Neural Computing and Applications (2021) 33:4457–4481
https://doi.org/10.1007/s00521-020-05256-0

ORIGINAL ARTICLE

A novel density-based adaptive *k* nearest neighbor method for dealing with overlapping problem in imbalanced datasets

Bo-Wen Yuan[1,3] · Xing-Gang Luo[2] · Zhong-Liang Zhang[2] · Yang Yu[1] · Hong-Wei Huo[3] · Tretter Johannes[3] · Xiao-Dong Zou[3]

Received: 5 October 2019 / Accepted: 27 July 2020 / Published online: 9 August 2020

Abstract

Although a large number of solutions have been proposed to handle imbalanced classification problems over past decades, many researches pointed out that imbalanced problem does not degrade learning performance by its own but together with other factors. One of these factors is the overlapping problem which plays an even larger role in the classification performance deterioration but is always ignored in previous study. In this paper, we propose a density-based adaptive *k* nearest neighbor method, namely DBANN, which can handle imbalanced and overlapping problems simultaneously. To do so, a simple but effective distance adjustment strategy is developed to adaptively find the most reliable query neighbors. Concretely, we first partition training data into six parts by density-based method. Next, for each part, we modify distance metric by considering both local and global distribution. Finally, output is made by the query neighbors selected in the new distance metric. Noticeably, the query neighbors of DBANN are adaptively changed according to the degree of imbalance and overlap. To show the validity of our proposed method, experiments are carried out on 16 synthetic datasets and 41 real-world datasets. The results supported by the proper statistical tests show that our proposed method significantly outperforms the state-of-the-art methods.

Keywords Nearest neighbor classification · Imbalanced datasets · Overlapping problem · Density-based method

1 Introduction

Imbalance classification, which has been widely applied in different scenarios including industrial manufacturing [1, 2], financial management [3, 4], biomedical engineering [5], information technology [6] and etc., is one of the critical issues in machine learning and data mining. Imbalanced datasets indicate a skewed distribution, namely the instances of one class outnumber the instances of other classes. Most of the standard classifiers tend to bias toward the majority class thus leading to high misclassification rate of minority instances. Imbalanced problem is commonly viewed as a main challenge to classification and has attracted great attention [7].

Existing solutions for imbalanced problem can be roughly grouped into four categories: resampling techniques, algorithm modification methods, cost-sensitive learning approaches and ensemble learning methods.

- Resampling techniques aim to rebalance the training dataset by means of some mechanisms to generate a more or less balanced class distribution which is suitable for standard classifiers [8, 9].
- Algorithm modification methods try to adjust the structure of standard classifiers to diminish the effect caused by class imbalance [10].
- Cost-sensitive learning approaches generally consider higher cost for minority class to compensate for the scarcity of minority data [11].
- Ensemble learning methods are originally developed to enhance the classification ability by combining different single classifiers. Moreover, researchers modify

✉ Xing-Gang Luo
xgluo@mail.neu.edu.cn

[1] Department of Information Science and Engineering, Northeastern University, Shenyang 110819, China

[2] Department of Management, Hangzhou Dianzi University, Hangzhou 310018, China

[3] Department of Foundry, BMW Brilliance Automotive Ltd, Shenyang 110143, China

ensemble algorithms to adapt to imbalanced problem and show promising results [12–14].

In addition to imbalanced problem, overlapping between classes is convinced as another factor which degrades the learning performance [15, 16]. Overlap appears when a region contains almost equal numbers of instances from different classes. This situation results in a roughly same prior probability for each class and thus brings a strong handicap for classification. Overlapping problem is pervasive in many real-world applications such as fault diagnosis [17], character recognition [18], speech classification [19] and drug design [20]. In these scenarios, instances from different classes usually have similar characteristics in the feature space. For example, in character recognition, letters ‘O’, ‘o’ and numeral ‘0’ have almost identical shape which results in an overlapping region in the feature space therefore hard to separate. Previous investigations have shown that overlap degrades classification performance even more severely than imbalance [21]. To clearly present the relationship between two factors, a series of experiments are conducted by varying the degree of imbalance and overlap in the dataset. The conclusions state that learning algorithms can yield competitive performance when dataset has low overlapping degree combined with high imbalance ratio, but they are hard to achieve desirable results in high overlapping degree even imbalance ratio is low. This demonstrates that overlap is the main factor of classification degeneration [21–23]. Furthermore, Denil and Trappenberg [15] took size of dataset into consideration, and the study revealed that in small datasets the learning process is hindered by imbalance and overlap, respectively. However, when training data are sufficient, two factors jeopardize the learning performance interact.

Currently, most of the researches deal with overlapping and imbalanced problems separately. However, in practical application level, overlapping problem frequently occurs in imbalanced data which poses a greater challenge to classification. Although a few papers attempt to consider both factors as a whole [24, 25], the structures of related algorithms are too complex to implement. To fill this gap, we propose a density-based adaptive nearest neighbor method (DBANN) which can deal with these two problems simultaneously with a simple structure. The main idea of DBANN is to develop an adaptive distance adjustment strategy which devotes to defining and making use of reliable query neighbors. To the best of our knowledge, our approach is the first *k*NN-based method which aims to combat both imbalanced and overlapping problems. The main contributions of our study can be summarized as follows:

- We propose a density-based *k*NN method named DBANN which can handle imbalanced and overlapping problems simultaneously.
- To enhance classification ability, we develop a distance adjustment strategy using density-based methods to adaptively find out the most reliable query neighbors.
- To validate the effectiveness of DBANN, we compare with other state-of-the-art methods on 16 synthetic datasets and 41 real-world datasets, respectively.

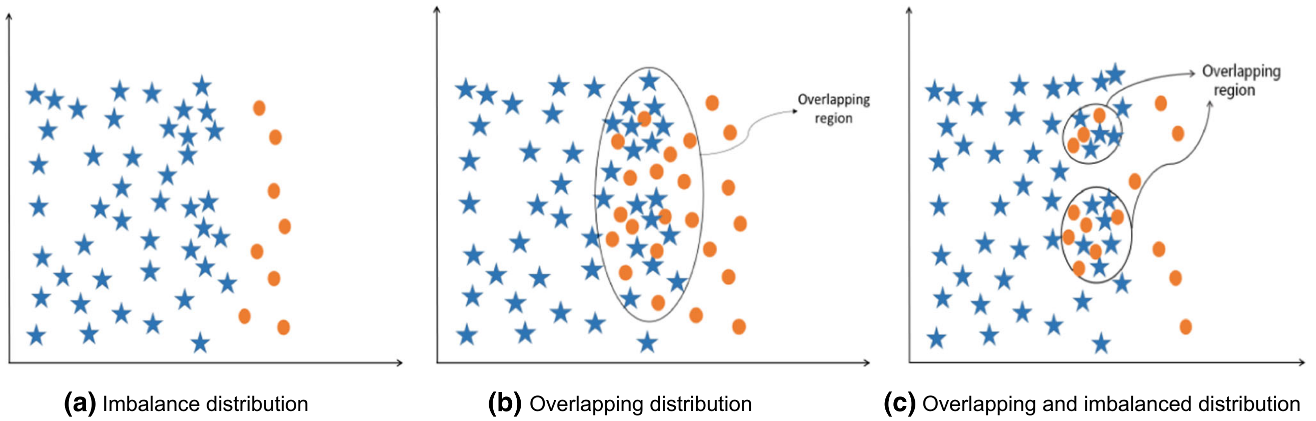
The outline of this paper is organized as follows: The related works are described in Sect. 2. In Sect. 3, we introduce our proposed method DBANN. Section 4 presents extensive experiments on both synthetic and real-world datasets. Results and discussion are shown in Sect. 5. Finally, Conclusions are presented in Sect. 6.

2 Related works

2.1 Overlapping problem in imbalanced datasets

In binary classification, imbalanced data refer to the distribution when instances of one class outnumber the other one, as shown in Fig. 1a. In this situation, minority class is hard to be recognized by standard classifiers which prefer to take a good coverage of majority class for achieving desirable global performance. However, in real applications, minority class always contains critical information we need, such as scrap part data within all products, patient information among all people. Therefore, it is imperative to take a deep insight into the data intrinsic characteristic in imbalanced data. With this in mind, we realize that imbalanced data dose not hinder learning ability by its own but together with some other factors, such as size of a dataset [21, 26], noise [27, 28], small disjunct [29, 30] and data shift [31, 32].

Overlap occurs in the region where both classes co-exist and is viewed as one of the main obstacles to classification [15], as depicted in Fig. 1b. In such region, the probability of each class is approximately equal which gives rise to high misclassification rate. In order to quantify overlapping degree for individual feature dimension, Ho and Basu [33] proposed a metric called maximum Fisher’s discriminant ratio ($F1$), as shown in Eq. (1) where $\mu_1, \mu_{2,} \sigma_1, \sigma_2$ indicate the means and variances of the two classes, respectively. For a multidimensional dataset, the maximal f among all the features is defined as $F1$. Datasets with a low value of $F1$ will have a high degree of overlap and vice versa. To overcome overlapping problem, some researchers tended to change data distribution before modeling. Tang [16] firstly transformed original data into a more separated data distribution throughout overlapping pattern extraction and



(a) Imbalance distribution **(b)** Overlapping distribution **(c)** Overlapping and imbalanced distribution

Fig. 1 Examples of imbalanced and overlapping distribution

rough set theory, and then a proposed DR-SVM is implemented on the transformed data. Batista et al. [34] used data cleaning techniques to cope with highly overlapping data and achieved desirable results. Similarly, in order to better prepare the data for classification, other pre-processing methods such as data selection and feature selection are proposed in [35, 36]. However, the pre-processing methods may involve the risk of noise introduction or information loss. Xiong et al. [37] found that modeling the overlapping and non-overlapping regions, respectively, is a promising scheme for solving class overlapping problem. Follow this line of thinking, Vorraboot et al. [38] first partitioned training data into non-overlapping region, borderline region and overlapping region. Afterward, different techniques were employed for different regions. Finally, the outputs of all techniques were combined. Nevertheless, the study is only suitable for two Gaussian classes with independent and identical distributions.

$$f = \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2} \tag{1}$$

In the real-world application cases, overlapping problem and imbalanced problem co-exist frequently in the dataset but are always ignored in previous studies. Figure 1c marks such two overlapping regions with circles. Comparing with Fig. 1b, we can find that overlapping regions here exhibit comparative high density in the perspective of global distribution. That is partly attributed to the sparsity of minority class which inversely accentuates the compactness of two overlapping regions. Additionally, the imbalance ratio is different in overlapping region and other regions. It is worth noting that for some learning algorithms which are based on a divide and conquer strategy [39], the variation of class distribution in different regions may pose a threat to the classification performance. Besides, overlapping and imbalanced problems also influence those algorithms which are sensitive to data density such as *k* nearest neighbors.

To address overlapping and imbalanced problems, Alejo et al. [25] developed a hybrid method which combines a modified back propagation (MBP) with a gabriel graph editing technique (GGE). MBP copes with imbalanced issue and GGE is responsible for overlapping problem. Vuttipittayamongkol et al. [40] proposed an overlap-based under-sampling method (OBU). Based on elimination of majority instances from overlapping region, OBU improves the visibility of minority instances. So far, the related studies are far from enough.

2.2 *k*NN-based methods for dealing with overlapping or imbalanced datasets

*k*NN is one of the typical non-parametric approaches which is widely applied in diverse domains due to its simple but powerful decision rule [41, 42]. However, when encountering imbalanced class distribution, *k*NN tends to lose power on yielding competitive results [43, 44]. To cope with it, *k*NN is modified to incorporate immunity against the influence of imbalance. Concretely, Kriminger et al. [43] proposed a class conditional nearest neighbor distribution algorithm (CCNND). To mitigate the effect of imbalanced distribution, for each class, CCNND calculates the number of training instances which satisfies a specified distance condition within the *k* nearest neighbors of a query instance. Afterward, an empirical cumulative distribution function (CDF) is built and the probability for each class is computed. Dubey and Pudi [45] provided a weighting scheme (here after called W-*k*NN) to address imbalanced issue. W-*k*NN assigns weight to each class based on the misclassification rate obtained by traditional *k*NN. Patel [46] developed a hybrid weighted strategy (here after called H-*k*NN). The main advantage is the use of dynamic *k* value, i.e., small *k* for minority class and large *k* for majority class. Therefore, H-*k*NN improves the ability to fully mine the information from imbalanced distribution. On this basis, the same author took fuzzy rule-based classification into consideration and proposed an improved fuzzy *k*-nearest neighbor (here after called F-*k*NN) [47]. Based on fuzzy membership, the query instance is allowed to know prior that how much its neighbors belong to a class. Zhang and Li [48] presented a minority-biased nearest neighbor algorithm called PNN. In order to handle the inappropriate probability estimation for minority class, PNN fixes the number of minority query neighbors. For example, *m*-PNN means that there must be *m* minority instances in query neighbors. Therefore, the number of query neighbors changes dynamically to ensure enough instances for probability estimation for both classes. *k* rare-class nearest neighbor classification (*k*RNN) [49] boosts PNN by updating the dynamic query neighbors strategy. The new strategy reinforces the analysis of distributions around query instances. *k*RNN can handle not only inter-class imbalance but also within class imbalance. Mullick et al. [50] proposed an adaptive learning *k*NN method called Ada-*k*NN. It uses a class-based global imbalance handling scheme (GIHS) to compensate for the disadvantage of minority data scarcity. To assign global weight for each class, GIHS considers both ideal class probability (balanced distribution) and reality class probability (imbalanced data distribution).

