms	20,019 ms	59 ms	18,677 ms	18,654 ms	16,905 ms
German	143 ms	6117 ms	15 ms	9012 ms	4562 ms	9666 ms
Heberman	69 ms	138 ms	15 ms	297 ms	124 ms	231 ms
Oil Spill	479 ms	39,300 ms	68 ms	35,012 ms	112 ms	34,971 ms
Pima Diabetic	131 ms	178 ms	37 ms	1644 ms	29,644 ms	1419 ms
Wine	15 ms	717 ms	15 ms	398 ms	150 ms	169 ms
Breast cancer Wisconsin	69 ms	902 ms	31 ms	1203 ms	721 ms	1488 ms
Wheat-seeds	15 ms	485 ms	6 ms	128 ms	387 ms	122 ms
Satimage	1390 ms	39,262 ms	328 ms	56,574 ms	27,164 ms	69,593 ms
IRIS_3	15 ms	140 ms	15 ms	105 ms	156 ms	93 ms
Bupa	78 ms	78 ms	15 ms	250 ms	131 ms	218 ms
Heart	78 ms	153 ms	15 ms	160 ms	133 ms	93 ms
Transfusion	79 ms	156 ms	31 ms	1860 ms	104 ms	1500 ms
ILPD	109 ms	62 ms	31 ms	563 ms	78 ms	437 ms
Mammography	772 ms	2500 ms	254 ms	612,687 ms	2133 ms	551,018 ms
Abalone	296 ms	2015 ms	171 ms	129,755 ms	1988 ms	115,253 ms
Yeast	171 ms	515 ms	31 ms	9769 ms	480 ms	10,358 ms
Banknote authentication	156 ms	203 ms	62 ms	2297 ms	265 ms	2140 ms

Table 11 Friedman test based on MCC

Comparison	Statistic	P value	Result
OUBoost vs RUSBoost	6.93008	0.00000	H0 is rejected
OUBoost vs AdaBoost	6.52717	0.00000	H0 is rejected
OUBoost vs RHSBoost	3.86795	0.00033	H0 is rejected
OUBoost vs SMOTEBoost	3.62620	0.00058	H0 is rejected
OUBoost vs FESBoost	2.73980	0.00615	H0 is rejected

Table 12 Friedman test based on Gmean

Algorithm	Rank
OUBoost	1.27273
FESBoost	2.81818
SMOTEBoost	3.31818
RHSBoost	3.45455
AdaBoost	4.95455
RUSBoost	5.18182

Table 13 . Friedman test based on Fscore

Algorithm	Rank
OUBoost	1.29545
FESBoost	2.77273
RHSBoost	3.02273
SMOTEBoost	3.18182
RUSBoost	5.13636
AdaBoost	5.59091

Table 14 Holm post-hoc test based on the MCC (Using OUBoost as control method)

Comparison	Statistic	p-value	Result
OUBoost vs AdaBoost	7.61503	0.00000	H0 is rejected
OUBoost vs RUSBoost	6.80920	0.00000	H0 is rejected
OUBoost vs SMOTEBoost	3.34417	0.00248	H0 is rejected
OUBoost vs RHSBoost	3.06213	0.00440	H0 is rejected
OUBoost vs FESBoost	2.61892	0.00882	H0 is rejected

Table 15 Holm post-hoc test based on the Gmean (Using OUBoost as control method)

Algorithm	Rank
OUBoost	1.18182
FESBoost	2.81818
RHSBoost	3.04545
SMOTEBoost	3.27273
RUSBoost	4.81818
AdaBoost	5.86364

Table 16 Holm post-hoc test based on the Fscore (Using OUBoost as control method)

Comparison	Statistic	p-value	Result
OUBoost vs AdaBoost	8.29998	0.00000	H0 is rejected
OUBoost vs RUSBoost	6.44658	0.00000	H0 is rejected
OUBoost vs SMOTEBoost	3.70679	0.00063	H0 is rejected
OUBoost vs RHSBoost	3.30387	0.00191	H0 is rejected
OUBoost vs FESBoost	2.90096	0.00372	H0 is rejected

to sixth columns show the classification performance of the algorithms based on the desired metric. We use AdaBoost, SMOTEBoost, RUSBoost, RHSBoost, FESBoost, and the proposed algorithm in our tests. It should be noted that in all tables and figures, the classification results of the AdaBoost algorithm are presented to observe the improvement of the algorithms in the imbalanced data classification.

