Nearest Neighbor with Double Neighborhoods Algorithm for Imbalanced Classification

Caiwen Wang, Youlong Yang

Abstract—**Classification of imbalanced data is a challenge in data mining and pattern recognition tasks. The over-advantage of the majority classes often lead to poor performance of traditional classifiers, when imbalanced data is processed. In this paper, we propose an algorithm called nearest neighbor with double neighborhoods algorithm (NNDN) to deal with binary-class imbalanced data classification. In classification step, a double neighborhoods scheme is presented based on data distribution to judge the sparsity of the main neighborhood, and a tendency weighting scheme is used to increase the sensitivity of the algorithm to minority instances. Finally, we compare our method with six well-known algorithms on forty benchmark data sets. The results show that the proposed algorithm is suitable for imbalanced data classification, and outperforms the re-sampling and cost-sensitive learning strategies with generality-oriented base learners in most data sets.**

Index Terms—**Nearest neighbor classification, Imbalanced data, Classification, Data distribution.**

I. Introduction

IMBALANCED data classification has become a hot topic in machine learning and pattern classification. It has been successfully applied in various real-life application domains, such as medical diagnosis [1], [2], credit card fraud detection [3], [4], rare event detection [5], user rating prediction [6] and spam email detection[7]. It seems more interesting and meaningful for users to research on the minority instances in these fields. For example, more attention is paid to the correct identification of fraudulent credit cards and the correct detection of spam emails, and so on.

Existing imbalanced learning strategies can be roughly grouped into data-level approaches and algorithm-level approaches. Data-level methods focus on adjusting the size of each class to balance the data set. As a commonly used technique, resampling methods resize the proportion of each class to balance the sample size of positive and negative class in data set. It includes over-sampling, under-sampling and hybrid sampling. Over-sampling methods [8], such as the synthetic minority over-sampling technique SMOTE [9], balance various sample sizes by synthesizing positive instances. (Hereafter minority class and positive class are used interchangeably, and majority class and negative class are similar.) However, SMOTE may lead to over generalization, variance [10] and increase amount of additional calculations. Under-sampling methods [11] balance a training set by selectively discarding part of negative instances. However, it may lose important information. The incomplete existing information may result in a distortion of decision boundary. Hybrid sampling methods [12] are combinations of over-sampling and under-sampling methods.

Algorithm-level methods are improved on the basis of existing classifiers to enhance their learning ability and sensitivity for positive instances, and mitigate the inherent bias of the original classifier towards negative instances. There are many existing classifiers can be used, such as Decision Tree [13], [14], K Nearest Neighbors, Bayesian Network [15], [16], Neural Networks [17], [18], [19] and Support Vector Machines [20], [21]. Cost-sensitive learning [22] is a commonly used algorithm-level technique. It increases the learning ability from positive instances by assigning larger misclassification cost to false negative errors. MetaCost [23] is widely used in cost-sensitive learning. However, the misclassification costs of most data sets are unknown or even impossible to measure with specific values. Ensemble learning [24], [25] is another algorithm-level technique. It increases the learning ability from positive instances by a comprehensive decision made by multiple classifiers. In general, the comprehensive model outperforms each base classifier. It may be affected by probability and uncertainty.

The size of each class varies greatly in an imbalanced data set. Imbalanced positive and negative information leads to minority instances not being correctly identified by traditional classifiers in classification process. Therefore, the skewed class distribution of imbalanced data creates difficulties for the classification task of the classifiers. Existence of positive sub-concepts also cause classification difficulties [26]. The single class is composed of various sub-concepts that contain various number of examples, which causes within-class imbalance [27]. Unfortunately, classification errors are often heavily concentrated toward the smaller disjuncts[28]. These positive instances contained in sub-concepts are surrounded by a large number of negative instances, resulting in extremely low sensitivity to positive instances.

As a local-oriented classifier, K nearest neighbor (KNN) makes decisions based on local information of the query instances. It stores entire training set and classifies according to the characteristics of local distribution of the query instances. Therefor, it is a good choice for dealing with classification of imbalanced data. However, a frequent type of error is to erroneously classify as a negative instance due to the positive sparse distribution of neighborhood in classification process of KNN. It leads to a lower classification precision of positive instances. There are many improved algorithms based on KNN are used for imbalanced data, such as K Rare-class Nearest Neighbor (KRNN) [29] and Positive-biased Nearest Neighbor (PNN) [30]. The two methods increase their learning ability from positive instances by directly adjusting the posterior probability of positive class. For example, the positive posterior probability is roughly adjusted to $[\mathrm{K}/2]/\mathrm{K}$

Manuscript received December 5, 2018; revised May 20, 2019. This work was supported by the National Natural Science Foundation of China (Grant No. 61573266).

Caiwen Wang is with the Department of Mathematics and Statistics, Xidian University, Xian, Shaanxi 710071, China e-mail:caiwen805@qq.com.

Youlong Yang is with the Department of Mathematics and Statistics, Xidian University, 266 Xinglong Section of Xifeng Road, Xian, Shaanxi 710126, China, e-mail: ylyang@mail.xidian.edu.cn.

in PNN algorithm, when there is positive tendency in the query neighborhood with fewer half of positive neighbors.

In this paper, we will provide more convincing judgment conditions and adjustment schemes. We propose a KNN-based method named NNDN for dealing with binary classification of imbalanced data sets. The main neighborhood and the auxiliary neighborhood are dynamically formed for the query instance based on data distribution. They are used to determine if the query neighborhood has positive sparsity and if the query instance has a positive tendency. The neighbors are given different tendency weights depending on whether the two conditions are met after being weighted by the distance. Finally, a classification decision is made by the weighted posterior probability.

The main contributions of this paper can be summarized as follows:

• We propose a double neighborhoods scheme in NNDN that can effectively judge the sparsity of the nearest neighbors of the query neighborhood.

• The tendency weighting scheme can achieve better classification performance for imbalanced data.

• To demonstrate the usefulness and applicability of NNDN, we compare it with six other methods using 40 benchmark data sets. Experiments show that NNDN improves overall classification performance of KNN, and outperforms KNN family algorithms. At the same time, NNDN performs better than algorithms that SMOTE and MetaCost combined with C4.5 algorithm respectively.

The rest of the paper is organized as follows. A brief introduction of related works is reviewed in Section II. In Section III, the formation of the double neighborhoods is described in detail. The NNDN algorithm is provided in Section IV. Experimental results and analysis of NNDN and comparison with other methods are presented in Section V. In Section VI, conclusions are drawn and some suggestions of the future work are given.

II. Related work

KNN is a simple and easy-to-understand algorithm. Given a K value of the number of neighbors, the nearest K instances of the query instance t are found in training set as its neighbors. Finally, a majority voting mechanism is used to obtain the classification decision. It can be expressed as

$$L(t)=\arg\max_c \sum_{i=1}^{K} \mathrm{I}(L(x_i)=c) \quad (c=0,1), \tag{1}$$

where I(A) is an indicator function. When the condition A is true, it has a value of 1 and 0 otherwise. Label of the ith instance x_i is represented as $L(x_i)$. Value of c is 0 or 1 (where 0 is the negative class and 1 is the positive class).

The neighbors are often weighted by the reciprocal of the distance squared, when K neighbors vary widely in distances and closer neighbors are more reliable. The distance weight of x_i can be calculated as $w_i=\frac{1}{d(x_i,t)^2}$, where $d(x_i,t)$ is the Euclidean distance [31] of x_i to t. Neighbors that are very close to t may gain very large weights, which may cause an overwhelming superiority on them. This is not conducive to finding the real label of t, so a more gentle weighting technique is used to alleviate this problem.

The Gaussian function with a height of 1, a width of 1 and an offset of 0 is used to distance weight in the NNDN algorithm. The distance weight of x_i can be calculated as

$$w_i^1=e^{-\frac{d(x_i,t)^2}{2}}. \tag{2}$$

Finally, $L(t)$ is predicted by a weighted voting mechanism. The mechanism can be expressed as

$$L(t)=\arg\max_c \sum_{i=1}^{K} \mathrm{I}(L(x_i)=c)\cdot w_i. \tag{3}$$

The strategy limits the distance weight to between 0 and 1. The smaller the distance is, the greater weight will be given. However, there may be an extreme decision when instances in a class are very rare or non-existent. To avoid this situation, the Laplace Smoothing approach [32] is used in the decision-making scheme. $L(t)$ is given by

$$L(t)=\arg\ \max_c \frac{\sum\limits_{i=1}^{K} \mathrm{I}(L(x_i)=c)\cdot w_i+\frac{1}{|T|}}{\sum\limits_{i=1}^{k} \cdot w_i+\frac{2}{|T|}}, \tag{4}$$

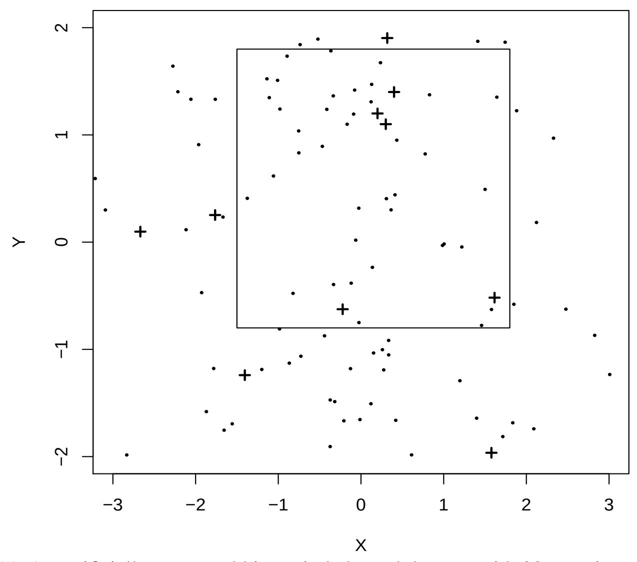
where $|T|$ is the size of training set.