Overlapping problem is another obstacle for learning algorithms with no exception to *k*NN, as stated in Sect. 2.1. Garcia et al. [44] investigated the behavior of *k*NN when overlap exists in imbalanced data. The results reveal that when imbalance ratio in overlapping region equals to global imbalance ratio, i.e., majority class dominates the overlapping region, true positive rate (TPR) drops with the increase in overlapping degree. Conversely, when minority class turns to be the most represented class in overlapping region, the TPR increases on the opposite. Additionally, they pointed out that imbalance ratio in overlapping region accounts more than the size of overlapping region and global imbalance ratio. Wang et al. [51] proposed an extremely simple but well performed algorithm called A-*k*NN. It aims to form reliable query neighbors for final decision. To do so, A-*k*NN modifies the distance metric to move the reliable instances closer to the query instance. Hence, even the query instance locates in the overlapping region which is viewed as ambiguous and untrusted area, the query neighbors selected after distance adjustment are reliable. Although A-*k*NN is an effective solution for overlapping problem, it cannot handle imbalanced issue.

Even though some efforts are made for *k*NN to enhance the classification performance, there are still some drawbacks remaining to be improved in further research. Firstly, previous *k*NN-based methods treat imbalanced and overlapping problems separately though the two factors always co-exist in the real-world applications. Secondly, some modified *k*NN methods choose pivot instances as query neighbors for decision making, nevertheless, the choice criterion they use considers either global or local information. When data density varies a lot in different regions, especially when imbalance ratio is significantly different among these regions, the choice criterion cannot work effectively anymore. Finally, previous works ignore the influence of noisy instances which can jeopardize the classification performance significantly. In the following sections, we will solve above concerns with a novel method.

3 Combating overlapping and imbalanced problems using density-based adaptive *k* nearest neighbor method (DBANN)

In this section, we expect to conquer overlapping and imbalanced problems by a density-based strategy. For ease of discussion, in this paper, we focus on binary class problem even though it can be generalized to multi-class problem. Considering a given training dataset D with N instances, $D = \{(x_1, y_1), (x_2, y_2)\ldots(x_n, y_n)\}$, where $x_i \in X$ is a training instance in m dimensions, x is a query instance. Traditional *k*NN first determines the *k* nearest neighbors of query instance based on euclidean distance in Eq. (2). Then, majority vote in Eq. (3) is used to classify x. Besides, IR in Eq. (4) indicates the imbalance ratio of a dataset, where N_{minority} refers to the number of minority instances and N_{majority} refers to the number of majority instances. It is worth noting that we use $\text{IR}_{\text{global}}$ as the imbalance ratio of the whole dataset and IR_{local} as the imbalance ratio in a specified region in the sequent sections.

$$d(x, x_i) = \left(\sum_{j=1}^{m} \left| x^j - x_i^j \right|^2 \right)^{1/2} \tag{2}$$

$$f(x) = \text{sgn}\left(\sum_{i}^{k} (y_i) \right) \tag{3}$$

$$\text{IR} = \frac{N_{\text{majority}}}{N_{\text{minority}}} \tag{4}$$

3.1 Description of reliable query neighbors

Previous studies have shown the significance of query neighbors in classification for *k*NN [48, 52]. Inspired by this, the main idea of DBANN is to seek for reliable query neighbors which are used for majority vote. Different from traditional *k*NN, DBANN selects query neighbors depending not only on distance but also data characteristic, i.e., imbalanced and overlapping distribution in local and global region. Besides, we expect the query neighbors can adaptively change to adapt different imbalanced and overlapping degrees. Therefore, first of all, we describe the characteristics of reliable query neighbors as follows:

- There is no doubt that the query neighbors should be the instances locating near around the query instance so as to include more representative information.
- Due to the scarcity of minority data, the query neighbors should be biased toward the minority class.
- Since instances in overlapping region are hard to separate, it is suggested to view these instances unreliable and lower the probability of the selection for query neighbors.
- In most cases, noisy instance is an obstacle for classification [28, 53, 54]. Hence, it should be avoided to be selected as query neighbors.
- Since *k*NN is proved to be sensitive to data complexity and class density [44], it is desirable to involve density and class distribution factors into consideration.

3.2 A density-based adaptive *k* nearest neighbor method (DBANN)

Most of previous researches of *k*NN method concentrate on addressing imbalanced problem but overlook the influence of overlap. Literature [51] proposes an adaptive *k*NN method (A-*k*NN) to deal with overlapping problem. Different from traditional *k*NN, A-*k*NN modifies distance metric according to overlapping degree. First of all, for each training instance x_i, A-*k*NN creates a reliable coefficient r_i. It is the distance from a training instance x_i to another training instance x_j which is the nearest neighbor belonging to different class from x_i, as listed in Eq. (5). Based on observation, we can easily find that a lower r_i value means $\left(x_i, x_j\right)$ locate closer to each other which implies that high overlapping degree exists in this region; therefore, x_i is viewed as an unreliable instance and vice versa. With this in mind, we know that r_i value can measure reliable degree of x_i. A high r_i value implies that x_i has high reliable degree and helpful in classification, on the contrary, a low r_i value indicates that x_i is useless and unreliable. After obtaining r_i for each training instance, A-*k*NN adjusts and forms a new distance metric by Eq. (6). Finally, the output is obtained by Eq. (3). Technically speaking, A-*k*NN is a local method to handle overlapping problem. However, the imbalance issue is not considered.

$$r_i = \min_{l:y_i \neq y_j} d\left(x_i, x_j\right) \tag{5}$$

$$d_{\text{new}}(x, x_i) = \frac{d(x, x_i)}{r_i} \tag{6}$$

In this study, we extend the concept of r_i to handle both imbalanced and overlapping problems.

Concretely, in the first step, we cluster training data into several clusters and noisy instances by a density-based method (we will introduce it in Sect. 3.3). To further characterize the clusters, we consider overlapping issue and divide clusters into two types with definitions listed as follows.

Definition 1 Overlapping cluster indicates that the cluster contains both majority and minority instances.

Definition 2 Non-overlapping cluster indicates that the cluster contains only majority or minority instances.

Therefore, after clustering, training data can be divided into six parts: (a) minority noisy instances, (b) majority noisy instances, (c) majority instances in overlapping cluster, (d) minority instances in overlapping cluster, (e) majority instances in non-overlapping cluster and (f) minority instances in non-overlapping cluster. Figure 2 depicts the six parts in detail.

Afterward, in the second step, we assign reliable coefficient r_i to each training instance x_i. Different from A-*k*NN, we capture the distribution variation and take noise factor into consideration. Specially, we assign r_i for training instances in each part as follows:

(a) For minority noisy instances, r_i is assigned as the distance from x_i to the nearest majority neighbor.

(b) For majority noisy instances, r_i is assigned as a randomly little positive value.

(c) For minority instances in overlapping cluster, r_i is assigned as the distance from x_i to the IR_{local}th nearest majority neighbor. IR_{local} equals to the imbalance ratio in the corresponding cluster which represents the local distribution. Obviously, high imbalance ratio expands the detection radius of x_i and obtains larger r_i accordingly. Noticeably, r_i also relates to the density of a cluster. A cluster with a high density means a large amount of instances locating in a small region together, and even detection radius of r_i expands to the IR_{local}th nearest majority neighbor it may probably increase by only a small value compared with low-density clusters.

(d) For majority instances in overlapping cluster, r_i is assigned as the distance from x_i to the nearest minority neighbor.

(e) For minority instances in non-overlapping cluster, r_i is assigned as the distance from x_i to the $\text{IR}_{\text{global}}$ th nearest majority neighbor. Different from IR_{local} in overlapping situation, $\text{IR}_{\text{global}}$ here is calculated by global imbalance ratio.

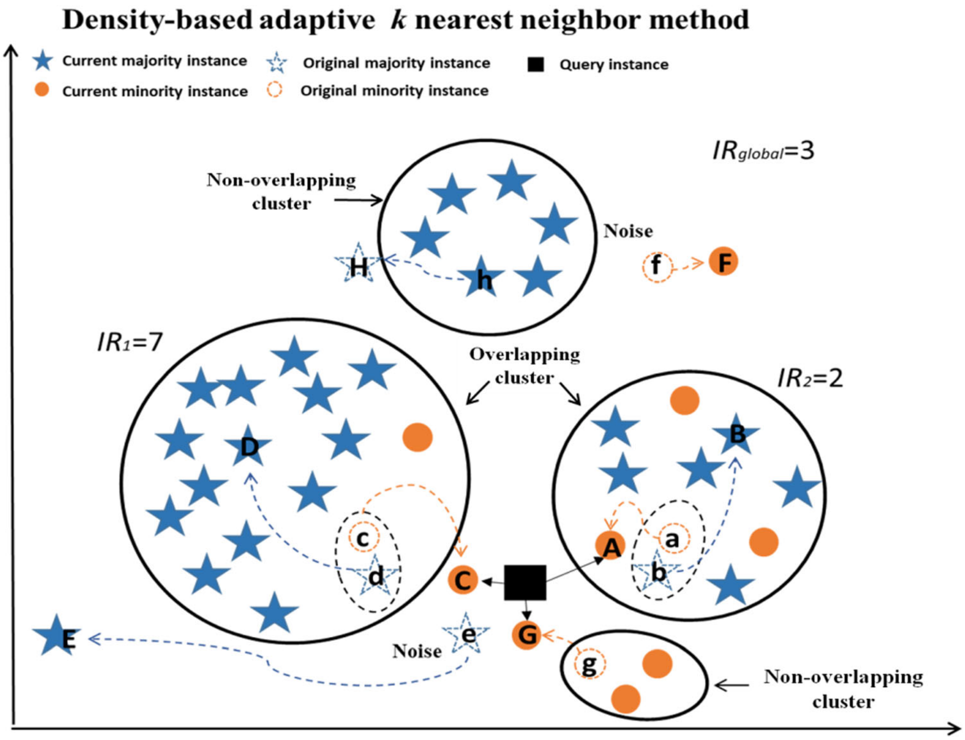

Density-based adaptive k nearest neighbor method
Current majority instance
Original majority instance
Query instance
Current minority instance
Original minority instance
IRglobal=3
Non-overlapping cluster
Noise
H
h
f
F
IR1=7
Overlapping cluster
IR2=2
D
B
A
a
b
c
d
C
E
Noise
e
G
g
Non-overlapping cluster

Fig. 2 Description of DBANN method

(f) For majority instances in non-overlapping cluster, r_i is also assigned as the distance from x_i to the nearest minority neighbor.

In the next step, we adjust the distance metric by Eq. (6). We can find that the new distance metric depends on two conditions, r_i and the euclidean distance $d(x, x_i)$. A training instance x_i is reliable when it locates near to the query instance x as well as possessing higher r_i value. It is obvious that, after distance adjustment the reliable instances are pulled closer to query instance and the unreliable ones are pushed away.

Finally, majority vote is implemented in new distance metric by Eq. (3). From the procedure introduced above, we can find that in r_i assigning procedure, our method considers not only local but also global distribution of the dataset.

Figure 2 illustrates the process of DBANN. The star and circle represent the majority and minority class, respectively, and the shape of hollow and solid represent the original and current distribution (after distance adjustment). Here, we only focus on the change of the instances marked with letters. Firstly, we concentrate on points a and b which distribute in the overlapping cluster. From the graph, we can see two instances distributing closer to each other which indicates that high overlapping degree exists. Therefore, they are viewed as unreliable points and ought to be pushed far away from the query instance by Eqs. (5) and (6). Nevertheless, except for overlapping degree, imbalanced issue is also considered. As a result, only majority point b is pushed to B whereas point a is pulled to A which is closer to query instance due to local imbalanced ratio ($\mathrm{IR}_{\mathrm{local2}} = 2$). Similar distance adjustment method is also suitable for points c and d, and the only difference is that point c is put closer to query instance due to higher local imbalance ratio ($\mathrm{IR}_{\mathrm{local1}} = 7$). Additionally, majority noisy point e is pushed far away due to the little number setting and minority noisy point f is pushed to F. As for minority point g in non-overlapping cluster, global imbalance ratio ($\mathrm{IR}_{\mathrm{global}} = 3$) is used to pull it to G, and majority point h is moved to H. Consequently, after distance adjustment, the query neighbors are point A, C, G and query instance is predicted as minority. Details of DBANN are listed in Algorithm 1.

