


Improving undersampling-based ensemble with rotation forest for imbalanced problem

Huaping GUO*, Xiaoyu DIAO, Hongbing LIU

School of Computer and Information Technology, Xinyang Normal University, Xinyang, P.R. China

Received: 23.05.2018

Accepted/Published Online: 08.01.2019

Final Version: 22.03.2019

Abstract: As one of the most challenging and attractive issues in pattern recognition and machine learning, the imbalanced problem has attracted increasing attention. For two-class data, imbalanced data are characterized by the size of one class (majority class) being much larger than that of the other class (minority class), which makes the constructed models focus more on the majority class and ignore or even misclassify the examples of the minority class. The undersampling-based ensemble, which learns individual classifiers from undersampled balanced data, is an effective method to cope with the class-imbalance data. The problem in this method is that the size of the dataset to train each classifier is notably small; thus, how to generate individual classifiers with high performance from the limited data is a key to the success of the method. In this paper, rotation forest (an ensemble method) is used to improve the performance of the undersampling-based ensemble on the imbalanced problem because rotation forest has higher performance than other ensemble methods such as bagging, boosting, and random forest, particularly for small-sized data. In addition, rotation forest is more sensitive to the sampling technique than some robust methods including SVM and neural networks; thus, it is easier to create individual classifiers with diversity using rotation forest. Two versions of the improved undersampling-based ensemble methods are implemented: 1) undersampling subsets from the majority class and learning each classifier using the rotation forest on the data obtained by combing each subset with the minority class and 2) similarly to the first method, with the exception of removing the majority class examples that are correctly classified with high confidence after learning each classifier for further consideration. The experimental results show that the proposed methods show significantly better performance on measures of recall, g-mean, f-measure, and AUC than other state-of-the-art methods on 30 datasets with various data distributions and different imbalance ratios.

Key words: Undersampling, ensemble, rotation forest, imbalanced problem

1. Introduction

The class imbalance problem, which is also named the skewed or rare class problem, refers to a dataset with at least the examples of one of its classes outnumbered by other classes [1–3]. For two-class data, the examples are commonly categorized into the majority class or the minority class, and the cost of misclassifying the minority class examples is often higher than the contrary cases, particularly for medical datasets, where high-risk patients tend to be the minority class. For example, in cancer detection, most patients have the common disease, while rare patients may have cancer, and how to effectively recognize cancer patients is notably meaningful. However, most conventional classification methods attempt to train models with high accuracy by assuming that the number of examples of each class is similar to each other, which makes the minority class examples often be ignored and misclassified as the majority class [4].

*Correspondence: hpguo@xynu.edu.cn

Many methods to handle imbalanced problems have been proposed, and the sampling technique and ensemble learning are two candidates of the most often used approaches. The sampling techniques, including undersampling and oversampling, aim to balance the dataset with a skew distribution to remove the harm of the imbalanced problem by sampling the data space. Undersampling techniques such as random undersampling [4, 5], which is the simplest sampling method, attempt to eliminate the harms of the imbalanced distribution by removing the intrinsic examples in the majority class. Contrary to undersampling, oversampling techniques aim to create new minority class examples [6–10]. For example, the random oversampling technique increases the number of examples of the minority class by randomly duplicating the minority class examples, and the synthetic minority oversampling technique (SMOTE) increases the ones by creating new synthetic examples along the line between the minority examples and their selected nearest neighbors [7, 8].

Ensemble learning, which has often been used to solve challenging issues such as medical detection [11], image recognition [12, 13], and software defect prediction [14], is an effective technique for the imbalanced problem. Ensemble methods can be grouped into two levels: iterative-based ensembles and parallel-based ensembles. Boosting is the most common and effective iterative-based ensemble learning method and is usually used to handle imbalanced data by combining data-preprocessing techniques such as the undersampling technique into the learning process. In this manner, these methods alter and bias the data distribution to train the next classifier toward the minority class, e.g., RUSBoost [15], EUSBoost [3, 16], and BNU-SVMs [17]. BalanceCascade [5] is a boost-like method for class-imbalance data; after learning an individual classifier, it removes the majority class examples that are correctly classified by the classifier for further consideration. The parallel-based ensembles refer to ensemble models where each classifier can be trained in parallel. The undersampling technique is often combined with a parallel-based ensemble such as UnderBagging [18] and EasyEnsemble [5, 19], where UnderBagging uses undersampling techniques to create diverse bags to learn individual classifiers. The UnderBagging method has been used with different names such as asymmetric bagging [20] and roughly balanced bagging [21]. Like UnderBagging, EasyEnsemble learns each individual class in each diverse bag, but EasyEnsemble uses AdaBoost to train the individual classifiers. Therefore, EasyEnsemble is an ensemble of ensembles.

In this paper, an improved undersampling-based ensemble method is proposed by embedding rotation forest [22] into the ensemble learning process, i.e. learning each classifier using rotation forest. The idea is inspired by two observations: 1) rotation forest has higher generalization ability than other ensemble methods such as bagging [23], boosting [24], and random forest [25], particularly for small-sized data; and 2) it is more sensitive to the sampling technique with diversity than some robust methods including SVM and neural network. Thus, it is easier to create individual classifiers using rotation forest. Like the undersampling-based ensemble methods proposed by Liu et al. [5, 19], this paper implements two versions of undersampling-based ensemble methods: 1) undersampling subsets from the majority class and learning each classifier using rotation forest on the data obtained by combining each subset with the minority class and 2) similarly to the first method, with the exception of removing the examples of the majority class that are correctly classified with high confidence after learning each classifier for further consideration. For the prediction, individual classifiers produce estimates of the posterior probabilities for each class. These probabilities are averaged across the classifiers, and the most probable class is assigned. The experimental results show that the proposed methods show significantly better performance on measures of recall, g-mean, f-measure, and AUC than other state-of-the-art undersampling-based methods on 30 datasets with various data distributions and different imbalance ratios.

The remainder of this paper is organized as follows: after presenting the strategies to handle the

imbalanced problem in Section 2, Section 3 describes the proposed method; Section 4 presents the experimental results; and, finally, Section 5 concludes this work.

2. Related work

Imbalance problems arise from the scarce representation of the most important examples, which leads to the fact that the learned models tend to focus more on the majority class examples, overlooking the minority class ones. Numerous techniques have been developed to solve the problem of class-imbalance datasets, and the sampling technique and ensemble learning are two of the most often used methods.

2.1. Sampling technique

Sampling techniques aim to alleviate the effect of the class imbalance distribution through sampling data space to rebalance the corresponding imbalanced dataset, so that the conventional learning methods pay more attention to the minority class. Resampling methods are notably versatile because they are independent of the selected classifier [26]. Commonly used sampling techniques belong to three categories depending on the method to balance the class distribution: undersampling, oversampling, and hybrid methods.

2.1.1. Undersampling techniques

Undersampling techniques eliminate the harms of a skewed distribution by discarding the intrinsic examples of the majority class. Random undersampling, which is a commonly used sampling method, randomly discards the majority class examples until a relatively balanced distribution is reached. The problem in random undersampling is that useful examples may be eliminated, causing a worse classification performance [27]. To overcome this drawback, many methods have been proposed to retain useful information in the majority class, such as the condensed nearest neighbor (CNN) rule [28] and Tomek links [29]. The CNN attempts to remove redundant examples of the majority class, and Tomek links discard the borderline and noisy examples. Japkowicz [30] discussed the sampling technique and observed that the technique was notably effective; furthermore, she noted that using sophisticated sampling techniques does not provide any clear advantage in solving the class imbalance problem. The studies of Sun et al. [31] showed that the random undersampling approaches often outperform complicated undersampling methods.

In this paper, we consider the undersampling technique to learn an ensemble instead of a single model to avoid the problem of the technique: useful examples may be eliminated and cause a worse classification performance.