5.3 MCC results

In this experiment, we compare the performance of the proposed algorithm to the state-of-the-art methods when the evaluation metric is MCC. Table 7 shows the results of each algorithm for each dataset using the MCC metric. The best classification performance is boldfaced for each dataset in the tables. The results show that the proposed algorithm performs better than the other boosting based algorithms on 17 out of 22 datasets. We further observe that OUBoost outperforms the AdaBoost nearly in all used datasets.

5.4 G-mean results

In this experiment, we report the result when the evaluation metric is G-mean. The results in Table 8 show that OUBoost gives better results than all five boosting algorithms AdaBoost, SMOTEBoost, RUSBoost, FESBoost, and RHSBoost on 18 out of 22 datasets. The improvement of OUBoost in most of the used datasets is significant, and the same results can be seen with the other evaluation metrics.

5.5 F-score results

The F-score metric is used to evaluate the performance in this experiment. The obtained results for algorithms are reported in Table 9. As can be seen in Table 9, OUBoost performs better than the other boosting algorithms on 19 out of 22 datasets. A high F-score indicates that both the minority recall and the majority recall are high.

Based on the results of these evaluation metrics, we can be relatively confident that the proposed algorithm works better than other methods in most datasets. However, each

of these algorithms performed well in imbalanced data classification.

5.6 Execution time

Table 10 shows the execution time of different algorithms in the classification of 22 imbalanced datasets from real world datasets. Algorithms execution time are calculated based on milliseconds (ms), which includes both training and testing phase. As expected, the RUSBoost algorithm with the lowest execution time is the fastest algorithm among the algorithms due to the random elimination operation. The second, third, and fourth lowest execution times are for AdaBoost, SMOTEBoost, and RHSBoost respectively. For other algorithms such as FESBoost and the proposed OUBoost, when the imbalance ratio is high, the algorithm runs slower than others. This is related to the DPC algorithm in detecting the majority class samples from the clusters that have the maximum distance from the minority class samples. However, in working with imbalanced data, the classification of minority class samples is more important than the execution time of the algorithms.

5.7 Statistical test results

In this section, we analyze the results of statistical tests in terms of MCC, Gmean, and F-score from Tables 7, 8, 9. Tables 11, 12, 13 show the ranking of each algorithm according to the Friedman test based on MCC, Gmean, and F-score metrics respectively. As can be seen, the proposed method with the highest ranking has the best performance compared to other algorithms, and FESBoost is the second-best algorithm. Tables 14, 15, 16 show the results of Holm's test. Holm's method rejects all hypotheses because the corresponding p-values are smaller than 0.05. We conclude that the performance of OUBoost is significantly different from each other boosting-based algorithms in classifying imbalanced data.

5.8 The results of large-scale datasets

In the third part of the experiments, we use several datasets with high imbalance ratios to compare the performance of the proposed algorithm with other boosting-based algorithms. In this experiment, we report the recall and F-score values of the minority class to evaluate the performance of algorithms. For this purpose, we use three large datasets with a high imbalance ratio. For each dataset, we create 5 new datasets with different IRs of 5, 15, 25, 50, and 100. Then we evaluate the algorithms using these datasets. Figures 5, 6, 7, 8, 9, 10 shows the results of these experiments.

5.9 Recall results

In this experiment, we report the recall value of the minority class to evaluate the performance of the algorithms when we have different imbalance ratios. Figures 5, 6, 7 show the minority class recall for algorithms in different datasets with various IRs. In the figures, the horizontal bar is related to different IRs, and the vertical bar is for a minority class recall. As shown in Figs. 5, 6, 7, our proposed algorithm effectively learns data from the minority class and improves the recall rate of the minority class in the imbalance dataset.