Algorithm1: DBANN

```
Input: Training data $D$
Output: Classification result
Cluster training data $D$ by density-based clustering method
Calculate global imbalance ratio $IR_{global}$
for each instance $x_i \in D$ do
    if $x_i \in$ noise $\bigcap$ minority class then
        $r_i \leftarrow$ calculate the distance to nearest majority neighbor
    else if $x_i \in$ noise $\bigcap$ majority class then
        $r_i \leftarrow$ assign a little positive real value
    else if $x_i \in$ overlapping cluster $\bigcap$ minority class then
        $IR_{local} \leftarrow$ calculate imbalance ratio in cluster
        $r_i \leftarrow$ calculate the distance to $IR_{local}^{th}$ nearest majority neighbor
    else if $x_i \in$ non-overlapping cluster $\bigcap$ minority class then
        $r_i \leftarrow$ calculate the distance to $IR_{global}^{th}$ nearest majority neighbor
    else if $x_i \in$ cluster $\bigcap$ majority class then
        $r_i \leftarrow$ calculate the distance to nearest minority neighbor
    end if
end for
Adjust distance metric by Eq. (6)
Implement majority vote in new distance by Eq. (3)
```

3.3 Density-based clustering algorithm

In Sect. 3.2, we take advantage of density-based clustering method to divide training data into different parts. Technically speaking, it is a framework of our method which implies that many existing density-based clustering algorithms are optional in DBANN. In this paper, we choose DBSCAN as our method.

DBSCAN is a typical density-based clustering algorithm which defines cluster as a region of high dense points separated by regions of lower dense points. It has attracted much attention by its desirable properties including arbitrary shaped clusters, automatic cluster number identification and noise detection [55, 56]. Additionally, as a useful method to capture data distribution, DBSCAN is always implemented by combing with other algorithms to enhance classification ability, especially when facing severely complex data distribution [38, 57]. Generally, DBSCAN characterizes density variation by resorting to two input parameters, a positive value *eps* and a positive constant integer *Minpts*. On this basis, some definitions of DBSCAN are listed as follows:

Definition 1 *eps-neighborhood* of a point *p* indicates the points within the radius *eps* around *p*.

Definition 2 A point p is a core point if the number of its *eps-neighborhood* is more than *Minpts*.

Definition 3 A point p is directly density-reachable from a point q if q is a core point and p is its *eps-neighborhood*.

Definition 4 A point p is density-reachable from a point q if there is a chain of points $p_1, \ldots p_n, p_1 = q, p_n = p$ which satisfies that p_{i+1} is directly density-reachable from p_i.

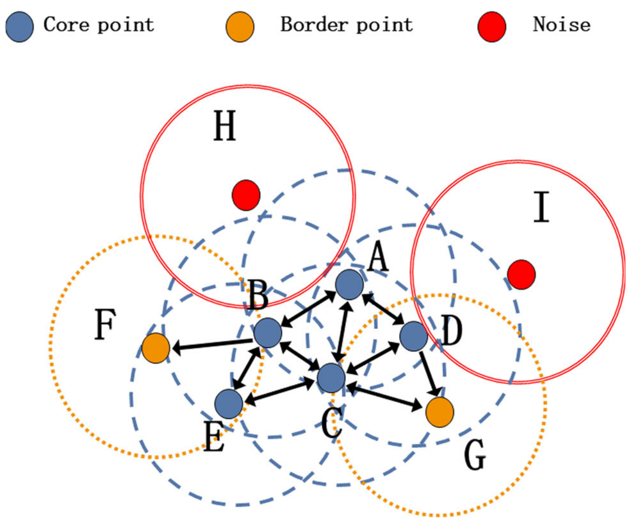


Fig. 3 Process of DBSCAN

Definition 5 A border point p is the *eps-neighborhood* of a core point q which has fewer neighbors than *Minpts* within the same *eps* radius.

Definition 6 A noisy point p is the point neither a core point nor a border point.

Initially, DBSCAN arbitrarily selects a point p and retrieves all *eps-neighborhood*, this process is defined as QueryNeighbour. If the number of *eps-neighborhood* is larger than *Minpts*, point p is assigned as core point and thus forming a new cluster, otherwise, p is assigned as a noisy point. Subsequently, the cluster expands by adding unvisited density-reachable points iteratively. The process is repeated until every unvisited point is marked either in a cluster or a noisy point. Noticeably, even a point is marked as a noisy point initially, it may be transformed to a border point of other cluster during cluster expanding process. Finally, DBSCAN forms several clusters and noisy points. Figure 3 demonstrates the process of DBSCAN (*Minpts* = 4, *eps* is indicated by the circles). As can be seen from the graph, point A is marked as core point at the beginning and thus creates a new cluster. Afterward, the cluster expands based on density measurement. It involves all blue points in the cluster until it reaches the yellow border points (F, G) which are the edge of the cluster. Due to low density, points (H, I) are marked as noisy points. The detail of DBSCAN is shown in Algorithm 2.

Table 1 Introduction of 16 synthetic datasets

Datasets	Size of positive instance	Size of negative instance	Imbalance ratio	Two centers	Overlapping degree
A1	333	667	1:2	(0.00, 0.05)	Severe overlap
A2	200	800	1:4	(0.00, 0.05)	
A3	100	900	1:9	(0.00, 0.05)	
A4	50	950	1:19	(0.00, 0.05)	
B1	333	667	1:2	(0.00, 0.50)	Moderate overlap
B2	200	800	1:4	(0.00, 0.50)	
B3	100	900	1:9	(0.00, 0.50)	
B4	50	950	1:19	(0.00, 0.50)	
C1	333	667	1:2	(0.00, 1.30)	Slight overlap
C2	200	800	1:4	(0.00, 1.30)	
C3	100	900	1:9	(0.00, 1.30)	
C4	50	950	1:19	(0.00, 1.30)	
D1	333	667	1:2	(0.00, 2.70)	Rare overlap
D2	200	800	1:4	(0.00, 2.70)	
D3	100	900	1:9	(0.00, 2.70)	
D4	50	950	1:19	(0.00, 2.70)	

Algorithm2: DBSCAN

```
Input: Training data , $eps$ , $MinPts$
Output: Sub-clusters, Noisy points
Initialize cluster id $C = 0$ , $S = \phi$
for each unvisited data point $p \in P$ do
    Neighbors $N \leftarrow$ QueryNeighbours $(p, eps, MinPts)$
    if $|N| < MinPts$ then
        label$(p) \leftarrow$ noise
        continue
    else
        label $(p) \leftarrow$ cluster $C$
        $S \leftarrow$ all unvisited $N$
        for each unvisited data point $q \in S$ do
            label $(q) \leftarrow$ cluster $C$
            Neighbors $N \leftarrow$ QueryNeighbours $(q, eps, MinPts)$
            if $q$ is a core point then
                for each data point $w$ in $N$ do
                    label $(w) \leftarrow$ cluster $C$
                    if $w$ is unvisited then
                        $S \leftarrow S \cup w$
                    end if
                end for
                mark $q$ as visited
            end if
        end for
    end if
    mark $p$ as visited
    $C \leftarrow C + 1$
end for
```

4 Experiments

In this section, experiments are carried out on 16 synthetic datasets and 41 real-world datasets to validate the effectiveness of DBANN method. The details of datasets are described in Sect. 4.1. In Sect. 4.2, we list the comparative algorithms and the corresponding parameters setting. Besides, the evaluation metrics and statistical tests are introduced in Sect. 4.3. All experiments are carried out by using python (Version 2.7.14).

Table 2 Introduction of 41 real-world datasets

Datasets	Size	Attributes	IR	$F1$
poker8vs6	1476	10	85.82	0.0153
yeast1458vs7	693	8	22.10	0.1757
glass1	213	9	1.80	0.1935
yeast1	1484	8	2.46	0.2422
yeast0359vs78	506	8	9.12	0.3113
yeast1vs7	458	7	14.26	0.3522
yeast1289vs7	946	8	30.53	0.3654
abalone918	730	7	16.38	0.6311
glass0	214	9	2.06	0.6492
yeast0256vs3789	1004	8	9.14	0.6939
winequalityred3vs5	4173	7	68.10	0.7509
yeast05679vs4	527	8	9.33	1.0515
ecoli01vs235	244	7	9.17	1.1028
vehicle0	845	18	3.27	1.1220
yeast2vs8	482	8	23.10	1.1424
yeast4	1484	8	28.09	1.2516
ecoli0146vs5	279	6	12.95	1.3345
ecoli01vs5	240	6	11.00	1.3897
ecoli3	335	7	8.57	1.3513
yeast2vs4	514	9	9.08	1.5793
ecoli046vs5	203	6	9.15	1.6030
ecoli0234vs5	202	7	9.10	1.6179
ecoli034vs5	200	7	9.00	1.6323
yeast02579vs368	1003	8	9.13	1.6361
ecoli067vs5	220	6	10.00	1.6921
ecoli2	333	7	5.44	1.8199
glass016vs5	184	9	19.44	1.8505
yeast6	1484	8	41.40	1.9674
ecoli0137vs26	281	7	39.14	2.3018
glass6	214	9	6.38	2.3913
abalone21vs8	581	10	40.50	2.4359
ecoli1	335	7	3.35	2.6396
yeast3	1484	8	8.10	2.7512
ecoli4	335	7	15.75	3.2504
glass0123vs456	213	9	3.18	3.3137
Wisconsin	683	9	1.85	3.5676
newthyroid2	215	5	5.14	3.5793
new-thyroid1	215	5	5.14	3.5793
ecoli0vs1	219	7	1.84	9.7145
shuttlec0vsc4	1829	9	13.86	12.9722
iris0	149	4	2.04	16.7971

4.1 Datasets used in the experiments

In this section, experiments are conducted on both synthetic and real-world datasets. Specially, synthetic data are generated for both classes from bivariate normal distributions. On the one hand, to explore the classification performance in different overlapping degrees, the mean vector of minority class is fixed at [0.00, 0.00], and the mean vector of majority class is set at [0.05, 0.05], [0.50, 0.50], [1.30, 1.30], [2.70, 2.70], respectively which represents four overlapping degrees. In detail, mean vector of [0.05, 0.05] means the data centers of two classes are closed; thus, severe overlapping region exists. Mean vector of [2.7, 2.7] indicates that the two centers locate far away so that the overlapping degree is rare. On the other hand, synthetic data are also generated into four different imbalance ratios by changing numbers of both classes. The description of synthetic datasets is shown in Table 1. As for real-world applications, we select 41 datasets from KEEL repository [58] refers to previous research [35], as shown in Table 2. KEEL is an open source which provides benchmark datasets for assessing the behaviors of the algorithms in different scenarios [58]. Datasets are ordered by overlapping degree according to Fisher's discriminant ratio ($F1$), in which we can divide them into two parts, low overlapping datasets with $F1 > 1.6$ and high overlapping datasets with $F1 < 1.6$. Imbalance ratio (IR) of datasets ranges from 1.8 to 68.1, as shown in Table 1.

4.2 Algorithms and parameter settings

In our experiments, DBANN is compared with other algorithms and strategies. The comparative algorithms can be divided into three directions: (a) *k*NN-based methods: The algorithms derived from *k* nearest neighbor are modified to address imbalanced or overlapping problems, including W-*k*NN [45], *k*RNN [49], F-*k*NN [47], H-*k*NN [46] and standard classifier *k*NN. (b) Generality-oriented learning algorithms and strategies: CART decision tree, support vector machine (SVM) together with data balancing methods SMOTE [59] and overlap-based under sampling (OBU) [40] which are popular in handling imbalanced and overlapping problems. (c) Ensemble algorithms: Kd-tree-based efficient ensemble (KDE) [60], hybrid sampling with bagging (HSB) [61] and RUSBoost (RUS) [62] represent three effective ensemble methods of

Table 3 Confusion matrix for binary classification

	Predicted positive class	Predicted negative class
Actual positive class	True positive (TP)	False negative (FN)
Actual negative class	False positive (FP)	True negative (TN)

bagging and boosting, respectively, for imbalanced problem.

- For *k*NN-based method W-*k*NN, *k*NN, F-*k*NN, H-*k*NN, DBANN, *k*RNN, parameter *k* is chosen from original literature which is set to 3, 3, 3, 3,3, 3,1, respectively. The other parameter is set according to the original literature. For DBANN, *Minpts* is set to 4, and *eps* is chosen as the optimal value from the range [0.01, 200] by 10-fold cross-validation.
- For generality-oriented algorithms, support vector machine is implemented with linear kernel which shows desirable performance in selected datasets. SMOTE and OBU are conducted to generate an equal number between minority class and majority class before classification.
- For ensemble algorithms, the base classifier is decision tree, and the number of base classifier is {10,10,40} for KDE, HSB and RUS, respectively, according to original literature. In KDE, $k = 3$, $\varepsilon = 0.1$, and in HSB, $k = 3$, I = {0, 0.2, 0.4, 0.6, 0.8, 1}.