2.1.2. Oversampling techniques

Oversampling techniques aim to create new minority class examples to eliminate the harms of the imbalanced problem. Random oversampling is one of the most widely used oversampling methods, which randomly duplicates the minority class examples such that the class distribution is more balanced. Although the oversampling technique creates a more balanced distribution, it suffers from the drawback of overfitting [27]. SMOTE [7] was proposed to overcome this drawback, which synthetically generates new minority class examples along the line between two selected minority class examples. Borderline oversampling [32] only oversamples the borderline minority class examples, since the borderline region is more crucial for establishing the decision boundary. The majority weighted minority oversampling technique (MWMOTE) [10] identifies the hard-to-learn informative

minority examples, assigns weights according to their distances from the nearest majority class examples, and generates synthetic examples from the weighted minority class examples.

2.1.3. Hybrid method

A hybrid method is a combination of the oversampling and undersampling methods. As an example, DTE-SBD (decision tree ensemble based on SMOTE, bagging, and Dsr) [33] uses differentiated sampling rates for the minority and majority classes with different principles. SMOTE is used to increase the examples of the minority class without repeating, and bagging is used to draw the majority class subset with a certain degree of diversity. Estabrooksetal et al. [34] suggested that a combination of oversampling and undersampling might be more effective to solve the class-imbalance problem.

2.2. Ensemble methods

Classifier ensembles, also named multiple classifier systems, are known to enhance the performance of a single classifier by combining several base classifiers. According to Guo et al. [1], the classifier ensembles for the class-imbalance problem can be grouped into two categories: iterative-based and parallel-based ensembles.

2.2.1. Iterative-based ensembles

Boosting is the most common and most effective iterative-based method for ensemble learning, which attempts to embed techniques for data preprocessing into the boosting algorithms for class-imbalance data. In this manner, these methods alter and bias the weight distribution that is used to train the next classifier toward the minority class. An undersampling technique is often combined with boosting methods for imbalanced problems. Examples include RUSBoost [15] and BNU-SVMs [17], where RUSBoost removes examples from the majority class in each iteration, and the weights of the remaining examples are simply normalized in the new dataset with respect to their total sum of weights. BNU-SVM undersamples the majority class set to keep the borderline examples to build the boosted training sets.

BalanceCascade [5, 19] is a boosting-like method, which sequentially trains individual classifiers. For each iteration, BalanceCascade removes the majority class examples that are correctly classified by the current classifier, i.e. the examples are not considered in further iterations. In addition, BalanceCascade trains each classifier using AdaBoost instead of a single model; thus, BalanceCascade is a hybrid model, i.e. an ensemble of ensembles. Like BalanceCascade, Adaptive EUSBoost learns each classifier using AdaBoost on each undersampled data. Unlike BalanceCascade, Adaptive EUSBoost embeds a cost-sensitive weight modification and an adaptive boundary decision strategy into the learning process to improve the model performance on class-imbalance data.

This paper proposes a new iterative-based ensemble. Similar to BalanceCascade and EUSBoost, the proposed method learns each classifier on undersampled data, but it learns each classifier using rotation forest to improve the performance of the individual classifiers.

2.2.2. Parallel-based ensembles

In this study, parallel-based ensembles refer to ensemble models where each base classifier can be trained in parallel. Bagging is the most often used parallel-based ensemble method and is often combined with an undersampling technique for imbalanced learning. Examples include UnderBagging [18] and Ensemble Under-sampling [35], where UnderBagging trains each classifier using a balanced dataset obtained by undersampling

the majority class set. Let IR be the imbalance ratio defined by the ratio between the sizes of the majority class set and minority class set. Ensemble undersampling randomly partitions the majority class set into IR disjoint subsets and combines each subset with the minority class set as a balanced one to learn an individual classifier.

As a special version of UnderBagging, EasyEnsemble [5, 19] undersamples several subsets from the majority class, trains an individual classifier on each subset, and combines the outputs of individual classifiers. Unlike UnderBagging, the base classifier of EasyEnsemble is learned by the ensemble learning method AdaBoost instead of a single model, and AdaBoost is mainly used to reduce the bias, whereas bagging mainly reduces the variance. Therefore, like BalanceCascade, EasyEnsemble is an ensemble of ensembles.

This paper proposes a new parallel-based ensemble method. Similar to undersampling and EasyEnsemble, the proposed method learns each classifier on balanced data from the randomly sampling majority class, but it learns each classifier using rotation forest to improve the accuracy of the classifier on the small balanced data.

Table 1. Rotation forest.

Input: the training set \mathbf{X} in form of $N \times n$ matrix; the labels \mathbf{Y} of training set in form of $N \times 1$ matrix; the number of iterations I to train individual classifiers; and K , the number of splitting feature set \mathbf{F} into disjoint subsets

1. $H = \emptyset$;
2. **for** $i = 1$ to I **do**
3. Split the feature set \mathbf{F} into K subsets: $\mathbf{F}_{i,j}$ ($j = 1$ to $|\mathbf{F}|/K$);
4. **for** $j = 1$ to K **do**
5. Let $\mathbf{X}_{i,j}$ be the dataset \mathbf{X} for the features in $\mathbf{F}_{i,j}$;
6. Eliminate from $\mathbf{X}_{i,j}$ a random subset of classes;
7. Sample from $\mathbf{X}_{i,j}$ with 75% of the size of $\mathbf{X}_{i,j}$ denoted the new dataset by $\mathbf{X}'_{i,j}$;
8. Apply PCA to obtain the coefficients in a matrix $\mathbf{C}_{i,j}$;
9. **end for**
10. Arrange the $\mathbf{C}_{i,j}$ in a matrix \mathbf{C}_i , $j = 1$ to $/K$;
11. Construct \mathbf{R} by rearranging the columns of \mathbf{C}_i to match the order of features in \mathbf{F} ;
12. Build h_i using $(\mathbf{X}\mathbf{R}_i, \mathbf{Y})$;
13. $H = H \cup \{h_i\}$;
14. **end for**

Output: An ensemble $H(\mathbf{x}) = \arg_j \max_{j \in \{1,2,\dots,c\}} \{\mu(\mathbf{x}, j)\}$ where $\mu(\mathbf{x}, j) = \frac{1}{I} \sum_i^T h_i(\mathbf{x}, j)$ and $h_i(\mathbf{x}, j)$ is the probability that decision tree h_i predicts \mathbf{x} to be class j .

Table 2. IUE-I: the first version of the improved undersampling-based ensemble.

Input: the majority class set \mathbf{X}_{maj} ; the minority class set \mathbf{X}_{min} ; the number T of subsets sampled from \mathbf{X}_{maj} ; and I , the number of iterations for training rotation forest

1. $H = \emptyset$;
2. **for** $i = 1$ to T **do**
3. Randomly sampling a subset $\mathbf{X}_{maj,i}$ from \mathbf{X}_{maj} with $|\mathbf{X}_{maj,i}| = |\mathbf{X}_{min}|$;
4. $\mathbf{X}_i = \mathbf{X}_{maj,i} \cup \mathbf{X}_{min}$;
5. Let \mathbf{Y}_i be the label vector corresponding to \mathbf{X}_i ;
6. Call rotation forest to train the classifier H_i with I base classifiers using balanced data $(\mathbf{X}_i, \mathbf{Y}_i)$;
7. $H = H \cup \{H_i\}$;
8. **end for**

Output: An ensemble $H(\mathbf{x}) = \arg_j \max_{j \in \{maj,min\}} \{\mu(\mathbf{x}, j)\}$ where $\mu(\mathbf{x}, j) = \frac{1}{T} \sum_i^T H_i(\mathbf{x}, j)$ and $H_i(\mathbf{x}, j)$ is the probability that rotation forest H_i predicts \mathbf{x} to be class j .

3. Proposed method

The class-imbalance problem exists in many applications [1, 2] and decreases the performance of conventional classifier learning methods. Undersampling-based ensemble methods are often used to handle the imbalanced problem, and the key issue of these methods is how to improve the performance of individual classifiers on limit

Table 3. IUE-II: the second version of the improved undersampling-based ensemble.

Input: the majority class set \mathbf{X}_{maj} ; the minority class set \mathbf{X}_{min} ; the number T of subsets sampled from \mathbf{X}_{maj} ; and I , the number of iterations for training rotation forest.