The distance weighting mechanism makes full use of information of neighbors. But they are not strategies specifically for dealing with imbalanced data. This is partly because of the weighting technique is more bias towards the near-distance class of t, and treats positive instances and negative instances fairness. Besides, the distance weighting method alone is not enough to deal with the complex distribution of imbalanced data when positive neighbors are close to negative neighbors. In other words, the distance weighting does not change an inherent bias of KNN algorithm toward the negative class. So another mechanism needs to be proposed for dealing with imbalanced data.

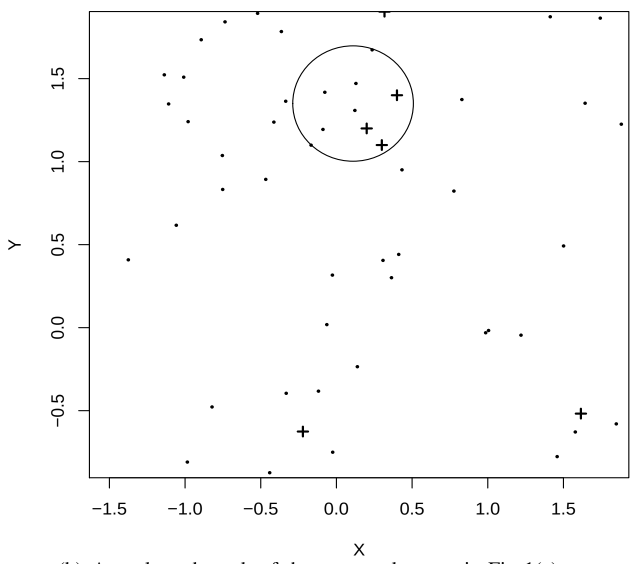
The common mistake is to classify a positive instance into negative class due to the positive sparsity in KNN algorithm. Therefore, a mechanism about positive propensity is proposed for dealing with imbalanced data in our method. When t tends to belong to positive class but positive neighbors are sparse, t is easily misjudged. At this time, if weights are added to these easily misjudged instances, false negative errors will be avoided. Thus improving the learning ability of the algorithm from positive instances. So the two problems need to be solved as follows:

• which query neighborhood is called a positive sparse neighborhood? Positive sparseness occurs when the size of positive neighbors of t is less than half of neighborhood's [30]. However, it is rough to judge positive sparseness only based on the frequency of the positive neighbors. We believe that the positive sparsity of a neighborhood is relative. Fig 1, as an example, illustrates this point well.

Fig 1(a) is an artificially generated binary imbalanced data set with 90 negative instances (denoted by "·") and 10 positive instances (denoted by "+"). Fig 1(b) enlarges the rectangular area in Fig 1(a). As can be seen from Fig 1(b), although there are less than half of positive neighbors in the circular neighborhood, positive neighbors in the neighborhood are relatively dense compared to distribution of positive instances outside of the neighborhood. This useful information cannot be ignored.



(a) An artificially generated binary imbalanced data set with 90 negative instances (denoted by "·") and 10 positive instances (denoted by "+")



(b) An enlarged result of the rectangular area in Fig 1(a)

Fig. 1. An example of positive sparse neighborhood

• How to determine whether a query instance has a tendency to belong to positive class? Intuitively, the more pronounced positive tendency the query instance has, the more likely it is to belong to positive class.

In view of the problems mentioned above, we propose the idea of the double neighborhoods. Considering that the positive sparsity of a neighborhood is relative, it seems more reasonable to judge positive sparsity of a neighborhood according to the positive frequency of a larger neighborhood than only to follow the positive and negative observation frequency. For the propensity of a neighborhood, we calculate it based on Pessimistic Estimate [33]. If the positive neighbors are sparse of the query instance with positive tendency, we will give greater weight to them. It will make the decision bias towards the positive class. The method can also be extended to multiple imbalanced classification problems by dividing multi-class problems into several easier-to-solve two-class sub-problems based on the well-known one-vs-one[34] or one-vs-all[35] decomposition strategies. Here we only study binary-class imbalanced data classification.

III. Construction of the double neighborhoods

In this section, we will detail the double neighborhoods mechanism. This strategy can well capture the distribution of positive instances near t, thereby increasing the sensitivity and learning ability of KNN from positive instances.

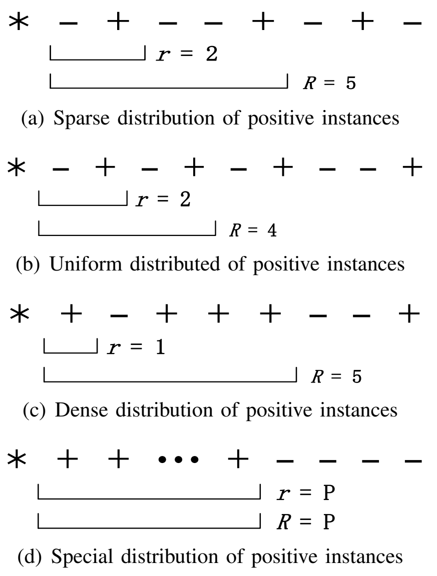
In standard KNN algorithm, $L(t)$ is determined according to whether the positive frequency is more than half of all neighbors. However, the frequency cannot fully represent the sparsity degree of positive instances near t. If t has positive tendency, a larger neighborhood which can capture the distribution of positive instances well needs to be created to determine the positive sparsity in the neighborhood. A neighborhood that is too large has no research significance.

In NNDN algorithm, two neighborhoods are dynamically constructed (one is called the main neighborhood and the other is called the auxiliary neighborhood). The two neighborhoods jointly capture the distribution information of positive neighbors around t. The main neighborhood is the key research object, and the auxiliary neighborhood is used to assist decision making. The first "positive-negative" dividing line serves as the boundary of the main neighborhood, and the second "positive-negative" dividing line serves as the boundary of the auxiliary neighborhood. The specific construction named Construct the Double Neighborhoods Algorithm (CDN) is given by Algorithm 1, where B_1 and B_2 are the first and second elements of B, respectively.

Algorithm 1 CDN

```
Input: Training set $T$; The query instance $t$.
Output: The main neighborhood $r(t)$ and the auxiliary neighborhood $R(t)$ of $t$.
 1: Initialize a collection of neighborhood boundaries $B$ into an empty set.
 2: for each $x_i \in T$ do
 3:     Calculate the Euclidean distance $d(x_i, t)$.
 4: end for
 5: The distance values are sorted in ascending order, and the sorted set is recorded as $T'$.
 6: for each $x_i \in T'$ do
 7:     if $L(x_i) = 1$, $L(x_{i+1}) = 0$ and $|B| < 2$ then
 8:         Add $x_i$ to $B$.
 9:     end if
10: end for
11: if $|B| = 2$ then
12:     Return $r(t) = \{x_j \in T' | d(x_j, t) <= d(B_1, t)\}$.
13:     Return $R(t) = \{x_j \in T' | d(x_j, t) <= d(B_2, t)\}$.
14: else
15:     Return $r(t) = R(t) = \{x_j \in T' | d(x_j, t) <= d(B_1, t)\}$.
16: end if
```

We analyze the time complexity of the CDN algorithm first. From the procedures of Algorithm 1, the complexity of the CDN algorithm proposed above depends mainly on step 5. The time complexity of sorting all distances generated by training set with size $|T|$ is $O(|T| \cdot \log |T|)$. In a sorted training set, two neighborhoods are found just by going through them once. The time complexity of this process is $O(|T|)$. In general, the time complexity of the CDN algorithm is $O(|T| \cdot \log |T|)$.



(a) Sparse distribution of positive instances

(b) Uniform distributed of positive instances

(c) Dense distribution of positive instances

(d) Special distribution of positive instances

Fig. 2. Examples of the main neighborhood and the auxiliary neighborhood under different distributions for the query instance t (denoted by "$*$"). Where, "$+$" represents a positive instance and "$-$" represents a negative instance in training set. R is the size of the auxiliary neighborhood, r the size of the main neighborhood and P is the number of positive instances in training set.

Although different locations can be used, we have the second position as the boundary of the auxiliary neighborhood. Later in this paper (see Section V-C), we calculate the classification performance with different locations on benchmark data sets. The result shows that the location did not significantly affect the classification performance of the NNDN algorithm. Unless all positive instances are closer to t than all negative instances, we can construct an auxiliary neighborhood. If this misfortune has already occurred, the boundary of the main neighborhood is used by the auxiliary neighborhood (as shown in step 15). Therefore, the main neighborhood is contained within the auxiliary neighborhood or the same as the auxiliary neighborhood. If positive instances centered on t are sparse, the size of the auxiliary neighborhood R may be large. In this case, many negative instances are likely to exist in the auxiliary neighborhood. On the contrary, if positive instances are dense, r and R may be small and R is even close to r. More positive instances are added during the formation of the auxiliary neighborhood. The strategy dynamically constructs neighborhoods based on data distribution and lays the foundation for more accurate classification.

Fig 2 shows four examples of the two neighborhoods for the query instance t based on some artificial data. Instances centered on t (denoted by "$*$") are arranged in ascending order of distance, where, "$+$" represents a positive instance and "$-$" represents a negative instance in training set to facilitate the description. The sizes of two neighborhoods vary with the distribution of positive instances.

Fig 2(a) shows the case when the positive instances near t are sparsely distributed. At this point, the main neighborhood contains two instances ($r = 2$), and one of which is a positive instance. There are five instances ($R = 5$) in the auxiliary neighborhood that contains two positive instances. Two negative instances are added during the formation of the auxiliary neighborhood. Positive instances in the auxiliary neighborhood are more sparse than that in the main neighborhood.