As can be seen in the figures, increasing IR reduces the performance of algorithms in classifying minority class samples. This is normal because as IR increases, the number of minority class samples will be much smaller than the majority class samples, and the minority class will become more difficult for algorithms to learn. However, in all IRs, it is clear that the proposed algorithm significantly outperforms other algorithms. We also observe that the proposed

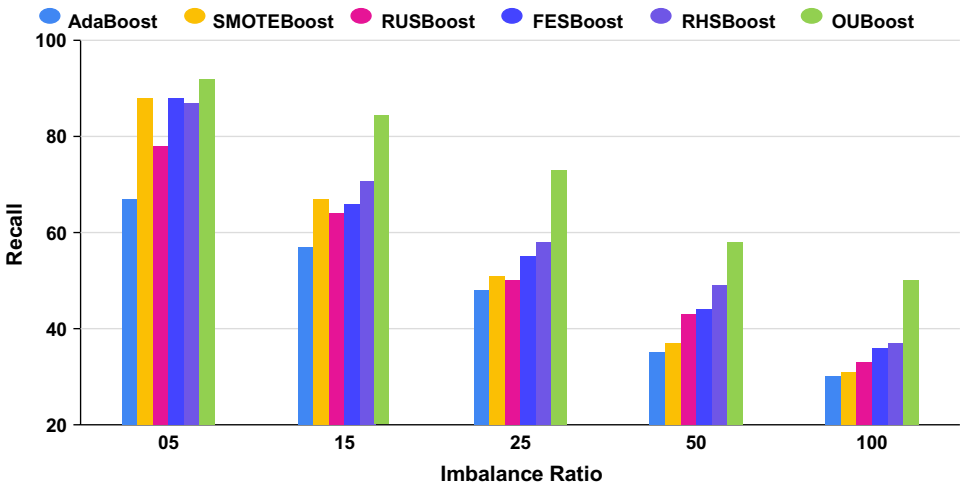

Fig. 5 The minority class recall averages of different algorithms on Mammography datasets

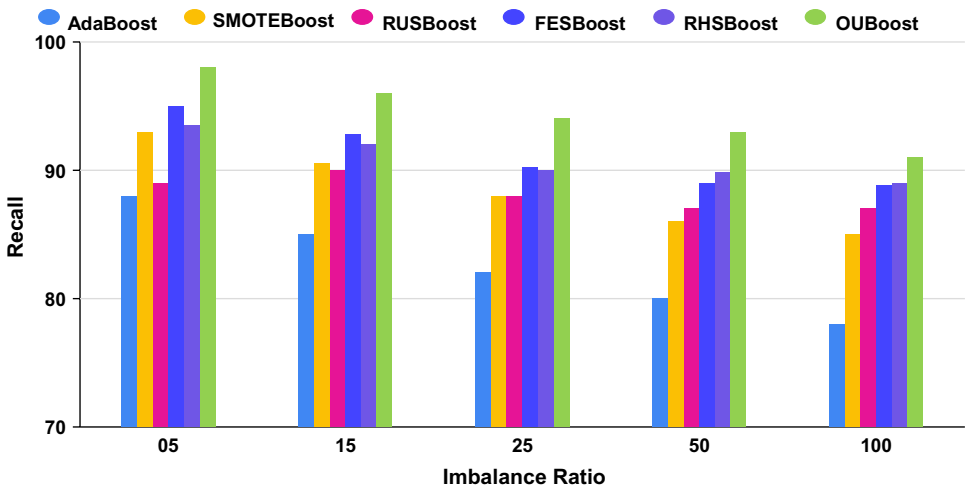

Fig. 6 The minority class recall averages of different algorithms on Credit card fraud datasets