1. $H = \emptyset$;
2. $f = \frac{|\mathbf{X}_{maj}| - |\mathbf{X}_{min}|}{T-1}$;
3. **for** $i = 1$ to T **do**
4. Randomly sampling a subset $\mathbf{X}_{maj,i}$ from \mathbf{X}_{maj} with $|\mathbf{X}_{maj,i}| = |\mathbf{X}_{min}|$;
5. $\mathbf{X}_i = \mathbf{X}_{maj,i} \cup \mathbf{X}_{min}$;
6. Let \mathbf{Y}_i be the label vector corresponding to \mathbf{X}_i ;
7. Call rotation forest to train the classifier H_i with I base classifiers using balanced data $(\mathbf{X}_i, \mathbf{Y}_i)$;
8. Removing from \mathbf{X}_{maj} the former f examples that are correctly classified by H_i with high confidence;
9. $H = H \cup \{H_i\}$;
10. **end for**

Output: An ensemble $H(\mathbf{x}) = \arg_j \max_{j \in \{maj, min\}} \{\mu(\mathbf{x}, j)\}$ where $\mu(\mathbf{x}, j) = \frac{1}{T} \sum_i^T H_i(\mathbf{x}, j)$ and $H_i(\mathbf{x}, j)$ is the probability that rotation forest H_i predicts \mathbf{x} to be class j .

balanced data obtained by undersampling the majority class, without harming the diversity among individual classifiers. This section proposes an improved undersampling-based ensemble (IUE) method using rotation forest. Two versions of IUE are implemented: IUE-I and IUE-II.

3.1. IUE-I

Suppose $\mathbf{x} = [x_1, x_2, \dots, x_n]$ is a data point described by n features, and let \mathbf{X}_{maj} be the majority class set containing objects in the form of an $N_{maj} \times n$ matrix and \mathbf{X}_{min} be the minority class set in a form of an $N_{min} \times n$ matrix. IUE-I uses a straightforward way to build an undersampling-based ensemble method. Like EasyEnsemble, IUE-I undersamples several subsets $\mathbf{X}_{maj,1}, \mathbf{X}_{maj,2}, \dots, \mathbf{X}_{maj,T}$ from \mathbf{X}_{maj} with the size of $\mathbf{X}_{maj,i}$ equal to that of \mathbf{X}_{min} for $i = 1, 2, \dots, T$, and learns a classifier H_i using $\mathbf{X}_{maj,i}$ and all of \mathbf{X}_{min} . Rotation forest [22], which is an ensemble learning method, is used to train each classifier (subensemble) H_i . The main heuristic of rotation forest is to apply feature extraction to subsets of features and reconstruct a full feature set for each base classifier, randomly splitting the feature set into K disjoint subsets, running principal component analysis (PCA) on each subset with a bootstrap sample of training set, and then organizing all the principal components in a sparse rotation matrix \mathbf{R} . A classifier is trained on the whole training set in the feature space defined by \mathbf{R} . A decision tree is selected as the base learner as it is sensitive to feature rotation. Table 1 shows the pseudocode for rotation forest. Therefore, IUE-I is an ensemble of ensembles.

For the prediction, unlike EasyEnsemble, individual classifiers of IUE-I produce estimates of the posterior probabilities for each class. These probabilities are averaged across the classifiers, and the most probable class is assigned. The pseudocode for IUE-I is shown in Table 2.

The idea behind IUE-I is simple. Similar to underbagging [18] and EasyEnsemble [5, 19], IUE-I generates T balanced subproblems and combines the T results to further exploit the majority class examples, which are ignored by the undersampling. In addition, since the decision tree is sensitive to the sampling technique, rotation forest, which is an ensemble of decision trees, is also sensitive to the sampling technique. Therefore, IUE-I guarantees the high diversity among individual classifiers learned by rotation forest, which is a key to the success of an ensemble. Furthermore, the minority class set \mathbf{X}_{min} is often notably small; thus, the training set \mathbf{X}_i is notably small, i.e. $|\mathbf{X}_i| = 2|\mathbf{X}_{min}| = 2N_{min}$. Therefore, how to learn individual classifiers with high performance is a challenge, which is another key to the success of an ensemble. Since rotation forest is observed to be notably accurate, it is used as the weak learner (although it is not weak) in this paper.

3.2. IUE-II

Unlike IUE-I, which uses an unsupervised strategy to learn each classifier, IUE-II learns each classifier in a supervised manner. Specifically, after constructing classifier h_i , the examples of the majority class that are correctly classified by h_i with high confidence are removed to train classifier h_{i+1} following BalanceCascade. The rationality is that given classifier h_i , the examples that are correctly classified by h_i are somewhat redundant. Like IUE-I, rotation forest is used as the weak learner. Table 3 describes the pseudocode of IUE-II.

IUE-II is similar to IUE-I with the exception of lines 2 and 8 of Table 3. Line 8 removes the true majority class examples with the top f highest confidence, and line 2 specifies how many majority class examples can be removed, i.e. $f = \frac{|\mathbf{X}_{maj}| - |\mathbf{X}_{min}|}{T-1}$. Therefore, at the beginning of the T th iteration, \mathbf{X}_{maj} has been shrunk $T - 1$ times, and the current size of \mathbf{X}_{maj} is $|\mathbf{X}_{maj}| - f \times (T - 1) = |\mathbf{X}_{min}|$.

For prediction, individual classifiers of IUE-II produce the estimates of the posterior probabilities for each class. These probabilities are averaged across the classifiers, and the most probable class is assigned.

Table 4. The description of the datasets used for experiments.

ID	Name	#Attr	Size	#IR	ID	Name	#Attr	Size	#IR
id1	glass-0-1-2-3_vs_4-5-6	9	214	3.20	id16	glass4	9	214	15.47
id2	vehicle0	18	846	3.25	id17	ecoli4	7	336	15.80
id3	ecoli1	7	336	3.36	id18	page-blocks-1-3_vs_4	10	472	15.86
id4	new-thyroid1	5	215	5.14	id19	abalone9-18	8	731	16.40
id5	new-thyroid2	5	215	5.14	id20	dermatology-6	34	358	16.90
id6	ecoli2	7	336	5.46	id21	shuttle-c2-vs-c4	9	129	20.50
id7	segment0	19	2308	6.02	id22	shuttle-6_vs_2-3	9	230	22.00
id8	glass6	9	214	6.38	id23	yeast-1-4-5-8_vs_7	8	693	22.10
id9	ecoli3	7	336	8.60	id24	car-good	6	1728	24.04
id10	page-blocks0	10	5472	8.79	id25	car-vgood	6	1728	25.58
id11	yeast-0-5-6-7-9_vs_4	8	528	9.35	id26	yeast4	8	1484	28.10
id12	vowel0	13	988	9.98	id27	yeast-1-2-8-9_vs_7	8	947	30.57
id13	glass-0-1-6_vs_2	9	192	10.29	id28	yeast5	8	1484	32.73
id14	glass2	9	214	11.59	id29	ecoli-0-1-3-7_vs_2-6	7	281	39.14
id15	yeast-1_vs_7	7	459	14.30	id30	yeast6	8	1484	41.40

4. Experiments

4.1. Datasets and experimental setup

Thirty binary datasets are selected from the KEEL dataset repository [36] for comparison, and the details are described in Table 4, where #Attr and #IR are the attributes number and imbalanced ratio, respectively. The imbalanced ratio is defined as the ratio between the size of the majority class set and that of the minority class set. The datasets are from various application domains, and the imbalanced degree of the datasets varies from 41.4 (highly imbalanced) to 3.2 (only slightly imbalanced). In this section, a 5×2 -fold cross-validation strategy [37] is conducted to evaluate the performance of the proposed method. Ten methods were selected as candidates to test the performance of the proposed method:

- *Undersampling-C45* (UC) [38] preprocesses the training set using the random undersampling technique to obtain a relatively balanced dataset and learns a model using C4.5 [39] on the balanced set.
- *UnderBagGing* (UBG) [18] learns each member on the undersampled subset from the majority class set, and C4.5 is selected as the base learner. The number of members T is set to be 100.