Fig 2(b) shows the case when the positive instances near t are relatively uniform distributed. At this point, the main neighborhood contains two instances ($r = 2$), and one of which is a positive instance. There are four instances ($R = 4$) in the auxiliary neighborhood that contains two positive instances. Positive and negative instances are added equally during the formation of the auxiliary neighborhood, and the sparse degree of positive instances in the two neighborhoods is identical.

Fig 2(c) shows the case when the positive instances near t are densely distributed. At this point, the main neighborhood contains one instance ($r = 1$). There are five instances ($R = 5$) in the auxiliary neighborhood that contains four positive instances. Three positive instances are added during the formation of the auxiliary neighborhood. Positive instances in the auxiliary neighborhood are denser than that in the main neighborhood.

Fig 2(d) represents an extreme case where all positives are closer to t than all negatives. At this point, instances in the two neighborhood are positive instances. The size of them are equal to the sample size of the positive instance P in the training set.

Subsequent experimental results indicate that the strategy is effective.

IV. The NNDN algorithm

In this section, we introduce the definitions of positive tendency, positive sparsity and the calculation method of tendency weight first. These theories make the NNDN algorithm have a stronger ability to learn from positive instances. They are the cornerstone for the algorithm to more accurately classify imbalanced data. Then the detailed pseudo code of the proposed algorithm is shown and analyze about its time complexity are given.

A. Positive tendency of the query instance

A positive instance-intensive neighborhood of t is likely to belong to a positive sub-concept. That is the higher positive frequency is, the greater the probability that the region belongs to a positive sub-concept, and the higher positive tendency of t. Obviously, it is inaccurate to quantify the sparse degree of positive instances simply by using the observed positive frequency in the region.

Pessimistic Estimates are used to estimate the error rate at internal nodes as well as at leaf nodes of a decision tree [33]. It also is used to estimate the false positive error rate for a region [29]. Inspired by them, the true positive frequency in a neighborhood can be estimated by Pessimistic Estimates. And then the sparse degree of positive instances in the neighborhood is measured by the true positive frequency. In a neighborhood, each neighbor is either a positive instance or a negative instance. Then if $L(t)$ is treated as a random variable, it is subject to the binomial distribution $B(n, q)$, where n is the size of the neighborhood, and q is the actual positive probability of it. If the observed frequency of positive instances in the region is assumed to be $\tilde{q}$, the true positive frequency q can be estimated as

$$q \approx \frac{\tilde{q} + \frac{z^2}{2n} + z\sqrt{\frac{\tilde{q}(1-\tilde{q})}{n} + \frac{z^2}{4n^2}}}{1 + \frac{z^2}{n}}, \tag{5}$$

where z is the z-score corresponding to a given confidence level c, which $z = 1.28$ for c=90%. The greater the confidence level c is, the closer the q is to $\tilde{q}$. The larger q is, the more likely the neighborhood belongs to a positive sub-concept and t is to be a positive instance. Therefore, q can be regarded as the degree of positive tendency of t when using the actual positive frequency in the training set as a criterion.

Definition 1. When the degree of the positive tendency q_r in the neighborhood of t is greater than the degree q_T of the training set in which it is located, t has a tendency to belong to positive class. At the same time,

$$q_r > q_T \tag{6}$$

is called the tendency condition, where q_r and q_T can be estimated by Eq. 4, respectively.

B. Positive sparsity of the main neighborhood

Instances with positive tendency are often surrounded by negative instances, which makes them susceptible to misclassification in an imbalanced data set. If t has a positive tendency, it is necessary to further judge the positive sparsity of its neighborhood. Predicting the label $L(t)$ of t simply based on whether the positive frequency is greater than the threshold of 0.5 in a neighborhood where positive instances are very rare is very rough and can easily generate errors in the classification. A dynamic description method is adopted to dynamically describe it by the double neighborhoods scheme (given in Section III).

Definition 2. After constructing the double neighborhoods, if the positive frequency in the main neighborhood of t is not greater than that in the auxiliary neighborhood, the main neighborhood is a positive sparse neighborhood. At the same time,

$$\frac{p}{r} \leq \frac{P}{R} \tag{7}$$

is called the sparsity condition, where p and P are the number of positive neighbors in them, respectively.

If frequencies of positive instances in the main neighborhoods of two query instances are identical but that is different in the auxiliary neighborhoods, the positive sparse degree of the main neighborhoods is different. Later expements proves that the proposed method is effective.

C. Calculation of the tendency weight

More attention should be paid to the false negative error in the positive sparse neighborhood. When t has positive tendency and its main neighborhood is a positive sparse neighborhood, neighbors in the main neighborhood are weighted by tendency. The weight is known as tendency weight. Thus, the sensitivity of the algorithm to positive instances is improved. In the NNDN algorithm, if t satisfies both the tendency condition and the sparsity condition, a tendency weight is given to each neighbors in the main neighborhood. It can be calculated as

$$w_i^2 = \mathrm{I}(L(x_i) = c) \cdot \frac{q_r / q_T}{\frac{P}{R} / \frac{p}{r}} + 1. \tag{8}$$

Therefore, the decision of the NNDN algorithm is made by

$$L(t) = \arg\max_c \frac{\sum_{i=1}^{r} \mathrm{I}(L(x_i) = c) \cdot {w_i}^1 \cdot {w_i}^2 + \frac{1}{|T|}}{\sum_{i=1}^{r} \cdot {w_i}^1 \cdot {w_i}^2 + \frac{2}{|T|}}, \tag{9}$$

where w_i^1 and w_i^2 are the distance weight and the tendency weight of the ith neighbor in the main neighborhood, respectively. The Laplace Smoothing approach is used in the decision, so the denominator is added $2/|T|$ and the numerator is added $1/|T|$.

D. The pseudo code of the NNDN *algorithm*

Each step of the NNDN algorithm is described in Algorithm 2. All the strategies used in the algorithm have been discussed previously. Input training set T, the query instance t, the confidence level c_m of the main neighborhood and the confidence level c_T of training set. The NNDN algorithm will output the label of the query instance t. Fig 3 shows the flowchart of the NNDN algorithm.

Algorithm 2 NNDN

```
Input: Training set $T$; The query instance $t$; The confidence
    level $c_m$ of the main neighborhood and the confidence
    level $c_T$ of training set.
Output: The label $L(t)$ of $t$.
 1: Construct the main neighborhood $r(t)$ of $t$.
 2: for each $x_i \in r(t)$ do
 3:     $x_i$ is weighted by $w_i^1$ computed by Eq. 2.
 4: end for
 5: Estimated the real positive frequency $q_r$ in $r(t)$ by Eq. 5.
 6: Estimated the real positive frequency $q_T$ in $T$ by Eq. 5.
 7: if $q_r < q_T$ then
 8:     $L(t) = \arg\max_c \frac{\sum_{i=1}^{r} I(L(x_i)=c) \cdot {w_i}^1 + \frac{1}{|T|}}{\sum_{i=1}^{r} w_i^1 + \frac{2}{|T|}}$.
 9: else
10:     Construct the auxiliary neighborhood $R(t)$ of $t$.
11:     if $\frac{p}{r} \leq \frac{P}{R}$ then
12:         The tendency weights $w_i^2$ are computed by Eq. 8.
13:     else
14:         The tendency weights $w_i^2 = 1$.
15:     end if
16:     $L(t)$ is given by Eq. 9.
17: end if
18: Return $L(t)$.
```

Finally, we focus on analyzing the time complexity of the NNDN algorithm. The complexity of NNDN depends mainly on the construction of the two neighborhoods. The time complexity of this process is $O(|T| \cdot \log|T|)$. If the size of test set is m, the final complexity of NNDN algorithm is $O(m \cdot |T| \cdot \log|T|)$. It is equal to that of KNN. However, it should be noted that some extra time need to be used to select the appropriate K value by cross validation method before using KNN.

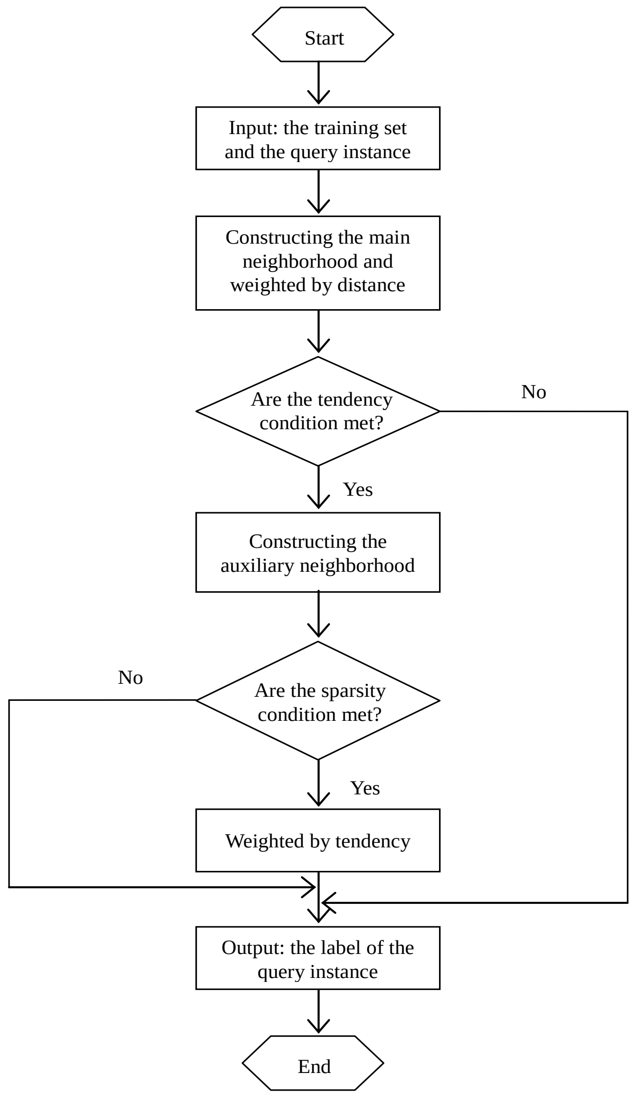

Start
Input: the training set and the query instance
Constructing the main neighborhood and weighted by distance
Are the tendency condition met?
No
Yes
Constructing the auxiliary neighborhood
Are the sparsity condition met?
No
Yes
Weighted by tendency
Output: the label of the query instance
End

Fig. 3. Flowchart of the NNDN algorithm

V. Experimental results and analysis

This section verifies the classification performance of the NNDN algorithm on 40 imbalanced data sets. All algorithms are developed using R language. We have conducted five-fold cross validation and the mean is recorded.