4.3 Performance measures and significance statistical test

All experiments are carried out by employing 10-fold cross-validation. The confusion matrix in Table 3 shows four types of classification results. On this basis, two indicators geometric means metric (GM) and F-measure metric ($F1$) are used to evaluate the classification performance, and detailed definitions are shown in Eqs. (7)–(11). It can be seen from Eqs. (10) and (11) that GM considers the proportion of correctly classified instances in both minority and majority classes, while $F1$ focuses more on the average performance of precision and recall.

To evaluate if significant difference exists among experimental algorithms, it is necessary to use statistical tests. Here we adopt non-parametric statistical Friedman test and Bonferroni–Dunn post hoc test [63]. Friedman test is first employed to detect differences among all the algorithms in two indicators. After that, Bonferroni–Dunn is applied to check out if DBANN performs significantly better than comparative algorithms.

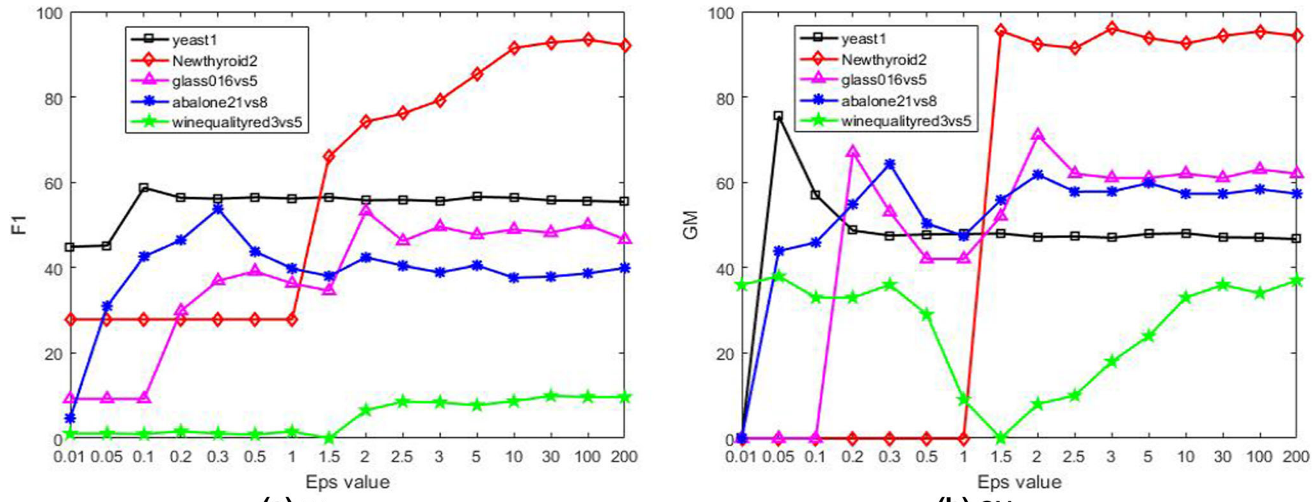
$$\text{Recall} = \text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}} \qquad (7)$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \qquad (8)$$

Table 4 Relationship among *eps*, clustering situation and classification performance on glass016vs5

No	Eps	C1	C2	C3	C4	C5	Noise	$F1$	GM
1	0.01						167	9.23	0.00
3	0.10						167	9.23	0.00
4	0.20	5	4				158	31.00	68.00
5	0.30	24	20	18	4	4	96	35.59	53.00
7	1.00	116	20	6			24	37.06	42.00
8	1.50	132	18				16	34.03	52.00
9	**2.00**	**152**	**4**				**10**	**53.53**	**72.96**
10	2.50	158	4				4	45.80	64.00
11	3.00	163					3	49.40	61.00
12	5.00	167						47.63	61.00
13	200.00	167						46.56	60.00

In boldface the best result is stressed



(a) F1 measure

(b) GM measure

Fig. 4 Performance of DBANN with different eps in $F1$ and GM

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{9}$$

$$F1 = \frac{2 \cdot \text{Recall} \cdot \text{Precision}}{\text{Recall} + \text{Precision}} \tag{10}$$

$$\text{GM} = \sqrt{\text{Sensitivity} \cdot \text{Specificity}} \tag{11}$$

To implement Friedman test, we first rank the performance of K algorithms on each dataset, the best performance ranks 1, the worst ranks K. When tie appears, average rank is assigned to each algorithm. Subsequently, we compute the Friedman statistic χ_F^2 by Eqs. (12) and (13). Specifically, r_{ij} denotes the rank of the jth of K algorithms on the ith of N datasets. As a result, R_j represents the average rank of the jth algorithm. Moreover, Iman and Davenport [64] found that Friedman's χ_F^2 was undesirably conservative and created a better statistic value F_F according to F-distribution with $(K-1)$ and $(K-1)(N-1)$ degrees of freedom as shown in Eq. (14). Critical value q_β is calculated by $q_\beta = F(\alpha, \mathrm{K}-1, (\mathrm{K}-1)(N-1))$. When $F_F > q_\beta$, null-hypothesis is rejected, i.e., significant difference exists among the comparative algorithms and vice versa. Furthermore, once the null-hypothesis is rejected, the post hoc tests Bonferroni–Dunn test is proceeded to conduct pairwise comparisons between DBANN and other algorithms. Here, critical value q_γ is based on the studentized range statistic divided by $\sqrt{2}$ [63]. The significant differences exist when average ranks of two algorithms differ by at least the critical difference (CD) [63]

$$R_j = \frac{1}{N}\sum_i r_i^j \tag{12}$$

$$\chi_F^2 = \frac{12N}{K(K+1)}\left(\sum_j R_j^2 - \frac{K(K+1)^2}{4}\right) \tag{13}$$

$$F_F = \frac{(N-1)\cdot\chi_F^2}{N(K-1)-\chi_F^2} \tag{14}$$

$$\text{CD} = q_\gamma\sqrt{\frac{K(K+1)}{6 \cdot N}} \tag{15}$$

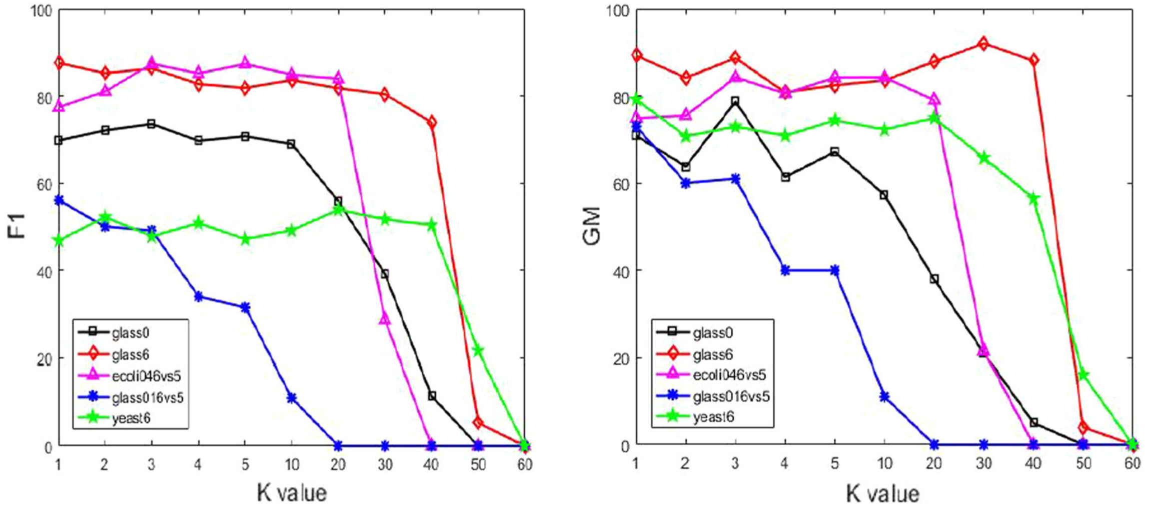
5 Results and discussion

5.1 Analyzing the critical parameters and property of DBANN

In this section, we provide an insight into detailed properties of DBANN. We first discuss the influence of parameter *eps* and parameter *k* on classification performance. Afterward, we investigate the distribution of query neighbors in DBANN. Finally, we analyze the advantage of DBANN over other *k*NN-based methods.

5.1.1 *eps* value

eps and *Minpts* are two input parameters in DBSCAN. Previous researches [55, 65] reported that *Minpts* has little impact on the clustering results. Therefore, in this section *Minpts* is set to 4 and *eps* is varied from 0.01 to 200 to analyze its influences on classification performance. Here we choose five real-world datasets for experiments and the



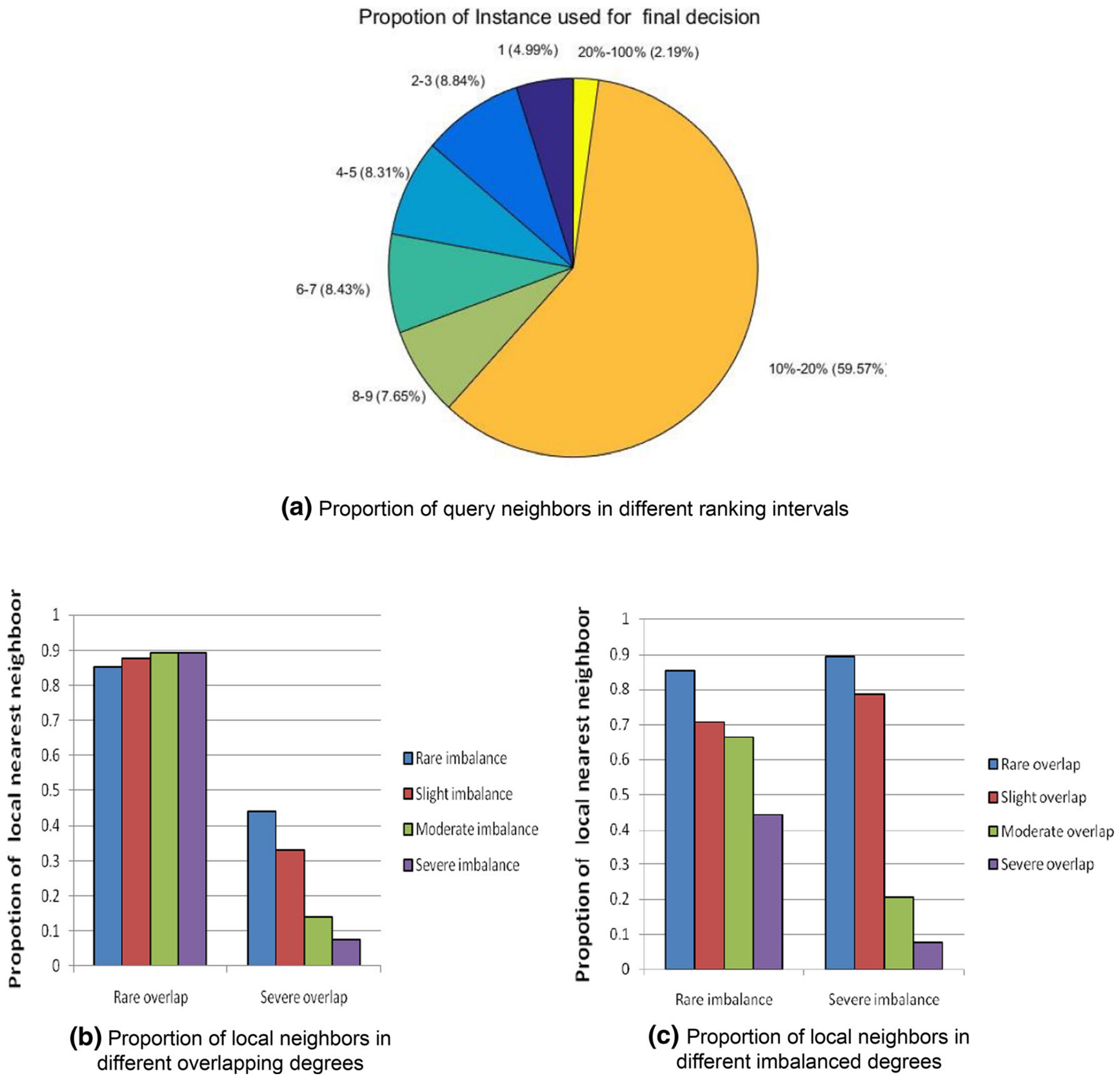
(a) F1 measure

(b) GM measure

Fig. 5 Performance of DBANN with different k in $F1$ and GM

results are shown in Fig. 4. It is easy to realize that *eps* is a sensitive parameter which dominates the performance of DBANN. In general, with the increase of *eps*, $F1$ and GM experience an increasing trend with fluctuation (here we call it phase I), and then the performance tends to be stable at a fixed range in the end (here we call it phase II). Noticeably, for some datasets (yeast1, glass016vs5, abalone21vs8) the optimal *eps* value exists in phase I, as for others (Newthyroid2, winequalityred3vs5) the optimal *eps* value exists in phase II. In this study, grid search is used to determine the optimal *eps* value.