- **RUSBoosT** (RBT) [15] combines the undersampling technique into the boost learning process. The number of members T is set to be 100; for each member, a subset with size equal to the minority class is sampled (without replacement) from the majority class. Then C4.5 is used to train a classifier using the subset and minority class set.
- **EasyEnsemble** (EE) [5, 22] samples T subsets from the majority class and trains AdaBoost with J weak learners using each subset. We select C4.5 as the weak classifier and set $T = 10$ and $J = 10$.
- **Ensemble UnderSampling** (EUS) [35] randomly partitions the majority class set into several disjoint subsets, and the size of each subset is equal to that of the minority class set. EUS combines the minority class set and each subset to train a base classifier. Therefore, the number of ensemble members is equal to the imbalanced ratio. We set C4.5 as the base learner.
- **Adaptive Ensemble Undersampling-Boost** (AEUB) [16] randomly partitions the majority class set into several disjoint subsets (like EUS), learns an individual classifier using AdaBoost, and determines the weight of each individual classifier. After learning an ensemble, the optimal adaptive decision boundary is determined using validation data. In this section, the number of the base classifiers of AdaBoost is set to 10. Following [16], one-third of the original training set is used as the validation data and the others as the training set.
- **BalanceCascade** (BC) [5, 22] is similar to EasyEnsemble with the exception of removing the correctly classified major class examples from further consideration. C4.5 is selected to train the weak classifiers, and we set $T = 10$ and $J = 10$.
- **DTE_SBD** [33] constructs a decision tree ensemble based on SMOTE, undersampling, and differentiated sampling rates. The number of individual classifiers is set to 100. The k -nearest neighbor parameter of SMOTE is set to 5, and C4.5 is selected to train the weak classifiers.
- **IUE-I**. The number of subsets is $T = 10$, and the number of iterations of rotation forest is $I = 10$ (refer to Table 2). C4.5 is used as the base learner of rotation forest and K (the number of splitting feature set \mathbf{F} of rotation forest) is set to be $\mathbf{F}/3$ (refer to Table 1).
- **IUE-II**. The experimental setting is the same as that of IUE-I.

Table 5. Confusion matrix.

	Predicted as positive	Predicted as negative
Actually positive	TP	FP
Actually negative	FN	TN

4.2. Evaluation measure

Evaluation measure is extremely essential to assess the effectiveness of an algorithm, and for imbalanced problems, precision, recall, f-measure, g-mean, and AUC are the most frequently used ones. The examples predicted by a classifier can be categorized into four groups as shown in Table 5, and the precision and recall are defined as $precision = TP/(TP + TF)$ and $recall = TP/(TP + FN)$. F-measure is a harmonic mean between recall and precision, defined as

$$f - measure = \frac{(1 + \delta^2) \times recall \times precision}{\delta^2 \times recall + precision}, \tag{1}$$

where δ , often set to be 1, is a coefficient to adjust the relative importance of precision versus recall.

G-mean is another metric considering both the positive class and negative class, defined as

$$g - mean = \sqrt{\frac{TP}{TP + TF} \times \frac{TN}{TN + TF}}. \tag{2}$$

Therefore, g-mean uses the geometric mean of the recall of the positive class and that of the negative class to measure a classifier performance. Besides, AUC is a commonly used measure to evaluate models' performances on imbalanced problems. According to [27], AUC can be estimated by

$$AUC = \left(\frac{TP}{TP + TF} + \frac{TN}{TN + TF} \right) / 2. \tag{3}$$

In this paper, recall, f-measure, g-mean, and AUC are employed to evaluate the classification performance on imbalanced datasets.

Table 6. The recall of comparing methods.

ID	IUE-II	IUE-I	AEUB	DTE_SBD	RBT	BC	EE	EUS	UBG	UC
id1	0.9457	0.9378	0.8940	0.8825	0.8665	0.9138	0.9100	0.8945	0.8905	0.8472
id2	0.9950	0.9900	0.9769	0.9447	0.9236	0.9839	0.9860	0.9527	0.9578	0.9308
id3	0.9169	0.8961	0.9093	0.9196	0.7745	0.8913	0.8887	0.8912	0.9354	0.8881
id4	0.9778	0.9722	0.9317	0.8814	0.8765	0.9598	0.9598	0.8984	0.9206	0.9039
id5	0.9824	0.9944	0.9490	0.9317	0.9154	0.9608	0.9549	0.9373	0.9261	0.9261
id6	0.8923	0.8808	0.8846	0.8692	0.8000	0.9000	0.9038	0.8654	0.8731	0.8538
id7	0.9885	0.9885	0.9909	0.9812	0.9818	0.9897	0.9897	0.9824	0.9824	0.9830
id8	0.9167	0.9233	0.8738	0.8895	0.8610	0.8962	0.8824	0.8681	0.8895	0.8610
id9	0.8925	0.8925	0.8928	0.8807	0.6402	0.9150	0.8980	0.9157	0.9271	0.8474
id10	0.9810	0.9778	0.9746	0.9603	0.9131	0.9742	0.9717	0.9617	0.9685	0.9367
id11	0.7725	0.7488	0.7765	0.7568	0.5526	0.7729	0.7688	0.7763	0.7923	0.7451
id12	1.0000	0.9978	0.9822	0.9711	0.9556	0.9822	0.9800	0.9622	0.9622	0.9467
id13	0.7944	0.7542	0.5653	0.5639	0.6222	0.5903	0.5819	0.5764	0.6500	0.7153
id14	0.8569	0.8333	0.6292	0.6139	0.6000	0.6875	0.6208	0.6014	0.6917	0.4847
id15	0.7800	0.6933	0.7400	0.6933	0.5333	0.7067	0.7133	0.7067	0.7200	0.6400
id16	0.9381	0.9381	0.7548	0.8000	0.6905	0.8167	0.8167	0.8000	0.8000	0.7095
id17	0.9200	0.9100	0.9200	0.8800	0.7800	0.9200	0.9100	0.8800	0.8800	0.8300
id18	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
id19	0.7857	0.7524	0.6571	0.6238	0.5857	0.6762	0.6857	0.7048	0.6905	0.5429
id20	1.0000	1.0000	0.9900	0.9900	0.9600	0.9900	0.9800	0.9900	0.9900	0.9800
id21	0.9667	0.9333	0.8333	0.6333	0.6000	0.6333	0.6333	0.6333	0.6333	0.5333
id22	0.9800	0.9600	0.9200	0.9200	1.0000	0.9200	0.9200	0.9200	0.9200	0.9000
id23	0.6933	0.6467	0.5200	0.6000	0.6667	0.6800	0.7000	0.6800	0.6733	0.6867
id24	1.0000	1.0000	1.0000	0.9441	0.9855	1.0000	1.0000	1.0000	1.0000	0.9249
id25	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
id26	0.8500	0.8497	0.8615	0.8577	0.7195	0.8655	0.8695	0.8692	0.8734	0.8615
id27	0.6267	0.6600	0.5667	0.5667	0.6267	0.6733	0.6533	0.6000	0.6600	0.5733
id28	1.0000	0.9909	1.0000	0.9818	0.8227	0.9909	0.9909	0.9909	1.0000	0.9818
id29	0.8583	0.8750	0.7167	0.7750	0.6500	0.7417	0.7750	0.8333	0.8333	0.7417
id30	0.8686	0.8458	0.8974	0.8745	0.7879	0.8974	0.8918	0.8918	0.8863	0.8510
average	0.9060	0.8948	0.8536	0.8396	0.7897	0.8643	0.8612	0.8528	0.8642	0.8209
Win	14	7	6	2	3	3	6	4	7	2
Top-3	23	18	10	3	3	13	13	8	12	4

4.3. Experimental results

To evaluate the performance of IUE (the proposed method), AEUB, DTE_SBD, RBT, BC, EE, EUS, UBG, and UC were selected as the comparison methods (for more details of the methods, refer to Section 4.1). The

Table 7. The g-mean of comparing methods.