A. Algorithm settings, evaluation metrics and data sets

The NNDN algorithm is validated by comparison with six existing algorithms, namely KRNN, PNN, over-sampling strategy SMOTE and cost-sensitive learning strategy MetaCost using KNN and J48 (the C4.5 decision tree in R language) as the base learners. Parameters of these algorithms ensure that algorithms for comparison have good performance on most data sets.

The parameters of these algorithms are set as follows:

• For KRNN, $K = 1$ and $c_g = c_r = 0.9$. For others in the KNN family, $K = 3$. For NNDN $c_m = c_T = 0.9$.

• J48 is set with the "-M" option (minimum one instance is allowed for a leaf node without pruning).

• Setting for SMOTE and MetaCost imbalanced learning strategies: SMOTE minority over-sampling of 3 times more instances for the minority class. Misclassification cost of MetaCost for the positive class is set to the negative-to-positive ratio in training population.

Accuracy is the most commonly used evaluation metric. However, in the framework of imbalanced learning, accuracy may not be a good choice because it often has a bias toward the majority class [36]. To measure the performance of different algorithms, we have employed three evaluation metrics commonly used in imbalanced data classification, i.e. the Area Under ROC (AUC), Precision (Pre) and F_1-Measure (F_1) [39]. These are comprehensive metrics that measure the overall classification performance of classifiers in imbalanced data classification .

Receiver Operating Characteristic (ROC) curve is used to check the trade-off between finding true positives and avoiding false positives. AUC has been widely used to evaluate the performance of classifiers. It is an objective measure which is not affected by subjective factors on account of its independence from the decision criterion selected and prior probabilities of class distributions and it is used to quantify the trade-off [37], [38].

The Pre is a useful measure that evaluates the detection performance. It can be calculated as

$$\mathrm{Pre} = \frac{TP}{TP+FP}. \tag{10}$$

F_1 is the harmonic mean of Precision and Recall. The F_1 is calculated as

$$\mathrm{F}_1 = 2 \cdot \frac{\mathrm{Precision} \cdot \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}}, \tag{11}$$

where Recall can be calculated as

$$\mathrm{Recall} = \frac{TP}{TP+FN}, \tag{12}$$

where TP, FP, TN and FN are true positives, false positives, true negatives and false negatives, respectively. Apparently, the greater the values of the three metrics is, the better the performance of the algorithm.

To validate the proposed NNDN, we perform experiments on forty commonly used imbalanced benchmark data sets. The detailed characteristics of these data sets are summarized in Table I. Some of them (Clean1, German, Wine, SPE, Yeast, Glass) can be downloaded from UCI machine learning repository [41], and the rest can be downloaded from the KEEL-dataset repository [40]. These publicly available data sets vary in size, attributes and IR to guarantee the reliability of performance measurements. Data sets range from lowly imbalanced (with an imbalance-ratio of 1.82) to highly imbalanced (with an imbalance-ratio of 85.88). With multi-class data sets, one class is chosen as positive, and the others are the negative class. Besides, the abbreviations are shown in the second column, and they will be used in the rest of this paper.

B. The overall performance of the NNDN algorithm

AUC values obtained from NNDN and other six comparison methods on the 40 imbalanced data sets are given in Table II. Pre values and F_1 values are given in Table III and Table IV, respectively. We have conducted five-fold cross validation and the mean is recorded. The best values are bolded.

TABLE I
CHARACTERISTICS OF IMBALANCED DATA SETS USED IN EXPERIMENTAL RESULT

Dataset(Pos,Neg)	Abbr	Size	Min	Attr	IR	Dataset(Pos,Neg)	Abbr	Size	Min	Attr	IR
Clean1	–	476	207	166	1.30	Led7digit02456789vs1	L7891	443	37	7	10.97
Glass1	–	214	76	9	1.82	Glass2	–	214	17	9	11.59
Wisconsin	Wis	683	239	9	1.86	Ecoli0146vs5	E1465	280	20	6	13.00
Pima	–	768	268	8	1.87	Yeast1vs7	Y17	459	30	7	14.30
German(2,1)	–	1000	300	20	2.33	Glass(5,other)	–	214	13	9	15.46
Yeast1	–	1484	429	8	2.46	Pageblocks13vs4	P134	472	28	10	15.86
Wine(3,other)	–	178	48	13	2.71	Thyroid(2,other)	T2	720	37	21	18.46
Vehicle0	V0	846	199	18	3.25	Yeast1458vs7	Y4587	693	30	8	22.10
SPECT_F(0,1)	SPE	267	55	44	3.85	Yeast2vs8	Y28	482	20	8	23.10
Yeast(MIT,other)	–	1484	244	8	5.00	Yeast4	–	1484	51	8	28.10
Newthyroid2	N2	215	35	5	5.14	Winequalityred4	W4	1599	53	11	29.17
Glass6	–	214	29	9	6.38	Yeast1289vs7	Y2897	947	30	8	30.57
Paw02a-800-7-60-BI	PBI	800	100	2	7.00	Yeast5	–	1484	44	8	32.73
04clover5z-800-7-50-BI	CBI	800	100	2	7.00	Winequalityred8vs6	W86	656	18	11	35.44
Yeast3	–	1484	163	8	8.10	Yeast6	–	1484	35	8	41.40
Ecoli3	–	336	35	7	8.60	Winequalityred8s67	W867	855	18	11	46.50
Yeast02579vs368	Y9368	1004	99	8	9.14	Winequalitywhite39vs5	W395	1482	25	11	58.28
Ecoli0347vs56	E4756	257	25	7	9.28	Poker89vs6	P896	1485	25	10	58.40
Yeast05679vs4	Y6794	528	51	8	9.35	Poker89vs5	P895	2075	25	10	82.00
Ecoli0147vs2356	E2356	336	29	7	10.59	Poker8vs6	P86	1477	17	10	85.88

TABLE II
AUC RESULTS OBTAINED FROM NNDN AND OTHER SIX COMPARISON METHODS FOR THE 40 IMBALANCED DATA SETS, AND THE BEST RESULT IS IN **BOLD** FACE

Datasets	KRNN	PNN	SMOTE		MetaCost		NNDN	Datasets	KRNN	PNN	SMOTE		MetaCost		NNDN
			KNN	J48	KNN	J48					KNN	J48	KNN	J48	
Clean1	95.10	91.07	85.74	77.71	66.71	66.88	**96.29**	L7891	94.37	94.79	90.03	88.02	89.63	86.93	**94.84**
Glass1	86.24	84.82	80.51	71.03	70.95	65.04	**87.10**	Glass2	72.75	69.63	66.75	64.99	62.82	67.25	**76.50**
Wis	**99.27**	98.49	97.19	94.53	97.02	95.12	99.26	E1465	97.90	96.98	92.55	84.93	93.73	82.84	**98.11**
Pima	76.74	**77.92**	72.29	68.14	74.06	72.63	76.78	Y17	**79.00**	75.92	74.00	64.55	70.88	64.28	78.75
German	69.92	**71.99**	66.09	62.73	68.08	65.26	70.18	Glass5	96.80	93.11	93.70	85.11	84.91	74.87	**97.34**
Yeast1	75.70	**76.45**	72.45	68.40	72.89	69.66	76.11	P134	99.88	99.57	99.89	99.64	93.33	91.11	**99.92**
Wine	99.88	99.12	98.85	95.52	98.24	93.97	**99.90**	T2	65.39	61.89	59.63	**98.16**	58.81	95.46	65.74
V0	98.22	97.55	95.90	91.69	91.45	91.43	**98.30**	Y4587	73.87	66.74	66.63	55.35	65.25	59.12	**73.95**
SPE	73.91	74.76	74.31	63.14	69.61	66.01	**76.45**	Y28	**88.11**	83.24	82.12	76.00	79.51	63.71	87.47
Yeast	83.36	83.46	79.65	73.24	81.70	76.38	**83.77**	Yeast4	**91.22**	88.84	78.55	65.35	84.90	81.14	91.11
N2	**99.63**	99.27	99.09	92.94	98.37	88.55	**99.63**	W4	67.20	**70.94**	59.51	60.04	64.86	63.05	68.28
Glass6	95.85	94.65	91.60	87.83	85.59	88.84	**95.99**	Y2897	71.87	71.15	64.08	60.78	64.65	64.90	**73.40**
PBI	90.25	89.92	82.31	79.58	83.25	70.09	**90.61**	Yeast5	**99.29**	97.87	94.86	90.46	97.00	92.86	99.13
CBI	88.43	87.53	81.71	79.35	80.56	59.75	**88.62**	W86	82.48	78.88	67.14	63.52	76.09	63.37	**84.00**
Yeast3	95.96	96.17	91.51	87.33	93.43	91.82	**96.26**	Yeast6	**93.17**	90.09	86.08	75.62	88.20	86.90	92.89
ecoli3	93.12	92.98	88.28	84.31	89.29	77.20	**93.56**	W867	72.58	72.70	54.33	63.77	66.60	62.27	**75.19**
Y9368	94.30	**94.39**	91.79	85.8	91.49	84.75	94.36	W395	65.70	**68.78**	60.79	58.18	59.25	62.28	65.31
E4756	96.16	95.67	90.69	88.68	88.00	79.34	**96.68**	P896	98.40	94.08	96.29	72.36	88.81	61.88	**98.60**
Y6794	87.90	86.67	80.73	73.13	83.70	75.67	**88.16**	P895	73.19	68.35	55.97	55.94	65.69	58.51	**75.68**
E2356	94.51	**95.55**	90.67	85.07	89.29	76.75	94.62	P86	96.98	80.55	94.72	64.78	84.63	65.34	**98.25**
								Average	86.86	85.31	81.22	76.44	80.33	75.08	**87.43**