In order to reveal the root cause behind the sensitivity of *eps* value, a further analysis is provided on relationship among *eps*, clustering situation and classification performance. We demonstrate this issue based on dataset glass016vs5 and the results are shown in Table 4. We find



(a) Proportion of query neighbors in different ranking intervals

(b) Proportion of local neighbors in different overlapping degrees

(c) Proportion of local neighbors in different imbalanced degrees

Fig. 6 Distribution of query neighbors in DBANN

Table 5 Imbalanced ratio (IR) and Fisher's discriminant ($F1$) ratio in different regions

Datasets	Whole region		Overlapping region		Non-overlapping region	
	IR	$F1$	IR	$F1$	IR	$F1$
glass1	1.80	0.1935	0.97	0.1766	3.00	0.2088
yeast2vs4	9.08	1.5793	1.07	1.0714	12.10	1.9242

that *eps* directly affects the clustering situation. When $eps < 0.1$, all instances are defined as noisy instances and no clusters are formed. With the increase of *eps*, more noisy instances are transferred to form clusters. When $eps = 0.3$ DBANN forms at most five clusters. Subsequently, clusters expand and merge until forming one big cluster with no noisy instance exists at last. As a result, $F1$ and GM also vary according to *eps* as shown in the last two columns in Table 4. Particularly, when $eps = 2$, DBANN achieves the optimal performance ($F1$ = 53.53%, GM = 72.96%) with two clusters and ten noisy instances.

As stated in Sect. 3.2, clustering results directly decide the choice of query neighbors. Therefore, *eps* which dominates the clustering situation is sensitive to classification performance of DBANN.

5.1.2 *k* value

To analyze the influence of k value, we fix the setting of *eps* at the optimal value and vary k, $k = 1.2.3, \ldots, 60$ on five real-world datasets. The classification results in Fig. 5 show that in most datasets, the classification performance

Table 6 Performance of *k*NN-based methods in different regions

Datasets	Method	Whole region		Overlapping region		Non-overlapping region	
		$F1$	GM	$F1$	GM	$F1$	GM
glass1	DBANN	76.23	79.72	**72.50**	**45.20**	78.44	82.93
	W-*k*NN	79.89	84.63	61.57	31.12	81.23	86.57
	*k*RNN	77.12	84.74	58.35	14.08	84.24	90.56
	*k*NN	72.96	75.98	33.16	14.08	86.92	89.63
	F-*k*NN	61.52	80.94	33.55	12.50	64.40	87.96
	H-*k*NN	82.48	90.65	71.22	20.00	86.92	89.63
yeast2vs4	DBANN	75.36	86.28	**71.00**	**82.79**	79.62	83.62
	W-*k*NN	71.48	83.09	55.95	53.44	80.16	81.70
	*k*RNN	78.87	82.84	64.16	65.03	80.16	81.70
	*k*NN	74.31	78.86	62.16	60.81	80.16	81.70
	F-*k*NN	42.05	80.25	60.47	63.19	40.18	84.68
	H-*k*NN	73.71	81.84	51.40	46.89	80.16	81.70

In boldface the best result in overlapping region is stressed

Table 7 Comparative results between DBANN and *k*NN-based methods in $F1$ on synthetic datasets

Datasets	DBANN	W-*k*NN	*k*RNN	*k*NN	F-*k*NN	H-*k*NN
A1	42.84	35.23	39.30	33.69	41.37	**43.68**
A2	**26.39**	19.97	15.75	12.42	18.64	17.57
A3	7.75	4.91	6.78	6.15	**8.48**	5.73
A4	**4.83**	2.03	3.73	0.00	1.08	0.60
B1	**47.23**	41.02	44.57	38.89	44.36	46.69
B2	**34.66**	29.24	29.29	24.73	27.57	26.05
B3	**16.87**	12.45	8.25	6.23	12.89	14.60
B4	**12.80**	2.71	11.40	1.72	4.03	8.25
C1	**75.66**	69.53	74.54	73.70	67.11	72.67
C2	**67.40**	59.33	65.23	60.37	55.80	57.37
C3	**56.28**	43.53	56.36	53.97	45.39	46.24
C4	**32.85**	22.83	28.46	26.50	27.18	23.32
D1	94.13	92.63	**94.44**	94.35	86.86	92.26
D2	95.88	**96.50**	94.88	94.93	89.56	95.18
D3	**89.37**	88.33	89.34	86.80	83.94	87.54
D4	85.53	85.13	**88.99**	88.64	82.56	83.90
Average rank	**1.5000**	4.1250	2.6250	4.5000	4.3750	3.8750
Final rank	**1**	4	2	6	5	3

In boldface the best result in *F1* is stressed drops with the increasing k value. Especially, when k reaches up to 60, the performances reduce to 0. This can be partly explained by the imbalanced solution of DBANN. Essentially, DBANN does not generate additional synthetic instances to compensate for minority class but increase the probability of minority class in query neighbors selection process. However, when k is too large, the proportion of minority instances in query neighbors cannot increase more even if the selection probability is 100% due to the imbalanced distribution, which results in performance loss. Based on our experience, when $k = 3$, the performance is desirable.

5.1.3 Distribution of reliable query neighbors

In this section, we investigate the distribution of query neighbors by a series of experiments. We set $k = 3$, *eps* at the optimal value in the whole experiments. We define the ranking for each training instance by sorting the distance from training instances to a query instance in ascending order, i.e., the nearest instance ranks 1.

We take glass0 dataset as an example to demonstrate this issue. We first implement DBANN on glass0 dataset by 10-fold cross-validation. For each run in the fold, we record the rankings of query neighbors for all query instances. After 10-fold cross-validation, we get the whole rankings. For example, we have 90 training instances and 10 query instances in onefold. In each run, we record the rankings of the three query neighbors for each query instance thus there are 30 rankings. After 10-fold cross-validation, there are totally 300 rankings which are used here for analysis. To facilitate the observation, we calculate the proportion of rankings in different intervals (1st, 2nd–3rd, 4th–5th, 6th–7th, 8th–9th, 10%th–20%th, 20%th–100%th), and the result is shown in Fig. 6a. Obviously, different from traditional *k*NN, the distribution of query neighbors of DBANN is not 100% k nearest neighbors anymore. Actually, the proportion of first three rankings (traditional *k*NN) only accounts for 13.83%, and the largest proportion of query neighbors distributes in the rankings in 10%th–20%th among all the training instances. Noticeably, the instance locates far away from the query instance

Table 8 Comparative results between DBANN and *k*NN-based methods in GM on synthetic datasets

Datasets	DBANN	W-*k*NN	*k*RNN	*k*NN	F-*k*NN	H-*k*NN
A1	54.38	48.66	54.66	45.14	57.94	**63.28**
A2	36.98	**40.25**	32.08	24.81	37.30	36.16
A3	**21.19**	15.81	14.05	11.40	20.17	15.66
A4	**13.12**	6.99	5.16	0.00	2.37	1.31
B1	62.44	56.31	59.39	49.96	62.23	**67.08**
B2	**56.37**	50.44	48.80	40.09	48.23	45.91
B3	**39.68**	31.80	18.08	11.65	30.45	32.10
B4	**33.60**	8.49	17.93	2.81	9.37	17.12
C1	85.31	79.84	84.07	79.36	87.55	**87.61**
C2	**83.86**	74.97	78.65	70.10	75.23	70.23
C3	**79.83**	64.86	71.65	65.46	69.28	65.02
C4	**66.04**	43.03	42.38	36.54	46.72	40.71
D1	96.61	95.71	96.81	95.99	**98.24**	96.97
D2	98.05	**98.36**	97.39	96.66	97.46	96.82
D3	**97.65**	95.04	95.03	91.11	95.28	93.28
D4	**96.37**	92.41	92.70	92.14	91.64	91.23
Average rank	**1.7500**	3.5625	3.4375	5.6875	2.875	3.6875
Final rank	**1**	4	3	6	2	5

In boldface the best result in GM is stressed

Table 9 Results of the Friedmen test and the Bonferroni–Dunn test among *k*NN-based methods on synthetic datasets ($\mathrm{CD} = 1.7038, q_\beta = 2.9013$)

Table	χ_F^2	F_F	W-*k*NN	*k*RNN	*k*NN	F-*k*NN	H-*k*NN	FR
Table 7	32.2857	10.1497	**2.6250**	1.1250	**3.0000**	**2.8500**	**2.3750**	$> q_\beta$
Table 8	37.8571	13.4746	**1.8125**	1.6875	**3.9375**	1.1250	**1.9375**	$> q_\beta$

In boldface the algorithm which has significant difference from DBANN is stressed (rankings in 20%th–100%th) also can be selected as query neighbors even though the proportion is only 2.19%. This partly shows that DBANN considers not only the local but also the global distribution in the selection of query neighbors.

To make a better understanding of the query neighbor selection mechanism, we expand the experiments on 16

Table 10 Comparative results between DBANN and *k*NN-based methods in $F1$ on real-world datasets

Datasets	IR	$F1$	DBANN	W-*k*NN	*k*RNN	*k*NN	F-*k*NN	H-*k*NN
poker8vs6	85.82	0.0153	2.27	**19.00**	0.00	0.00	0.00	5.00
yeast1458vs7	22.10	0.1757	**21.43**	10.58	7.33	0.00	0.00	10.71
glass1	1.80	0.1935	76.23	79.89	77.12	72.96	61.52	**82.48**
yeast1	2.46	0.2422	**56.26**	50.93	54.80	51.59	49.21	55.36
yeast0359vs78	9.12	0.3113	42.91	32.51	**49.30**	46.87	33.09	38.65
yeast1vs7	14.26	0.3522	**44.19**	30.76	27.71	24.71	23.88	35.74
yeast1289vs7	30.53	0.3654	**30.44**	25.78	13.00	14.00	4.86	19.17
abalone918	16.38	0.6311	**49.33**	35.53	41.81	28.24	42.64	45.06
glass0	2.06	0.6492	**74.60**	67.73	72.59	68.34	53.91	72.16
yeast0256vs3789	9.14	0.6939	59.20	56.04	**65.93**	62.32	41.30	57.15
winequalityred3vs5	68.10	0.7509	**7.66**	1.33	0.98	0.00	0.00	0.00
yeast05679vs4	9.33	1.0515	**55.02**	41.05	46.00	43.86	20.83	42.63
ecoli01vs235	9.17	1.1028	74.81	68.90	79.67	**79.67**	52.83	70.57
vehicle0	3.27	1.1220	85.03	87.17	88.31	89.78	83.16	**90.11**
yeast2vs8	23.10	1.1424	**66.33**	52.71	61.67	65.00	65.00	47.33
yeast4	28.09	1.2516	**47.66**	25.66	33.42	29.76	18.40	41.53
ecoli0146vs5	12.95	1.3345	77.33	83.33	**86.67**	**86.67**	55.14	**86.67**
ecoli01vs5	11.00	1.3897	82.33	81.33	**90.00**	**90.00**	56.02	82.67
ecoli3	8.57	1.3513	**69.00**	60.89	66.10	52.44	47.52	51.57
yeast2vs4	9.08	1.5793	75.36	71.48	**78.87**	74.31	42.05	73.71
ecoli046vs5	9.15	1.6030	82.05	81.33	**88.00**	84.67	60.10	81.33
ecoli0234vs5	9.10	1.6179	80.33	82.00	**87.33**	**87.33**	58.24	78.00
ecoli034vs5	9.00	1.6323	81.33	84.00	**88.33**	**88.33**	58.76	80.67
yeast02579vs368	9.13	1.6361	78.22	74.61	81.11	**82.14**	38.02	77.81
ecoli067vs5	10.00	1.6921	73.33	66.05	**79.67**	69.67	57.26	70.33
ecoli2	5.44	1.8199	87.08	84.13	**87.94**	87.53	75.19	83.07
glass016vs5	19.44	1.8505	41.67	35.00	38.33	28.33	21.67	**50.00**
yeast6	41.40	1.9674	50.67	54.09	**56.15**	54.10	43.94	51.69
ecoli0137vs26	39.14	2.3018	**46.67**	45.00	**46.67**	**46.67**	38.33	41.67
glass6	6.38	2.3913	84.05	82.67	79.67	82.33	55.03	**88.50**
abalone21vs8	40.50	2.4359	40.38	26.67	40.00	36.67	**48.33**	46.67
ecoli1	3.35	2.6396	77.43	70.97	**78.73**	76.47	66.28	69.42
yeast3	8.10	2.7512	**77.02**	67.80	75.53	72.88	74.46	64.74
ecoli4	15.75	3.2504	**86.00**	74.33	83.33	83.33	56.76	81.67
glass0123vs456	3.18	3.3137	90.27	83.13	87.39	86.40	76.30	**90.57**
wisconsin	1.85	3.5676	95.91	96.08	95.45	94.96	94.03	**96.15**
newthyroid2	5.14	3.5793	95.56	93.33	90.22	89.03	69.00	**98.57**
new-thyroid1	5.14	3.5793	95.14	92.57	94.66	93.24	89.62	**100.00**
ecoli0vs1	1.84	9.7145	**97.98**	94.88	97.90	97.23	79.98	95.54
shuttlec0vsc4	13.86	12.9722	97.32	**99.57**	**99.57**	**99.57**	34.15	99.17
iris0	2.04	16.7971	**100.00**	**100.00**	**100.00**	**100.00**	97.07	**100.00**
Average rank			**2.3902**	3.9878	2.5487	3.3292	5.5609	3.1829
Final rank			1	5	2	4	6	3