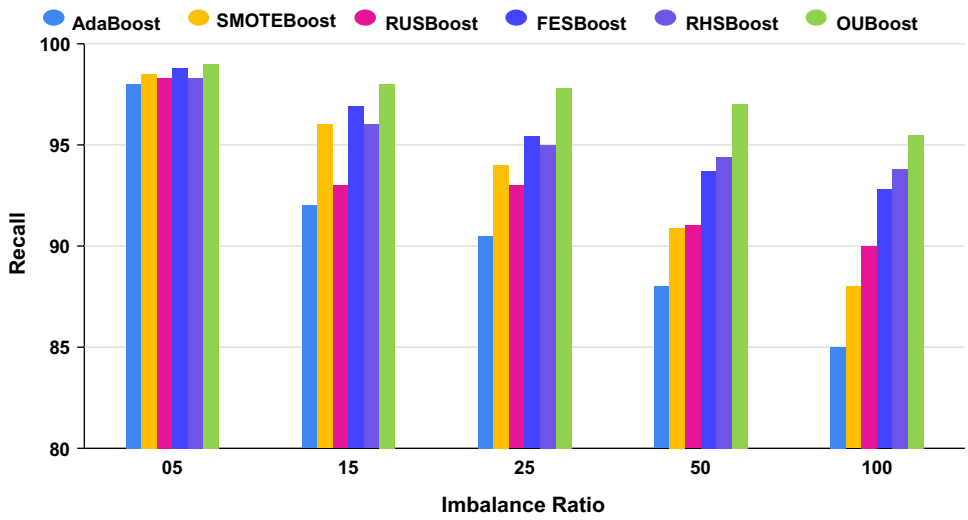

Fig. 7 The minority class recall averages of different algorithms on CICIDS2017 datasets

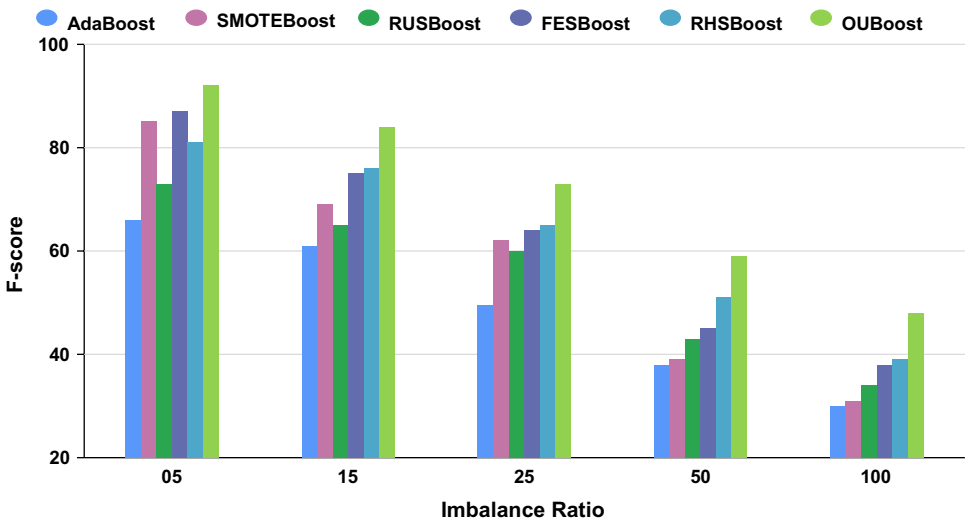

Fig. 8 The minority class F-score averages of different algorithms on Mammography datasets

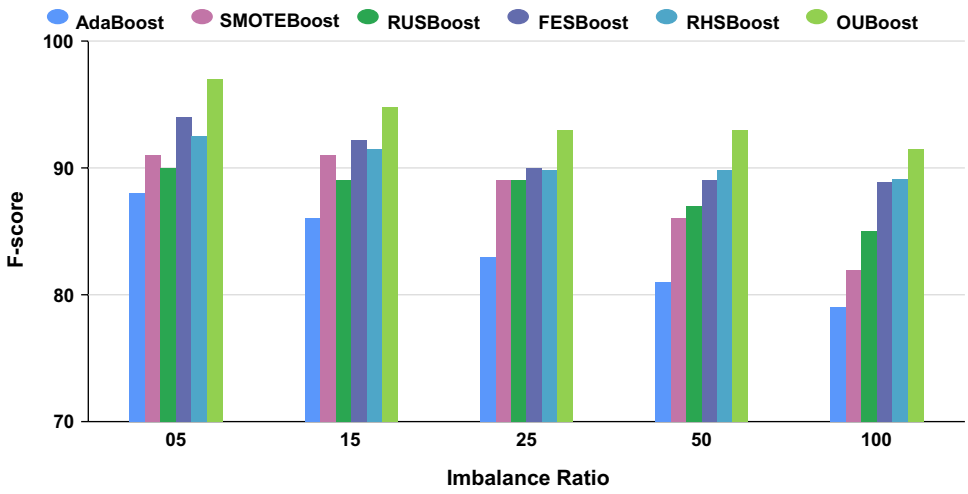

Fig. 9 The minority class F-score averages of different algorithms on Credit card fraud datasets