ID	IUE-II	IUE-I	AEUB	DTE_SBD	RBT	BC	EE	EUS	UBG	UC
id1	0.9267	0.9243	0.9062	0.8978	0.8997	0.9135	0.9121	0.8935	0.8949	0.8784
id2	0.9696	0.9691	0.9549	0.9387	0.9330	0.9620	0.9620	0.9334	0.9397	0.9172
id3	0.8851	0.8758	0.8722	0.8828	0.8437	0.8677	0.8666	0.8757	0.8851	0.8728
id4	0.9778	0.9724	0.9299	0.9173	0.9195	0.9606	0.9595	0.9164	0.9273	0.9077
id5	0.9809	0.9842	0.9425	0.9473	0.9397	0.9637	0.9609	0.9373	0.9330	0.9239
id6	0.9003	0.8880	0.8930	0.8933	0.8666	0.9027	0.9016	0.8687	0.8730	0.8436
id7	0.9938	0.9937	0.9916	0.9844	0.9879	0.9912	0.9915	0.9823	0.9830	0.9796
id8	0.9315	0.9338	0.8841	0.9173	0.9047	0.9127	0.9038	0.8943	0.9124	0.8656
id9	0.8764	0.8645	0.8707	0.8736	0.7459	0.8818	0.8705	0.8780	0.8788	0.8014
id10	0.9604	0.9578	0.9608	0.9561	0.9438	0.9596	0.9587	0.9520	0.9558	0.9355
id11	0.7995	0.7894	0.7877	0.7922	0.5291	0.7874	0.7859	0.7854	0.7986	0.7665
id12	0.9767	0.9719	0.9734	0.9623	0.9739	0.9714	0.9710	0.9530	0.9539	0.9313
id13	0.7086	0.6630	0.6089	0.6163	0.5061	0.5709	0.5660	0.5905	0.6096	0.3847
id14	0.7689	0.7016	0.6462	0.6681	0.4607	0.6365	0.5996	0.6152	0.6524	0.5349
id15	0.7607	0.7014	0.7233	0.7208	0.5371	0.7042	0.7084	0.7056	0.7102	0.6277
id16	0.8925	0.8732	0.7875	0.7978	0.6562	0.7999	0.8073	0.7745	0.7758	0.6720
id17	0.9346	0.9168	0.9208	0.8911	0.8439	0.9124	0.9067	0.8840	0.8748	0.7730
id18	0.9562	0.9489	0.9394	0.9774	0.9458	0.9221	0.9216	0.9479	0.9351	0.8996
id19	0.8015	0.7770	0.7166	0.7038	0.4639	0.7036	0.7058	0.7237	0.7222	0.6127
id20	0.9991	0.9994	0.9784	0.9823	0.9718	0.9781	0.9745	0.9784	0.9784	0.9722
id21	0.9808	0.9610	0.8450	0.6577	0.6394	0.6569	0.6519	0.6577	0.6577	0.5971
id22	0.9872	0.9171	0.8624	0.9085	0.9220	0.8854	0.8812	0.8759	0.8899	0.8059
id23	0.6446	0.6053	0.5868	0.6255	0.3361	0.6106	0.6228	0.6235	0.6225	0.5401
id24	0.9639	0.9654	0.9492	0.9043	0.8950	0.9429	0.9434	0.9266	0.9277	0.8330
id25	0.9833	0.9730	0.9512	0.9448	0.9173	0.9530	0.9522	0.9385	0.9416	0.9074
id26	0.8292	0.8239	0.8255	0.8344	0.5459	0.8344	0.8363	0.8228	0.8261	0.7998
id27	0.6647	0.6744	0.6512	0.6418	0.4500	0.6648	0.6519	0.6357	0.6742	0.4737
id28	0.9663	0.9585	0.9620	0.9570	0.8894	0.9610	0.9620	0.9555	0.9558	0.9399
id29	0.8257	0.8129	0.7467	0.7850	0.6438	0.7611	0.7832	0.7825	0.7851	0.7000
id30	0.8819	0.8576	0.8717	0.8766	0.7416	0.8710	0.8690	0.8751	0.8737	0.8309
average	0.8909	0.8752	0.8513	0.8485	0.7618	0.8481	0.8463	0.8395	0.8449	0.7843
Win	20	5	1	1	0	2	1	0	1	0
Top-3	28	20	9	11	2	9	5	4	5	0

results are reported in Tables 6, 7, 8, and 9, which report the summary results of the compared methods on the measures of recall, g-mean, f-measure, and AUC. The tables also report the average value on each measure for each algorithm, the winning times, and the times that an algorithm ranks in the top 3 of the 30 datasets.

Table 6 reports the summery results of the ten algorithms on recall. From Table 6, it can be seen that IUE outperforms the other undersampling-based ensemble methods. Specifically, IUE-II (IUE-I) has the best recall on 14 (7) of the 30 datasets and performs in the top 3 on 23 (18) of the sets. Furthermore, IUE-II performs better than IUE-I, which indicates that removing the redundant examples of the majority class is meaningful (refer to Section 3.2).

Figure a shows the average rank of the ten algorithms on recall, where the rank of the methods in each dataset is calculated as follows [40, 41]: on a dataset, the best performing algorithm gets the rank of 1.0, the second best ranks 2.0, and so on. In the case of ties, the average ranks are assigned. As shown in Figure a, IUE-II and IUE-I have much smaller average ranks than other methods. Specifically, IUE-II ranks first with the average rank of 2.75, followed by IUE-I with the average rank of 3.84, BC (4.12), EE (4.66), EBG (4.69), AEUB (5.05), EUS (5.81), DTE_SBD (7.13), UC (8.19), and RBT (8.74).

The Friedman test [42], which is a nonparametric statistical test, is used to determine whether the superiority of our methods is accidental. The Friedman test can be used to detect differences across multiple

Table 8. The fl-measure of comparing methods.

ID	IUE-II	IUE-I	AEUB	DTE_SBD	RBT	BC	EE	EUS	UBG	UC
id1	0.8461	0.8454	0.8311	0.8223	0.8387	0.8358	0.8368	0.8051	0.8092	0.7964
id2	0.9156	0.9182	0.8910	0.8739	0.8755	0.9040	0.9025	0.8545	0.8668	0.8309
id3	0.7661	0.7582	0.7463	0.7600	0.7645	0.7434	0.7426	0.7577	0.7568	0.7585
id4	0.9392	0.9225	0.8230	0.8491	0.8631	0.8950	0.8905	0.8218	0.8267	0.7808
id5	0.9425	0.9392	0.8475	0.8887	0.8816	0.9055	0.9014	0.8406	0.8378	0.8016
id6	0.7556	0.7209	0.7376	0.7582	0.7591	0.7501	0.7414	0.6818	0.6862	0.6309
id7	0.9918	0.9912	0.9733	0.9552	0.9735	0.9734	0.9754	0.9405	0.9444	0.9255
id8	0.8141	0.8138	0.7104	0.8018	0.7944	0.7713	0.7552	0.7381	0.7765	0.6559
id9	0.5867	0.5606	0.5731	0.5906	0.5178	0.5825	0.5659	0.5696	0.5630	0.4445
id10	0.7828	0.7769	0.7992	0.8061	0.8588	0.7941	0.7946	0.7795	0.7856	0.7468
id11	0.4607	0.4632	0.4325	0.4588	0.2862	0.4345	0.4367	0.4242	0.4429	0.4167
id12	0.8149	0.7947	0.8433	0.7992	0.9431	0.8283	0.8321	0.7647	0.7700	0.6872
id13	0.2947	0.2606	0.2320	0.2480	0.2019	0.1966	0.1964	0.2100	0.2182	0.1786
id14	0.3217	0.2584	0.2430	0.2670	0.1593	0.2247	0.1990	0.2116	0.2341	0.1670
id15	0.2985	0.2629	0.2587	0.2726	0.2081	0.2468	0.2500	0.2495	0.2533	0.1998
id16	0.4541	0.4185	0.4126	0.3857	0.2655	0.3635	0.3903	0.3138	0.3158	0.2690
id17	0.7196	0.6668	0.6027	0.5358	0.6084	0.5532	0.5535	0.4992	0.4660	0.3325
id18	0.6008	0.5833	0.5326	0.7484	0.6282	0.4621	0.4595	0.5768	0.5127	0.4863
id19	0.3433	0.3238	0.2558	0.2574	0.1188	0.2291	0.2267	0.2438	0.2478	0.1892
id20	0.9861	0.9909	0.7962	0.8428	0.8796	0.7931	0.8062	0.7962	0.7962	0.7773
id21	0.9657	0.9181	0.7390	0.6500	0.6300	0.6357	0.5962	0.6500	0.6500	0.5800
id22	0.9512	0.6593	0.4917	0.5335	0.5652	0.4387	0.4207	0.3987	0.4266	0.3476
id23	0.1370	0.1202	0.1304	0.1346	0.0777	0.1217	0.1253	0.1264	0.1265	0.1123
id24	0.5413	0.5536	0.4614	0.3823	0.3183	0.4322	0.4370	0.3712	0.3750	0.2486
id25	0.7220	0.6250	0.4525	0.4220	0.3483	0.4623	0.4577	0.3969	0.4091	0.3261
id26	0.2464	0.2378	0.2305	0.2497	0.1168	0.2407	0.2424	0.2239	0.2262	0.2040
id27	0.1258	0.1257	0.1345	0.1342	0.0676	0.1153	0.1104	0.1190	0.1379	0.0736
id28	0.4851	0.4594	0.4524	0.4741	0.6356	0.4733	0.4797	0.4372	0.4178	0.3928
id29	0.3812	0.3772	0.3749	0.3159	0.2634	0.2847	0.2962	0.2534	0.2599	0.3376
id30	0.2841	0.2543	0.2259	0.2586	0.2458	0.2234	0.2231	0.2359	0.2370	0.1961
Average	0.6158	0.5867	0.5412	0.5492	0.5232	0.5305	0.5282	0.5097	0.5125	0.4631
Win	19	4	0	3	4	0	0	0	1	0
Top-3	28	21	8	16	10	5	4	0	1	0