The proposed algorithm achieves outstanding overall performance compared with the other methods on most imbalanced data sets. If we take a closer look at the average values of the performance, the NNDN algorithm obtains the highest average AUC of 87.43, the highest averages Pre of 50.51 and F_1 of 54.93 than that of other methods. The NNDN algorithm achieves the best AUC score (including tie for first) in 26, achieves the best Pre score in 23 and the best Pre score in 22 out of the 40 imbalanced data sets. And the results are close to the best performances in the rest of cases. Furthermore, as the IR goes on, the NNDN algorithm still shows its advantage. The above results show that our strategy is very effective for imbalanced data.

The performance results of each algorithm for all imbalanced data sets are depicted in Fig 4 so that we can more directly demonstrate the performance advantages of the NNDN algorithm. From the results in the three figures, the boxplots of all other algorithms are visibly lower than the boxplot of the NNDN algorithm. Besides, if we look more closely at these boxplots, the scale of the box representing the NNDN algorithm is almost always the smallest in a same figure. Based on these two aspects, it can be concluded that our proposal is not only optimal in overall performance, but also the most robust. One critical fact that makes the NNDN suitable for the imbalanced dataset is that the training data distribution is adequately learned by the NNDN algorithm. The local instance distribution features of the query instance is captured by the double neighborhoods scheme.

TABLE III
PRE RESULTS OBTAINED FROM NNDN AND OTHER SIX COMPARISON METHODS FOR THE 40 IMBALANCED DATA SETS, AND THE BEST RESULT IS IN **BOLD** FACE

Datasets	KRNN	PNN	SMOTE		MetaCost		NNDN	Datasets	KRNN	PNN	SMOTE		MetaCost		NNDN
			KNN	J48	KNN	J48					KNN	J48	KNN	J48	
Clean1	63.09	68.19	69.48	**71.28**	57.36	58.86	66.67	L7891	49.89	51.80	57.32	**65.85**	54.76	32.92	60.52
Glass1	58.69	**67.39**	63.98	56.27	57.76	50.48	63.41	Glass2	18.45	12.36	22.85	**30.36**	9.86	6.66	25.40
Wis	95.87	95.72	95.69	93.67	96.28	94.54	**96.37**	E1465	62.91	49.98	**82.67**	58.74	45.67	21.69	81.81
Pima	52.63	**58.23**	51.28	50.98	56.17	56.81	55.62	Y17	23.68	16.30	21.37	24.55	13.75	14.43	**31.79**
German	41.95	**48.83**	42.01	44.08	40.39	42.21	44.90	Glass5	44.98	22.66	52.50	48.86	29.25	11.85	**52.89**
Yeast1	45.22	**51.36**	44.56	48.09	43.79	47.50	48.14	P134	70.53	40.75	**94.80**	92.56	41.82	25.64	73.16
Wine	89.07	89.07	**89.71**	88.03	88.24	79.21	88.59	T2	7.69	7.02	13.71	**80.51**	6.68	37.27	21.05
V0	76.48	79.55	79.05	**82.54**	65.77	69.71	75.94	Y4587	10.26	7.24	11.33	14.92	5.93	9.04	**15.47**
SPE	33.38	34.26	35.25	37.70	31.52	32.60	**38.03**	Y28	28.58	14.50	49.45	58.91	22.89	32.54	**64.14**
Yeast	42.38	45.44	42.70	45.83	39.08	41.53	**46.10**	Yeast4	20.10	15.62	28.15	25.63	18.27	18.45	**29.27**
N2	92.44	92.53	91.50	88.55	90.21	57.90	**93.96**	W4	7.83	8.14	9.62	9.69	6.82	7.53	**15.82**
Glass6	70.97	69.55	**80.98**	75.14	69.38	53.82	72.86	Y2897	9.98	7.55	12.43	17.63	6.49	8.90	**27.93**
PBI	39.79	36.42	**43.71**	37.75	33.24	28.22	43.34	Yeast5	42.64	28.00	60.84	**63.48**	34.64	25.49	48.78
CBI	39.13	36.68	40.42	35.09	29.79	28.97	**41.00**	W86	9.39	6.62	12.13	13.38	6.68	9.76	**27.76**
Yeast3	59.46	57.92	62.39	**66.02**	48.31	53.94	65.63	Yeast6	20.58	15.56	31.66	29.88	19.90	17.21	**31.68**
Ecoli3	47.78	43.27	49.21	51.43	40.80	26.09	**51.86**	W867	4.32	3.85	5.52	8.56	4.03	2.16	**12.55**
Y9368	61.82	58.88	60.06	66.55	41.76	44.07	**68.64**	W395	12.26	5.61	12.73	12.43	5.51	5.64	**15.48**
E4756	68.37	55.81	63.95	57.47	42.36	28.33	**70.90**	P896	29.87	5.47	**88.17**	21.05	21.22	8.15	70.84
Y6794	40.14	31.26	38.02	36.07	27.80	25.87	**44.65**	P895	4.43	2.02	5.60	6.48	2.74	3.76	**16.72**
E2356	64.03	56.19	59.71	62.51	33.66	23.08	**64.24**	P86	17.13	1.83	**78.20**	30.98	17.07	4.71	56.48
								Average	41.95	37.49	48.87	47.74	35.19	31.19	**50.51**

TABLE IV
F_1 RESULTS OBTAINED FROM NNDN AND OTHER SIX COMPARISON METHODS FOR THE 40 IMBALANCED DATA SETS, AND THE BEST RESULT IS IN **BOLD** FACE

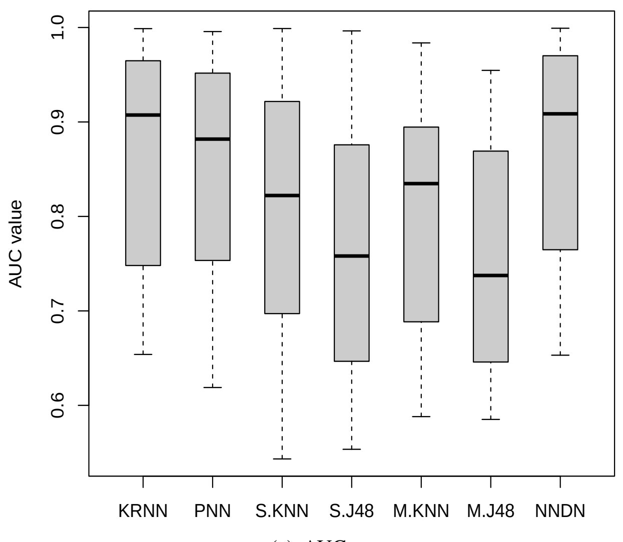
Datasets	KRNN	PNN	SMOTE		MetaCost		NNDN	Datasets	KRNN	PNN	SMOTE		MetaCost		NNDN
			KNN	J48	KNN	J48					KNN	J48	KNN	J48	
Clean1	77.26	**80.05**	78.02	77.10	66.29	64.52	79.33	L7891	61.95	62.38	68.71	**71.56**	54.18	47.14	64.19
Glass1	67.65	67.73	68.28	63.84	60.24	56.08	**70.82**	glass2	**29.08**	20.88	23.15	27.00	16.31	13.80	27.54
Wis	96.36	95.45	95.58	94.35	93.70	94.52	**96.93**	E1465	73.14	61.08	**83.48**	62.66	60.43	34.22	78.64
Pima	62.77	61.70	60.34	61.78	62.43	63.30	**63.78**	Y17	31.19	22.73	31.64	26.78	21.09	19.09	**34.91**
German	52.73	52.86	50.87	50.15	51.62	53.40	**53.89**	Glass5	56.56	35.26	**63.61**	59.30	31.25	21.63	61.52
Yeast1	56.01	54.88	55.83	55.30	56.57	56.64	**56.85**	P134	81.22	56.37	90.63	**93.95**	43.69	42.19	83.13
Wine	94.37	94.75	94.42	93.39	93.20	88.05	**95.04**	T2	12.79	11.86	12.14	90.08	11.16	54.11	14.15
V0	85.97	**86.07**	84.47	85.24	73.70	80.70	84.73	Y4587	17.87	13.57	17.45	12.24	12.68	12.34	**20.09**
SPE	47.68	48.10	47.25	42.41	41.86	43.36	**50.88**	Y28	39.79	25.01	55.05	50.34	18.01	9.26	**56.81**
Yeast	52.69	54.03	52.67	52.74	52.42	51.26	**55.28**	Yeast4	29.85	25.94	32.43	31.78	22.22	28.22	**35.24**
N2	94.63	94.53	93.00	86.55	88.32	64.90	**95.10**	W4	12.15	13.92	13.03	12.67	10.32	12.85	**15.16**
Glass6	74.81	72.95	**81.41**	78.49	58.80	66.25	72.77	Y2897	15.75	13.32	18.71	**20.31**	8.04	12.26	18.95
PBI	51.04	51.20	50.66	48.06	44.34	37.10	**53.10**	Yeast5	58.9	43.35	65.29	**70.07**	43.55	39.94	63.35
CBI	49.95	50.07	50.33	45.58	41.16	28.60	**51.18**	W86	15.13	12.14	15.32	18.64	11.63	12.23	**27.39**
Yeast3	69.96	70.22	72.37	**73.91**	68.03	66.61	72.23	Yeast6	32.24	26.02	**47.27**	42.75	20.54	27.15	40.11
Ecoli3	59.16	59.29	56.53	58.44	52.53	37.17	**64.68**	W867	7.14	7.28	8.16	11.46	6.31	6.33	**12.32**
Y9368	72.25	70.11	74.62	73.83	66.09	55.56	**74.79**	W395	**16.99**	9.35	14.92	15.05	6.78	9.36	12.27
E4756	71.65	67.63	72.03	68.43	50.08	37.15	**74.03**	P896	43.93	10.34	**81.26**	36.20	15.56	7.93	76.42
Y6794	51.34	45.29	45.61	41.78	40.45	36.47	**55.61**	P895	7.34	3.93	3.59	**7.77**	4.04	3.44	5.15
E2356	68.06	66.00	**70.50**	65.00	50.47	34.22	68.72	P86	27.34	3.54	**76.87**	29.44	9.86	7.77	59.96
								Average	50.67	45.53	54.44	52.66	41.00	38.43	**54.93**