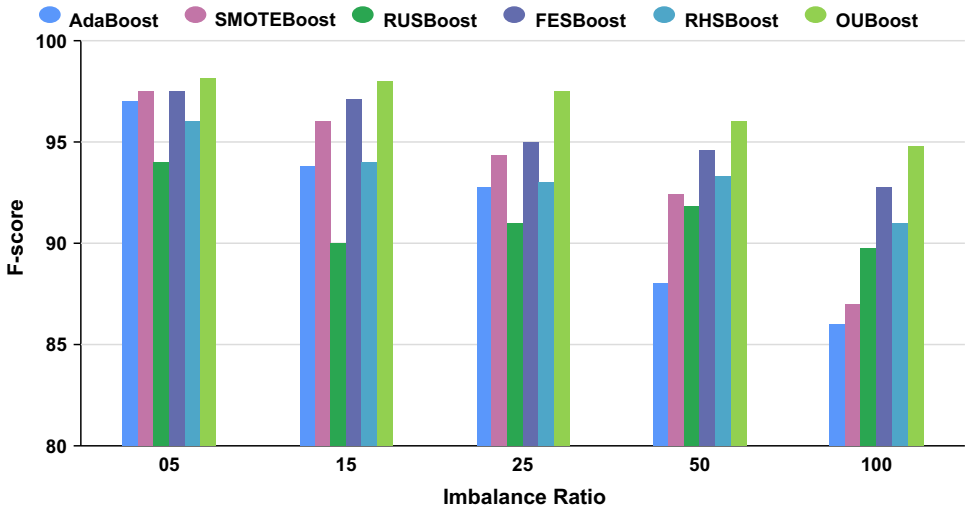

Fig. 10 The minority class F-score averages of different algorithms on CICIDS2017 datasets

algorithm gives good improvements when there is a high imbalance ratio in most used datasets.

5.10 F-score results

In this experiment, we report the result when the evaluation metric is F-score. The F-score value indicates that only a good precision and good recall together result in a good F-measure. In Figs. 8, 9, 10, the horizontal bar is related to different IRs, and the vertical bar is for a minority class F-score. The results in the figures show that OUBoost gives better results than all five boosting algorithms AdaBoost, SMOTEBoost, RUSBoost, FESBoost, and RHSBoost on datasets.

As these figures show, the OUBoost algorithm improves the classification performance of imbalance learning. We further observe that the OUBoost algorithm gives the best result.

According to the figures, the RUSBoost algorithm performs relatively better in high IRs, while the SMOTEBoost algorithm loses its performance. Other hybrid algorithms FESBoost, RHSBoost, try to maintain their efficiency when increasing IR. This indicates that the use of hybrid methods is more appropriate for classifying datasets with high imbalance ratios.

6 Conclusions

In this paper, we first propose a new advanced technique for undersampling the majority class in imbalanced datasets. The proposed peak-based undersampling algorithm uses the peak density clustering technique to select the majority class samples with the highest density and distance from the minority class. This algorithm intelligently ignores low-value and noisy samples from the majority class and preserves useful and informative samples for learning. Then, we propose a novel boosting-based algorithm to deal with the imbalance problem. Our OUBoost algorithm uses the proposed peak-based undersampling method and oversampling technique in the boosting process. In the proposed algorithm, for creating temporary new datasets in each iteration, the number of majority class samples can be determined based on the minority class ratio. As a result, temporary new datasets are created with lower imbalance ratios than the original dataset.