In boldface the best result in *F1* is stressed synthetic datasets (introduced in Sect. 4.1) which can be divided into four overlapping levels as well as four imbalanced levels. We define *k* nearest neighbors of a query instance (traditional *k*NN query neighbors) as local neighbors and then analyze the proportion of local neighbors on all the query neighbors in DBANN. Higher

Table 11 Comparative results between DBANN and *k*NN based methods in GM on real-world datasets

Datasets	IR	*F*1	DBANN	W-*k*NN	*k*RNN	*k*NN	F-*k*NN	H-*k*NN
poker8vs6	85.82	0.0153	**46.07**	31.14	0.00	0.00	0.00	7.05
yeast1458vs7	22.10	0.1757	**35.15**	21.88	11.37	0.00	0.00	19.47
glass1	1.80	0.1935	79.72	84.63	84.74	75.98	80.94	**90.65**
yeast1	2.46	0.2422	69.01	65.55	**69.61**	61.75	60.16	75.89
yeast0359vs78	9.12	0.3113	**67.80**	54.82	66.60	60.43	50.95	60.68
yeast1vs7	14.26	0.3522	**62.39**	55.56	36.27	30.64	38.68	49.96
yeast1289vs7	30.53	0.3654	**49.08**	44.79	17.13	17.13	11.42	28.55
abalone19	129.41	0.5292	**40.83**	4.98	0.00	0.00	0.00	0.00
glass0	2.06	0.6492	87.75	78.10	**91.42**	76.18	83.98	90.89
yeast0256vs3789	9.14	0.6939	78.22	74.79	**78.36**	72.25	74.70	72.40
winequalityred3vs5	68.10	0.7509	**40.83**	4.98	2.36	0.00	0.00	0.00
yeast05679vs4	9.33	1.0515	**73.05**	59.21	60.98	55.12	40.21	60.08
ecoli01vs235	9.17	1.1028	83.78	77.58	83.79	73.79	**87.47**	77.58
vehicle0	3.27	1.1220	**97.39**	93.29	96.65	95.33	92.59	95.01
yeast2vs8	23.10	1.1424	**71.97**	71.96	68.96	68.97	68.97	58.96
yeast4	28.09	1.2516	**76.39**	46.36	49.26	40.27	30.87	61.23
ecoli0146vs5	12.95	1.3345	**90.80**	84.69	87.76	87.76	90.78	87.76
ecoli01vs5	11.00	1.3897	90.74	84.58	**90.75**	**90.75**	90.70	87.66
ecoli3	8.57	1.3513	**91.37**	79.28	84.76	66.58	63.35	66.89
yeast2vs4	9.08	1.5793	**86.28**	83.09	82.84	78.86	80.25	81.84
ecoli046vs5	9.15	1.6030	90.65	84.42	**90.66**	87.54	90.59	84.42
ecoli0234vs5	9.10	1.6179	90.63	87.53	**90.64**	**90.64**	90.59	84.40
ecoli034vs5	9.00	1.6323	90.62	87.51	**90.64**	**90.64**	90.59	84.39
yeast02579vs368	9.13	1.6361	89.48	88.23	90.02	88.71	**90.37**	88.14
ecoli067vs5	10.00	1.6921	83.78	77.58	83.79	73.79	**87.47**	77.58
ecoli2	5.44	1.8199	93.91	93.89	**95.13**	91.82	94.05	90.60
glass016vs5	19.44	1.8505	**60.00**	50.00	50.00	40.00	30.00	**60.00**
yeast6	41.40	1.9674	**84.21**	74.19	74.87	67.85	66.15	70.65
ecoli0137vs26	39.14	2.3018	**50.00**	**50.00**	**50.00**	**50.00**	**50.00**	**50.00**
glass6	6.38	2.3913	91.36	91.37	85.20	85.22	**91.62**	91.38
abalone21vs8	40.50	2.4359	**64.02**	27.01	44.02	37.01	54.02	51.03
ecoli1	3.35	2.6396	**90.50**	83.11	88.50	84.01	84.55	77.98
yeast3	8.10	2.7512	**89.22**	81.35	86.21	81.51	87.62	77.64
ecoli4	15.75	3.2504	**93.93**	80.88	87.85	87.85	90.86	87.84
glass0123vs456	3.18	3.3137	**98.66**	89.28	93.25	90.58	91.75	93.32
wisconsin	1.85	3.5676	98.39	97.75	97.44	96.16	96.44	**98.71**
newthyroid2	5.14	3.5793	97.93	94.64	93.39	90.28	92.63	**98.43**
new-thyroid1	5.14	3.5793	96.38	93.06	94.66	93.08	94.32	**100.00**
ecoli0vs1	1.84	9.7145	97.98	97.00	97.02	96.06	**100.00**	97.98
shuttlec0vsc4	13.86	12.9722	**100.00**	99.55	99.55	99.55	96.77	99.13
iris0	2.04	16.7971	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
Average rank			**1.9268**	4.0243	2.8902	4.6097	3.9268	3.6219
Final rank			1	5	2	6	4	3

In boldface the best result in GM is stressed proportion indicates decision making depends more on local distribution. Conversely, lower proportion indicates decision making is more relied on global distribution.

From the results in Fig. 6b, we can see that when overlapping degree is rare, DBANN relies more on local neighbors with the proportion approximate to 0.85. In contrast, when overlap is severe, the proportion of local neighbors is significant lower on an average. This can partly show our advantage in query neighbors selection, i.e., when data distribution is tough, DBANN adaptively expands the detection radius to search for more reliable instances even though they locate far away. Besides, Fig. 6c shows the comparisons in different imbalanced degrees, and the graph clearly demonstrates that on rare and severe imbalanced degrees, the proportion of local neighbors drops with the increase in overlapping degree. However, on severe imbalanced datasets, the proportion of local neighbors on moderate and severe overlapping datasets is lower than that on rare imbalanced datasets. Above discussions imply that DBANN can adaptively select query neighbors from local to global region in different scenarios.

5.1.4 Advantage of DBANN over other *k*NN-based methods

To further study the advantage of the query neighbor selection mechanism in DBANN, we compare DBANN with other *k*NN-based algorithms on two typical datasets glass1 and yeast2vs4, which have different overlapping degrees and imbalance ratio, as a case study.

To better demonstrate this issue, we divide each dataset into overlapping region and non-overlapping region so as to take a closer look at the performance of each algorithm in different regions (whole region, overlapping region and non-overlapping region). Inspired by [66], we use *k*NN ($k = 5$) to separate the two regions. First of all, for each instance, it is considered to be in non-overlapping region if the instance and all its 5 nearest neighbors belong to the same class, otherwise, it is considered to be in overlapping region. Secondly, we calculate imbalance ratio (IR) and Fisher's discriminant ratio ($F1$) in different regions, respectively (Table 5). Finally, we run 6 *k*NN-based algorithms on two datasets and record their performances in different regions (Table 6). It is worth noting that $F1$ in Table 5 indicates overlapping degree while $F1$ in Table 6 indicates F-measure.

By observing Table 5, we note that the local distribution differs in different regions, in which the imbalance ratio is approximate to 1 in overlapping region while it is much higher in non-overlapping region. The overlapping degree ($F1$) is higher in overlapping region than other regions. All these results support the previous conclusion that the distribution of overlapping region is complex and hard to learn. This conclusion is also proved when we compare the performance of all *k*NN-based algorithms on two datasets in different regions in Table 6, in which $F1$ and GM value in overlapping region is significant lower than non-overlapping regions as well as the whole region. However, it is worth noting that DBANN performs better than other algorithms in overlapping region on both datasets. In glass1, DBANN achieves the best results ($F1$: 72.50, GM: 45.20) in overlapping region although its final result ($F1$: 76.23, GM: 79.72) only ranks fourth among all the algorithms. Meanwhile, it is witnessed that the performance of DBANN in non-overlapping region does not drop significantly compared with other algorithms. The same situation also occurs in yeast2vs4. Above results indicate that DBANN is able to excel in overlapping region at the cost of a small loss in non-overlapping region. This property is convinced as the main advantage of DBANN over remaining algorithms and we believe this property is beneficial from the adaptive query neighbors selection mechanism which is sensitive to the variation of local distribution.

5.2 Overall performance of DBANN

5.2.1 Performance on synthetic datasets

In this part, we validate the effectiveness of our proposed method by a bunch of experiments. Tables 7 and 8 show the comparison results of DBANN with *k*NN-based methods in $F1$ and GM on 16 synthetic datasets. The optimal result in each dataset is highlighted in bold-face. It can be found that DBANN performs better than other methods in

Table 12 Results of the Friedmen test and the Bonferroni–Dunn test among *k*NN-based methods on real-world datasets ($\mathrm{CD} = 1.0643, q_\beta = 2.45$)

Table	χ_F^2	F_F	W-*k*NN	*k*RNN	*k*NN	F-*k*NN	H-*k*NN	FR
Table 10	79.0924	25.1272	**1.5976**	0.1585	0.9390	**3.1707**	0.7927	$> q_\beta$
Table 11	53.3059	14.0562	**2.0975**	0.9634	**2.6829**	**2.0000**	**1.6951**	$> q_\beta$

In boldface the algorithm which has significant difference from DBANN is stressed

Table 13 Comparative results between DBANN and generality-oriented methods in *F*1 on real-world datasets

Datasets	IR	*F*1	DBANN	CART + SMOTE	CART + OBU	SVM + SMOTE	SVM + OBU	RUS	HSB	KEC
poker8vs6	85.82	0.0153	2.27	**41.89**	4.02	3.79	0.00	8.48	3.25	5.41
yeast1458vs7	22.10	0.1757	**21.43**	17.22	9.52	14.17	15.77	17.67	13.34	11.39
glass1	1.80	0.1935	**76.23**	71.37	62.81	56.07	57.95	69.98	60.50	72.54
yeast1	2.46	0.2422	56.26	50.73	51.16	**58.44**	53.36	49.89	49.06	51.89
yeast0359vs78	9.12	0.3113	**42.91**	23.06	28.84	37.04	32.62	33.47	30.53	26.54
yeast1vs7	14.26	0.3522	**44.19**	24.86	35.32	30.48	5.00	21.47	21.91	31.62
yeast1289vs7	30.53	0.3654	**30.44**	15.52	22.47	12.97	10.63	15.76	8.51	23.36
abalone918	16.38	0.6311	**49.33**	29.21	26.08	45.57	24.67	25.78	22.21	30.09
glass0	2.06	0.6492	74.60	75.06	68.30	65.39	65.69	76.93	70.12	**78.51**
yeast0256vs3789	9.14	0.6939	**59.20**	46.11	47.52	52.89	43.56	41.34	40.45	49.75
winequalityred3vs5	68.10	0.7509	7.66	1.25	0.00	5.07	0.00	1.90	2.82	**7.86**
yeast05679vs4	9.33	1.0515	**55.02**	43.39	39.09	43.27	26.27	42.05	39.48	46.19
ecoli01vs235	9.17	1.1028	74.81	61.36	61.81	72.31	**75.74**	66.33	47.40	65.67
vehicle0	3.27	1.1220	85.03	86.69	73.27	**94.72**	91.77	90.51	85.63	91.19
yeast2vs8	23.10	1.1424	**66.33**	42.60	34.89	47.52	65.00	32.21	13.88	45.38
yeast4	28.09	1.2516	**47.66**	32.46	20.13	27.47	19.52	29.62	22.65	30.20
ecoli0146vs5	12.95	1.3345	**77.33**	52.67	56.33	62.90	74.67	67.00	51.45	73.00
ecoli01vs5	11.00	1.3897	**82.33**	69.67	58.05	72.67	78.00	75.67	63.33	72.78
ecoli3	8.57	1.3513	**69.00**	46.74	44.84	58.82	42.08	55.00	49.69	63.57
yeast2vs4	9.08	1.5793	75.36	67.76	74.51	69.48	54.05	66.93	72.74	**79.52**
ecoli046vs5	9.15	1.6030	**82.05**	66.00	64.00	79.71	65.05	80.67	57.00	76.00
ecoli0234vs5	9.10	1.6179	**80.33**	69.05	74.67	72.19	68.24	76.67	53.90	73.33
ecoli034vs5	9.00	1.6323	**81.33**	67.67	74.33	72.57	69.05	76.33	61.54	74.67
yeast02579vs368	9.13	1.6361	78.22	74.09	71.66	71.09	77.44	70.45	58.46	**78.63**
ecoli067vs5	10.00	1.6921	**73.33**	64.38	69.33	63.00	72.05	70.67	51.73	69.33
ecoli2	5.44	1.8199	**87.08**	78.66	70.25	69.47	66.98	81.12	69.67	83.32
glass016vs5	19.44	1.8505	41.67	56.67	49.07	57.33	19.19	**73.33**	16.86	60.00
yeast6	41.40	1.9674	**50.67**	39.52	33.58	28.11	13.33	40.28	15.94	48.31
ecoli0137vs26	39.14	2.3018	**46.67**	38.33	29.00	31.19	38.33	40.00	21.62	23.33
glass6	6.38	2.3913	84.05	86.03	67.19	82.57	77.19	**86.50**	72.64	85.71
abalone21vs8	40.50	2.4359	40.38	37.57	55.33	**68.00**	36.67	46.52	23.36	47.54
ecoli1	3.35	2.6396	77.43	70.23	72.45	**78.00**	70.90	77.77	75.35	72.85
yeast3	8.10	2.7512	**77.02**	71.70	68.41	66.96	56.34	71.91	68.04	73.82
ecoli4	15.75	3.2504	**86.00**	74.00	60.67	82.38	43.33	82.67	42.00	68.00
glass0123vs456	3.18	3.3137	90.27	86.01	83.99	87.16	86.28	**91.36**	83.64	88.10