In spite of our method has the overall good performance in direct comparison with the rest, the result must be statistically validated. We conduct a non-parametric Friedman's test which is a favorable statistical test for comparing more than two algorithms over multiple data sets and Holm's post-hoc procedure. AUC, Pre and F1 are selected as evaluation criteria, respectively.

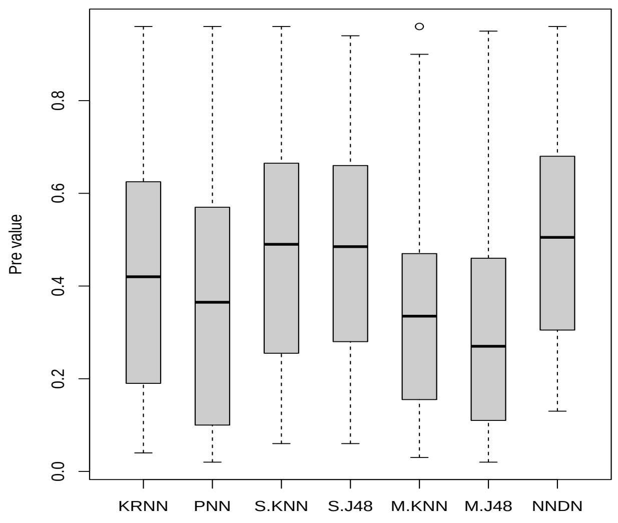
The first step in the Friedman's test is to calculate average rankings of the algorithms in terms of the three metrics. We report the average rank of each comparing algorithms on forty data sets in Table V, Table VI and Table VII, respectively. A ranking of 1 is the best.

TABLE V
AVERAGE RANKINGS OF THE ALGORITHMS FOR AUC METRIC OBTAINED BY APPLYING THE FRIEDMAN TEST

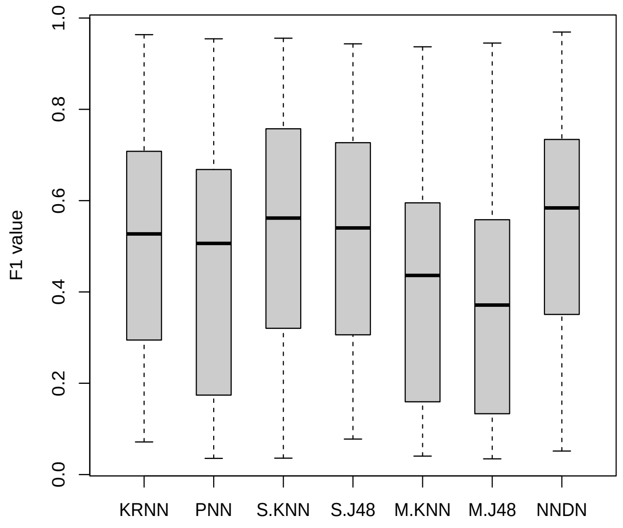
Method	Average rank
NNDN	**1.4125**
KRNN	2.2125
PNN	2.7250
SMOTE-KNN	4.6250
MetaCost-KNN	4.9250
MetaCost-J48	6.0250
SMOTE-J48	6.0750



(a) AUC



(b) Pre



(c) F1

Fig. 4. Boxplots of the overall performance of all methods

It can be seen that NNDN is ranked better than that of other algorithms in terms of each metric. We could highlight the robustness of NNDN with respect to the other approaches since there is a clear difference for the average rankings.

TABLE VI
AVERAGE RANKINGS OF THE ALGORITHMS FOR PRE METRIC OBTAINED BY APPLYING THE FRIEDMAN TEST

Method	Average rank
NNDN	**1.8875**
SMOTE-KNN	2.7375
SMOTE-J48	2.9000
KRNN	4.0125
PNN	4.7250
MetaCost-KNN	5.8500
MetaCost-J48	5.8875

TABLE VII
AVERAGE RANKINGS OF THE ALGORITHMS FOR F1 METRIC OBTAINED BY APPLYING THE FRIEDMAN TEST

Method	Average rank
NNDN	**1.8000**
SMOTE-KNN	2.9750
KRNN	3.4250
SMOTE-J48	3.6000
PNN	4.2500
MetaCost-KNN	5.9750
MetaCost-J48	5.9750

TABLE VIII
HOLM'S TEST RESULTS FOR ALL ALGORITHMS ON THE AUC METRICS

Method	z	p	Holm's adjusted alph
KRNN	$1.52e$-09	0.0083	Rejected
PNN	$1.52e$-09	0.0100	Rejected
SMOTE-KNN	$7.50e$-09	0.0125	Rejected
SMOTE-J48	$7.50e$-09	0.0167	Rejected
MetaCost-KNN	$7.88e$-05	0.0250	Rejected
MetaCost-J48	$7.88e$-05	0.0500	Rejected

TABLE IX
HOLM'S TEST RESULTS FOR ALL ALGORITHMS ON THE PRE METRICS

Method	z	p	Holm's adjusted alph
KRNN	$4.24e$-09	0.0083	Rejected
PNN	$9.87e$-09	0.0100	Rejected
SMOTE-KNN	$8.35e$-08	0.0125	Rejected
SMOTE-J48	$7.40e$-05	0.0167	Rejected
MetaCost-KNN	$2.64e$-03	0.0250	Rejected
MetaCost-J48	$1.37e$-02	0.0500	Rejected

Then we verify that the average ranks are significantly different in all data sets. Friedman's statistics are calculated as 181.52, 130.69 and 122.10, respectively, and have a chi-square distribution with 6 degrees of freedom after applying the Friedman's test. The p-value of the test on each metric is less than $2.2e$-16, indicating that the null hypothesis of equal performance should be rejected with an α=0.05.

According to the statistical studies shown in Table VIII, Table IX and Table X, the null hypothesis of equality is rejected. It indicates that the proposed NNDN approach statistically outperforms all the algorithms of comparison. It is a competitive technique to solve imbalanced classification in a general scenario. We must also stress that the p-values associated with the comparisons support our conclusions with a high degree of confidence. So we can state that our approach is obviously better than its competitors from the view of statistical point.

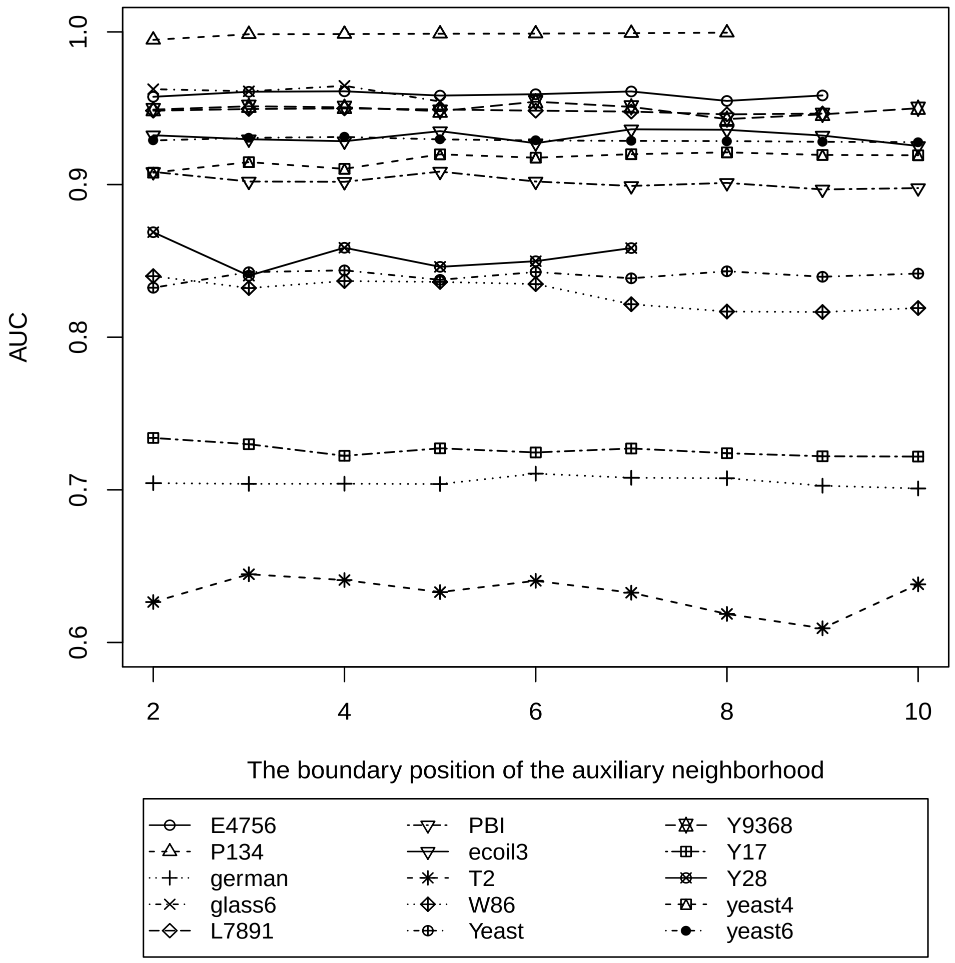


Fig. 5. AUC results with settings of the boundary of the auxiliary neighborhood. the horizontal axis represents the boundary position of the auxiliary neighborhood, and the vertical axis represents the corresponding AUC value.