In addition, we investigated boosting-based algorithms to deal with imbalanced data. We analyzed the performance of boosting algorithms using 30 imbalanced datasets with various effects such as dataset size, imbalance ratio, noise samples, and outliers. For this purpose, we use several evaluation metrics including Recall, MCC, G-mean, and F-Score. We also report time comparisons and statistical test analyses of the algorithms. By observing the results, it can be concluded that the use of hybrid methods, including the proposed OUBoost algorithm, works better than other simple boosting methods. In fact, the use of oversampling and undersampling simultaneously has better results in the boosting process than simple methods. However, oversampling is an effective approach to deal with the imbalanced data problem, but undersampling alone does not yield acceptable results compared to other methods. This is due to the loss of information in the majority class.

Possible future work for this study includes analyzing the impact of other oversampling and undersampling methods on the performance of boosting-based algorithms to deal with the imbalance problem.

Data Availability We have used a set of general datasets and cite them in the manuscript.

References

1. Majid A, Ali S, Iqbal M, Kausar N (2014) Prediction of human breast and colon cancers from imbalanced data using nearest neighbor and support vector machines. Comput Methods Programs Biomed 113(3):792–808
2. Di Martino M, Decia F, Molinelli J, Fernández A (2012) Improving electric fraud detection using class imbalance strategies. ICPRAM (2).
3. Chawla NV, Japkowicz N, Kotcz A (2004) Special issue on learning from imbalanced data sets. ACM SIGKDD Explor Newsl 6(1):1–6
4. Weiss GM (2004) Mining with rarity: a unifying framework. ACM SIGKDD Explor Newsl 6(1):7–19
5. Ganganwar V (2012) An overview of classification algorithms for imbalanced datasets. Int J Emerg Technol Adv Eng 2(4):42–47
6. Hernandez J, Carrasco-Ochoa JA, Martínez-Trinidad JF (2013) An empirical study of oversampling and undersampling for instance selection methods on imbalance datasets. In: Iberoamerican Congress on Pattern Recognition.
7. Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP (2002) SMOTE: synthetic minority over-sampling technique. J Artif Intell Res 16:321–357
8. Estabrooks A (2000) A combination scheme for inductive learning from imbalanced data sets [DalTech].
9. Kubat M, & Matwin S (1997) Addressing the curse of imbalanced training sets: one-sided selection. Icml.
10. Pazzani M, Merz C, Murphy P, Ali K, Hume T, Brunk C (1994) Reducing misclassification costs. In: Machine Learning Proceedings 1994 (pp. 217–225). Elsevier.
11. Japkowicz N (2001) Supervised versus unsupervised binary-learning by feedforward neural networks. Mach Learn 42(1):97–122
12. Batista GE, Prati RC, Monard MC (2004) A study of the behavior of several methods for balancing machine learning training data. ACM SIGKDD Explor Newsl 6(1):20–29
13. Drummond C, Holte RC (2003) C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In: Workshop on learning from imbalanced datasets II.
14. Freund Y, Schapire RE (1996) Experiments with a new boosting algorithm. icml.
15. Freund Y, Schapire R, Abe N (1999) A short introduction to boosting. J Jpn Soc Artif Intell 14(771–780):1612
16. Seiffert C, Khoshgoftaar TM, Van Hulse J, Napolitano A (2008) Resampling or reweighting: a comparison of boosting implementations. In: 2008 20th IEEE International Conference on Tools with Artificial Intelligence.
17. Seiffert C, Khoshgoftaar TM, Van Hulse J, Napolitano A (2009) RUSBoost: a hybrid approach to alleviating class imbalance. IEEE Trans Syst Man Cybern Part A Syst Hum 40(1):185–197
18. Chawla NV, Lazarevic A, Hall LO, Bowyer KW (2003) SMOTEBoost: improving prediction of the minority class in boosting. In: European conference on principles of data mining and knowledge discovery.
19. Hasib KM, Iqbal M, Shah FM, Mahmud JA, Popel MH, Showrov M, Hossain I, Ahmed S, Rahman O (2020) A survey of methods for managing the classification and solution of data imbalance problem. arXiv preprint arXiv:2012.11870.
20. Gong J, Kim H (2017) RHSBoost: improving classification performance in imbalance data. Comput Stat Data Anal 111:1–13
21. Popel MH, Hasib KM, Habib SA, Shah FM (2018) A hybrid under-sampling method (HUSBoost) to classify imbalanced data. In: 2018 21st international conference of computer and information technology (ICCIT).
22. Ahmed S, Rayhan F, Mahbub A, Jani R, Shatabda S, Farid DM. (2019). LIUBoost: locality informed under-boosting for imbalanced data classification. In: *Emerging Technologies in Data Mining and Information Security* (pp. 133–144). Springer.
23. Raghuwanshi BS, Shukla S (2019) Classifying imbalanced data using ensemble of reduced kernelized weighted extreme learning machine. Int J Mach Learn Cybern 10(11):3071–3097