Table 13 (continued)

Datasets	IR	$F1$	DBANN	CART + SMOTE	CART + OBU	SVM + SMOTE	SVM + OBU	RUS	HSB	KEC
wisconsin	1.85	3.5676	**95.91**	92.07	90.90	95.67	94.18	95.22	91.29	94.58
newthyroid2	5.14	3.5793	95.56	93.17	96.03	**97.46**	**97.46**	92.07	81.57	92.13
new-thyroid1	5.14	3.5793	95.14	92.70	90.70	**98.89**	**98.89**	90.16	81.73	90.13
ecoli0vs1	1.84	9.7145	97.98	95.54	95.55	97.90	96.47	97.32	95.55	**98.56**
shuttlec0vsc4	13.86	12.9722	97.32	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
iris0	2.04	16.7971	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
Average rank			**2.2561**	4.9024	5.4390	4.0122	5.4024	3.8415	6.7317	3.4146
Final rank			1	5	7	4	6	3	8	2

In boldface the best result in $F1$ is stressed

almost all datasets in terms of average rank in $F1$ and GM. Particularly, when data distribution is severely overlapping, i.e., in datasets A1–A4, DBANN obtains the best average rank in both $F1$ (1.50) and GM (2.25). This implies the advantage of query neighbors selection mechanism in the face of extreme tough data distribution. When overlapping degree is moderate or slight (B1–B4, C1–C4), DBANN obtains the optimal results in all datasets except for GM in B1 and C1. As for imbalance issue, we observe that DBANN performs better in high imbalance ratio (1:9,1:19) with average rank 1 in GM while the average rank in low in imbalance ratio datasets (1:2,1:4) is 2.5. This demonstrates that DBANN has the ability to handle the high imbalanced distribution.

Moreover, to analyze statistical significance differences in comparative methods, Friedmen test (FR) is carried out. According to F-distribution, the critical value q_β is $F(0.05,\ 5 \times 15) = 2.9013$. From the results in Table 9, we can see $F_F > q_\beta$ in both $F1$ and GM which indicates that there are significant differences existing among all compared methods. Subsequently, the pairwise comparisons are conducted by Bonferroni–Dunntest. The critical value q_γ for two-tailed Bonferroni–Dunn test ($\alpha = 0.05$) with 6 algorithms is 2.576 [63]. We highlight the algorithms which are significantly different from DBANN in boldface. Concretely, differences exist in W-*k*NN, *k*NN, F-*k*NN, H-*k*NN in $F1$, and W-*k*NN, *k*NN, H-*k*NN in GM. Additionally, we notice that DBANN seems similar with *k*RNN with regard to $F1$ and GM in statistical test. However, we know that the targets and structures of two algorithms are totally different. *k*RNN tends to bias the posterior probability estimation toward the minority class based on local distribution to handle imbalanced problem whereas DBANN aims to boost performance by searching for reliable query neighbors in both local and global distribution by additionally considering overlapping issue.

5.2.2 Performance on real-world datasets

In this section, we compare DBANN with *k*NN-based methods as well as generality-oriented methods on 41 real-world datasets. Tables 10 and 11 show that the average ranks of GM and $F1$ of DBANN are 2.3902 (1) and 1.9268 (1), respectively, indicating that DBANN achieves better performance than other *k*NN-based methods. In order to obtain clear insights into the behaviors of DBANN, we analyze the results in different distributions by means of a statistical study. In the first place, by observing the overlapping issue (high overlapping degree: $F1 < 1.6$, low overlapping degree: $F1 \geq 1.6$), we note that the average ranks of DBANN are 2.15 and 2.2 with respect to $F1$ and GM in the high overlapping datasets while the average

Table 14 Comparisons of DBANN with generality-oriented methods in GM on real-world datasets

Datasets	IR	*F*1	DBANN	CART + SMOTE	CART + OBU	SVM + SMOTE	SVM + OBU	RUS	HSB	KEC
poker8vs6	85.82	0.0153	46.07	54.90	19.93	24.96	0.00	31.14	**67.03**	31.13
yeast1458vs7	22.10	0.1757	35.15	22.81	13.02	65.63	0.00	38.92	**76.82**	25.15
glass1	1.80	0.1935	**79.72**	59.13	77.66	68.12	70.33	72.63	70.56	73.90
yeast1	2.46	0.2422	**69.01**	43.06	55.70	63.88	53.84	61.10	68.25	61.16
yeast0359vs78	9.12	0.3113	67.80	26.24	33.75	71.84	20.45	54.19	**76.66**	43.38
yeast1vs7	14.26	0.3522	62.39	32.17	45.19	79.02	31.28	44.26	**80.85**	54.97
yeast1289vs7	30.53	0.3654	49.08	16.37	29.40	**69.29**	28.11	42.35	59.46	50.48
abalone918	16.38	0.6311	71.46	2.48	41.81	76.15	30.49	48.37	**76.79**	55.47
glass0	2.06	0.6492	**87.75**	32.42	71.07	82.11	78.09	83.80	86.63	83.95
yeast0256vs3789	9.14	0.6939	78.22	72.67	51.83	69.49	32.18	62.32	**82.32**	65.45
winequalityred3vs5	68.10	0.7509	40.83	49.84	0.00	**79.91**	0.00	10.74	60.39	12.04
yeast05679vs4	9.33	1.0515	73.05	37.95	42.25	72.30	5.74	56.49	**80.92**	57.54
ecoli01vs235	9.17	1.1028	**83.78**	50.30	74.26	79.50	79.51	80.68	83.77	80.67
vehicle0	3.27	1.1220	**97.39**	74.22	84.89	93.52	93.52	94.35	94.61	94.01
yeast2vs8	23.10	1.1424	71.97	**84.89**	44.44	54.23	54.26	54.96	64.84	61.97
yeast4	28.09	1.2516	76.39	54.23	32.04	73.50	31.22	62.19	**82.81**	49.97
ecoli0146vs5	12.95	1.3345	**90.80**	59.62	58.84	84.38	74.04	83.86	90.75	77.75
ecoli01vs5	11.00	1.3897	**90.74**	63.90	58.63	84.33	79.11	87.63	87.62	87.64
ecoli3	8.57	1.3513	**91.37**	48.48	83.60	90.86	76.38	70.73	81.29	73.17
yeast2vs4	9.08	1.5793	86.28	66.84	77.57	80.94	46.43	79.80	**95.26**	87.39
ecoli046vs5	9.15	1.6030	90.65	68.95	73.65	84.19	73.64	**93.74**	87.44	87.52
ecoli0234vs5	9.10	1.6179	**90.63**	68.39	73.65	84.15	68.36	87.49	84.32	87.51
ecoli034vs5	9.00	1.6323	90.62	68.33	73.67	84.14	68.34	84.37	**93.70**	87.49
yeast02579vs368	9.13	1.6361	89.48	78.51	75.30	86.93	69.01	82.50	**92.96**	85.53
ecoli067vs5	10.00	1.6921	**83.78**	74.22	74.26	79.50	79.51	80.68	83.77	80.67
ecoli2	5.44	1.8199	93.91	82.18	73.47	90.69	57.11	86.88	**94.18**	89.16
glass016vs5	19.44	1.8505	60.00	60.00	**90.00**	80.00	60.00	**90.00**	40.00	70.00
yeast6	41.40	1.9674	84.21	49.71	51.35	**84.83**	45.89	80.44	80.28	63.50
ecoli0137vs26	39.14	2.3018	50.00	50.00	40.00	50.00	50.00	50.00	**70.00**	30.00
glass6	6.38	2.3913	91.36	89.30	85.59	85.63	89.27	89.33	89.30	**93.41**
abalone21vs8	40.50	2.4359	64.02	54.91	69.82	**84.91**	4.89	67.01	81.01	74.01
ecoli1	3.35	2.6396	90.50	65.07	68.82	**95.14**	67.95	86.04	88.64	80.41
yeast3	8.10	2.7512	89.22	72.53	70.52	86.41	40.12	80.36	**93.21**	83.84
ecoli4	15.75	3.2504	**93.93**	74.22	69.68	94.85	34.23	90.89	76.95	80.88
glass0123vs456	3.18	3.3137	**98.66**	82.86	81.06	90.40	90.96	94.92	90.62	91.07

Table 14 (continued)

Datasets	IR	$F1$	DBANN	CART + SMOTE	CART + OBU	SVM + SMOTE	SVM + OBU	RUS	HSB	KEC
wisconsin	1.85	3.5676	98.39	85.98	98.70	95.68	**99.35**	96.81	93.53	96.14
newthyroid2	5.14	3.5793	**97.93**	94.21	97.11	97.11	97.11	96.38	91.81	94.66
newthyroid1	5.14	3.5793	96.38	93.40	89.72	**100.00**	**100.00**	96.36	93.89	92.60
ecoli0vs1	1.84	9.7145	**97.98**	94.33	96.13	96.13	90.90	97.96	97.96	**97.98**
shuttlec0vsc4	13.86	12.9722	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
iris0	2.04	16.7971	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
Average rank			**2.3780**	6.3780	6.0365	3.7195	6.4024	4.0975	2.8780	4.10975
Final rank			1	7	6	3	8	4	2	5

In boldface the best result in GM is stressed

ranks in the low overlapping datasets are 2.61 and 2.68, respectively. These results support the ability of DBANN in face of high overlapping distribution. Moreover, it is also witnessed that when high overlap and high imbalance co-occur, i.e., IR > 20 and $F1 < 1.6$, DBANN still outperforms most of other methods. Especially, in datasets yeast1458vs7, yeast1289vs7, winequalityred3vs5, yeast2-vs8 and yeast4, DBANN obtain the optimal results. This good behavior is due to the query neighbors selection mechanism of DBANN which helps to provide query neighbors with more reliable information when minority class is scarce and distribution is overlapping. Again, we implement Friedmen test (FR), Bonferroni-Dunntest (BD) ($q_\beta = F(0.05, 5 \times 40) = 2.45$,$q_\gamma = 2.576$) on real-world datasets and find that DBANN presents significant difference from W-*k*NN, F-*k*NN in $F1$ and W-*k*NN, *k*NN, F-*k*NN and H-*k*NN in GM among *k*NN-based methods, as shown in Table 12. As for generality-oriented methods, DBANN also achieves superior performance which is listed in Tables 13 and 14. Especially, in terms of $F1$, DBANN gets the smallest average rank 2.2561 which is superior to the second rank 3.4146 by a large margin. Likewise, the significant test is listed in Table 15 which indicates that there are differences between DBANN and most of the algorithms except for HSB, SVM + SMOTE in GM, and KEC in $F1$.

6 Conclusions

In this study, we propose a novel method DBANN to deal with both imbalanced and overlapping problems. The main idea of DBANN is to find the most reliable query neighbors by using density-based methods. We first divide the training data into six parts by DBSCAN, and then in each part we assign reliable degree to instances based on density, class imbalance as well as overlapping situation. Afterward, we adjust the distance metric according to reliable degree to make reliable instances more likely to be selected as query neighbors. Finally, output is made by reliable query neighbors.

Different from existing *k*NN-based methods, DBANN takes advantage of both local and global information in query neighbors selection. Additionally, noise factor is also considered in DBANN to boost the classification performance. It is worth noting that the query neighbors in our method change adapt according to data distribution. To validate the effectiveness of DBANN, we implement experiments on both synthetic datasets and real-world datasets. The results show that our method outperforms *k*NN-based methods as well as generality-oriented methods in terms of $F1$ and GM.

Table 15 Results of the Friedmen test and the Bonferroni–Dunn test among generality-oriented methods on real-world datasets ($\mathrm{CD} = 1.4552, q_\beta = 2.25$)

Table	χ_F^2	F_F	RUS	HSB	KEC	CART + SMOTE	CART + OBU	SVM + SMOTE	SVM + OBU	FR
Table 13	93.769	19.4108	**1.5854**	**4.4757**	1.1586	**2.6464**	**3.1830**	**1.7561**	**3.1464**	$> q_\beta$
Table 14	120.0164	28.7493	**1.7195**	0.5000	**1.7317**	**4.0000**	**3.6585**	1.3415	**4.0244**	$> q_\beta$

In boldface the algorithm which has significant difference from DBANN is stressed

Further research is required to extend DBANN to multi-class classification problems in the future. Moreover, we also plan to implement other density-based clustering methods in the framework of DBANN. Besides, it is interesting to set up a specific public datasets for algorithms comparisons on overlapping problems.