algorithms based on the ranks of the algorithms on multiple datasets. STAC [43], which is a web platform to compare algorithms using statistical tests, was used for the experiments. We assume that the performances of all ten methods for comparison are identical and set the P-value at 0.05. The experimental result shows that the hypothesis that all algorithms have identical performance is rejected with an extremely low P-value ($P < 0.00001$).

To further differentiate these algorithms, the Nemenyi post hoc test was also used after the hypothesis “the performance of the comparisons of the groups of data is similar” was rejected. The Nemenyi test computes an average ranking difference threshold CD , and the hypothesis will be rejected if the average ranking difference is larger than $CD/2$ [44], where CD is defined as

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}}, \tag{4}$$

where k is the number of algorithms, N is the number of datasets, and q_α is the critical range of Tukey’s distribution. We set $\alpha = 0.05$. Figure b shows the Nemenyi figure of the compared algorithms on the recall. From Figure b, IUE-II performs similarly to IUE-I and significantly outperforms other undersampling-based ensemble methods. IUE-I significantly outperforms DTE_SBD, RBT, EUS, and UC, and there is no significant difference between IUE-I and each of the other methods.

Table 9. The AUC of comparing methods.

ID	IUE-II	IUE-I	AEUB	DTE_SBD	RBT	BC	EE	EUS	UBG	UC
id1	0.9274	0.9247	0.9071	0.8995	0.9013	0.9140	0.9132	0.8950	0.8961	0.8801
id2	0.9700	0.9693	0.9552	0.9390	0.9332	0.9623	0.9624	0.9337	0.9401	0.9176
id3	0.8870	0.8782	0.8751	0.8849	0.8506	0.8704	0.8691	0.8769	0.8874	0.8753
id4	0.9783	0.9728	0.9314	0.9207	0.9238	0.9610	0.9599	0.9192	0.9286	0.9108
id5	0.9812	0.9844	0.9434	0.9481	0.9416	0.9643	0.9613	0.9381	0.9336	0.9247
id6	0.9011	0.8883	0.8934	0.8941	0.8718	0.9032	0.9019	0.8693	0.8735	0.8452
id7	0.9938	0.9937	0.9917	0.9845	0.9880	0.9912	0.9915	0.9823	0.9830	0.9797
id8	0.9319	0.9341	0.8856	0.9183	0.9062	0.9135	0.9044	0.8951	0.9129	0.8668
id9	0.8775	0.8672	0.8733	0.8753	0.7696	0.8841	0.8726	0.8805	0.8822	0.8081
id10	0.9606	0.9581	0.9609	0.9561	0.9444	0.9598	0.9588	0.9520	0.9559	0.9356
id11	0.8013	0.7926	0.7901	0.7954	0.6149	0.7900	0.7888	0.7867	0.7995	0.7696
id12	0.9769	0.9723	0.9735	0.9624	0.9742	0.9715	0.9711	0.9532	0.9541	0.9318
id13	0.7252	0.6725	0.6214	0.6327	0.5742	0.5820	0.5778	0.6007	0.6198	0.5431
id14	0.7761	0.7156	0.6618	0.6760	0.5492	0.6476	0.6123	0.6230	0.6626	0.5484
id15	0.7656	0.7343	0.7279	0.7236	0.5780	0.7077	0.7115	0.7107	0.7134	0.6378
id16	0.8944	0.8780	0.8067	0.8164	0.7057	0.8203	0.8273	0.7950	0.7960	0.7071
id17	0.9363	0.9202	0.9223	0.8928	0.8644	0.9138	0.9088	0.8849	0.8758	0.7900
id18	0.9572	0.9505	0.9414	0.9777	0.9482	0.9252	0.9248	0.9495	0.9374	0.9068
id19	0.8062	0.7850	0.7215	0.7125	0.5153	0.7088	0.7095	0.7254	0.7258	0.6292
id20	0.9991	0.9994	0.9787	0.9826	0.9723	0.9784	0.9749	0.9787	0.9787	0.9725
id21	0.9825	0.9642	0.9044	0.8167	0.8000	0.8159	0.8110	0.8167	0.8167	0.7667
id22	0.9877	0.9214	0.8882	0.9232	0.9309	0.9018	0.8982	0.8927	0.9055	0.8327
id23	0.6537	0.6105	0.5941	0.6326	0.4600	0.6212	0.6328	0.6316	0.6297	0.5903
id24	0.9646	0.9660	0.9505	0.9105	0.9000	0.9445	0.9451	0.9293	0.9304	0.8402
id25	0.9835	0.9735	0.9524	0.9463	0.9212	0.9541	0.9534	0.9404	0.9433	0.9123
id26	0.8327	0.8281	0.8289	0.8376	0.5924	0.8366	0.8386	0.8278	0.8308	0.8064
id27	0.6750	0.6807	0.6637	0.6664	0.5089	0.6716	0.6589	0.6527	0.6870	0.5620
id28	0.9669	0.9591	0.9627	0.9574	0.8987	0.9616	0.9625	0.9562	0.9568	0.9413
id29	0.8317	0.8225	0.7879	0.8098	0.6838	0.7843	0.8065	0.8101	0.8127	0.7460
id30	0.8829	0.8600	0.8734	0.8773	0.7746	0.8728	0.8706	0.8761	0.8745	0.8334
Average	0.8936	0.8792	0.8590	0.8590	0.7932	0.8578	0.8560	0.8495	0.8548	0.8071
Win	20	4	1	1	0	2	1	0	2	0
Top-3	29	20	9	12	2	8	6	2	6	0

Table 7 and Figure c report the summary results and average rank of the ten algorithms on g-mean, respectively. Similar to the results on recall, Table 7 shows that IUE-II (IUE-I) has the best g-mean on 19 (4) of the 30 datasets and performs in the top 3 on 28 (21) datasets. In addition, Figure c shows that the average ranks of IUE-II and IUE-I on most datasets are much smaller than those of the other methods, and IUE-II ranks first with the average rank of 1.58, followed by IUE-I with the average rank of 3.40, DTE_SBD with 4.75, BC with 4.87, AEUB with 5.05, EE with 5.37, UBG with 5.38, EUS with 6.73, RBT with 8.43, and UC with 9.43. In addition, combining the results of Table 6, we observe that the proposed methods show better performance on g-mean than on recall compared to the other methods, which indicates that the proposed methods are more accurate than other undersampling-based methods on the majority class.