24. Hsiao Y-H, Su C-T, Fu P-C (2020) Integrating MTS with bagging strategy for class imbalance problems. Int J Mach Learn Cybern 11(6):1217–1230
25. Raghuwanshi BS, Shukla S (2020) SMOTE based class-specific extreme learning machine for imbalanced learning. Knowl Based Syst 187:104814
26. Raghuwanshi BS, Shukla S (2021) Classifying imbalanced data using SMOTE based class-specific kernelized ELM. Int J Mach Learn Cybern 12(5):1255–1280
27. Jiang M, Yang Y, Qiu H (2022) Fuzzy entropy and fuzzy support-based boosting random forests for imbalanced data. Appl Intell 52(4):4126–4143
28. Dong J, Qian Q (2022) A density-based random forest for imbalanced data classification. Fut Internet 14(3):90
29. Kamalov F, Moussa S, Avante Reyes J (2022) KDE-based ensemble learning for imbalanced data. Electronics 11(17):2703
30. Puri A, Kumar Gupta M (2022) Improved hybrid bag-boost ensemble with K-means-SMOTE–ENN technique for handling noisy class imbalanced data. Comput J 65(1):124–138
31. Zhai J, Qi J, Zhang S (2022) Imbalanced data classification based on diverse sample generation and classifier fusion. Int J Mach Learn Cybern 13(3):735–750
32. Rodriguez A, Laio A (2014) Clustering by fast search and find of density peaks. Science, 344(6191): 1492–1496.
33. Li Z, Tang Y (2018) Comparative density peaks clustering. Expert Syst Appl 95:236–247
34. Mohseni M, Tanha J (2021) A density-based undersampling approach to intrusion detection. In: 2021 5th International Conference on Pattern Recognition and Image Analysis (IPRIA).
35. Bache K, Lichman M (2017) UCI machine learning repository. In: University of California, School of Information and Computer Science, Irvine, CA (2013)
36. Alcalá-Fdez J, Fernández A, Luengo J, Derrac J, García S, Sánchez L, Herrera F (2011) Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. J Multiple Valued Logic Soft Comput: 17.
37. Machine Learning Mastery repository, available on: https://github.com/jbrownlee.
38. Douzas G, Bacao F, Last F (2018) Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE. Inf Sci 465:1–20
39. Sharafaldin I, Lashkari AH, Ghorbani AA (2018) Toward generating a new intrusion detection dataset and intrusion traffic characterization. ICISSp 1:108–116
40. Mammogaphy dataset, available on: https://www.bcsc-research.org/data/mammography_dataset/digitial-mammo-dataset-download.
41. Creditcardfraud dataset, available on: https://www.kaggle.com/mlg-ulb/creditcardfraud.
42. Chicco D, Jurman G (2020) The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. BMC Genom 21(1):1–13
43. Sun Y, Kamel MS, Wang Y (2006) Boosting for learning multiple classes with imbalanced class distribution. In: Sixth international conference on data mining (ICDM'06).
44. Rahman MS, Rahman MK, Kaykobad M, Rahman MS (2018) isGPT: an optimized model to identify sub-Golgi protein types using SVM and Random Forest based feature selection. Artif Intell Med 84:90–100
45. Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30
46. Holm S (1979) A simple sequentially rejective multiple test procedure. Scand J Stat: 65–70.