Acknowledgements The authors would like to thank the editor and reviewers for their useful comments and suggestions, which are of great help in improving the quality of the paper. This work is financially supported by the National Science Foundation of China (NSFC Proj. 71831006, 71801065, and 71771070), Zhejiang Provincial Natural Science Foundation of China under Grant No. LZ20G010001 and the Promotion China Ph.D Program from BMW Briliance Automotive.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Qiwei H, Chakhar S, Siraj S, Labib A (2017) Spare parts classification in industrial manufacturing using the dominance-based rough set approach. Eur J Oper Res 262(3):1136–1163
2. Li Z, Wang Y, Wang K (2019) A deep learning driven method for fault classification and degradation assessment in mechanical equipment. Comput Ind 104:1–10
3. Lei K, Xie Y, Zhong S, Dai J, Yang M, Shen Y (2019) Generative adversarial fusion network for class imbalance credit scoring. Neural Comput Appl 32:8451–8462
4. Villuendas-Rey Y, Rey-Benguría CF, Ferreira-Santiago Á, Camacho-Nieto O, Yáñez-Márquez C (2017) The naïve associative classifier (NAC): a novel, simple, transparent, and accurate classification model evaluated on financial data. Neurocomputing 265:105–115
5. Shoaran M, Haghi BA, Taghavi M, Farivar M, Emami-Neyestanak A (2018) Energy-efficient classification for resource-constrained biomedical applications. IEEE J Emerg Sel Top Circuits Syst 8(4):693–707
6. Lowrance CJ, Lauf AP (2019) An active and incremental learning framework for the online prediction of link quality in robot networks. Eng Appl Artif Intell 77:197–211
7. Guo H, Li Y, Shang J, Mingyun G, Huang Y, Bing G (2017) Learning from class-imbalanced data: review of methods and applications. Expert Syst Appl 73:220–239
8. Nekooeimehr I, Lai-Yuen SK (2016) Adaptive semi-unsupervised weighted oversampling (A-SUWO) for imbalanced datasets. Expert Syst Appl 46:405–416
9. Jian C, Gao J, Ao Y (2016) A new sampling method for classifying imbalanced data based on support vector machine ensemble. Neurocomputing 193:115–122
10. Raj V, Magg S, Wermter S (2016) Towards effective classification of imbalanced data with convolutional neural networks. In: IAPR workshop on artificial neural networks in pattern recognition. Springer, pp 150–162
11. Khan SH, Hayat M, Bennamoun M, Sohel FA, Togneri R (2018) Cost-sensitive learning of deep feature representations from imbalanced data. IEEE Trans Neural Netw Learn Syst 29(8):3573–3587
12. García S, Zhang Z-L, Altalhi A, Alshomrani S, Herrera F (2018) Dynamic ensemble selection for multiclass imbalanced datasets. Inf Sci 445:22–37
13. Zhang Z, Krawczyk B, Garcìa S, Rosales-Pérez A, Herrera F (2016) Empowering one-vs-one decomposition with ensemble learning for multi-class imbalanced data. Knowl Based Syst 106(C):251–263
14. Zhang ZL, Luo XG, González S, García S, Herrera F (2018) DRCW-ASEG: one-versus-one distance-based relative competence weighting with adaptive synthetic example generation for multi-class imbalanced datasets. Neurocomputing 285(12):176–187
15. Denil M, Trappenberg T (2010) Overlap versus imbalance. In: Canadian conference on artificial intelligence. Springer, pp 220–231
16. Tang Y, Gao J (2007) Improved classification for problem involving overlapping patterns. IEICE Trans Inf Syst 90(11):1787–1795
17. Peng P, Wang J (2019) Wear particle classification considering particle overlapping. Wear 422(423):119–127
18. Liu CL (2006) Artificial neural networks in pattern recognition. In: Second IAPR workshop on artificial neural networks in pattern recognition (ANNPR 2006), pp 37–146
19. Chowdhury SA, Stepanov EA, Danieli M et al (2019) Automatic classification of speech overlaps: feature representation and algorithms. Comput Speech Lang 55:145–167
20. Podder A, Latha N (2017) Data on overlapping brain disorders and emerging drug targets in human Dopamine Receptors Interaction Network. Data Br 12:277–286
21. López V, Fernández A, García S, Palade V, Herrera F (2013) An insight into classification with imbalanced data: empirical results and current trends on using data intrinsic characteristics. Inf Sci 250:113–141
22. García V, Sánchez J, Mollineda R (2007) An empirical study of the behavior of classifiers on imbalanced and overlapped data sets. In: Iberoamerican congress on pattern recognition. Springer, pp 397–406
23. Prati RC, Batista GE, Monard MC (2004) Class imbalances versus class overlapping: an analysis of a learning system behavior. In: Mexican international conference on artificial intelligence. Springer, pp 312–321
24. Yu Q, Hongye S, Guo L, Chu J (2011) A novel svm modeling approach for highly imbalanced and overlapping classification. Intell Data Anal 15(3):319–341

25. Alejo R, Valdovinos RM, García V, Horacio Pacheco-Sanchez J (2013) A hybrid method to face class overlap and class imbalance on neural networks and multi-class scenarios. Pattern Recogn Lett 34(4):380–388
26. Wasikowski M, Chen X (2010) Combating the small sample class imbalance problem using feature selection. IEEE Trans Knowl Data Eng 22(10):1388–1400
27. Xia S-Y, Xiong Z-Y, He Y, Li K, Dong L-M, Zhang M (2014) Relative density-based classification noise detection. Optik Int J Light Electron Opt 125(22):6829–6834
28. Sáez JA, Luengo J, Stefanowski J, Herrera F (2015) SMOTE–IPF: addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering. Inf Sci 291:184–203
29. Orriols-Puig A, Bernadó-Mansilla E, Goldberg DE, Sastry K, Lanzi PL (2009) Face twise analysis of XCS for problems with class imbalances. IEEE Trans Evol Comput 13(5):1093–1119
30. Prati RC, Batista GE, Monard MC (2004) Learning with class skews and small disjuncts. In: Brazilian symposium on artificial intelligence. Springer, pp 296–306
31. Adams N (2010) Dataset shift in machine learning. J R Stat Soc Ser A (Stat Soc) 173(1):274
32. Subbaswamy A, Saria S (2018) Counterfactual normalization: proactively addressing dataset shift and improving reliability using causal mechanisms. arXiv preprint arXiv:1808.03253
33. Ho TK, Basu M (2002) Complexity measures of supervised classification problems. IEEE Trans Pattern Anal Mach Intell 24(3):1–300
34. Batista GE, Prati RC, Monard MC (2004) A study of the behavior of several methods for balancing machine learning training data. ACM SIGKDD Explor Newsl 6(1):20–29
35. Fernández A, del Jesus MJ, Herrera F (2015) Addressing overlapping in classification with imbalanced datasets: a first multi-objective approach for feature and instance selection. In: International conference on intelligent data engineering and automated learning. Springer, pp 36–44
36. Alshomrani S, Bawakid A, Shim S-O, Fernández A, Herrera F (2015) A proposal for evolutionary fuzzy systems using feature weighting: dealing with overlapping in imbalanced datasets. Knowl Based Syst 73:1–17
37. Xiong H, Wu J, Liu L (2010) Classification with class overlapping: a systematic study. In: Proceedings of the 1st international conference on E-business intelligence (ICEBI2010). Atlantis Press
38. Vorraboot P, Rasmequan S, Chinnasarn K, Lursinsap C (2015) Improving classification rate constrained to imbalanced data between overlapped and non-overlapped regions by hybrid algorithms. Neurocomputing 152:429–443
39. Weiss GM (2004) Mining with rarity: a unifying framework. ACM SIGKDD Explor Newsl 6(1):7–19
40. Vuttipittayamongkol P, Elyan E, Petrovski A, Jayne C (2018) Overlap-based undersampling for improving imbalanced data classification. In: International conference on intelligent data engineering and automated learning. Springer, Cham, 2018
41. Liu N, Xing X, Li Y, Zhu A (2019) Sparse representation based image super-resolution on the *k*nn based dictionaries. Opt Laser Technol 110:135–144
42. Kuzhali SE, Suresh DS (2018) Patch-based denoising with *k*-nearest neighbor and SVD for microarray images. In: Computer science on-line conference. Springer, pp 132–147
43. Kriminger E, Principe JC, Lakshminarayan C (2012) Nearest neighbor distributions for imbalanced classification. In: The 2012 international joint conference on neural networks (IJCNN). IEEE, pp 1–5
44. García V, Mollineda RA, Sánchez JS (2008) On the *k*-nn performance in a challenging scenario of imbalance and overlapping. Pattern Anal Appl 11(3–4):269–280
45. Dubey H, Pudi V (2013) Class based weighted *k*-nearest neighbor over imbalance dataset. In: Pacific-Asia conference on knowledge discovery and data mining. Springer, pp 305–316
46. Harshita P, Thakur GS (2016) A hybrid weighted nearest neighbor approach to mine imbalanced data. In: Proceedings of the international conference on data mining (DMIN). The Steering Committee of the World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), p 106
47. Harshita P, Thakur GS (2018) An improved fuzzy K-nearest neighbor algorithm for imbalanced data using adaptive approach. IETE J Res 2018:1–10
48. Zhang X, Li Y (2011) A positive-biased nearest neighbor algorithm for imbalanced classification. In: Pacific-Asia conference on knowledge discovery and data mining. Springer, pp 293–304
49. Zhang X, Li Y, Kotagiri R, Lifang W, Tari Z, Cheriet M (2017) *k* rare-class nearest neighbor classification. Pattern Recogn 62:33–44
50. Mullick SS, Datta S, Das S (2018) Adaptive learning-based *k*-nearest neighbor classifiers with resilience to class imbalance. IEEE Trans Neural Netw Learn Syst 99:1–13
51. Wang J, Neskovic P, Cooper LN (2007) Improving nearest neighbor rule with a simple adaptive distance measure. Pattern Recogn Lett 28(2):207–213
52. İnkaya T (2015) A density and connectivity based decision rule for pattern classification. Expert Syst Appl 42(2):906–912
53. Van Hulse J, Khoshgoftaar TM, Napolitano A (2010) A novel noise filtering algorithm for imbalanced data. In: 2010 9th international conference on machine learning and applications. IEEE, pp 9–14
54. Kang Q, Chen XS, Li S, Zhou M (2017) A noise filtered under-sampling scheme for imbalanced classification. IEEE Trans Cybern 47(12):4263–4274
55. Schubert E, Sander J, Ester M, Kriegel HP, Xiaowei X (2017) Dbscan revisited, revisited: why and how you should (still) use dbscan. ACM Trans Database Syst (TODS) 42(3):19
56. Czerniawski T, Sankaran B, Nahangi M, Haas C, Leite F (2017) 6D DBSCAN-based segmentation of building point clouds for planar object classification. Autom Constr 88:44–58
57. Das B, Krishnan NC, Cook DJ (2014) Handling imbalanced and overlapping classes in smart environments prompting dataset. In: Yada K (ed) Data mining for service. Springer, Berlin, pp 199–219
58. Alcalafdez J, Sánchez L, García S, Del Jesus MJ, Ventura S, Garrell JM, Otero J, Romero C, Bacardit J, Rivas VM (2009) KEEL: a software tool to assess evolutionary algorithms for data mining problems. Soft Comput 13(3):307–318
59. Chawla NV, Bowyer KW, Hall LO, Philip Kegelmeyer W (2002) Smote: synthetic minority over-sampling technique. J Artif Intell Res 16:321–357
60. Zhang J, Shi H (2019) Kd-tree based efficient ensemble classification algorithm for imbalanced learning. In: 2019 international conference on machine learning, big data and business intelligence (MLBDBI), pp 203–207
61. Lu Y, Cheung YM, Tang YY (2016) Hybrid sampling with bagging for class imbalance learning. In: Pacific-Asia conference on knowledge discovery and data mining. Springer International Publishing
62. Seiffert C, Khoshgoftaar TM, Van Hulse J, Napolitano A (2010) RUSBoost: a hybrid approach to alleviating class imbalance. IEEE Trans Syst Man Cybern Part A Syst Hum 40(1):185–197
63. Demšar J (2010) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30

64. Iman RL, Davenport JM (1980) Approximations of the critical region of the fbietkan statistic. Commun Stat Theory Methods 9(6):571–595
65. Ester M, Kriegel H-P, Sander J, Xu X et al (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. KDD 96:226–231
66. Bader-El-Den M, Teitei E, Perry T (2019) Biased random forest for dealing with the class imbalance problem. IEEE Trans Neural Netw Learn Syst 30(7):2163–2172