TABLE X
HOLM'S TEST RESULTS FOR ALL ALGORITHMS ON THE F1 METRICS

Method	z	p	Holm's adjusted alph
KRNN	$1.52e$-09	0.0083	Rejected
PNN	$9.37e$-09	0.0100	Rejected
SMOTE-KNN	$3.05e$-07	0.0125	Rejected
SMOTE-J48	$6.30e$-06	0.0167	Rejected
MetaCost-KNN	$8.85e$-03	0.0250	Rejected
MetaCost-J48	$1.14e$-02	0.0500	Rejected

C. *The properties and parameters of* NNDN

We analyze the properties and parameters of NNDN in this section. The influence of the rules for constructing the auxiliary neighborhood on classification performance of NNDN will be demonstrated. We will explain the reasons for not taking other positions of the auxiliary neighborhood from the view of experimental point. Subsequently, the size of the main neighborhood and the auxiliary neighborhood generated by CDN is given.

As shown in Fig 5, 15 data sets with different IR are selected from Table I for the experiment. The horizontal axis represents the boundary position of the auxiliary neighborhood. (For example, “3” indicates that the third “positive-negative” dividing line is served as the boundary.) The vertical axis represents the corresponding AUC value of NNDN. Some curves like Y28, P134 and E4756 are incomplete, due to the fact that the location of boundary for the auxiliary neighborhood cannot be further. It is obvious that the performance of NNDN is not sensitive to the boundary. Even when further boundary is taken, the negative effects will follow. For example, AUC value of Y28 and W86 all have been negatively affected. Therefore, the second location is chosen as the boundary of the auxiliary neighborhood.

TABLE XI
THE PROPORTION OF INSTANCES THAT THE SIZE OF THE MAIN NEIGHBORHOOD IS LESS THAN OR EQUAL TO 9 AND THAT THE AUXILIARY NEIGHBORHOOD IS EXTENDED BY ITS MAIN NEIGHBOR

Datasets	The proportion 1 (%)	The proportion 2 (%)
German	84.50	84.90
Yeast	49.12	66.51
Glass6	11.21	29.44
PBI	38.38	51.00
Ecoli3	26.79	32.14
Y9368	21.31	43.13
E4756	24.12	35.41
L7891	19.19	32.96
P134	9.96	27.97
Y28	12.86	62.24
Yeast4	15.50	45.75
Y17	20.91	77.61
W86	16.01	50.46
T2	35.00	88.61
Yeast6	10.38	39.35

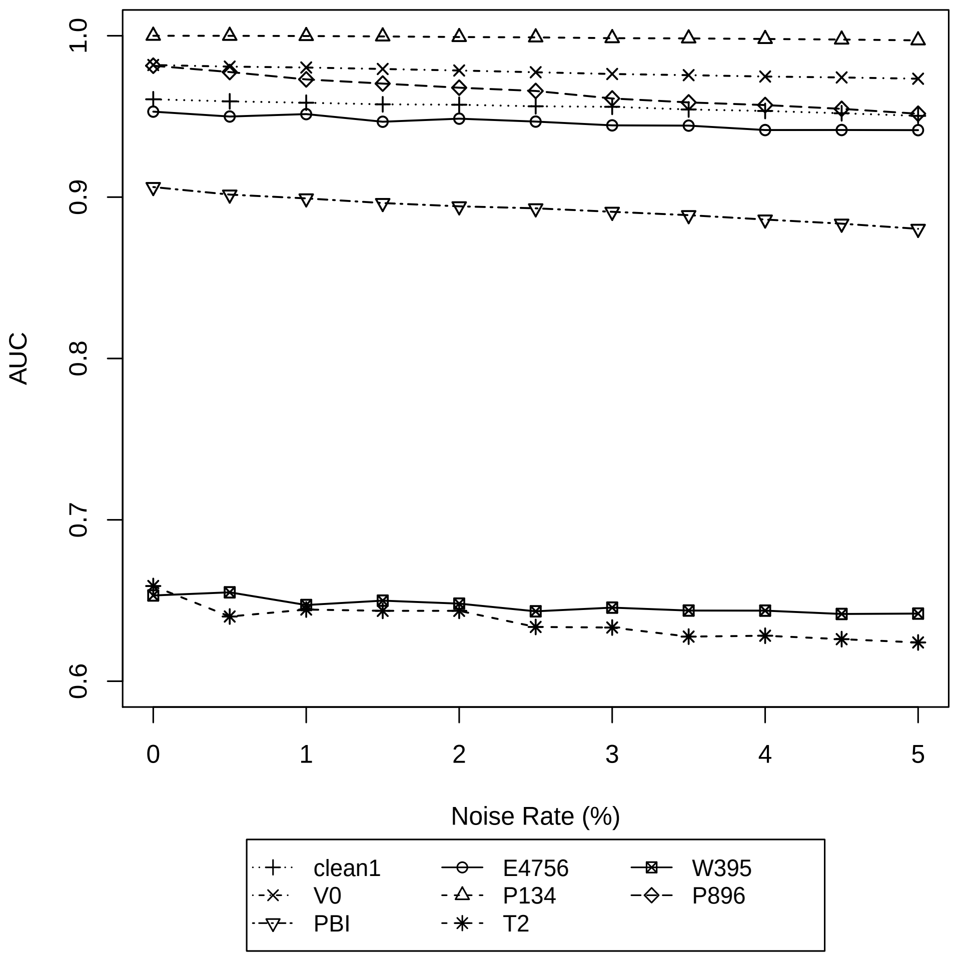


Fig. 6. Classification performance of the NNDN algorithm on imbalanced data sets with different noise rates. the horizontal axis represents the noise rate of each training set, and the vertical axis represents the corresponding AUC value of it.

The proportion of instances whose main neighborhood's size is less than or equal to 9 is counted (abbreviated as the proportion 1). Then the proportion are counted for all instances whose auxiliary neighborhood is extended by its main neighborhood (abbreviated as the proportion 2). Detailed contents are shown in the Table XI.

As shown in Table XI, the proportion 1 and the proportion 2 are described in the last two columns, respectively. For a lowly imbalanced data set, such as German, 84.50% of all main neighborhoods with a size is less than or equal to 9, and 84.90% of all auxiliary neighborhoods extended by its main neighbor. This computational complexity is very small due to the low IR. For a moderately imbalanced data set, both of proportions are relatively small. A data set with a high IR requires more computational costs. For example, IR of Yeast6 is 41.10, and 10.38% of all main neighborhoods with a size less than or equal to 9. However, 39.35% of auxiliary neighborhoods are expanded from the main neighborhood.

D. The robustness of the NNDN *algorithm on noise data sets*

Experiments on robustness of the NNDN algorithm in imbalanced data sets with varying noise ratios are conducted in this section. Due to space constraints, only eight data sets with different IR are selected from Table I for the experiment. when each data set is classified, 0.5% to 5% of the labels in the training set are artificially changed to error labels.

As shown in Fig 6, the horizontal axis represents the noise rate of each training set, and the vertical axis represents the corresponding AUC value of the NNDN algorithm. On data sets P896, PBI and T2, as the noise rates increase, the performance of the NNDN algorithm is slightly reduced. However, most data sets correspond to smooth lines. It is shown that the NNDN algorithm is robust to most data sets with the increase of noise data. In summary, the proposed algorithm has a good robustness when classifying on imbalanced data sets with low noise rates.

VI. Conclusion and future work

False negative errors caused by positive sparsity often occur in imbalanced data classification. In the NNDN algorithm, more attention is paid to it. We put forward the idea of the double neighborhoods, which is more convenient to describe the distribution of positive neighbor of the query instance. Instances closer to the query instance are usually more reliable to its classification, so the neighbors in the main neighborhood are weighted by distance first. After determining positive tendency by the positive frequencies of the main neighborhood and training set, we decide whether the sparsity of the main neighborhood needs to be further judged by constructing the auxiliary neighborhood. The neighbors in the main neighborhood that satisfies two conditions are given a weight by tendency. Experiments on 40 benchmark data sets showed that NNDN is significantly better than the resampling and cost-sensitive strategies for imbalanced data. It is also superior to the most advanced algorithms in the KNN family for dealing with imbalanced data. At the same time, the proposed algorithm has a good robustness when classifying on imbalanced data sets with low noise rates.

We intend to extend the NNDN algorithm to multi-class data imbalance in the future. We are also working on more efficient estimation methods of the positive frequency of the neighborhood.

Acknowledgments

We thank the anonymous reviewers whose comments and suggestions helped improve and clarify this manuscript to a large extent.