The Friedman test shows that the hypothesis of all algorithms with identical performance on g-mean is rejected with an extremely low P-value ($P < 0.00001$). The Nemenyi post hoc test was used for further comparison. Figure d reports the Nemenyi figure of compared algorithms on the g-mean. From Figure d, both IUE-II and IUE-I significantly outperform the other methods, since the average ranking difference exceeds $CD/2 = 1.215$. In addition, in Figure d, FIUE-II significantly outperforms IUE-I on the g-mean.

Table 8 and Figure e report the summary results and average ranks of the compared methods on f1-measure, respectively. Table 8 shows that IUE-II and IUE-I have much larger average f1-measures than the other methods, and IUE-II(IUE-I) performs the best for f1-measure on 18 (4) of 30 datasets and performs in

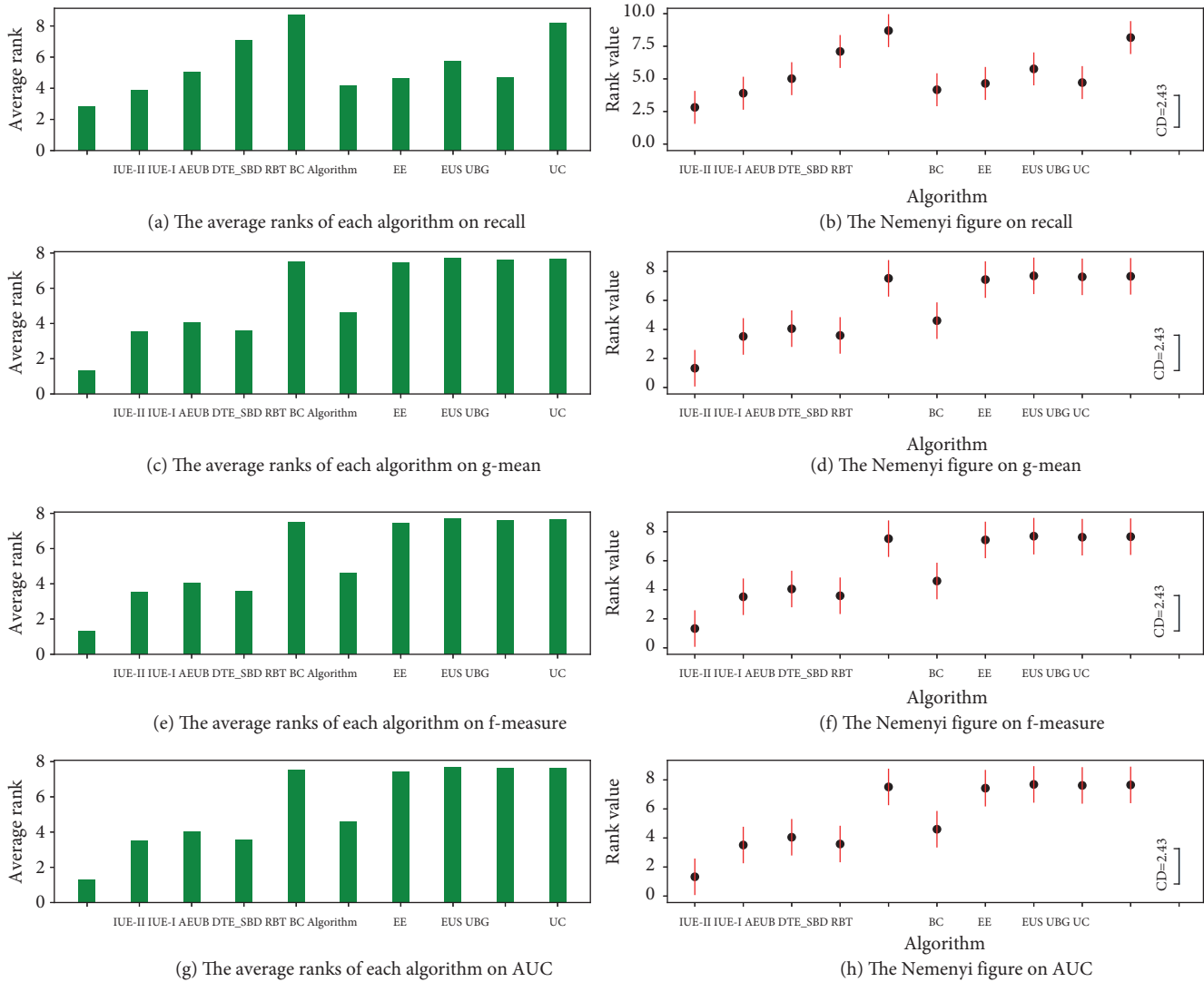


Figure. Average ranks and Nemenyi figure of the compared algorithms on measures of recall, g-mean, f1-measure, and AUC.

the top 3 on 28 (21) of the sets. From Figure e, the average ranks of the proposed methods are much smaller than those of other methods; IUE-II ranks first with the average rank of 1.80, followed by IUE-I with 3.43, DTE_SBD (3.90), AEUB (5.03), BC (5.80), EE (5.80), RBT (5.93), UBG (6.53), EUS (7.47), and UC (9.30). The Nemenyi test was used for further comparison of the algorithms on the g-mean, since the hypothesis of all algorithms with identical performance on the g-mean is rejected by the Friedman test. Figure f reports the Nemenyi figure of the f1-measure. The corresponding results show that, similar to the results in Figures b and d, IUE-II is comparable to IUE-I and significantly outperforms the other methods, since the average ranking difference exceeds $CD/2 = 1.215$. IUE-I is comparable to DTE_SBD and significantly outperforms the other methods.

Table 9, Figure g, and Figure h depict the AUC, ranks, and Nemenyi figures of ten methods in a comparison of the AUC. Table 9 shows that our proposed methods are superior for most of the 30 sets compared to other methods. Specifically, IUE-II (IUE-I) performs best on 22 (4) of 30 datasets and performs in the top 3 on

29 (20) of the sets. Table 9 and Figure g show that average recalls (ranks) of IUE-II, IUE-I, AEUB, DTE_SBD, RBT, BC, EE, EUS, UBG, and UC are 0.8724 (1.79), 0.8471 (3.48), 0.8413 (4.31), 0.8392 (4.55), 0.8088 (7.65), 0.8009 (8.26), 0.8434 (4.23), 0.8029 (8.24), 0.8282 (6.13), 0.7975 (8.61), and 0.8005 (8.53), respectively. The Friedman test shows that the hypothesis of all algorithms with identical performance on AUC is rejected with an extremely low P-value ($P < 0.00001$). Figure h reports the Nemenyi figures of the compared methods for AUC. In Figure h, IUE-II significantly outperforms IUE-I, and both IUE-II and IUE-I significantly outperform the other methods.

5. Conclusion

Imbalanced learning has received sufficient attention in machine learning and pattern recognition, and undersampling-based ensemble methods are notably effective in handling imbalanced problems. However, the training sets obtained by undersampling techniques to train individual classifiers are often notably small because of the limit of the minority class examples. Therefore, how to improve individual classifiers on limited datasets is notably meaningful. In this paper, we propose an improved undersampling-based ensemble (IUE) using rotation forest, since it is notably accurate, particularly for small datasets. In addition, it is easier to create individual classifiers with diversity using rotation forest, since rotation forest is more sensitive to the sampling technique than robust methods such as SVM and neural networks. The experimental results show that IUE significantly outperforms other undersampling-based ensemble methods for imbalanced datasets on the measure of recall, g-mean, f-measure, and AUC.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (No. 61802329), the Project of Science and Technology Department of Henan Province (No. 182102210132), the Innovation Team Support Plan of the University of Science and Technology of Henan Province (No. 19IRTSTHN014), and the Nanhu Scholars Program for Young Scholars of XYNU.

References

- [1] Guo H, Li Y, Shang J, Mingyun G, Yuanyue H, Bing G. Learning from class-imbalanced data: review of methods and applications. *Expert Systems with Applications* 2017; 73: 220-239.
- [2] He H, Garcia EA. Learning from imbalanced Data. *IEEE Transactions on Knowledge and Data Engineering* 2009; 21: 1263-1284.
- [3] He H, Ma Y (editors). *Imbalanced Learning: Foundations, Algorithms, and Applications*. New York, NY, USA: IEEE Press, 2013.
- [4] Martin PD. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *Journal of Machine Learning Technologies* 2011; 2(1): 37-63.
- [5] Liu XY, Wu J, Zhou ZH. Exploratory under-sampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics Part B* 2009; 39(2): 965-969.
- [6] Castellanos FJ, Valero-Mas JJ, Calvo-Zaragoza J, Rico-Juan JR. Oversampling imbalanced data in the string space. *Pattern Recognition Letters* 2018; 103: 32-38.
- [7] Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research* 2002; 16: 321-357.

- [8] Han H, Wang W, Mao B. Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning. In: Proceedings of the First International Conference on Intelligent Computing (Part I); 23–26 August 2005; Hefei, China. Berlin, Germany: Springer. pp. 878-887.
- [9] Bunkhumpornpat C, Sinapiromsaran K, Lursinsap C. Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. In: Proceedings of the 13th Pacific-Asia Conference on Knowledge Discovery and Data Mining(PAKDD); 27–30 April 2009; Bangkok, Thailand. Berlin, Germany: Springer. pp. 475-482.
- [10] Barua S, Islam MM, Yao X, Murase K. MWMOTE-Majority weighted minority oversampling technique for imbalanced data set learning. *IEEE Transactions on Knowledge and Data Engineering* 2014; 26(2): 405-425.
- [11] Kazemi Y, Mirroshandel SA. A novel method for predicting kidney stone type using ensemble learning. *Artificial Intelligence in Medicine* 2018; 84: 117-126.
- [12] Chan YT, Wang SJ, Tsai C. Real-time foreground detection approach based on adaptive ensemble learning with arbitrary algorithms for changing environments. *Information Fusion* 2018; 39: 154-167.
- [13] Han M, Liu B. Ensemble of extreme learning machine for remote sensing image classification. *Neurocomputing* 2015; 149: 65-70.
- [14] Tong H, Liu B, Wang S. Software defect prediction using stacked denoising autoencoders and two-stage ensemble learning. *Information and Software Technology* 2018; 96: 94-111.
- [15] Seiffert C, Khoshgoftaar T, Hulse JV, Napolitano A. Rusboost: a hybrid approach to alleviating class imbalance. *IEEE Transactions on Systems, Man, and Cybernetics Part A* 2010; 40(1): 185-197.
- [16] Lu W, Li Z, Chu J. Adaptive Ensemble Undersampling-Boost: A novel learning framework for imbalanced data. *Journal of Systems and Software* 2017; 132: 272-282.
- [17] Bao L, Juan C, Li J, Zhang Y. Boosted near-miss under-sampling on SVM ensembles for concept detection in large-scale imbalanced datasets. *Neurocomputing* 2016; 172: 198-206.
- [18] Barandela R, Valdovinos RM, Sánchez JS. New applications of ensembles of classifiers. *Pattern Analysis and Applications* 2003; 6(3): 245-256.
- [19] Liu XY, Wu J, Zhou ZH. Exploratory under-sampling for class-imbalance learning. In: Proceedings of the 6th IEEE International Conference on Data Mining (ICDM); 18–22 December 2006; Hong Kong, China. New York, NY, USA: IEEE. pp. 965-969.
- [20] Tao D, Tang X, Li X, Wu X. Asymmetric bagging and random subspace for support vector machines-based relevance feedback in image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 2006; 28(7): 1088-1099.
- [21] Hido S, Kashima H, Takahashi Y. Roughly balanced bagging for imbalanced data. *Statistical Analysis and Data Mining* 2009; 2(5-6): 412-426.
- [22] Rodríguez JJ, Kuncheva LI, Alonso CJ. Rotation forest: a new classifier ensemble method. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 2006; 28(10): 1619-1630.
- [23] Breiman L. Bagging predictors. *Machine Learning* 1996; 24(2): 123-140.
- [24] Freund Y, Schapire RE. A Decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences* 1997; 55(1): 119-139.
- [25] Breiman L. Random forests. *Machine Learning* 2001; 45(1): 5-32.
- [26] López V, Fernández A, García S, Palade V, Herrera F. An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Information Sciences* 2013; 250: 113-141.
- [27] Branco P, Torgo L, Ribeiro RP. A Survey of Predictive Modelling under Imbalanced Distributions. *CoRR* abs/1505.01658 2015.

- [28] Zhai J, Zhai M, Kang X. Condensed fuzzy nearest neighbor methods based on fuzzy rough set technique. *Intelligent Data Analysis* 2014; 18(3): 429-447.
- [29] Devi D, Biswas SK, Purkayastha B. Redundancy-driven modified Tomek-link based undersampling: a solution to class imbalance. *Pattern Recognition Letters* 2017; 93: 3-12.
- [30] Japkowicz N. The class imbalance problem: significance and strategies. In: *Proceedings of the International Conference on Artificial Intelligence*; 2000; Las Vegas, NV, USA. pp. 111-117.
- [31] Sun Z, Song Q, Zhu X, Sun H, Xu B, Zhou Y. A novel ensemble method for classifying imbalanced data. *Pattern Recognition* 2015; 48(5): 1623-1637.
- [32] Nguyen HM, Cooper EW, Kamei K. Borderline over-sampling for imbalanced data classification. *International Journal of Knowledge Engineering and Soft Data Paradigms* 2011; 3(1): 4-21.
- [33] Sun J, Lang J, Fujita H, Li H. Imbalanced enterprise credit evaluation with DTE-SBD: Decision tree ensemble based on SMOTE and bagging with differentiated sampling rates. *Information Sciences* 2018; 425: 76-91.
- [34] Estabrooks A, Jo T, Japkowicz N. A multiple resampling method for learning from imbalanced data sets. *Computational Intelligence* 2004; 20: 18-36.
- [35] Kang P, Cho S. EUS SVMs: Ensemble of under-sampled SVMs for data imbalance problems. In: *Proceedings of the 13th International Conference on Neural Information Processing, Part I*; 3-6 October 2006; Hong Kong, China. Berlin, Germany: Springer. pp. 837-846.
- [36] Alcalá-Fdez J, Fernández A, Luengo J, Derrac J, García S, Sánchez L, Herrera F. KEEL data-mining software tool: data set repository integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic and Soft Computing* 2011; 17(2-3): 255-287.
- [37] García-Pddrajas N, García-Osorio C, Fyfe C. Nonlinear boosting projections for ensemble construction. *Journal of Machine Learning Research* 2007; 8: 1-33.
- [38] Chawla NV. C4.5 and imbalanced data sets: investigating the effective of sampling method, probabilistic estimate, and decision tree structure. In: *Proceedings of the ICML '03 Workshop on Learning from Imbalanced Data Sets*; 21-24 August 2003; Washington, DC, USA. Palo Alto, CA, USA: AAAI Press.
- [39] Quinlan JR. *C4.5: Programs for Machine Learning*. San Mateo, CA, USA: Morgan Kaufmann, 1993.
- [40] Demsar J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research* 2006; 6: 1-30.
- [41] García S, Fernández A, Luengo J, Herrera F. Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power. *Information Sciences* 2010; 180: 2044-2064.
- [42] Li J. A two-step rejection procedure for testing multiple hypotheses. *Journal of Statistical Planning and Inference* 2008; 138: 1521-1527.
- [43] Rodríguez-Fdez I, Canosa A, Mucientes M, Bugarín A. STAC: A web platform for the comparison of algorithms using statistical tests. In: *Proceedings of the IEEE International Conference on Fuzzy Systems*; 2-5 August 2015; İstanbul, Turkey. New York, NY, USA: IEEE. pp. 1-8.
- [44] Ren F, Cao P, Li W, Zhao D, Zaiane O. Ensemble based adaptive over-sampling method for imbalanced data learning in computer aided detection of microaneurysm. *Computerized Medical Imaging and Graphics* 2017; 55: 54-67.