References

[1] M. Khalilia, S. Chakraborty and M. Popescu, "Predicting disease risks from highly imbalanced data using random forest," *BMC Medical Informatics and Decision Making*, vol. 11, no. 1, pp. 51-51, 2011.

[2] R. J. Kuo, P. Y. Su, F. E. Zulvia and C. C. Lin, "Integrating cluster analysis with granular computing for imbalanced data classification problem–a case study on prostate cancer prognosis," *Computers and Industrial Engineering*, vol. 125, pp. 319-332, 2018.

[3] W. Wei, J. J. Li, L. B. Cao, Y. M. Ou and J. H. Chen, "Effective detection of sophisticated online banking fraud on extremely imbalanced data," *World Wide Web-internet and Web Information Systems*, vol. 16, no. 4, pp. 449-475, 2013.

[4] U. Fiore, D. S. Alfredo, P. Francesca, Z. Paolo and P. Francesco, "Using generative adversarial networks for improving classification effectiveness in credit card fraud detection," *Information Sciences*, vol. 479, pp. 448-455, 2019.

[5] H. Guo, Y. Li, S. Jennifer, M. Gu, Y. Huang and B. Gong, "Learning from class-imbalanced data: review of methods and applications," *Expert Systems with Applications*, vol. 73, pp. 220-239, 2017.

[6] Hendry and R. C. Chen, "Using deep learning to predict user rating on imbalance classification data," IAENG International Journal of Computer Science, vol. 46, no. 1, pp109-117, 2019.

[7] A. N. Amany, I. G. Neveen and A. S. Afaf, "Antlion optimization and boosting classifier for spam email detection," *Future Computing and Informatics Journal*, vol. 3, no. 2, pp. 436-442, 2018.

[8] K. Savetratanakaree, K. Sookhanaphibarn, S. Intakosum and R. Thawonmas, "Borderline over-sampling in feature space for learning algorithms in imbalanced data environments," IAENG International Journal of Computer Science, vol. 43, no. 3, pp363-373, 2016.

[9] N. V. Chawla, K. W. Bowyer, L. O. Hall and W. P. Kegelmeyer, "SMOTE: Synthetic minority over sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, no. 1, pp. 321-357, 2002.

[10] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263-1284, 2009.

[11] S. García and F. Herrera, "Evolutionary undersampling for classification with imbalanced datasets: proposals and taxonomy," *Evolutionary Computation*, vol. 17, no. 3, pp. 275-306, 2014.

[12] A. Estabrooks and N . Japkowicz, "A mixture-of-experts framework for learning from imbalanced data sets," in *International Symposium on Intelligent Data Analysis*, 2001, pp. 34-43.

[13] N. Prasad L V and M. M. Naidu, "CC-SLIQ: Performance enhancement with 2k split points in SLIQ decision tree algorithm," IAENG International Journal of Computer Science, vol. 41, no. 3, pp163-173, 2014.

[14] N. V. Kalyankar, "Effect of training set size in decision tree construction by using GATree and J48 algorithm," Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering 2018, WCE 2018, 4-6 July, 2018, London, U.K., pp193-196.

[15] C. F. Wang and K. Liu, "Learning bayesian network classifier based on artificial fish swarm algorithm," IAENG International Journal of Computer Science, vol. 42, no. 4, pp355-360, 2015.

[16] Y. Yang and Y. Wu, "On the properties of concept classes induced by multivalued bayesian networks," *Information Sciences*, vol. 184, no. 1, pp. 155-165, 2012.

[17] N. Shigei, K. Mandai, S. Sugimoto, R. Takaesu and Y. Ishizuka, "Land- use classification using convolutional neural network with bagging and reduced categories," Lecture Notes in Engineering and Computer Science: Proceedings of The International MultiConference of Engineers and Computer Scientists 2019, IMECS 2019, 13-15 March, 2019, Hong Kong, pp7-11.

[18] N. Singhal, Srishti, and V. Kalaichelvi, "Application of convolutional neural network to classify sitting and standing postures, "Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering and Computer Science 2017, WCECS 2017, 25-27 October, 2017, San Francisco, USA, pp140-144.

[19] M. Mohamadian, H. Afarideh, and F. Babapour, "New 2d matrix-based neural network for image processing applications," IAENG International Journal of Computer Science, vol. 42, no. 3, pp265-274, 2015.

[20] A. Hambarde, M. F. Hashmi and A. Keskar, "Robust image authentication based on hmm and svm classifiers," Engineering Letters, vol. 22, no. 4, pp183-193, 2014.

[21] J. Xie, D. F. Lu, J. H. Shu, J. Wang, H. Y. Wang, C. Meng, and W. Zhang, "A hybrid support vector machine method for protein remote homology detection," Lecture Notes in Engineering and Computer Science: Proceedings of The International MultiConference of Engineers and Computer Scientists 2018, IMECS 2018, 14-16 March, 2018, Hong Kong, pp57-62.

[22] Y. Sun, M. S. Kamel, A. K. C. Wong and Y. Wang, "Cost-sensitive boosting for classification of imbalanced data," *Pattern Recognition*, vol. 40, no. 12, pp. 3358-3378, 2007.

[23] P. Domingos, "MetaCost: A general method for making classifiers cost-sensitive," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1999, pp. 155-164.

[24] Z. Chen, T. Lin, X. Xia, H. Xu and S. Ding, "A synthetic neighborhood generation based ensemble learning for the imbalanced data classification," *Applied Intelligence*, vol. 48, no. 8, pp. 2441-2457, 2017.

[25] H. X. Guo, L. J. Li, Y. N. Li, X. Liu and J. L. Li, "BPSO-Adaboost-KNN ensemble learning algorithm for multi-class imbalanced data classification," *Engineering Applications of Artificial Intelligence*, vol. 49, pp. 176-193, 2016.

[26] N. Japkowicz and S. Stephen, "The class imbalance problem: a systematic study," *Intelligent Data Analysis*, vol. 6, no. 5, pp. 429-449, 2002.

[27] N. Japkowicz, "Concept-learning in the presence of between-class and within-class imbalances," *Biennial Conference of the Canadian Society on Computational Studies of Intelligence: Advances in Artificial Intelligence*, vol. 2056, pp. 67-77, 2001.

[28] T. Jo and N. Japkowicz, "Class imbalances versus small disjuncts," *Acm Sigkdd Explorations Newsletter*, vol. 6, no. 1, pp. 40-49, 2004.

[29] X. Zhang, Y. Li, R. Kotagiri, L. Wu and Z. Tari, "KRNN: K rare-class nearest neighbour classification," *Pattern Recognition*, vol. 62, pp. 33-44, 2017.

[30] X. Zhang and Y. Li, "A positive-biased nearest neighbor algorithm for imbalanced classification," *Advances in Knowledge Discovery and Data Mining*, vol. 7819, pp. 293-304, 2013.

[31] L. Wang, X. Zheng, L. Zhang and Q. Yue, "Notes on distance and similarity measures of dual hesitant fuzzy sets," IAENG International Journal of Applied Mathematics, vol. 46, no. 4, pp488-494, 2016.

[32] F. Provost and P. Domingos, "Well-trained pets: improving probability estimation trees," *Technical Report CDER 00-04-IS, Stern School of Business*, New York University, 2000.

[33] I. H. Witten and E. Frank, *Data mining: practical machine learning tools and techniques*, Morgan Kaufmann, 2005.

[34] Z. Zhang, B. Krawczyk, S. García, A. R. Pérez and F. Herrera, "Empowering one-vs-one decomposition with ensemble learning for multi-class imbalanced data," *Knowledge-Based Systems*, vol. 106, pp. 251-263, 2016.

[35] M. Galar, A. Fernández, E. Barrenechea, H. Bustince and F. Herrera, "An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes," *Pattern Recognition*, vol. 44, no. 8, pp. 1761-1776, 2011.

[36] S. Alshomrani, A. Bawakid, S. O. Shim, A. Fernndez and F. Herrera, "A proposal for evolutionary fuzzy systems using feature weighting: dealing with overlapping in imbalanced datasets," *Knowledge-Based Systems*, vol. 73, pp. 1-17, 2015.

[37] J. T. Wixted, L. Mickes, S. A Wetmore S. D. Gronlund and J. S. Neuschatz, "ROC analysis in theory and practice," *Journal of Applied Research in Memory and Cognition*, vol. 6, no. 3, pp. 343-351, 2017.

[38] O. Loyola-Gonzlez, M. A. Medina-Prez, J. F. Martnez-Trinidad, J. A. Carrasco-Ochoa, R. Monroy and M. Garca-Borroto, "PBC4cip: A new contrast pattern-based classifier for class imbalance problems," *Knowledge-Based Systems*, vol. 115, pp. 100-109, 2017.

[39] D. T. Larose and C. D. Larose, *Data mining and predictive analytics*, John Wiley and Sons Inc Publishing, 2015.

[40] “Knowledge extraction based on evolutionary learning,” http://www.keel.es/.
[41] “Ucivine machine learning repository,” http://archive.ics.uci.edu/ml/.

Caiwen Wang was born in Lanzhou, Gansu Province, China in 1994. She received her B. S. degree in the Department of Mathematics and Applied Mathematics from Tianshui Normal University in 2016. Now she is a master of mathematics and statistics at Xidian University. She is major in Probabilistic graphical models, data analysis and its application.

Youlong Yang received his B. S. and M. S. in Mathematics from Shaanxi Normal University, Xian, China in 1990 and 1993 respectively, and Ph.D. in System Engineering from Northwester Polytechnical University, Xian,China in 2003. Since 2004, he has been with the faculty at Xidian University, Xi’an, China. His research interests include Machine learning, Statistical data analysis and Probabilistic graphical models.